Sparse Recovery Using Sparse Matrices

Citation: Gilbert, Anna, and Piotr Indyk. "Sparse Recovery Using Sparse Matrices." Proceedings of the IEEE 98.6 (2010): 937–947. Web. ©2010 IEEE.

As Published: http://dx.doi.org/10.1109/jproc.2010.2045092

Publisher: Institute of Electrical and Electronics Engineers

Persistent URL: http://hdl.handle.net/1721.1/70932

Version: Final published version: final published article, as it appeared in a journal, conference proceedings, or other formally published context

INVITED PAPER

Sparse Recovery Using Sparse Matrices

Significant results in predicting the operation of equipment such as network routers, or the results of group testing for defective items, can often be obtained from a few samples.

By Anna Gilbert and Piotr Indyk

ABSTRACT | In this paper, we survey algorithms for sparse recovery problems that are based on *sparse* random matrices. Such matrices has several attractive properties: they support algorithms with low computational complexity, and make it easy to perform incremental updates to signals. We discuss applications to several areas, including compressive sensing, data stream computing, and group testing.

KEYWORDS | Compressive sensing; expanders; sparse matrices; sparse recovery; streaming algorithms

I. INTRODUCTION

The past several years have seen a new approach to the acquisition of compressible signals. Traditional approaches first capture the entire signal and then process it for compression, transmission, or storage. In comparison, the new approach obtains a succinct approximate representation directly by acquiring a small number of nonadaptive linear measurements of the signal. For any signal x, of length n, the representation is equal to Ax, where A is an $m \times n$ matrix. The vector Ax is often referred to as the *measurement vector* or *sketch* of x. Although m is typically much smaller than n, the sketch Ax contains plenty of useful information about the signal x. In particular, the sketch of x retains enough inherent information that we can directly obtain a *sparse approximation* or compressed form of the signal.

This approach has been discovered and explored extensively in several different research communities, including theoretical computer science, applied mathematics, and digital signal processing. The goal of that research is to obtain encoding and recovery schemes with good compression rate (i.e., short sketch lengths) as well as good algorithmic properties (i.e., low encoding, update and recovery times).

Linear sketches have found numerous uses in several areas, including *compressive sensing*, *data stream computing*, and *combinatorial group testing*.

- *Compressive sensing*. In this area [12], [16], the signal or image x is acquired using (analog or digital) hardware, which (approximately) computes a dot product of each row of the matrix A and the signal at a unit cost. Once we obtain the measurement vector Ax, we process it digitally to extract information about the signal, including significant coefficients in an orthonormal basis (e.g., wavelet or Fourier), as well as the original signal. Frequently, the number of measurements we obtain with compressed sensing hardware is much less than that of traditional hardware devices. Nevertheless, we can sample a bandlimited analog signal at a sub-Nyquist rate and still recover significant frequencies in the signal or the entire signal spectrum. For examples of compressive sensing hardware, see [13], [31], [41], and [42].
- *Data stream computing*. In this area [7], [24], [33], the vectors x are often very large, and cannot be represented explicitly. For example, in network measurement, x_i could denote the total number of packets with destination i passing through a network router. Storing such vector x itself is typically infeasible due to its large size: each destination is represented by an IP address that is 32-bit long, and therefore the vector x has dimension $n = 2^{32}$. Thus, it is preferable to maintain a lower dimensional sketch

Manuscript received April 20, 2009; revised November 22, 2009; accepted February 26, 2010. Date of current version May 19, 2010. This work was supported by a David and Lucille Packard Fellowship, the MADALGO (Center for Massive Data Algorithmics, funded by the Danish National Research Association), and the National Science Foundation (NSF) under Grants CCF-0728645, CCF-0910765, and DMS-0547744.

A. Gilbert is with the Department of Mathematics, University of Michigan, Ann Arbor, MI 48109-0937 USA (e-mail: annacg@umich.edu).

P. Indyk is with the Department of Electrical Engineering and Computer Science, MIT, Cambridge, MA 02139-4307 USA (e-mail: indyk@mit.edu).

Digital Object Identifier: 10.1109/JPROC.2010.2045092

0018-9219/$26.00 ©2010 IEEE

Ax instead and recover an approximation to x from the sketch. However, it must be possible to maintain such sketch under incremental updates to x. For example, if a new packet arrives, the corresponding coordinate of x is incremented by 1, which should be reflected in the sketch Ax. Fortunately, this can be easily done if the sketching procedure is linear. Specifically, let Δ_i denote the update to the vector x after seeing a packet with destination i (i.e., $\Delta_i = 1$ and $\Delta_j = 0$ for $j \neq i$). Then, we have $A(x + \Delta_i) = Ax + A\Delta_i$. Since $A\Delta_i$ is simply the ith column of A, updating the sketch can be accomplished by simply adding that column to the current sketch Ax. See, e.g., [18] and [30] for more information about using data stream algorithms for network measurement.

- *Combinatorial group testing*. In pooling designs or more generally combinatorial group testing [15], the vector x represents a universe of n items in total. Moreover, we know k of the elements are *defective*. More specifically, the vector x is the characteristic vector for the defective set so that $x \in \{0,1\}^n$ has exactly k entries that are 1 and $(n - k)$ zeros. The goal of combinatorial group testing is to construct a collection of tests (called a *design*) to minimize the number of tests needed to find the defective set for the worst case input. The tests are represented by a matrix A that is binary, with the jth column of the ith rows equal to 1 if and only if the jth item is used by the ith test. In the simplest (boolean) setting, each test returns 1 if at least one of the elements used in the test is defective. In our setting we assume the linear model, where each test returns the number of defective elements. Note that each such test corresponds to taking the dot product of x and a test vector, and therefore is captured in the linear sketching model.[1] See, e.g., [17], [28], and [36] for further discussion and recent developments in the area.

In each of these applications, it is useful (and often crucial) that the measurement matrix A be a *sparse matrix*, i.e., contain very few nonzero elements per column. In data stream processing, the time needed to update the sketch Ax under the update Δ_i is proportional to the number of nonzero elements in the vector $A\Delta_i$, which is equal to the number of nonzeros in the ith column of A. In experiment pooling, the design matrix A is a binary matrix that captures which compounds are pooled together and the measurements Ax reflect the activity levels of the pooled compounds. In many chemical and biological applications, the assumption that compound activity is a linear function of the concentration holds only when there are not many compounds mixed together in a single pool; thus, the design matrix A should be not only binary but also sparse. In other applications, sparsity can be useful for computational reasons: one can compute the matrix–vector product Ax very quickly.[2]

A. Definitions and Classification of the Results

Formally, we define the sparse recovery problem as follows. Let $\mathrm{Err}_q^k = \mathrm{Err}_q^k(x)$ be the smallest possible ℓ_q approximation error $\|x - x'\|_q$, where x' ranges over all k-sparse vectors (i.e., that have at most k nonzero entries). Our goal is, given Ax, to find a vector $\hat{x}$ such that the ℓ_p approximation error[3] $\|x - \hat{x}\|_p$ is at most $c > 0$ times $\mathrm{Err}_q^k(x)$, i.e.,

$$\|\hat{x} - x\|_p \leq c \cdot \mathrm{Err}_q^k(x). \tag{1}$$

Note that for any value of p, the error $\|x - \hat{x}\|_p$ is minimized when $\hat{x}$ consists of the k largest (in magnitude) coefficients of x. We refer to such $\hat{x}$ as the "head" of the signal x, while $x - \hat{x}$ will be called the "tail" of x.

As mentioned earlier, we aim to design sparse recovery schemes that achieve short sketches, have low algorithmic complexity, and provide "good" recovery guarantees. In addition, the schemes described in this survey can be classified based on other characteristics, such as 1) whether the schemes are randomized or deterministic, or 2) how general is the class of signals x supported by the schemes. In the following, we elaborate on both issues.

- Randomization: we distinguish between two classes of schemes: *for-each* and *for-all*. The latter describes a scheme in which *one* matrix A works for *all* signals x. In the former case, the matrix A is chosen at random from some distribution, and *for each* signal x, the recovery algorithm works "with high probability" (at least $1 - 1/n$).[4] Naturally, schemes with the for-all property are preferable to those with the for-each guarantee (if all other parameters are the same).We note that "for-all" does not mean that the matrix is constructed in an "explicit" or efficient manner. In fact, most of the constructions presented here use the probabilistic method. Although it is possible to construct recovery schemes explicitly [2], [14], [34], such schemes tend to require more measurements.

[1] In fact, we can assume an even more general setting, where we allow a *general* vector $x \in \mathbb{R}^n$, and our goal is to identify the top k most significant coefficients from the set of linear measurements. This is applicable in a setting where the entries in x represent the activity level of n compounds, or a genetic response in a biological sample.

[2] Specifically, the matrix–vector product can be computed in time $O(ns)$, where s is the column sparsity of A. As we will see in Section III, in many settings, one can achieve $s = O(\log(n/k))$, which leads to the running time of $O(n\log(n/k))$. This compares favorably to the $O(nm)$ time achievable for random Gaussian matrices, or to the $O(n \log n)$ time achievable for random Fourier-like matrices.

[3] It is natural to consider $p = q$. However, as we will see later, other guarantees are also possible.

[4] We adopt here the terminology frequently used in computer science. Note that one could require weaker probability bounds, e.g., $1 - o(1)$. However, all algorithms presented in this survey naturally achieve the stronger probability bound without changing the (asymptotic) bound on the number of measurements.

- Generality of supported signals: ideally, the recovery schemes should support arbitrary signals x. In this survey, we focus on describing such schemes. However, there has been plenty of work on algorithms supporting more restrictive classes of signals. In particular, there have been several schemes based on sparse matrices that work for (almost) exactly k-sparse signals [27], [29], [32], [37]–[39], [43], [44]. Although we do not cover them in detail, we point out relevant connections and references whenever possible.

B. Survey Summary

We present an overview of the algorithms for sparse recovery that utilize sparse measurement matrices. The description is divided into two sections: for-each algorithms are covered in Section II, while for-all algorithms are described in Section III. Historically, most of the for-each schemes have been developed in the data stream community during 2001–2004. In contrast, most of the algorithms with for-all guarantees have been discovered after 2004, during the process of unifying the ideas of compressive sensing and data stream algorithms. We present the algorithms in the same chronological order.

Almost all schemes described here offer sketch length bounds of $O(k \log n)$ or less, which matches or is close to the lower bound of $\Omega(k \log(n/k))$ shown in [4]. They are supported by efficient algorithms, with running times ranging from polynomial in n to near-linear in n. They offer a variety of approximation guarantees, starting from a"plain vanilla" guarantee of (1) with $p = q = 1$ (the l_1/l_1-guarantee) to more complex (but often stronger) ones. The exact sketch length bounds, approximation guarantees, and algorithm running times are stated in Theorems 1–10.

Due to lack of space, we focus on describing only those algorithms that achieve the best known bounds and solve the sparse recovery problem formulated earlier in this section. See [19] and [20] for some of the earlier work on closely related problems, such as recovering good piecewise constant approximations from a sketch of a signal.

II. ALGORITHMS WITH FOR-EACH GUARANTEES

In this section, we describe algorithms that provide for-each guarantees. The algorithms were discovered and described in the context of data stream computing. The descriptions provided here are sometimes simpler than the original versions, since we ignore various issues specific to data streams (such as how to generate the random matrix A using few random bits, how to update the sketch under incremental changes to x, etc.).

A. Count-Min and Count-Median

The count-min and count-median algorithms [9] utilize sparse random matrices where each entry is either 0 or 1. Both algorithms use the same distribution of matrices, and differ only in the details of the recovery algorithms.

Each matrix A is generated in the following way. Let w be a parameter, and let h be any function from the set H of all functions $h : \{1, \ldots, n\} \to \{1, \ldots, w\}$. Each such function defines a $w \times n$ 0–1 matrix $A(h)$, such that $(A(h))_{j,i}$ is equal to 1 if $j = h(i)$, and is equal to 0 otherwise. Note that each column has exactly one 1.

To create the matrix A, we choose d functions $h_1, \ldots, h_d$ independently and uniformly at random from H. Then, we define A to be a vertical concatenation of matrices $A(h_1), \ldots, A(h_d)$. Note that the number of rows in the matrix A is equal to $m = wd$.

For intuition about the construction, observe that, for any signal x, and $j = 1, \ldots, w$, $l = 1, \ldots, d$, we have

$$(Ax)_{(l-1)w+j} = (A(h_l)x)_j = \sum_{i:h_l(i)=j} x_i.$$

That is, the coordinate of the sketch corresponding to the function h_l and value j is simply the sum of all values x_i such that i is mapped to j by h_l. For a fixed value of i, the sums $\sum_{t:h_l(t)=h_l(i)} x_t$ contain approximations of x_i, contaminated by other coordinates mapped together with i. As a result, "aggregating" those sums over different h_l provides an approximation of x_i. Different aggregation methods will lead to different algorithms.

Count-Min: The count-min algorithm [9] (see also [18]) works under the assumption that $x \geq 0$. In this case, computing the approximation x^* from Ax is particularly simple: we define

$$x_i^* = \min_l (A(h_l)x)_{h_l(i)} = \min_l \sum_{i':h_l(i')=h_l(i)} x_{i'}.$$

The guarantees for the estimator x^* can be derived as follows. First, observe that $x_i \leq x_i^*$, since the entries $x_{i'}$ contaminating the estimation of x_i can only increase the value of the estimator x_i^*. Thus, the estimator $(A(h_l)x)_{h_l(i)}$ with the minimum value provides the smallest approximation error. Moreover, for any coordinate x_i and function index l, we have

$$\begin{aligned}\mathbf{E}\left[(A(h_l)x)_{h_l(i)} - x_i\right] &= \sum_{i \neq i'} \mathbf{Pr}[h_l(i) = h_l(i')] x_{i'} \\ &\leq \frac{1}{w}\|x\|_1.\end{aligned}$$

By Markov inequality

$$\mathbf{Pr}\left[(A(h_l)x)_{h_l(i)} - x_i \geq \frac{2}{w}\|x\|_1\right] \leq 1/2$$

and, therefore

$$\mathbf{Pr}\left[x_i^* - x_i \geq \frac{2}{w}\|x\|_1\right] \leq 1/2^d.$$

For $d = C\log n$, we have that the above guarantee holds for *all* $i = 1, \ldots, n$ with probability $1 - n/2^d = 1 - 1/n^{C-1}$. Thus, with the same probability, we have

$$\|x^* - x\|_\infty \leq \frac{2}{w}\|x\|_1.$$

The disadvantage of the above guarantee is that the error is a function of the norm of the whole vector x, not its tail. However, the probability that any of the entries in the head of x contaminate an estimator of a specific x_i is at most k/w. Thus, a slightly more refined analysis[5] shows that, for $w = 4/\alpha \cdot k$, $\alpha \in (0,1)$, we have

$$\mathbf{Pr}\left[x^* - x_i \geq \alpha/k \cdot \mathrm{Err}_1^k\right] \leq 1/2^d.$$

For $d = C\log n$, this implies

$$\|x^* - x\|_\infty \leq \alpha/k \cdot \mathrm{Err}_1^k$$

with probability $1 - n/2^d = 1 - 1/n^{C-1}$.

Count-Median: The count-min algorithm can be extended to work for general signals [9]; the extension is often referred to as the *count-median* algorithm. The main issue to take care of is that, for general vectors x, the inequality $x_i^* \geq x_i$ no longer holds, since the entries contaminating the estimator might be negative. As a result, we cannot aggregate using min. Instead, we replace the estimator x^* by

$$\left(x^*_{\mathrm{med}}\right)_i = \mathrm{median}_l (A(h_l)x)_{h_l(i)}.$$

By using the Chernoff bound, we show that, with high probability, the majority of the estimators $(A(h_l)x)_{h_l(i)}$ (and therefore their median) have small error. Specifically, we can show that for any constant $C' > 0$, there exists C such that if we set $d = C\log n$, then

$$\left\|x^*_{\mathrm{med}} - x\right\|_\infty \leq \alpha/k \cdot \mathrm{Err}_1^k$$

with probability $1 - 1/n^{C'}$.

Theorem 1: There exists a distribution over $m \times n$ matrices A, $m = O(k/\alpha \cdot \log n)$, such that for any signal x, given Ax, we can recover $\hat{x} = x^*_{\mathrm{med}}$ such that

$$\|\hat{x} - x\|_\infty \leq \alpha/k \cdot \mathrm{Err}_1^k$$

with high probability. The column sparsity of A is $O(\log n)$, and the time needed to recover $\hat{x}$ from Ax is $O(n\log n)$.

We conclude by observing that the approximation guarantee in the above theorem implies a weaker but perhaps more intuitive guarantee about the l_1 approximation error. Consider the vector $\hat{x}$ consisting of the k largest (in magnitude) elements of x^*_{med}. Then, we have

$$\|x - \hat{x}\|_1 \leq (1 + 3\alpha)\mathrm{Err}_1^k.$$

To show this, let S be the set of the k largest in magnitude coordinates of x, and let $\hat{S}$ be the support of $\hat{x}$. Note that $\|\hat{x}_S\|_1 \leq \|\hat{x}_{\hat{S}}\|_1$. We have

$$\begin{aligned}\|x - \hat{x}\|_1 &\leq \|x\|_1 - \|x_{\hat{S}}\|_1 + \|x_{\hat{S}} - \hat{x}_{\hat{S}}\|_1\\ &\leq \|x\|_1 - \|\hat{x}_{\hat{S}}\|_1 + 2\alpha\mathrm{Err}_1^k\\ &\leq \|x\|_1 - \|\hat{x}_S\|_1 + 2\alpha\mathrm{Err}_1^k\\ &\leq \|x\|_1 - \|x_S\|_1 + 3\alpha\mathrm{Err}_1^k\\ &\leq (1 + 3\alpha)\mathrm{Err}_1^k.\end{aligned}$$

For more detailed descriptions of the algorithms, see [6], [9], and [18].

B. Count-Sketch

The next[6] algorithm, called *count-sketch* [6], provides error guarantees that are a function of Err_2^k as opposed to Err_1^k. This is accomplished by using a distribution over matrices A very similar to those used by count-min, with one difference: each nonzero entry is chosen independently and uniformly at random from $\{-1, 1\}$ (instead just being equal to 1). Formally, let $r_{i,l}$ be independent random variables with values chosen uniformly at random from $\{-1, 1\}$, and let the functions $h_1, \ldots, h_d$ be defined as in the previous section. Then, the matrix A is a vertical concatenation of matrices $A(h_1), \ldots, A(h_d)$, where $(A(h_l))_{j,i}$ is equal to $r_{i,l}$ if $j = h_l(i)$, and is equal to 0 otherwise. To estimate the coordinate x_i, one then uses the median estimator

$$x^*_{\mathrm{med}'} = \mathrm{median}_l\ r_{i,l}(A(h_l)x)_{h_l(i)}.$$

[5] The argument is essentially a simplified version of the argument used in [6]. See [10] or [24, Lecture 4] for the proof.

[6] Chronologically, the count-sketch algorithm has been invented before count-min. It is easier, however, to describe the ideas in the reverse order.

The analysis of count-sketch relies on the observation that

$$\mathbf{Pr}\left[\left(\left(r_{i,l}A(h_l)x\right)_{h_l(i)} - x_i\right)^2 \geq C/w \cdot \mathrm{Err}_2^k\right] \leq 1/4$$

for some absolute constant $C > 0$. The final guarantee is captured by the following theorem.

Theorem 2: There exists a distribution over $m \times n$ matrices A, $m = O(k/\alpha \log n)$, such that for any signal x, given Ax, we can recover $\hat{x}$ such that

$$\|\hat{x} - x\|_\infty^2 \leq \alpha/k \cdot \left(\mathrm{Err}_2^k\right)^2$$

with high probability. The column sparsity of A is $O(\log n)$, and the time needed to recover $\hat{x}$ from Ax is $O(n \log n)$.

As before, the approximation guarantee in the theorem implies a weaker but more intuitive guarantee, this time about the l_2 approximation error. Consider the vector $\hat{x}$ consisting of the k largest (in magnitude) elements of $x^*_{\mathrm{med}'}$. Then, we have [11]

$$\|x - \hat{x}\|_2^2 \leq (1 + 9\sqrt{\alpha})\left(\mathrm{Err}_2^k\right)^2.$$

The proof proceeds as follows. Let $E = \mathrm{Err}_2^k$. Let S be the set of k largest (in magnitude) coordinates of x, and let $\hat{S}$ be the support of $\hat{x}$. Moreover, for any set P, let $-P$ denote the complement of P. We have

$$\|x - \hat{x}\|_2^2 \leq \left\|(x - \hat{x})_{\hat{S}}\right\|_2^2 + \|x_{S-\hat{S}}\|_2^2 + \left\|x_{-(S\cup\hat{S})}\right\|_2^2. \qquad (2)$$

The first term is bounded by $k\alpha/k \cdot E^2 = \alpha E^2$. To bound the second term, we proceed as follows. Consider any $i \in S - \hat{S}$ and $j \in \hat{S} - S$. We have

$$|x_i| - |x_j| \leq |\hat{x}_i| - |\hat{x}_j| + 2\sqrt{\alpha/k}E \leq 2\sqrt{\alpha/k}E. \qquad (3)$$

Let $a = \max_{i \in S-\hat{S}} |x_i|$ and $b = \min_{j \in \hat{S}-S} |x_j|$. From (3), we have $a \leq b + 2\sqrt{\alpha/k}E$. Thus

$$\|x_{S-\hat{S}}\|_2^2 \leq a^2|S - \hat{S}| \leq (b + 2\sqrt{\alpha/k}E)^2|S - \hat{S}|.$$

Since $\|x_{\hat{S}-S}\|_2^2 \geq b^2|\hat{S} - S| = b^2|S - \hat{S}|$, we continue

$$\begin{aligned}\|x_{S-\hat{S}}\|_2^2 &\leq \left(\|x_{\hat{S}-S}\|_2 \Big/ \sqrt{|S - \hat{S}|} + 2\sqrt{\alpha/k}E\right)^2 |S - \hat{S}| \\ &\leq \left(\|x_{\hat{S}-S}\|_2 + 2\sqrt{\alpha}E\right)^2 \\ &\leq \|x_{\hat{S}-S}\|_2^2 + 4\|x_{\hat{S}-S}\|_2\sqrt{\alpha}E + 4\alpha E^2 \\ &\leq \|x_{\hat{S}-S}\|_2^2 + 4\sqrt{\alpha}E^2 + 4\alpha E^2 \\ &\leq \|x_{\hat{S}-S}\|_2^2 + 8\sqrt{\alpha}E^2.\end{aligned}$$

Plugging into (2), we get

$$\begin{aligned}\|x - \hat{x}\|_2^2 &\leq \alpha E^2 + \|x_{\hat{S}-S}\|_2^2 + 8\sqrt{\alpha}E^2 + \left\|x_{-(S\cup\hat{S})}\right\|_2^2 \\ &\leq 9\sqrt{\alpha}E^2 + \|x_{-S}\|_2^2 \\ &= (1 + 9\sqrt{\alpha})E^2.\end{aligned}$$

C. Sublinear Algorithms

The above algorithms all run in time at least linear in the signal size as they entail estimating a value for each coordinate in the signal, even those that are insignificant. If our goal is to just report k nonzero terms of k-sparse approximation, then it is sufficient to find (or approximate) the top k values only to achieve similar error guarantees. Sublinear algorithms aim to do just that and to do so in time that scales polynomially with the number of terms k desired and logarithmically with the length of the input signal.

We start with the simplest example of a sublinear algorithm and its associated binary measurement matrix to find the unique nonzero entry in a signal of length n and sparsity 1. Let B be the binary matrix with ith column given by the binary representation of i, beginning with the first column $i = 0$. We refer to this matrix as a *bit-tester* matrix.[7] We add a row of 1's to the bit-tester matrix (to estimate the signal value) and refer to this matrix as B_1. It has $\log(n) + 1$ rows and n columns and from the measurements $B_1 x$ of a vector x with a single large entry, we can determine both the value of the entry and its position in time $\log(n) + 1$. The measurements are simply the position in binary plus an estimate of the signal value and the recovery algorithm is trivial. It also can be seen that a similar approach applies even if the signal x is not exactly 1-sparse, but contains some "small" amount of noise.

For general signals, the approach is to "augment" the algorithms and measurement matrix constructions from previous sections with the matrix B_1. Recall that those algorithms used simple hash functions which map signal

[7] Readers familiar with coding theory might recognize B as the parity-check matrix of the Hamming code.

coordinates x_i to rows j of the measurements. Implicit in the correctness proofs was the ability of those hash functions to *isolate* a few significant signal values from one another. More precisely, if h is chosen uniformly at random from a prespecified family H of hash functions, then

$$\mathbf{Pr}[h(i) = h(i')] = \frac{O(1)}{w}$$

for some w; that is, the probability that positions i and i' are hashed into the same measurement is low. Using arguments similar to those above, we can show that if there are only k large (or nonzero) entries that are hashed into more than k measurements, then with high probability, a large fraction of the significant entries are hashed into separate measurements. We can view this process as a random masking of the original signal, leaving a signal with only one significant entry, to which we can apply the bit-tester matrix. More precisely, each row of our final matrix M is the pointwise (Hadamard) product between a row in A and a row in B_1. We say that M is the row tensor product of B_1 and A, $M = B_1 \bigotimes_r A$. Note that M has approximately $k \log(n)$ rows.

Once we have a good estimate of a large fraction of the significant entries, we can subtract their contribution from the original measurements (exploiting the linearity of the measurement process algorithmically, in addition to its role in the application). We then repeat the process, using "fresh" measurements.

By using the above techniques, we obtain the following result [21].

Theorem 3: There exists a distribution over $m \times n$ matrices A, $m = O(k \log n)$, such that for any signal x, given Ax, we can recover $\hat{x}$ such that $\|\hat{x} - x\|_1 \le C\mathrm{Err}_1^k$ with high probability. The column sparsity of A is $O(\log^c n)$ for some constant c, and the time needed to recover $\hat{x}$ from Ax is polynomial in k and $\log n$.

III. ALGORITHMS WITH FOR-ALL GUARANTEES

In this section, we describe algorithms that provide for-all guarantees. The algorithms have been discovered during the process of unifying the ideas of compressive sensing with those from data stream algorithms. The key part of that process has been to identify concrete properties that 1) hold for a random sparse matrix with a nonzero probability and 2) are sufficient to support efficient and accurate recovery algorithms.

One such property is based on the notion of *graph expansion* [2], [44]. Consider a bipartite graph $G = G(A)$ between two node sets U and V, with $|U| = n$ and $|V| = m$, such that an edge (i, j) belongs to G if and only if $A_{j,i} = 1$. Informally, such a graph is an *expander*, if each small enough set of the nodes in U has many neighbors in V (the formal definition is provided below).

The notion of expansion has been known to be useful for some related problems, such as constructing low-density parity-check codes. In fact, iterative decoding algorithms for such codes have been used, e.g., in [25], [27], and [44], to design sparse recovery algorithms. However, those algorithms were designed and proven to work only for the case where the signal x is either exactly k-sparse or "almost" k-sparse. In contrast, the algorithms we present here work for arbitrary input signals x.

Formally, we define *unbalanced expander graphs* as follows. Consider a bipartite graph $G = (U, V, E)$, where $E \subset U \times V$ is the set of edges. We refer to U as the "left" part, and refer to V as the "right" part of the graph. A vertex belonging to the left (right) part is called a left (right) vertex. In our constructions, the left part will correspond to the set $\{1, 2, \ldots, n\}$ of coordinate indexes of vector x, and the right part will correspond to the set of row indexes of the measurement matrix. A bipartite graph is called *left-d-regular* if every vertex in the left part has exactly d neighbors in the right part.

Definition: A bipartite, left-d-regular graph $G = (U, V, E)$ is an (s, d, ϵ)-*expander* if any set $S \subset U$ of at most s left vertices has at least $(1 - \epsilon)d|S|$ neighbors.

The algorithms described in this section use adjacency matrices A of the expanders graphs G: we simply set $A_{j,i} = 1$ if and only if $(i, j) \in E$. Note that the resulting matrices are sparse, with exactly d ones per column.

What are the achievable expansion parameters? Since expander graphs are meaningful only when $|V| < d|U|$, some vertices must share neighbors, and hence the parameter ϵ cannot be smaller than $1/d$. Using the probabilistic method one can show that there exist (s, d, ϵ)-expanders with $d = O(\log(n/s)/\epsilon)$ and $m = |V| = O(s \log(n/s)/\epsilon^2)$. Since our constructions require $s = O(k)$ and ϵ strictly bounded away from zero, the resulting matrices will have $O(k \log(n/k))$ rows.

For many applications, one often needs an *explicit* expander, i.e., an expander for which we can efficiently compute the neighbor set of a given left vertex. No explicit constructions with the aforementioned parameters are known. However, it is known [23] how to explicitly construct expanders with left degree $d = O((\log |U|)(\log s)/\epsilon)^{1+1/\alpha}$ and right set size $(d^2 s^{1+\alpha})$, for any fixed $\alpha > 0$. For simplicity, in the remainder of this paper, we will assume expanders with the optimal parameters.

Unlike in the for-each case,[8] the algorithms in this section are known to be resilient to the measurement noise. That is, we could assume that we are given a *noisy* sketch vector $b = Ax + \mu$, where μ is the "measurement noise" vector. In that case, the error bounds in the approximation

[8] It should be noted that, although the for-each algorithms have not been typically analyzed for the case of noisy sketches, the algorithm themselves could very well be quite resilient to various forms of noise.

guarantees would have an additional term depending on $\eta = \|\mu\|_1/d$. However, for the sake of consistency, we will focus the description on the noise-free case where $b = Ax$. The reader is referred to the original papers for the bounds for the noise-resilient variants of the algorithms.

A. RIP(1) and ℓ_1-Minimization

In this section, we give an overview of the "geometric" approach to sparse recovery using sparse matrices, introduced in [2]. The approach uses the ℓ_1-minimization algorithm that has been earlier shown to work for random dense matrices [12], [16]. In the noiseless case $b = Ax$, the algorithm proceeds by finding $\hat{x}$ such that $A\hat{x} = b$ and $\|\hat{x}\|_1$ is minimized.

To understand when the above algorithm performs an accurate recovery, we need the following generalized definition of the *restricted isometry property*.

Definition 2: An $m \times n$ matrix A is said to satisfy RIP(p, k, δ) if, for any k-sparse vector x, we have

$$\|x\|_p(1-\delta) \leq \|Ax\|_p \leq \|x\|_p.$$

For the case of $p = 2$, the notion was introduced[9] in [12], which also showed that if a matrix A satisfies this property, then the ℓ_1-minimization procedure produces an accurate solution. Since then there has been a tremendous amount of study of the properties and construction of RIP$(2, k, \delta)$ [or RIP(2), for short] matrices. Unfortunately, sparse matrices cannot satisfy the RIP(2) property, unless their number of rows is "large" [8]. In particular, sparse 0–1 matrices must have at least $\Omega(k^2)$ rows.

However, it was shown [2] that such matrices *can* satisfy RIP(p) for p equal (or very close) to 1. In particular, the adjacency matrices of expander graphs do have this property.[10] By earlier arguments, such matrices have $O(k \log(n/k))$ rows, which translates into $O(k\log(n/k))$ sketch length bound.

Lemma 4: Consider any $m \times n$ matrix A that is the adjacency matrix of an (k, d, ϵ)-unbalanced expander $G = (U, V, E)$. Then, the scaled matrix A/d satisfies the RIP$(1, k, \delta)$ property for $\delta = 2\epsilon$.

Proof: Let $x \in \mathbb{R}^n$ be a k-sparse vector. Without loss of generality, we assume that the coordinates of x are ordered such that $|x_1| \geq \ldots \geq |x_n|$. We order the edges $e_t = (i_t, j_t)$, $t = 1, \ldots, dn$ of G in a lexicographic manner. It is helpful to imagine that the edges $e_1, e_2, \ldots$ of E are being added to the (initially empty) graph. An edge $e_t = (i_t, j_t)$ causes a *collision* if there exists an earlier edge $e_s = (i_s, j_s)$, $s < t$, such that $j_t = j_s$. We define E' to be the set of edges which do *not* cause collisions, and $E'' = E - E'$.

Claim 5: We have

$$\sum_{(i,j)\in E''} |x_i| \leq \epsilon d \|x\|_1.$$

Proof: For each $t = 1, \ldots, dn$, we use an indicator variable $r_t \in \{0, 1\}$, such that $r_t = 1$ iff $e_t \in E''$. Define a vector $z \in \mathbb{R}^{dn}$ such that $z_t = |x_{i_t}|$. Observe that

$$\sum_{(i,j)\in E''} |x_i| = \sum_{e_t=(i_t,j_t)\in E} r_t |x_{i_t}| = r \cdot z$$

To upper bound the latter quantity, observe that the vectors satisfy the following constraints:

- the vector z is nonnegative;
- the coordinates of z are monotonically nonincreasing, and at most kd of them are nonzero;
- for each *prefix set* $P_i = \{1, \ldots, di\}$, $i \leq k$, we have $\|r_{|P_i}\|_1 \leq \epsilon di$—this follows from the expansion properties of the graph G;
- $r_{|P_1} = 0$, since the graph is simple.

It follows that for any r, z satisfying the above constraints, we have $r \cdot z \leq \|z\|_1 \epsilon$. Since $\|z\|_1 = d\|x\|_1$, the lemma follows. ■

Since

$$\|Ax\|_1 \geq \sum_{e_t=(i_t,j_t)\in E'} |x_{i_t}| - \sum_{e_t=(i_t,j_t)\in E''} |x_{i_t}|$$

then Claim 5 immediately implies that $\|Ax\|_1 \geq d\|x\|_1(1-2\epsilon)$. Since for any x we have $\|Ax\|_1 \leq d\|x\|_1$, it follows that A/d satisfies the RIP$(1, k, 2\epsilon)$ property. ■

We now need to show that the RIP(1) property of the matrix A is sufficient to guarantee that the ℓ_1-minimization works. First, we show that any vector from the kernel of an adjacency matrix A of an expander graph (i.e., such that $Ax = 0$) is "smooth," i.e., the ℓ_1-norm of the vector cannot be concentrated on a small subset of its coordinates. An analogous result for RIP(2) matrices and with respect to the ℓ_2-norm has been used before to show guarantees for LP-based recovery procedures.

Lemma 6: Consider any $y \in \mathbb{R}^n$ such that $Ay = 0$, and let S be any set of k coordinates of y. Then, we have

$$\|y_S\|_1 \leq \alpha(\epsilon)\|y\|_1$$

where $\alpha(\epsilon) = (2\epsilon)/(1-2\epsilon)$.

[9] The original paper [12] employed a slightly different notation using "double sided error," i.e., requiring that $\|x\|_2(1-\delta') \leq \|Ax\|_2 \leq \|x\|_2(1+\delta')$. The two definitions can be seen to be equivalent, by scaling A and setting $(1+\delta) = (1+\delta')/(1-\delta')$.

[10] In fact, for some range of parameters, the opposite holds, i.e., 0–1 matrices that satisfy RIP(1) are adjacency matrices of expander graphs. See [2] and [8] for more details.

The proof proceeds by showing that any vector y whose ℓ_1-norm is concentrated on a small set S of coordinates cannot satisfy $Ay = 0$. This is because [by the RIP(1) property] the ℓ_1-norm of the vector $A(y_S)$ is "large," and (from the expansion property of the underlying graph) the contribution of the coordinates in the complement of S is not sufficient to reduce Ay to 0. See [2] for the formal proof.

The "smooth kernel" property is then used, as in prior work, to provide recovery guarantees for the ℓ_1-minimization. This is achieved by the following lemma, by setting $u = x$ and $v = \hat{x}$.

Lemma 7: Consider any two vectors u, v, such that for $y = v - u$, we have $Ay = 0$, and $\|v\|_1 \leq \|u\|_1$. Let S be the set of k largest (in magnitude) coefficients of u, then

$$\|v - u\|_1 \leq 2/(1 - 2\alpha(\epsilon)) \cdot \|u - u_S\|_1.$$

The following theorem summarizes the discussion.

Theorem 8: There exists an $m \times n$ (expander) matrix A, $m = O(k \log(n/k)/\epsilon^2)$, such that for any signal x, given Ax, we can recover $\hat{x}$ such that

$$\|x - \hat{x}\|_1 \leq c(\epsilon)\mathrm{Err}_1^k$$

where $c(\epsilon) \to 2$ as $\epsilon \to 0$. The column sparsity of A is $O(\log(n)/\epsilon^2)$, and the recovery involves solving a linear program with $O(n)$ variables and $O(m + n)$ constraints.

This concludes the overview of the results of [2]. Further studies of ℓ_1-minimization algorithms for sparse matrices have been done in [43] and [29], where the authors obtained tight estimates for the number of measurements needed to recover signals of given sparsity. The papers consider somewhat different setups: in [43], one allows arbitrary sparse signals x and measurements contaminated by random Gaussian noise; in [29], the authors consider sparse nonnegative signals.

B. EMP, SMP, and Other Near-Linear Time Algorithms

In this section, we describe a family of iterative algorithms for performing sparse recovery. Their key feature is that they enable performing sparse recovery in near-linear time while still using $O(k \log(n/k))$ measurements. The algorithms do not use linear programming; instead, they exploit various forms of voting mechanisms to converge to a solution. The specific algorithms covered are: expander matching pursuit (EMP) [26] and sparse matching pursuit (SMP) [5].[11]

To describe the algorithms we need some notation. For a set S of nodes of a graph G, the ordered set of its neighbors in G is denoted by $\Gamma_G(S)$. The subscript G will be omitted when it is clear from the context, and we write $\Gamma(u)$ as a shorthand for $\Gamma(\{u\})$.

Both EMP and SMP proceed in a sequence of steps, where each step is similar to the median estimation process of the count-median algorithm. A minor technical difference is that the algorithms are constructed for general sparse matrices A, as opposed to block-structured ones used by count-median. Therefore, for a given sketch vector $b = Ax$, the median estimation vector $E_{\mathrm{med}}(b)$ is defined as

$$E_{\mathrm{med}}(b)_i = \mathrm{median}_{j \in \Gamma(i)} b_j.$$

That is, each vertex i selects the entries b_j where j is a neighbor of i in G, and then computes the median of those entries. One can observe that for the matrices used by count-median, the new and old estimators are identical. The basic intuitions behind the choice of the estimator transfer as well.

There is, however, one important difference: unlike in the for-each setup, here we cannot guarantee that *each* coordinate $E_{\mathrm{med}}(b)_i$ differs from x_i by only a small term. In fact, due to the deterministic nature of the process, it might be possible that, for some coordinate i, all sketch coordinates $b_j, j \in \Gamma(i)$, could be highly "contaminated" by other entries of x. Thus, the algorithms do not provide guarantees for the l_∞ error of the recovered approximation. However, it is nevertheless possible to directly give guarantees for the l_1 approximation error.

1) EMP: The first algorithm that achieved the $O(k \log(n/k))$ sketch length bound and recovery time near-linear in n was the EMP. The algorithm and its analysis are somewhat complicated, so instead of a detailed description we provide only an overview.

EMP consists of two phases. In the first phase, the algorithm identifies a set $\bar{I}$ of coordinates of x that 1) contains "most" of the k largest (in magnitude) coefficients of x and 2) for all nodes $i \notin \bar{I}$ the neighborhood sets $\Gamma(i)$ and $\Gamma(\bar{I})$ have "small" intersection. The first constraint ensures that we can set the coordinates $\hat{x}_i$ of the approximation to zero for all $i \notin \bar{I}$. The second constraint ensures that the values of sketch coordinates $b_{\Gamma(\bar{I})}$ are not too contaminated by entries x_i for $i \notin \bar{I}$. Together, this implies that we can focus on decoding $\hat{x}_{\bar{I}}$ from $b_{\Gamma(\bar{I})}$. This is accomplished during the second phase, which proceeds in a sequence of iterations. In each iteration, the algorithm identifies coordinates $i \in \bar{I}$ such that most of elements of $\Gamma(i)$ do not have any other neighbors in $\Gamma(\bar{I})$. The algorithm then estimates the values $\hat{x}_i$ of such coordinates (using the median estimator), eliminates them from the $\bar{I}$, and subtracts their

[11] There is a very recent variant of SMP called sequential sparse matching pursuit [3]. We do not cover it in this survey due to lack of space.

contribution to the sketch. The process is continued until the set $\bar{I}$ becomes empty.

Since each coordinate of the approximation is estimated only once, and is never revised again, the EMP algorithm is very efficient: it runs in time proportional to the number of edges in the graph G, which is $O(n \log(n/k))$. The recovered vector $\hat{x}$ provides an approximation in the ℓ_1-norm, i.e., we have the following.

Theorem 9: There exists an $m \times n$ (expander) matrix A, $m = O(k \log(n/k)/\alpha^2)$, such that for any signal x, given Ax, we can recover $\hat{x}$ such that

$$\|x - \hat{x}\|_1 \leq (1 + \alpha)\mathrm{Err}_1^k.$$

The column sparsity of A is $O(\log(n)/\alpha^2)$, and the recovery algorithm (EMP) has $O(n \log(n/k)/\alpha^2)$ running time.

Although EMP offers excellent asymptotic guarantees, its empirical performance is not so great. Specifically, the number of measurements required by the algorithm to achieve correct recovery is suboptimal. For example, our recovery experiments on random signed k-sparse signals of length n, for $k = 50$ and $n = 20\,000$, show that one typically needs at least 5000 measurements to recover the signal correctly using the EMP algorithm. In comparison, the linear-programming-based recovery algorithm for sparse matrices described earlier requires only about 450 measurements to perform the same task.[12]

2) SMP: The SMP borrows some of the ideas present in EMP, but it has been also influenced by the recent iterative algorithms for sparse recovery using dense matrices, such as [35]. The running time of the new algorithm is slightly higher (by a logarithmic factor) than of EMP. However, empirically, the algorithm performs successful recovery from a significantly smaller number of measurements. In particular, for the instances described above, SMP typically needs about 2000 measurements. The asymptotic bound on the number of required measurements is still $O(k \log(n/k))$.

The recovery algorithm is iterative, in the spirit of *matching pursuit* [40]. In each iteration, the algorithm estimates the difference between the current approximation $\hat{x}^j$ and the signal x from the sketch $A\hat{x}^j - b$. The estimation, denoted by u^*, is obtained by using the median estimator as in EMP. The approximation $\hat{x}^j$ is updated by u, and the process is repeated.

Let $H_l[y]$ be a "thresholding operator," which zeros out all but the l largest in magnitude coefficients of the argument y. Also, let $C > 0$ be some constant. The details of the algorithm, together with remarks about the properties used in the analysis, are depicted in Fig. 1.

The remarks rely on the following trick, borrowed from [35]: we can decompose the input signal x into the "head" x' (containing the k most significant components of x) and the "tail" $x - x'$. Then, we can interpret the "sketch of the tail" term $A(x - x')$ as measurement noise. That is, we can assume that the sketch b is equal to $Ax' + \mu'$, where $\mu' = A(x - x')$ and x' is k-sparse. Note that the RIP(1) property of A implies that $\|A(x - x')\|_1 \leq d\|x - x'\|_1 = d\mathrm{Err}_1^k$. We define $\eta' = \|\mu'\|_1/d \leq \mathrm{Err}_1^k$.

```
1) Let $j = 0$
2) Let $\hat{x}^j = 0$
3) Repeat $T$ times
    a) Let $j = j + 1$
    b) Let $b = b - A\hat{x}^{j-1}$
       Remark: $b = A(x' - \hat{x}^{j-1}) + \mu'$
    c) Let $u^* = E_{med}(b)$
    d) Let $u^j = H_{2k}[u^*]$
       Remark: $\|u^j - (x' - \hat{x}^{j-1})\|_1 \leq \|x' - \hat{x}^{j-1}\|/4 + C\eta'$
    e) Let $\hat{x}^j = \hat{x}^{j-1} + u^j$
       Remark: $\|x' - \hat{x}^j\|_1 \leq \|x' - \hat{x}^{j-1}\|/4 + C\eta'$
    f) Let $\hat{x}^j = H_k[\hat{x}^j]$
       Remark: $\|x' - \hat{x}^j\|_1 \leq \|x' - \hat{x}^{j-1}\|/2 + 2C\eta'$
```

Fig. 1. ***Sparse matching pursuit algorithm: pseudocode and remarks on the analysis.***

From the remarks in the algorithm description, we conclude that for any $j = 1, 2, \ldots, T$, we have

$$\|\hat{x}^j - x'\|_1 \leq \|x'\|_1/2^j + O(\eta').$$

Thus, setting the number of iterations to $T = \log(\|x'\|_1/\eta')$ guarantees that

$$\|\hat{x}^T - x'\|_1 = O(\eta') = O\left(\mathrm{Err}_1^k\right).$$

The following theorem summarizes the discussion.

Theorem 10: There exists an $m \times n$ (expander) matrix A, $m = O(k \log(n/k))$, such that for any signal x, given Ax, we can recover $\hat{x}$ such that

$$\|x - \hat{x}\|_1 \leq c\ \mathrm{Err}_1^k$$

for an absolute constant $c > 0$. The column sparsity of A is $O(\log n)$, and the recovery algorithm (SMP) has $O(n \log(n/k)T)$ running time, for T defined as above.

3) Connections to Message-Passing Algorithms: The SMP algorithm described above, as well as the aforementioned algorithms from [25], [27], and [44], can be interpreted in

[12] For both algorithms, we used randomly generated 0–1 matrices with column sparsity equal to 20.

a general framework of *message-passing* algorithms. Such algorithms structure their operations based on the bipartite graph G underlying the matrix A. Specifically, each node of the graph can be viewed as a separate processing unit, and the algorithm proceeds by the units sending messages to each other along the edges of the graph. Message-passing algorithms have numerous advantages over the "centralized" ones: their computational complexity is low (if the underlying graph is sparse); they also can be easily implemented in a parallel or distributed manner.

There have been several papers on message-passing algorithms for sparse recovery problems using sparse random matrices. In [37] and [39], the authors introduced the belief propagation approach to compressive sensing, and applied it to the recovery of *random* signals, modeled by a two-state mixture of Gaussians. In a more recent paper [1], the authors used belief propagation on signals modeled as Gaussian-scale mixtures to obtain algorithms with an excellent empirical performance.

Message passing framework has been also used to design randomized algorithms that work in the worst case. In particular, the paper [32] introduced and analyzed such algorithms that work for arbitrary k-sparse signals. That algorithm can be viewed as an iterative generalization of the count-min algorithms described in earlier sections.

C. HHS and Sublinear Algorithms

As in Section II, there are versions of the above algorithms with sublinear running times. The main example is heavy hitters on steroids (HHS) [22]. The output of the HHS algorithm is $\hat{x}$ where $\|x-\hat{x}\|_2 \leq C(\mathrm{Err}_2 + 1/\sqrt{k}\mathrm{Err}_1)$ and its running time is $k^2(\log n)^{O(1)}$. It retains the same overall architecture as the iterative algorithms: within each step, it isolates significant entries by hashing, estimates their values, and then updates the measurements accordingly. It shares a "voting" procedure for determining significant signal entries with the EMP and SMP algorithms; however, these votes are derived from the bit tests rather than from the signal estimates directly. HHS differs from the simple sublinear algorithm we sketched in Section II in three major parts. First, in order to obtain a strong guarantee for all signals, we must hash k significant entries into $O(k)$ measurements repeatedly, for $O(\log n)$ repetitions. The adjacency matrix of a (s,d,ϵ) expander with $s=O(k)$ is a way to achieve this. Second, because we use a simple bit tester B_1 to identify the significant entries, we must ensure that it is applied to a signal that is sufficiently filtered; the contribution of the insignificant entries must be small enough not to pollute our estimates of the significant entry (recall that because the algorithm is iterative, estimation errors at one stage can accumulate at further iterations). Furthermore, we must carefully balance the ℓ_1 and ℓ_2 errors. To this end, we employ a second hash matrix that reduces the noise in each measurement after the first hash. In each iteration j, we keep a list of signal positions for which we have at least $\sqrt{k/j \log k}\log(n/j)\log(n)$ votes. Third, we use a separate matrix to estimate the values of the identified signal positions with the desired mixed norm error guarantee. Finally, in each iteration, we prune the list of signal positions to retain the top $O(k)$ positions. ■

Acknowledgment

The authors would like to thank J. Nelson, G. Cormode, and the anonymous reviewers for very helpful and insightful comments.

REFERENCES

[1] M. Akcakaya, J. Park, and V. Tarokh. (2009). Compressive sensing using low density frames. [Online]. Available: http://arxiv.org/abs/0903.0650

[2] R. Berinde, A. Gilbert, P. Indyk, H. Karloff, and M. Strauss, "Combining geometry and combinatorics: A unified approach to sparse signal recovery," in *Proc. 46th Annu. Allerton Conf. Commun. Control Comput.*, 2008, pp. 798–805.

[3] R. Berinde and P. Indyk, "Sequential sparse matching pursuit," in *Proc. 47th Annu. Allerton Conf. Commun. Control Comput.*, 2009, pp. 36–43.

[4] K. Do Ba, P. Indyk, E. Price, and D. Woodruff, "Lower bounds for sparse recovery," in *Proc. 19th Annu. ACM-SIAM Symp. Discrete Algorithms*, 2010.

[5] R. Berinde, P. Indyk, and M. Ruzic, "Practical near-optimal sparse recovery in the l_1 norm," in *Proc. 46th Annu. Allerton Conf. Commun. Control Comput.*, 2008, pp. 198–205.

[6] M. Charikar, K. Chen, and M. Farach-Colton, "Finding frequent items in data streams," in *Proc. Int. Coll. Autom. Lang. Programm.*, pp. 693–703, 2002.

[7] G. Cormode and M. Hadjieleftheriou, "Finding the frequent items in streams of data," *Commun. ACM*, vol. 52, pp. 97–105, 2009.

[8] V. Chandar, "A negative result concerning explicit matrices with the restricted isometry property," 2008, *preprint*.

[9] G. Cormode and S. Muthukrishnan, "Improved data stream summaries: The count-min sketch and its applications," *J. Algorithms*, vol. 55, no. 1, pp. 58–75, 2005.

[10] G. Cormode and S. Muthukrishnan, "Summarizing and mining skewed data streams," in *Proc. SIAM Int. Data Mining Conf.*, 2005.

[11] G. Cormode and S. Muthukrishnan, "Combinatorial algorithms for compressed sensing," in *Proc. 40th Annu. Conf. Inf. Sci. Syst.*, Princeton, NJ, Mar. 2006, pp. 198–201.

[12] E. Candès, J. Romberg, and T. Tao, "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information," *IEEE Trans Inf. Theory*, vol. 52, no. 2, pp. 489–509, Feb. 2006.

[13] M. Duarte, M. Davenport, D. Takhar, J. Laska, T. Sun, K. Kelly, and R. Baraniuk, "Single-pixel imaging via compressive sampling," *IEEE Signal Process. Mag.*, vol. 25, no. 2, pp. 83–91, Mar. 2008.

[14] R. DeVore, "Deterministic constructions of compressed sensing matrices," 2007, *preprint*.

[15] D.-Z. Du and F. K. Hwang, *Combinatorial Group Testing and Its Applications*. Singapore: World Scientific, 1993.

[16] D. L. Donoho, "Compressed sensing," *IEEE Trans. Inf. Theory*, vol. 52, no. 4, pp. 1289–1306, Apr. 2006.

[17] Y. Erlich, N. Shental, A. Amir, and O. Zuk, "Compressed sensing approach for high throughput carrier screen," in *Proc. 47th Annu. Allerton Conf. Commun. Control Comput.*, 2009, pp. 539–544.

[18] C. Estan and G. Varghese, "New directions in traffic measurement and accounting: Focusing on the elephants, ignoring the mice," *ACM Trans. Comput. Syst.*, vol. 21, no. 3, pp. 270–313, Aug. 2003.

[19] A. C. Gilbert, S. Guha, P. Indyk, Y. Kotidis, S. Muthukrishnan, and M. J. Strauss, "Fast, small-space algorithms for approximate histogram maintenance," in *Proc. 34th Annu. ACM Symp. Theory Comput.*, 2002, pp. 389–398.

[20] A. C. Gilbert, Y. Kotidis, S Muthukrishnan, and M. Strauss, "One-pass wavelet decompositions of data streams," *IEEE Trans. Knowl. Data Eng.*, vol. 15, no. 3, pp. 541–554, May/Jun. 2003.

[21] A. Gilbert, Y. Li, E. Porat, and M. Strauss. (2009). Approximate sparse recovery: Optimizing time and measurements. [Online]. Available: http://arxiv.org/abs/0912.0229

[22] A. C. Gilbert, M. J. Strauss, J. A. Tropp, and R. Vershynin, "One sketch for all: Fast algorithms for compressed sensing," in *Proc. 39th Annu. ACM Symp. Theory Comput.*, 2007, pp. 237–246.

[23] V. Guruswami, C. Umans, and S. P. Vadhan, "Unbalanced expanders and randomness extractors from Parvaresh-Vardy codes," in *Proc. IEEE Conf. Comput. Complex.*, 2007, pp. 96–108.

[24] P. Indyk. (2007). Sketching, streaming and sublinear-space algorithms. *graduate course notes*. [Online]. Available: http://stellar.mit.edu/S/course/6/fa07/6.895/

[25] P. Indyk, "Explicit constructions for compressed sensing of sparse signals," in *Proc. 19th Annu. ACM-SIAM Symp. Discrete Algorithms*, 2008, pp. 30–33.

[26] P. Indyk and M. Ruzic, "Near-optimal sparse recovery in the l_1 norm," in *Proc. 49th Annu. Symp. Found. Comput. Sci.*, 2008, pp. 199–207.

[27] S. Jafarpour, W. Xu, B. Hassibi, and A. R. Calderbank. (2008). Efficient and robust compressed sensing using high-quality expander graphs. [Online]. Available: http://arxiv.org/PS_cache/arxiv/pdf/0806/0806.3802v1.pdf

[28] R. Kainkaryam, A. Bruex, A. Gilbert, P. Woolf, and J. Schiefelbein, "Poolmc: Smart pooling of MRNA samples in microarray experiments," 2010, *Manuscript*.

[29] M. A. Khajehnejad, A. G. Dimakis, W. Xu, and B. Hassibi. (2008). Sparse recovery of positive signals with minimal expansion. [Online]. Available: http://arxiv.org/abs/0902.4045

[30] B. Krishnamurthy, S. Sen, Y. Zhang, and Y. Chen, "Sketch-based change detection," in *Proc. 3rd ACM SIGCOMM Conf. Internet Meas.*, 2003, pp. 234–247.

[31] J. Laska, S. Kirolos, Y. Massoud, R. Baraniuk, A. Gilbert, M. Iwen, and M. Strauss, "Random sampling for analog-to-information conversion of wideband signals," in *Proc. IEEE Dallas CAS Workshop Design Appl. Integr. Softw.*, 2006, pp. 119–122.

[32] Y. Lu, A. Montanari, B. Prabhakar, S. Dharmapurikar, and A. Kabbani, "Counter braids: A novel counter architecture for per-flow measurement," in *Proc. ACM SIGMETRICS Int. Conf. Meas. Model. Comput. Syst.*, 2008, pp. 121–132.

[33] S. Muthukrishnan. (2003). Data streams: Algorithms and applications, Invited Talk at the 14th Annu. ACM-IAM Symp. Discrete Algorithms. [Online]. Available: http://athos.rutgers.edu/~muthu/stream-1-1.ps

[34] S. Muthukrishnan, "Some algorithmic problems and results in compressed sensing," in *Proc. Annu. Allerton Conf. Commun. Control Comput.*, 2006.

[35] D. Needell and J. A. Tropp, "CoSaMP: Iterative signal recovery from incomplete and inaccurate samples," *Appl. Comput. Harmonic Anal.*, vol. 26, no. 3, pp. 301–321, 2009.

[36] N. Shental, A. Amir, and O. Zuk. (2009). Rare-Allele detection using compressed se(que)nsing. [Online]. Available: arXiv:0909.0400

[37] S. Sarvotham, D. Baron, and R. G. Baraniuk, "Compressed sensing reconstruction via belief propagation," Electr. Comput. Eng. Dept., Rice Univ., Houston, TX, Tech. Rep. ECE-0601, 2006.

[38] S. Sarvotham, D. Baron, and R. G. Baraniuk, "Sudocodes—Fast measurement and reconstruction of sparse signals," in *Proc. IEEE Int. Symp. Inf. Theory*, 2006.

[39] S. Sarvotham, D. Baron, and R. G. Baraniuk. (2008). Bayesian compressive sensing via belief propagation. [Online]. Available: http://arxiv.org/abs/0812.4627

[40] J. A. Tropp and A. C. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Trans. Inf. Theory*, vol. 53, no. 12, pp. 4655–4666, Dec. 2007.

[41] J. Tropp, M. Laska, M. Duarte, J. Romberg, and R. Baraniuk, "Beyond Nyquist: Efficient sampling of sparse bandlimited signals," *IEEE Trans. Inf. Theory*, vol. 56, no. 1, pp. 520–544, Jan. 2010.

[42] D. Takhar, J. Laska, M. B. Wakin, M. F. Duarte, D. Baron, S. Sarvotham, K. Kelly, and R. G. Baraniuk, "A new compressive imaging camera architecture using optical-domain compression," in *Proc. IS&T/SPIE Symp. Electron. Imaging*, 2006, vol. 6065, DOI:10.1117/12.659602606509.

[43] W. Wang, M. J. Wainwright, and K. Ramchandran. (2008). Information-theoretic limits on sparse signal recovery: Dense versus sparse measurement matrices. [Online]. Available: http://arxiv.org/abs/0806.0604

[44] W. Xu and B. Hassibi, "Efficient compressive sensing with deterministic guarantees using expander graphs," in *Proc. IEEE Inf. Theory Workshop*, 2007, pp. 414–419.

ABOUT THE AUTHORS

Anna Gilbert received the S.B. degree from the University of Chicago, Chicago, IL and the Ph.D. degree from Princeton University, Princeton, NJ, in 1997, both in mathematics.

In 1997, she was a Postdoctoral Fellow at Yale University and AT&T Labs-Research. From 1998 to 2004, she was a member of the technical staff at AT&T Labs-Research, Florham Park, NJ. In 2004, she joined the Department of Mathematics, University of Michigan, Ann Arbor, where she is now an Associate Professor. Her research interests include analysis, probability, networking, and algorithms. She is especially interested in randomized algorithms with applications to harmonic analysis, signal and image processing, networking, and massive data sets.

Dr. Gilbert received a National Science Foundation (NSF) CAREER Award, a Sloan Research Fellowship, and a NAS Award for Initiatives in Research.

Piotr Indyk received the M.S. degree from Uniwersytet Warszawski, Warsaw, Poland, in 1995 and the Ph.D. degree from Stanford University, Stanford, CA, in 2000, both in computer science.

Currently, he is an Associate Professor of Electrical Engineering and Computer Science at the Massachusetts Institute of Technology (MIT), Cambridge. His research interests include high-dimensional computational geometry, sketching and streaming algorithms, sparse recovery and compressive sensing.

Dr. Indyk received a National Science Foundation (NSF) CAREER Award, a Sloan Fellowship, and a Packard Fellowship.