Sparse Recovery with Partial Support Knowledge*

Khanh Do Ba and Piotr Indyk

Massachusetts Institute of Technology, Cambridge MA 02139, USA

Abstract. The goal of sparse recovery is to recover the (approximately) best k-sparse approximation $\hat{x}$ of an n-dimensional vector x from linear measurements Ax of x. We consider a variant of the problem which takes into account *partial knowledge* about the signal. In particular, we focus on the scenario where, after the measurements are taken, we are given a set S of size s that is supposed to contain most of the "large" coefficients of x. The goal is then to find $\hat{x}$ such that

$$\|x - \hat{x}\|_p \leq C \min_{\substack{k\text{-sparse } x' \\ \text{supp}(x') \subseteq S}} \|x - x'\|_q \ . \quad (1)$$

We refer to this formulation as the *sparse recovery with partial support knowledge problem (SRPSK)*. We show that SRPSK can be solved, up to an approximation factor of $C = 1 + \epsilon$, using $O((k/\epsilon)\log(s/k))$ measurements, for $p = q = 2$. Moreover, this bound is tight as long as $s = O(\epsilon n/\log(n/\epsilon))$. This completely resolves the asymptotic measurement complexity of the problem except for a very small range of the parameter s.

To the best of our knowledge, this is the first variant of $(1+\epsilon)$-approximate sparse recovery for which the asymptotic measurement complexity has been determined.

1 Introduction

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

A particularly useful and well-studied problem is that of *stable sparse recovery*. We say that a vector x' is k-sparse if it has at most k non-zero coordinates.

* This material is based upon work supported by the Space and Naval Warfare Systems Center Pacific under Contract No. N66001-11-C-4092, David and Lucille Packard Fellowship, MADALGO (Center for Massive Data Algorithmics, funded by the Danish National Research Association) and NSF grants CCF-0728645 and CCF-1065125.

The sparse recovery problem is typically defined as follows: for some norm parameters p and q and an approximation factor $C > 0$, given Ax, recover an "approximation" vector $\hat{x}$ such that

$$\|x - \hat{x}\|_p \leq C \min_{k\text{-sparse } x'} \|x - x'\|_q \tag{2}$$

(this inequality is often referred to as ℓ_p/ℓ_q *guarantee*). If the matrix A is random, then (2) should hold for each x with some probability (say, 3/4). Sparse recovery has a tremendous number of applications in areas such as compressive sensing of signals [4, 11], genetic data acquisition and analysis [23, 3] and data stream algorithms [20, 17].

It is known [4] that there exist matrices A and associated recovery algorithms that produce approximations $\hat{x}$ satisfying (2) with $p = q = 1$, constant approximation factor C, and sketch length

$$m = O(k \log(n/k)) \ . \tag{3}$$

A similar bound, albeit using random matrices A, was later obtained for $p = q = 2$ [15] (building on [6–8]). Specifically, for $C = 1 + \epsilon$, they provide a distribution over matrices A with

$$m = O((k/\epsilon) \log(n/k)) \tag{4}$$

rows, together with an associated recovery algorithm.

It is also known that the bound in (3) is asymptotically optimal for some constant C and $p = q = 1$ (see [10] and [13], building on [14, 16, 18]). The bound of [10] also extends to the randomized case and $p = q = 2$. For $C = 1 + \epsilon$, a lower bound of (roughly) $m = \Omega(k/\epsilon^{p/2})$ was recently shown [22].

The necessity of the "extra" logarithmic factor multiplying k is quite unfortunate: the sketch length determines the "compression rate", and for large n any logarithmic factor can worsen that rate tenfold.

In this paper we show that this extra factor can be reduced if we allow the recovery process to take into account some *partial knowledge* about the signal. In particular, we focus on the scenario where, after the measurements are taken, we are given a set S of size s (s is known beforehand) that is supposed to contain most of the "large" coefficients of x. The goal is then to find $\hat{x}$ such that

$$\|x - \hat{x}\|_p \leq C \min_{\substack{k\text{-sparse } x' \\ \text{supp}(x') \subseteq S}} \|x - x'\|_q \ . \tag{5}$$

We refer to this formulation as the *sparse recovery with partial support knowledge problem (SRPSK)*.

Results We show that SRPSK can be solved, up to an approximation factor of $C = 1 + \epsilon$, using $O((k/\epsilon) \log(s/k))$ measurements, for $p = q = 2$. Moreover, we show that this bound is tight as long as $s = O(\epsilon n / \log(n/\epsilon))$. This completely resolves the asymptotic measurement complexity of the problem except for a very small range of the parameter s.

To the best of our knowledge, this is the first variant of $(1+\epsilon)$-approximate sparse recovery for which the asymptotic measurement complexity has been determined.

Motivation The challenge of incorporating external knowledge into the sparse recovery process has received a fair amount of attention in recent years [9]. Approaches include *model-based compressive sensing* [2, 12] (where the sets of large coefficients are known to exhibit some patterns), *Bayesian compressive sensing* [5] (where the signals are generated from a known distribution) and support restriction.

There are several scenarios where our formulation (SRPSK) could be applicable. For example, for tracking tasks, the object position typically does not change much between frames, so one can limit the search for current position to a small set. The framework can also be useful for exploratory tasks, where there is a collection $\mathcal{S}$ of sets, one of which is assumed to contain the support. In that case, setting the probability of failure to $\frac{1}{|\mathcal{S}|}$ enables exploring all sets in the family and finding the one which yields the best approximation.

From a theoretical perspective, our results provide a smooth tradeoff between the $\Theta(k\log(n/k))$ bound for "standard" sparse recovery and the $\Theta(k)$ bound known for the *set query problem* [21]. In the latter problem we have the full knowledge of the signal support, i.e., $s=k$.

Our techniques Consider the *upper bound* first. The general approach of our algorithm is to reduce SRPSK to the *noisy sparse recovery problem* (NSR). The latter is a generalization of sparse recovery where the recovery algorithm is given $Ax+\nu$, where ν is the *measurement noise*. The reduction proceeds by representing Ax as $Ax_S + Ax_{\bar{S}}$, and interpreting the second term as noise. Since the vector x_S has dimension s, not n, we can use A with only $O(k\log(s/k))$ rows. This yields the desired measurement bound.

To make this work, however, we need to ensure that for any fixed S, the sub-matrix A_S of A (containing the columns with indices in S) is a valid sparse recovery matrix for s-dimensional vectors. This would be immediate if (as often happens, e.g. [4]) each column of A was an i.i.d. random variable chosen from some distribution: we could simply sample the n columns of A from the distribution parametrized by k and s. Unfortunately, the algorithm of [15] (which has the best known dependence on ϵ) does not have this independence property; in fact, the columns are highly dependent on each other. However, we show that it is possible to modify it so that the independence property holds.

Our *lower bound* argument mimics the approach of [10]. Specifically, consider fixing $s=\Theta(\epsilon n/\log(n/\epsilon))$; we show how to encode $\alpha=\Theta(\log(n/\epsilon)/\epsilon)$ code words $x_1,\ldots,x_\alpha$, from some code C containing $2^{\Theta(k\log(s/k))}$ code words, into a vector x, such that a $(1+\epsilon)$-approximate algorithm for SRPSK can iteratively decode all x_i's, starting from x_α and ending with x_1. This shows that one can "pack" $\Theta(\log(n/\epsilon)/\epsilon \cdot k\log(s/k))$ bits into Ax. Since one can show that each coordinate of Ax yields only $O(\log(n/\epsilon))$ bits of information, it follows that Ax has to have $\Theta((k/\epsilon)\log(s/k))$ coordinates.

Unfortunately, the argument of [10] applied only to the case of when ϵ is a constant strictly greater than 0 (i.e., $\epsilon = \Omega(1)$). For $\epsilon = o(1)$, the recovery algorithm could return a convex combination of several x_i's, which might not be decodable. Perhaps surprisingly, we show that the formulation of SRPSK avoids this problem. Intuitively, this is because different x_i's have different supports, and SRPSK enables us to restrict sparse approximation to a particular subset of coordinates.

2 Preliminaries

For positive integer n, let $[n] = \{1, 2, \ldots, n\}$. For positive integer $s \leq n$, let $\binom{[n]}{s}$ denote the set of subsets of cardinality s in $[n]$.

Let $v \in \mathbb{R}^n$. For any positive integer $k \leq s$ and set $S \in \binom{[n]}{s}$, denote by $v_k \in \mathbb{R}^n$ the vector comprising the k largest components of v, breaking ties by some canonical ordering (say, leftmost-first), and 0 everywhere else. Denote by $v_S \in \mathbb{R}^n$ the vector comprising of components of v indexed by S, with 0 everywhere else, and denote by $v_{S,k} \in \mathbb{R}^n$ the vector comprising the k largest components of v among those indexed by S, with 0 everywhere else.

Let $\mathit{\Pi}_S \in \mathbb{R}^{s \times n}$ denote the projection matrix that keeps only components indexed by S (the dimension n will be clear from context). In particular, $\mathit{\Pi}_S v \in \mathbb{R}^s$ consists of components of v indexed by S, and for any matrix $A \in \mathbb{R}^{m \times n}$, $A\mathit{\Pi}_S^T \in \mathbb{R}^{m \times s}$ consists of the columns of A indexed by S.

Define the ℓ_p/ℓ_p *sparse recovery with partial support knowledge problem* (denoted SRPSK_p) to be the following:

Given parameters (n, s, k, ϵ), where $1 \leq k \leq s \leq n$ and $0 < \epsilon < 1$, design an algorithm and a distribution over matrices $A \in \mathbb{R}^{m \times n}$, where $m = m(n, s, k, \epsilon)$, such that for any $x \in \mathbb{R}^n$, the algorithm, given Ax and a specified set $S \in \binom{[n]}{s}$, recovers (with knowledge of A) a vector $\hat{x} \in \mathbb{R}^n$ such that, with probability 3/4,

$$\|x - \hat{x}\|_p^p \leq (1 + \epsilon)\|x - x_{S,k}\|_p^p \ . \tag{6}$$

Define the ℓ_p/ℓ_p *noisy sparse recovery problem* (NSR_p) to be the following:

Given parameters (n, k, ϵ), where $1 \leq k \leq n$ and $0 < \epsilon < 1$, design an algorithm and a distribution over matrices $A \in \mathbb{R}^{m \times n}$, where $m = m(n, k, \epsilon)$, such that for any $x \in \mathbb{R}^n$ and $\nu \in \mathbb{R}^m$, the algorithm recovers from $Ax + \nu$ (with knowledge of A) a vector $\hat{x} \in \mathbb{R}^n$ such that, with probability 3/4,

$$\|x - \hat{x}\|_p^p \leq (1 + \epsilon)\|x - x_k\|_p^p + \epsilon\|\nu\|_p^p \ . \tag{7}$$

The distribution of A must be "normalized" so that for any $v \in \mathbb{R}^n$, $\mathbb{E}[\|Av\|_p] \leq \|v\|_p$.

For all four problems, we will denote a solution by a pair $(A, \mathcal{R})$, where A is the measurement matrix and $\mathcal{R}$ is the recovery algorithm. For SRPSK_1 and SRPSK_2, we will also often denote the recovery algorithm with the parameter S as a subscript, e.g., $\mathcal{R}_S$.

3 Lower bounds

We will need a result from communication complexity. Consider the following two-party communication game involving Alice and Bob: Alice is given a string $y \in \{0,1\}^d$. Bob is given an index $i \in [d]$, together with $y_{i+1}, y_{i+2}, \ldots, y_d$. They also share an arbitrarily long common random string r. Alice sends a single message $M(y,r)$ to Bob, who must output y_i correctly with probability at least $3/4$, where the probability is taken over r. We refer to this problem as the *augmented indexing problem* (AIP). The communication cost of AIP is the minimum, over all correct protocols, of the length of the message $M(y,r)$ on the worst-case choice of r and y. The following lemma is well-known (see, e.g., [19] or [1]):

Lemma 1. *The communication cost of AIP is $\Omega(d)$.*

We will also make use of Lemma 5.1 of [10], which we reproduce below:

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

Now we can prove our lower bounds for SRPSK_1 and SRPSK_2:

Theorem 3. *Any solution to SRPSK$_1$ requires, for $s = O(\epsilon n / \log(n/\epsilon))$, at least $\Omega\left((k/\epsilon)\log(s/k)\right)$ measurements.*

Proof. For $\alpha = n/s$, divide $[n]$ into α equal-sized disjoint blocks, S_i for $i = 1, \ldots, \alpha$. For each block S_i, we will choose a binary error-correcting code $C_i \subseteq \{0,1\}^n$ with minimum Hamming distance k, where all the code words have Hamming weight exactly k and support contained in S_i. Since $|S_i| = s = n/\alpha$, we know each C_i can be chosen big enough that

$$\log |C_i| = \Theta(k \log(n/(\alpha k))) \ . \tag{8}$$

Now, we will use any solution to SRPSK_1 to design a protocol for AIP with instance size

$$d = \Theta(\alpha k \log(n/(\alpha k))) \ . \tag{9}$$

The protocol is as follows:
Alice divides her input y into α equal-sized blocks each of size

$$d/\alpha = \Theta(k \log(n/(\alpha k))) \ . \tag{10}$$

Interpreting each block y_i as a binary number, she uses it to index into C_i (notice that C_i has sufficiently many code words for each y_i to index a different one), specifying a code word $x_i \in C_i$. She then computes

$$x = D^1 x_1 + D^2 x_2 + \cdots + D^\alpha x_\alpha \tag{11}$$

for some fixed D dependent on ϵ. Then, using shared randomness, and following the hypothetical protocol, Alice and Bob agree on a matrix A (wlog, and for technical reasons, with orthonormal rows), which they both round to A' so that each entry has b bits. Alice computes $A'x$ and sends it to Bob.

Bob, knowing his input i, can compute the $j = j(i)$ for which block y_j of Alice's input contains i, and hence knows the set S_j. Moreover, he knows $y_{j'}$, and thereby $x_{j'}$, for every $j' > j$, so he can compute

$$z = D^{j+1}x_{j+1} + \cdots + D^{\alpha}x_{\alpha} \ . \tag{12}$$

From Alice's message, using linearity, he can then compute $A'(x - z)$. Now, by Lem. 2, there must exist some $\sigma \in \mathbb{R}^n$ with $A'(x - z) = A(x - z - \sigma)$ and

$$\|\sigma\|_1 < n^2 2^{-b} \|x - z\|_1 = n^2 2^{-b} \sum_{i'=1}^{j} kD^{i'} < n^2 2^{-b} k \tfrac{D^{j+1}}{D-1} \ . \tag{13}$$

Now, let $w = x - z - \sigma$, so that Bob has $Aw = A'(x - z)$. He then runs $\mathcal{R}_{S_j}$ on Aw to recover $\hat{w}$ with the properties that $\mathrm{supp}(\hat{w}) \subseteq S_j$ and

$$\begin{aligned} \|w - \hat{w}\|_1 &\leq (1+\epsilon)\|w - w_{S_j,k}\|_1 \leq (1+\epsilon)\|w - D^j x_j\|_1 \\ &\leq (1+\epsilon)(\|D^1 x_1 + \cdots + D^{j-1}x_{j-1}\|_1 + \|\sigma\|_1) \\ &= (1+\epsilon)\left(k\tfrac{D^j - D}{D-1} + \|\sigma\|_1\right) \ . \end{aligned} \tag{14}$$

Bob then finds the code word in C_j that is closest in ℓ_1-distance to $\hat{w}/D^j$ (which he hopes is x_j) and, looking at the index of that code word within C_j (which he hopes is y_j), he returns the bit corresponding to his index i.

Now, suppose that Bob was wrong. This means he obtained a $\hat{w}$ that, appropriately scaled, was closer or equidistant to another code word in C_j than x_j, implying that $\|x_j - \hat{w}/D^j\|_1 \geq k/2$. Since $\mathrm{supp}(\hat{w}) \subseteq S_j$, we can write

$$\begin{aligned} \|w - \hat{w}\|_1 &\geq \|x - z - \hat{w}\|_1 - \|\sigma\|_1 \\ &= \|D^1 x_1 + \cdots + D^{j-1}x_{j-1}\|_1 + D^j \|x_j - \hat{w}/D^j\|_1 - \|\sigma\|_1 \\ &\geq k\left(\tfrac{D^j - D}{D-1} + D^j/2\right) - \|\sigma\|_1 \ . \end{aligned} \tag{15}$$

We will show that for appropriate choices of D and b, (14) and (15) contradict each other, implying that Bob must have correctly extracted his bit and solved AIP. To this end, it suffices to prove the following inequality:

$$\|\sigma\|_1 < \tfrac{k}{3}\left(D^j/2 - \epsilon\tfrac{D^j}{D-1}\right) \ , \tag{16}$$

where we used the fact that $\epsilon < 1$. Now, let us fix $D = 1 + 4\epsilon$. The above inequality becomes

$$\|\sigma\|_1 < \tfrac{k}{3}\left((1+4\epsilon)^j/2 - (1+4\epsilon)^j/4\right) = k(1+4\epsilon)^j/12 \ . \tag{17}$$

Now, from (13) we know that

$$\|\sigma\|_1 < n^2 2^{-b} k \frac{D^{j+1}}{D-1} = n^2 2^{-b} k (1+4\epsilon)^{j+1}/(4\epsilon) \ , \tag{18}$$

so we need only choose b large enough that $2^b \geq 15n^2/\epsilon$, i.e., $b = O(\log(n/\epsilon))$ suffices. Recall that b is the number of bits per component of A', and each component of $x-z$ can require up to $\alpha \log D = O(\epsilon\alpha)$ bits, so the message $A'(x-z)$ which Alice sends to Bob contains at most $O(m(b+\epsilon\alpha)) = O(m(\log(n/\epsilon)+\epsilon\alpha))$ bits, with which they solve AIP with $d = \Theta(\alpha k \log(n/(\alpha k)))$. It follows from Lem. 1 that

$$m = \Omega\left(\frac{\alpha k \log(n/(\alpha k))}{\log(n/\epsilon) + \epsilon\alpha}\right) \ . \tag{19}$$

Finally, as long as $\epsilon\alpha = \Omega(\log(n/\epsilon))$, or equivalently, $s = n/\alpha = O(\epsilon n/\log(n/\epsilon))$, this simplifies to

$$m = \Omega((k/\epsilon)\log(s/k)) \ . \tag{20}$$

□

Theorem 4. *Any solution to SRPSK$_2$ requires, for $s = O(\epsilon n/\log(n/\epsilon))$ and $\epsilon \leq 1/6$,*[1] *requires at least $\Omega((k/\epsilon)\log(s/k))$ measurements.*

We omit the proof, which involves only algebraic modifications from the proof of Thm. 3, due to space constraints.

4 Upper bounds

First we prove a general black box reduction from SRPSK$_1$ to NSR$_1$ that works if the solution to NSR$_1$ has certain additional properties:

Lemma 5. *Suppose we have a solution to NSR$_1$ with parameters (n,k,ϵ), where the $m \times n$ measurement matrix A' has $m = m(n,k,\epsilon)$ rows. Suppose in addition that the columns of A' are generated i.i.d. from some distribution. Then there exists a solution $(A,\mathcal{R})$ to SRPSK$_1$ with parameters (n,s,k,ϵ) that uses $O(m(s,k,\Theta(\epsilon)))$ measurements. Moreover, if A' has, in expectation, $h(n,k,\epsilon)$ non-zeros per column, and the NSR$_1$ recovery time is $t(n,k,\epsilon)$, then A has, in expectation, $O(h(s,k,\Theta(\epsilon)))$ non-zeros, and $\mathcal{R}$ runs in $O(t(s,k,\Theta(\epsilon)))$ time.* [2]

Proof. We construct our solution $(A,\mathcal{R})$ to SRPSK$_1$ as follows:

[1] The assumption that $\epsilon \leq 1/6$ is not necessary, but makes the proof simpler, so we leave it in the theorem statement.

[2] Note that this recovery time is based on the assumption that the solution to NSR generated the columns of its measurement matrix i.i.d. In our application of this reduction (Lems. 7 and 8), we will need to modify the NSR solution to enforce this requirement, which will increase its recovery time.

1. Let $\delta > 0$ be a constant to be specified later. Consider an instantiation of the solution to NSR_1 with parameters $(s, k, \delta\epsilon)$, so that its measurement matrix A' is $m \times s$, where $m = m(s, k, \delta\epsilon)$. Generate the n columns of our $m \times n$ measurement matrix A i.i.d. from the same distribution used to generated the i.i.d. columns of A' (note that the number of rows m is the same for both A and A').
2. Given $S \subseteq [n]$, $|S| = s$, let $\mathcal{R}'_S$ denote the recovery algorithm for NSR_1 corresponding to the parameters $(s, k, \delta\epsilon)$ and given the matrix $A\Pi_S^T$ (recall that a recovery algorithm for NSR_1 is allowed to behave differently given different instances of the measurement matrix). Define our recovery procedure $\mathcal{R}_S$ by $\mathcal{R}_S(y) = \Pi_S^T(\mathcal{R}'_S(y))$; in words, we run $\mathcal{R}'_S$ on our m-dimensional measurement vector y to obtain an s-dimensional vector, which we embed into an n-dimensional vector at positions corresponding to S, filling the rest with zeros.

Note that the number of non-zeros per column of A and the running time of $\mathcal{R}$ follow immediately.

Observe that, thanks to the independence of the columns of A, the submatrix comprised of any s of them (namely, $A\Pi_S^T$) is a valid $m \times s$ measurement matrix. Thus we have the guarantee that for any signal $x' \in \mathbb{R}^s$ and noise vector $\nu \in \mathbb{R}^m$, $\mathcal{R}'_S$ recovers from $A\Pi_S^T x' + \nu$ a vector $\hat{x}' \in \mathbb{R}^s$ satisfying, with probability $3/4$,

$$\|x' - \hat{x}'\|_1 \leq (1+\epsilon)\|x' - x'_k\|_1 + \delta\epsilon\|\nu\|_1 \ . \tag{21}$$

Now, let $x \in \mathbb{R}^n$ be our signal for SRPSK_1. We interpret $\Pi_S x \in \mathbb{R}^s$ to be the sparse signal and $Ax_{\bar{S}} \in \mathbb{R}^m$ to be the noise, so that running $\mathcal{R}'_S$ on $A\Pi_S^T(\Pi_S x) + Ax_{\bar{S}}$ returns $\hat{x}' \in \mathbb{R}^s$ satisfying, with probability $3/4$,

$$\begin{aligned} \|\Pi_S x - \hat{x}'\|_1 &\leq (1+\epsilon)\|\Pi_S x - (\Pi_S x)_k\|_1 + \delta\epsilon\|Ax_{\bar{S}}\|_1 \\ &= (1+\epsilon)\|x_S - x_{S,k}\|_1 + \delta\epsilon\|Ax_{\bar{S}}\|_1 \ . \end{aligned} \tag{22}$$

Finally, consider the $\hat{x} \in \mathbb{R}^n$ recovered by $\mathcal{R}_S$ in our procedure for SRPSK_1 when run on

$$Ax = Ax_S + Ax_{\bar{S}} = A\Pi_S^T(\Pi_S x) + Ax_{\bar{S}} \ . \tag{23}$$

We have $\hat{x} = \Pi_S^T \hat{x}'$, or, equivalently, $\Pi_S \hat{x} = \hat{x}'$, so

$$\begin{aligned} \|x - \hat{x}\|_1 &= \|x_{\bar{S}}\|_1 + \|x_S - \hat{x}\|_1 = \|x_{\bar{S}}\|_1 + \|\Pi_S x - \hat{x}'\|_1 \\ &\leq \|x_{\bar{S}}\|_1 + (1+\epsilon)\|x_S - x_{S,k}\|_1 + \delta\epsilon\|Ax_{\bar{S}}\|_1 \\ &= \|x_{\bar{S}}\|_1 + (1+\epsilon)(\|x - x_{S,k}\|_1 - \|x_{\bar{S}}\|_1) + \delta\epsilon\|Ax_{\bar{S}}\|_1 \\ &= (1+\epsilon)\|x - x_{S,k}\|_1 - \epsilon\|x_{\bar{S}}\|_1 + \delta\epsilon\|Ax_{\bar{S}}\|_1 \ . \end{aligned}$$

Thus, if we can ensure that $\|Ax_{\bar{S}}\|_1 \leq (1/\delta)\|x_{\bar{S}}\|_1$, we would obtain the desired guarantee for SRPSK_1 of

$$\|x - \hat{x}\|_1 \leq (1+\epsilon)\|x - x_{S,k}\|_1 \ . \tag{24}$$

But we know that $\mathbb{E}[\|Ax_{\bar{S}}\|_1] \leq \|x_{\bar{S}}\|_1$, so by the Markov bound

$$\Pr\left[\|Ax_{\bar{S}}\|_1 > (1/\delta)\|x_{\bar{S}}\|_1\right] \leq \delta \ . \tag{25}$$

Choosing, say, $\delta = 1/12$ would give us an overall success probability of at least $2/3$, which can be amplified by independent repetitions and taking a componentwise median in the standard way. □

Straightforward modification of the above proof yields the ℓ_2/ℓ_2 version:

Lemma 6. *Suppose we have a solution to NSR_2 with parameters (n, k, ϵ), where the $m \times n$ measurement matrix A' has $m = m(n, k, \epsilon)$ rows. Suppose in addition that the columns of A' are generated i.i.d. from some distribution. Then there exists a solution $(A, \mathcal{R})$ to $SRPSK_2$ with parameters (n, s, k, ϵ) that uses $O(m(s, k, \Theta(\epsilon)))$ measurements. Moreover, if A' has, in expectation, $h(n, k, \epsilon)$ non-zeros per column, and the NSR_2 recovery time is $t(n, k, \epsilon)$, then A has, in expectation, $O(h(s, k, \Theta(\epsilon)))$ non-zeros, and $\mathcal{R}$ runs in $O(t(s, k, \Theta(\epsilon)))$ time.* [3]

By a modification of the algorithm of [15], we prove the following result:

Lemma 7. *There exist a distribution on $m \times n$ matrices A and a collection of algorithms $\{\mathcal{R}_S \mid S \in \binom{[n]}{s}\}$ such that for any $x \in \mathbb{R}^n$ and set $S \subseteq [n]$, $|S| = s$, $\mathcal{R}_S(Ax)$ recovers $\hat{x}$ with the guarantee that*

$$\|x - \hat{x}\|_2 \leq (1 + \epsilon)\|x - x_{S,k}\|_2 \tag{26}$$

with probability $3/4$. The matrix A has $m = O((k/\epsilon)\log(s/k))$ rows.

Proof. To apply a NSR_2 solution to SRPSK_2 using Lem. 6, we need the columns of the measurement matrix to be generated independently. However, this requirement does not hold with the algorithm in [15] as is. Therefore, we show how to modify it to satisfy this requirement without changing its recovery properties and asymptotic number of measurements. For simplicity, we will ignore pseudo-randomness considerations, and replace all k-wise independence by full independence in the construction of [15].

We begin by describing the measurement matrix A of [15] (denoted by Φ in that paper). At the highest level, A is formed by vertically stacking matrices $A^{(j)}$, for $j = 1, \ldots, \log k$. Each $A^{(j)}$ is formed by vertically stacking two matrices, $E^{(j)}$ and $D^{(j)}$. It will suffice for our purposes if the columns of each $E^{(j)}$ and each $D^{(j)}$ are independent.

Consider, first, $E^{(j)}$, which consists of several i.i.d. submatrices, again stacked vertically, in each of which every entry is set i.i.d. (to 1, -1 or 0). Thus, every entry, and therefore every column, of $E^{(j)}$ is already independent without modification.

Next, consider $D^{(j)}$, which consists of several similarly stacked i.i.d. submatrices. For some constant $c < 1$, each one of these submatrices consists of kc^j i.i.d. "blocks" $B_1^{(j)}, B_2^{(j)}, \ldots, B_{kc^j}^{(j)}$, which will be the smallest unit of vertically stacked submatrices we need to consider (see Fig. 1). Within each block

[3] See footnote to Lem. 5.

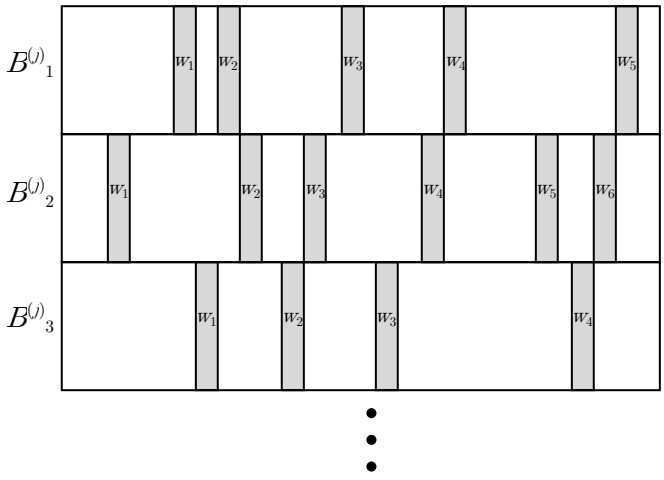

Fig. 1. Example of an i.i.d. submatrix in $D^{(j)}$ consisting of kc^j blocks. Each grey rectangle represents a code word, and white space represents zeros.

$B_i^{(j)}$, each column is independently chosen to be non-zero with some probability, and the i^{th} non-zero column is equal to the i^{th} code word w_i from some error-correcting code C. The code C has a constant rate and constant fractional distance. Therefore, each block has $O(\log h)$ rows (and C needs to have $O(h)$ code words), where h is the expected number of non-zero columns per block.

The problem with the construction of $D^{(j)}$ (from our perspective) is that each column chosen to be non-zero is not independently chosen, but instead is determined by a code word that depends on how many non-zero columns are to its left. In order to overcome this obstacle, we observe that the algorithm of [15] only requires that the codewords of the consecutive non-zero columns are *distinct*, not *consecutive*. Thus, we use as ECC C' with the same rate and error-correction, but with $O(h^3)$ code words instead of $O(h)$; for each column chosen to be non-zero, we set it to a code word chosen uniformly at random from C'. In terms of Fig. 1, each grey rectangle, instead of being the code word from C specified in the figure, is instead a random code word from a larger code C'. Note that each block has still $O(\log h)$ rows as before.

A block is *good* if all codewords corresponding to it are distinct. Observe that for any given block, the probability it is not good is at most $O(1/h)$. If there are fewer than $O(h)$ blocks in all of $D^{(j)}$, we could take a union bound over all of them to show that all blocks are good constant probability. Unfortunately, for $j = 1$, we have $h = O(n/k)$ while the number of blocks is $\Omega(k)$. The latter value could be much larger than h.

Instead, we will simply double the number of blocks. Even though we cannot guarantee that all blocks are good, we know that most of them will be, since each one is with probability $1 - O(1/h)$. Specifically, by the Chernoff bound, at

least half of them will be with high probability (namely, $1 - e^{-\Omega(k)}$). We can use only those *good* blocks during recovery and still have sufficiently many of them to work with.

The result is a solution to NSR_2 still with $O((k/\epsilon)\log(n/k))$ rows (roughly 6 times the solution of [15]), but where each column of the measurement matrix is independent, as required by Lem. 6. A direct application of the lemma gives us the theorem. □

Lemma 8. *The matrix A of Lem. 7 has, in expectation, $O(\log^2 k \log(s/k))$ non-zeros per column, and each algorithm $\mathcal{R}_S$ runs in $O(s\log^2 k + (k/\epsilon)\log^{O(1)} s)$ time.*

Proof. It suffices to show that the modifications we made to [15] do not change the asymptotic expected number of non-zeros in each column and does not increase the recovery time by more than an additive term of $O(n\log^2 k)$. Lem. 6 then gives us this lemma (by replacing n with s in both quantities).

Consider, first, the number of non-zeros. In both the unmodified and the modified matrices, this is dominated by the number of non-zeros in the (mostly dense) code words in the D^j's. But in the modified D^j, we do not change the asymptotic length of each code word, while only doubling, in expectation, the number of code words (in each column as well as overall). Thus the expected number of non-zeros per column of A remains $O(\log^2 k \log(n/k))$ as claimed.

Next, consider the running time. The first of our modifications, namely, increasing the number of code words from $O(h)$ to $O(h^3)$, and hence their lengths by a constant factor, does not change the asymptotic running time since we can use the same encoding and decoding functions (it suffices that these be polynomial time, while they are in fact poly-logarithmic time). The second of our modifications, namely, doubling the number of blocks, involves a little additional work to identify the *good* blocks at recovery time. Observe that, for each block, we can detect any collision in time linear in the number of code words. In $D^{(j)}$ there are $O(jkc^j)$ blocks each containing $O(n/(kc^j))$ code words, so the time to process $D^{(j)}$ is $O(jn)$. Thus, overall, for $j = 1, \ldots, \log k$, it takes $O(n\log^2 k)$ time to identify all *good* blocks. After that, we need only work with the same number of blocks as there had been in the unmodified matrix, so the overall running time is $O(n\log^2 k + (k/\epsilon)\log^{O(1)} n)$ as required. □

For completeness, we state the section's main result:

Theorem 9. *There exist a distribution on $m \times n$ matrices A and a collection of algorithms $\{\mathcal{R}_S \mid S \in \binom{[n]}{s}\}$ such that for any $x \in \mathbb{R}^n$ and set $S \subseteq [n]$, $|S| = s$, $\mathcal{R}_S(Ax)$ recovers $\hat{x}$ with the guarantee that*

$$\|x - \hat{x}\|_2 \leq (1+\epsilon)\|x - x_{S,k}\|_2 \tag{27}$$

with probability $3/4$. The matrix A has $m = O((k/\epsilon)\log(s/k))$ rows and, in expectation, $O(\log^2 k \log(s/k))$ non-zeros per column. Each algorithm $\mathcal{R}_S$ runs in $O(s\log^2 k + (k/\epsilon)\log^{O(1)} s)$ time.

References

1. Z. Bar-Yossef, T. S. Jayram, R. Krauthgamer, and R. Kumar. The sketching complexity of pattern matching. *RANDOM*, 2004.
2. R. G. Baraniuk, V. Cevher, M. F. Duarte, and C. Hegde. Model-based compressive sensing. *IEEE Transactions on Information Theory*, 56, No. 4:1982–2001, 2010.
3. A. Bruex, A. Gilbert, R. Kainkaryam, John Schiefelbein, and Peter Woolf. Poolmc: Smart pooling of mRNA samples in microarray experiments. *BMC Bioinformatics*, 2010.
4. E. J. Candès, J. Romberg, and T. Tao. Stable signal recovery from incomplete and inaccurate measurements. *Comm. Pure Appl. Math.*, 59(8):1208–1223, 2006.
5. V. Cevher, P. Indyk, L. Carin, and R.G Baraniuk. Sparse signal recovery and acquisition with graphical models. *Signal Processing Magazine*, pages 92 – 103, 2010.
6. M. Charikar, K. Chen, and M. Farach-Colton. Finding frequent items in data streams. *ICALP*, 2002.
7. G. Cormode and S. Muthukrishnan. Improved data stream summaries: The count-min sketch and its applications. *LATIN*, 2004.
8. G. Cormode and S. Muthukrishnan. Combinatorial algorithms for Compressed Sensing. In *Proc. 40th Ann. Conf. Information Sciences and Systems*, Princeton, Mar. 2006.
9. Defense Sciences Office. Knowledge enhanced compressive measurement. *Broad Agency Announcement*, DARPA-BAA-10-38, 2010.
10. K. Do Ba, P. Indyk, E. Price, and D. Woodruff. Lower bounds for sparse recovery. *SODA*, 2010.
11. D. L. Donoho. Compressed Sensing. *IEEE Trans. Info. Theory*, 52(4):1289–1306, Apr. 2006.
12. Y.C. Eldar and H. Bolcskei. Block-sparsity: Coherence and efficient recovery. *IEEE Int. Conf. Acoustics, Speech and Signal Processing*, 2009.
13. S. Foucart, A. Pajor, H. Rauhut, and T. Ullrich. The gelfand widths of lp-balls for $0 < p \leq 1$. *J. Complexity*, 2010.
14. A. Y. Garnaev and E. D. Gluskin. On widths of the euclidean ball. *Sov. Math., Dokl.*, page 200204, 1984.
15. Anna C. Gilbert, Yi Li, Ely Porat, and Martin J. Strauss. Approximate sparse recovery: optimizing time and measurements. In *STOC*, pages 475–484, 2010.
16. E. D. Gluskin. Norms of random matrices and widths of finite-dimensional sets. *Math. USSR-Sb.*, 48:173182, 1984.
17. P. Indyk. Sketching, streaming and sublinear-space algorithms. *Graduate course notes, available at* `http://stellar.mit.edu/S/course/6/fa07/6.895/`, 2007.
18. B. S. Kashin. Diameters of some finite-dimensional sets and classes of smooth functions. *Math. USSR, Izv.,*, 11:317333, 1977.
19. P. B. Milterson, N. Nisan, S. Safra, and A. Wigderson. On data structures and asymmetric communication complexity. *J. Comput. Syst. Sci.*, 57(1):37–49, 1998.
20. S. Muthukrishnan. Data streams: Algorithms and applications. *Foundations and Trends in Theoretical Computer Science*, 2005.
21. E. Price. Efficient sketches for the set query problem. *SODA*, 2011.
22. E. Price and D. Woodruff. $(1 + \epsilon)$-approximate sparse recovery. *Preprint*, 2011.
23. N. Shental, A. Amir, and Or Zuk. Identification of rare alleles and their carriers using compressed se(que)nsing. *Nucleic Acids Research*, 38(19):1–22, 2010.