Sequential Sparse Matching Pursuit

Radu Berinde
MIT
texel@mit.edu

Piotr Indyk
MIT
indyk@theory.csail.mit.edu

Abstract— **We propose a new algorithm, called *Sequential Sparse Matching Pursuit (SSMP)*, for solving sparse recovery problems. The algorithm provably recovers a k-sparse approximation to an arbitrary n-dimensional signal vector x from only $O(k \log(n/k))$ linear measurements of x. The recovery process takes time that is only near-linear in n. Preliminary experiments indicate that the algorithm works well on synthetic and image data, with the recovery quality often outperforming that of more complex algorithms, such as l_1 minimization.**

I. Introduction

Over the recent years, a new 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 a $m \times n$ 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. A particularly useful and well-studied problem is that of stable sparse recovery: given Ax, recover a *k-sparse* vector x^* (i.e. having at most k non-zero components) satisfying some approximation guarantee. In this paper we consider the ℓ_1/ℓ_1 guarantee[1] where we require

$$\|x - x^*\|_1 \leq C \min_{k\text{-sparse } x'} \|x - x'\|_1 \tag{1}$$

where C is an absolute constant. Sparse recovery has applications to numerous areas such as data stream computing [1], [2] and compressed sensing [3], [4], [5].

It is known [3], [6], [7], [8], [9], [10] that there exist matrices A and recovery algorithms satisfying Equation 1 with sketch length $m = O(k \log(n/k))$. In particular, the algorithms EMP [9] and SMP [10] achieve the aforementioned bound together with a near-linear (in n) recovery times. The computational efficiency of these algorithms is of key importance when solving massive recovery problems.

However, both EMP and SMP suffer from certain drawbacks. In case of EMP, the *empirical* number of measurements required by the algorithm to achieve correct recovery is suboptimal (i.e. the empirical "big-Oh" constant is large). While the SMP algorithm achieves much better empirical performance, its original version suffers from convergence problems when the input parameters (notably the sparsity k and the number of measurements m) fall outside of the theoretically guaranteed region. The SMP package[2] fixes the issue by forcing convergence; however, this requires that the user provides an additional and somewhat unintuitive convergence parameter.

In this paper we present a new algorithm, called *Sequential Sparse Matching Pursuit*, that bypasses the aforementioned problems. In a nutshell, it is simply a version of SMP where the updates to the candidate approximation vector are performed sequentially, as opposed to in parallel. As a result, the convergence is achieved automatically. Moreover, since the algorithm always performs the "best" update first, empirically SSMP performs significantly better than SMP (see section VI). At the same time, SSMP inherits the theoretical $O(k \log(n/k))$ measurement bound, and has a near-linear time of $O(\log(\|x\|_1/\eta) \cdot dn(d + \log n))$ where $d = O(\log(n/k))$ and η is the noise level (defined later in the paper).

A. Related work

The sparse recovery problem has been a subject of a tremendous amount of work. We refer the reader to our earlier paper [6] for a broad overview of the work in the area and comparison of different techniques. Below, we only briefly discuss the prior work of relevance to this paper.

Sparse recovery using sparse matrices. The most immediate ancestor of SSMP is the SMP algorithm introduced in [10]. Both SMP and SSMP use *sparse* binary sketching matrices, and iterative, voting-like mechanism to recover the signal approximation. The main difference between them is that SMP performs $O(k)$ updates of the approximation vector in parallel, while SSMP performs them sequentially.

More generally, there has been substantial amount of work utilizing sparse matrices to obtain sketches of high-dimensional vectors that support efficient sparse recovery (see, e.g., [11], [12], [13]). See [14] for a survey of those methods.

Iterative algorithms. Iterative algorithms are a popular approach to sparse recovery problem, see e.g., [7], [8], [15]. Our approach builds on that line of work, notably [8], who show show how Restricted Isometry Property can be used to analyze iteration steps.

[1] Some algorithms can achieve somewhat stronger properties, such as the ℓ_2/ℓ_1 guarantee. However, in this paper we focus exclusively on the ℓ_1/ℓ_1 case.

[2] Available at http://groups.csail.mit.edu/toc/sparse/wiki/index.php?title=Sparse_Recovery_Experiments

Coding theory. The iterative voting procedure used by SMP and SSMP is reminiscent of the iterative decoding algorithms used for *low-density parity-check* codes. In fact, iterative algorithms of this type have been used, e.g., in [16], [17], [18], to design sparse recovery algorithms. However, those algorithms were designed to work only for the case where the signal x is k-sparse or "almost" k-sparse. In contrast, SSMP is designed and guaranteed to work for arbitrary input signals x.

Message passing. There are interesting connections between SMP and SSMP and the message-passing-based recovery methods for sparse matrices [19], [20], [21]. In particular, the Counter Braids algorithm of [20] provides provable guarantees for the quality of recovered approximation. However, that scheme works only for non-negative signals x.

II. Preliminaries

A. Expanders

An essential tool for our constructions are *unbalanced expander graphs*. Consider a bipartite graph $G = (U, V, E)$. We refer to U as the "left" part, and refer to V as the "right" part; 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 $\{1, 2, \dots, n\}$ 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 1: 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.

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 $d = O(\log(|U|/s)/\epsilon)$ and $|V| = O(s \log(|U|/s)/\epsilon^2)$.

For many applications one usually 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 (optimal) parameters are known. However, it is known [22] 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 $\alpha > 0$. In the remainder of this paper, we will assume expanders with the optimal parameters.

B. RIP

A key property of matrices that enables sparse recovery is 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

$$\|x\|_p(1-\delta) \le \|Ax\|_p \le \|x\|_p$$

For the case of $p = 2$, the notion has been introduced in [23], where it was also shown that e.g. random gaussian matrices satisfy it with high probability. Unfortunately, sparse matrices that we use in this paper cannot satisfy the RIP(2) property, unless their number of rows is "large" [24]. However, it was shown [6] that such matrices *can* satisfy RIP(1). In particular, the adjacency matrices of expander graphs, scaled by a factor of $1/d$, do have this property. These are the matrices that we will use in this paper.

C. Notation

For a set S of nodes of G, the 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 $\Gamma(\{u\})$.

For any n-dimensional vector x, and $S \subset \{1 \dots n\}$, we use x_S to denote an $|S|$-dimensional projection of x on coordinates in S.

All uses of the norm notation $\|\cdot\|$ in this paper refer to the L1 norm.

We define $H_l[y]$ to be a "thresholding operator", which zeros out all but l largest in magnitude coefficients of the argument y.

III. Formal statement of the results

Theorem 1: There exists an algorithm (SSMP) that, for any k-sparse signal x and noise vector μ, given $b = Ax + \mu$, recovers x^* such that $\|x - x^*\|_1 = O(\|\mu\|_1)$.

Let $x^{(k)}$ be the best k-sparse approximation of x, i.e. $x^{(k)} = \text{argmin}_{k-sparse\ x'} \|x - x'\|_1$. Since $Ax + \mu = Ax^{(k)} + [\mu + A(x - x^{(k)})]$, and $\|A(x - x^{(k)})\|_1 \le \|x - x^{(k)}\|_1$ for any x, Theorem 1 immediately implies the following more general statement.

Corollary 2: For any parameter k, any vector x and noise vector μ, given $b = Ax + \mu$, the SSMP algorithm recovers x^* such that $\|x - x^*\|_1 = O(\|\mu\|_1 + \|x - x^{(k)}\|_1)$.

IV. Algorithm

In this section we describe the *Sequential Sparse Matching Pursuit* algorithm. We also present an overview of the algorithm analysis.

We start from a notation recap. We use x to denote the original k-sparse n-dimensional signal and μ to denote the m-dimensional measurement noise vector, where $\eta = \|\mu\|_1$. The $m \times n$ measurement matrix A is equal to $1/d \cdot A(G)$, where G is a $((c+1)k, (1-\epsilon/2)d)$-expander with left degree d, and $A(G)$ is its adjacency matrix. The matrix A satisfies RIP$(1, (c+1)k, \epsilon)$, i.e. for any $(c+1)k$-sparse vector x we have $(1-\epsilon)\|x\|_1 \le \|Ax\|_1 \le \|x\|_1$. The measurement vector is defined as $b = Ax + \mu$, where μ is the "noise vector".

The algorithm consists of two nested iterations: the inner one and the outer one. The goal of the inner iteration is to reduce the residual error $\|Ax^j - b\|_1$ by a constant factor, unless the error is already smaller than $c_l\|\mu\|_1$. This is done in $S = (c-1)k$ *update steps*, where each step reduces the

```
1) Let $j = 0$
2) Let $x^j = 0$
3) Repeat $T = O(\log(\|x\|_1/\eta))$ times
    a) Let $j = j + 1$
    b) Repeat $S = (c-1)k$ times
        • Find a coordinate $i$ and an increment $z$
          that minimizes $\|A(x^j + ze_i) - b\|_1$
        • Set $x^j$ to $x^j + ze_i$
        Remark: from Corollary 5 we have
        $\|x - x^j\|_1 \leq \|x - x^{j-1}\|_1/4 + C\eta$
    c) Let $x^j = H_k[x^j]$
        Remark: from Lemma 6 we have
        $\|x - x^j\|_1 \leq \|x - x^{j-1}\|_1/2 + 2C\eta$
4) Report $x' = x^T$
```

Fig. 1. Sequential Sparse Matching Pursuit algorithm

residual error by a factor $(1 - c_u/k)$ or better. In each update step, the algorithm finds a coordinate i and an increment z such that $\|A(x^j + ze_i) - b\|_1$ is minimized. For a given i, the value of z that minimizes the expression is equal to the median of the vector $(Ax^j - b)_N$, where N is the set of neighbors of i in G.

In each step of the outer loop, the inner loop is executed, and then the vector x^j is re-sparsified by keeping the largest (in the absolute value) k coordinates of x^j. As in SMP, this step approximately preserves the error of x^j.

A. Implementation and the running time

The algorithm can be efficiently implemented in the following way. For each i, we maintain the optimum increment $z = z_i$, together with the resulting change D_i to the L1 error norm. The D_i's are stored in a priority queue (e.g., in a heap), which enables finding the largest value of D_i, as well as update the value of each D_i in time $O(\log n)$.

When a value of some (say the i-th) coordinate of x^j is modified, this affects the values of d entries l of the vector $Ax^j - b$. In turn, each entry l can affect the values $D_{i'}$ of all $O(dn/m)$ neighbors i' of l. For each such i' we will need to recompute the median in $O(d)$ time, and update the heap. Thus, each coordinate update takes $O(d^2n/m(d + \log n))$ time. Therefore, the total running time of the algorithm is at most $O(\log(\|x\|_1/\eta) \cdot d^2nk/m(d + \log n))$, which simplifies to $O(\log(\|x\|_1/\eta) \cdot dn(d + \log n))$ since $m = \Theta(kd)$.

B. Correctness

The correctness proof is outlined in the remarks in the algorithm description. The key part of the argument is to show that if the residual error is at least $c_l\|\mu\|_1$, then each update step reduces it by a factor $(1 - c_u/k)$ or better. Specifically, we show the following lemma.

Lemma 3: If $\|Ax' - b\|_1 \geq c_l\|\mu\|_1$, then there exists an index i such that

$$\|A(x' - e_ix'_i + e_ix_i) - b\|_1 \leq (1 - c_u/k)\|Ax' - b\|_1$$

To relate the improvement in the residual error to a reduction in approximation error, we use the following claim:

Claim 4: It holds that:

$$(1-\epsilon)\|x' - x\|_1 \leq \|Ax' - b\|_1 + \|\mu\|_1 \leq \|x - x'\|_1 + 2\|\mu\|_1$$

Proof: From the RIP$(1, (c+1)k, \epsilon)$ property of A:

$$\begin{aligned} (1-\epsilon)\|x' - x\|_1 &\leq \|Ax' - Ax\|_1 \\ &\leq \|Ax' - (Ax + \mu) + \mu\|_1 \\ &\leq \|Ax' - b\|_1 + \|\mu\|_1 \\ &= \|A(x' - x) - \mu\|_1 + \|\mu\|_1 \\ &\leq \|x - x'\|_1 + 2\|\mu\|_1 \end{aligned}$$

■

We obtain the following corollary:

Corollary 5: There exist constants c and C such that, after the j-th inner loop, we have

$$\|x - x^j\|_1 \leq \|x - x^{j-1}\|_1/4 + C\eta$$

The sparsification step is handled by the following simple lemma.

Lemma 6: For any k-sparse vector x, and any vector x' we have

$$\|H_k[x'] - x\|_1 \leq 2\|x' - x\|_1$$

Proof: Observe that $H_k[x']$ is the closest k-sparse vector to x. Thus, we have $\|H_k[x'] - x'\|_1 \leq \|x - x'\|_1$. The lemma follows from triangle inequality. ■

Theorem 1 now follows from the above two lemmas.

V. Proof of Lemma 3

Let $\Delta = x' - x$. Since $b = Ax + \mu$, the thesis of the theorem can be rewritten as

$$\|A(\Delta - e_i\Delta_i) - \mu\|_1 \leq (1 - c_u/k)\|A\Delta - \mu\|_1$$

First, observe that, by triangle inequality, the assumption $\|Ax' - b\|_1 \geq c_l\|\mu\|_1$ implies that $\|Ax' - Ax\|_1 \geq (c_l - 1)\|\mu\|_1$. By the RIP1 property of A this implies

$$\|\Delta\|_1 = \|x' - x\|_1 \quad \geq \quad (1-\epsilon)(c_l - 1)\|\mu\|_1 \tag{2}$$

Let $T = \text{supp}(\Delta)$. Clearly, $|T| \leq (c+1)k$. Consider any $i \in T$, and let N_i be the set of neighbors of i in the graph G. The key idea in the proof is to split the neighborhood set N_i into a union of N_i^+ and N_i^-. This is done by proceeding as in the proof of the RIP1 principle in [6]. First, w.l.o.g., we reorder the coordinates so that $|\Delta_1| \geq |\Delta_2| \geq \ldots \geq |\Delta_n|$. Then, we enumerate the edges (i, j) of the graph G in lexicographic order. If (i, j) is the first edge from any node to the vertex j, then j is included in N_i^+, otherwise it is included in N_i^-. Note that the sets N_i^+ are pairwise disjoint.

From the expansion property of G, it follows that for any prefix of p first vertices i, we have $\sum_{i=1}^{p} |N_i^-| \leq \epsilon dp$. This in turn implies several other properties. In particular, for a constant $c_p > 1$, define $T^+ \subset T$ to contain all indices i such that $|N_i^-| \le c_p d\epsilon$. The following claim states that the coordinates in T^+ contain most of the ”L1 mass” of Δ.

Claim 7:

$$\sum_{i \in T^+} |\Delta_i| \ge (1 - 1/c_p)\|\Delta\|_1$$

Proof: Let $T^- = T \setminus T^+$. Consider all the indices $u_1, u_2, \ldots$ in T^- so that $u_1 < u_2 < \ldots$; by definition $|N_{u_i}^-| > c_p d\epsilon$ for all i. For any $k \ge 1$ consider the k-th index u_k. Then

$$k c_p d\epsilon < \sum_{j=1}^{k} |N_{u_j}^-| \le \sum_{i=1}^{u_k} |N_i^-| \le u_k d\epsilon$$

It follows that $u_k > k c_p$. Thus there are at least $k(c_p - 1)$ indices in $T^+ \cap \{1 \ldots u_k\}$ for all k. This allows us to partition T^+ in the following way: for any k let S_k be the set containing the smallest $c_p - 1$ elements of $(T^+ \cap \{1 \ldots u_k\}) \setminus \left(\bigcup_{i=1}^{k-1} S_i\right)$.

Notice that sets S_k are by construction disjoint. For any index u_k in T^-, we have a set of $c_p - 1$ unique indices in T^+, all of which are smaller than u_k; hence for any $v \in S_k$, $|\Delta_v| \ge |\Delta_{u_k}|$. Since

$$\begin{aligned} \|\Delta\|_1 &\ge \sum_{u_k \in T^-} \left(|\Delta_{u_k}| + \sum_{v \in S_k} |\Delta_v| \right) \\ &\ge \sum_{u_k \in T^-} c_p |\Delta_{u_k}| \end{aligned}$$

it follows that $\sum_{i \in T^+} |\Delta_i| \ge (1 - 1/c_p)\|\Delta\|_1$. ■

The above claim implies

$$\begin{aligned} \sum_{i \in T^+} \|(A\Delta_i e_i)_{N_i^+}\|_1 &\ge \sum_{i \in T^+} \Delta_i |N_i^+|/d \\ &\ge (1 - 1/c_p)(1 - c_p\epsilon)\|\Delta\|_1 \end{aligned} \tag{3}$$

The next claim concerns the amount of the L1 mass contributed to a coordinate j of the vector $A\Delta$ by the coordinates $i \in T$ such that $j \in N_i^-$. We can think about this mass as ”noise” contributed by the coordinates of Δ itself (as opposed to μ). Again, from the edge enumeration process, it follows that this contribution is low overall. Specifically:

Claim 8:

$$\sum_{i \in T^+} \|(A(\Delta - \Delta_i e_i))_{N_i}\|_1 \le \epsilon\|\Delta\|_1$$

Proof: Since sets N_i^+ are disjoint

$$\begin{aligned} \sum_{i \in T^+} \|(A(\Delta - \Delta_i e_i))_{N_i^+}\|_1 &\le \frac{1}{d} \sum_{i \in T^+} \sum_{j \in N_i^+} \sum_{i' | j \in N_{i'}^-} |\Delta_{i'}| \\ &\le \frac{1}{d} \sum_{j} \sum_{i | j \in N_i^-} |\Delta_i| \;\le\; \frac{1}{d} \sum_{i=1}^{n} |\Delta_i| \cdot |N_i^-| \end{aligned}$$

Define the value $s_p = pd\epsilon - \sum_{i=1}^{p} |N_i^-|$. We know that all $s_p \ge 0$.

$$\begin{aligned} \sum_{i=1}^{n} |\Delta_i| \cdot |N_i^-| &= d\epsilon|\Delta_1| - s_1|\Delta_1| + \sum_{i=2}^{n} |\Delta_i| \cdot |N_i^-| \\ &\le d\epsilon|\Delta_1| - s_1|\Delta_2| + \sum_{i=2}^{n} |\Delta_i| \cdot |N_i^-| \\ &\le d\epsilon(|\Delta_1| + |\Delta_2|) - s_2|\Delta_2| + \sum_{i=3}^{n} |\Delta_i| \cdot |N_i^-| \\ &\le d\epsilon(|\Delta_1| + |\Delta_2|) - s_2|\Delta_3| + \sum_{i=3}^{n} |\Delta_i| \cdot |N_i^-| \\ &\le \cdots \le d\epsilon \sum_{i=1}^{n} |\Delta_i| - s_n|\Delta_n| \le d\epsilon\|\Delta\|_1 \end{aligned}$$

■

The above claim implies

$$\sum_{i \in T^+} \|(A(\Delta - \Delta_i e_i))_{N_i^+}\|_1 \quad\le\quad \epsilon\|\Delta\|_1 \tag{4}$$

Now we proceed with the main part of the proof. Define

$$\text{gain}_i = \|A\Delta - \mu\|_1 - \|A(\Delta - e_i\Delta_i) - \mu\|_1$$

Observe that, equivalently, we have

$$\text{gain}_i = \|(A\Delta - \mu)_{N_i}\|_1 - \|(A(\Delta - e_i\Delta_i) - \mu)_{N_i}\|_1$$

Therefore

$$\begin{aligned} &\text{gain}_i = \|(A\Delta - \mu)_{N_i}\|_1 - \|(A(\Delta - e_i\Delta_i) - \mu)_{N_i}\|_1 \\ &= \|(A\Delta - \mu)_{N_i^+}\|_1 - \|(A(\Delta - e_i\Delta_i) - \mu)_{N_i^+}\|_1 + \\ &\quad + \left[\|(A\Delta - \mu)_{N_i^-}\|_1 - \|(A(\Delta - e_i\Delta_i) - \mu)_{N_i^-}\|_1\right] \\ &\ge \|(A\Delta - \mu)_{N_i^+}\|_1 - \|(A(\Delta - e_i\Delta_i) - \mu)_{N_i^+}\|_1 - \\ &\quad - \|(Ae_i\Delta_i)_{N_i^-}\|_1 \\ &\ge \|(A\Delta)_{N_i^+}\|_1 - \|(A(\Delta - e_i\Delta_i))_{N_i^+}\|_1 - 2\|\mu_{N_i^+}\|_1 - \\ &\quad - \|(Ae_i\Delta_i)_{N_i^-}\|_1 \\ &\ge \|(Ae_i\Delta_i)_{N_i^+}\|_1 - 2\|(A(\Delta - e_i\Delta_i))_{N_i^+}\|_1 - 2\|\mu_{N_i^+}\|_1 - \\ &\quad - \|(Ae_i\Delta_i)_{N_i^-}\|_1 \\ &\ge (1 - \frac{c_p\epsilon}{1 - c_p\epsilon})\|(Ae_i\Delta_i)_{N_i^+}\|_1 - \\ &\quad - 2\|(A(\Delta - e_i\Delta_i))_{N_i^+}\|_1 - 2\|\mu_{N_i^+}\|_1 \end{aligned}$$

Aggregating over all $i \in T^+$, we have

$$\begin{aligned} \sum_{i \in T^+} \text{gain}_i \ge \sum_{i \in T^+} \Big(&(1 - \frac{c_p\epsilon}{1 - c_p\epsilon})\|(Ae_i\Delta_i)_{N_i^+}\|_1 - \\ &- 2\|(A(\Delta - e_i\Delta_i))_{N_i^+}\|_1 - 2\|\mu_{N_i^+}\|_1 \Big) \end{aligned}$$

From Claims 7 and 8, and the fact that the sets N_i^+ are pairwise disjoint, we have

$$\begin{aligned}\sum_{i\in T^+} \text{gain}_i &\geq \big(1-\frac{c_p\epsilon}{1-c_p\epsilon}\big)(1-1/c_p)(1-c_p\epsilon)\|\Delta\|_1 - \\ &\quad -2\epsilon\|\Delta\|_1 - 2\|\mu\|_1 \\ &= \Big[\big(1-\frac{c_p\epsilon}{1-c_p\epsilon}\big)(1-1/c_p)(1-c_p\epsilon) - 2\epsilon - \\ &\quad - 2/((1-\epsilon)(c_l-1))\Big]\|\Delta\|_1 \\ &= C\|\Delta\|_1\end{aligned}$$

where we used Equation 2 to relate $\|\Delta\|_1$ and $\|\mu\|_1$. It follows that there exists a coordinate $i \in T^+$ such that

$$\text{gain}_i \geq \frac{C}{|T^+|}\|\Delta\|_1 \geq \frac{C}{(c+1)k}\|\Delta\|_1$$

At the same time

$$\begin{aligned}\|\Delta\|_1 &\geq \|A(x'-x)\|_1 \geq \|Ax'-Ax-\mu\|_1 - \|\mu\|_1 \\ &\geq \|Ax'-b\|_1 - \frac{\|\Delta\|_1}{(1-\epsilon)(c_l-1)}\end{aligned}$$

Therefore

$$\text{gain}_i \geq \frac{C}{(c+1)k}\|Ax'-b\|_1 / \left(1 - \frac{1}{(1-\epsilon)(c_l-1)}\right)$$

and we are done.

VI. Experimental results

A. Exact recovery of random sparse signals

We test SSMP on sparse signals in which k positions are randomly chosen and the value for each position is set to either $+1$ or -1. For a set of parameters m, k, the probability of correct recovery is estimated by randomly choosing a measurement matrix and attempting recovery of 100 random k-sparse signals, noting the fraction of successful recoveries. We show the results for signals of size $n = 20000$ in figure 2; graphs generated in the same manner are also shown for SMP [10] and LP [3], [25]. SSMP shows a significant improvement compared to SMP, requiring 40% less measurements for sparse signals. SSMP bridges the gap between SMP and the much slower method of ℓ_1-minimization using linear programming.

Our tests have shown that a small number of iterations T suffices for SSMP to recover sparse vectors; in this setting SSMP typically decodes signals faster than SMP. A benchmark comparing the running times of SSMP with SMP, GPSR [26], and ℓ_1−MAGIC [25] is shown in figure 3.

B. Approximate recovery of images

We also present experiments with recovery of natural images in the spirit of [3], [27], [21]. For a given image, we perform measurements on the vector containing the image's `db2` wavelet coefficients and use SSMP as well as other algorithms to recover the vector and reconstruct the image.

We use two 256×256 grayscale images: the boat image in [3], [27] and the peppers image in [21]. We evaluate the quality of each recovery by computing the ℓ_1 norm of the approximation error in the wavelet basis as well as the peak signal-to-noise ratio (PSNR). Note that the PSNR is a logarithmic function of the ℓ_2 approximation error which is the same in both the wavelet and the original (image) bases.

Figure 4 shows the results of our experiments on the two images. We plot the quality of the approximation with varying number of measurements m. For both SSMP and SMP we have to choose the sparsity of the recovery k. After experimenting with different values, we chose two values for k: one equal to 5% of the number of measurements m, which performs well with fewer measurements; and one equal to 10% of m which performs well with many measurements. We see that SSMP results in better quality that SMP, even exceeding the recovery quality of LP when the number of measurements is large. We also note that the behavior of the algorithms has been remarkably consistent between the two images.

Figure 5 shows how the three parameters S, T, and k affect the quality of the recovered image. For each plot, one parameter is fixed to a sensible value while the other two parameters are varied over a range. The tests are run on the peppers image, with $m = 17000$ measurements. Notice that increasing the number of inner steps S beyond a small multiple of k does not lead to significant improvement; on the other hand, increasing the number of iterations T generally increases the quality of the recovery. The running times are, as expected, proportional to the total number of steps $S \cdot T$.

References

[1] S. Muthukrishnan, "Data streams: Algorithms and applications (invited talk at soda'03)," *Available at http://athos.rutgers.edu/∼muthu/stream-1-1.ps*, 2003.

[2] P. Indyk, "Sketching, streaming and sublinear-space algorithms," *Graduate course notes, available at* `http://stellar.mit.edu/S/course/6/fa07/6.895`, 2007.

[3] E. J. Candès, J. Romberg, and T. Tao, "Stable signal recovery from incomplete and inaccurate measurements," *Comm. Pure Appl. Math.*, vol. 59, no. 8, pp. 1208–1223, 2006.

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

[5] M. Duarte, M. Davenport, D. Takhar, J. Laska, T. Sun, K. Kelly, and R. Baraniuk, "Single-pixel imaging via compressive sampling," *IEEE Signal Processing Magazine*, 2008.

[6] R. Berinde, A. Gilbert, P. Indyk, H. Karloff, and M. Strauss, "Combining geometry and combinatorics: a unified approach to sparse signal recovery," *Allerton*, 2008.

[7] W. Dai and O. Milenkovic, "Subspace pursuit for compressive sensing: Closing the gap between performance and complexity," *Arxiv:0803.0811*, 2008.

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

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

[10] R. Berinde, P. Indyk, and M. Ružić, "Practical near-optimal sparse recovery in the l_1 norm," *Allerton*, 2008.

[11] A. C. Gilbert, S. Guha, P. Indyk, Y. Kotidis, S. Muthukrishnan, and M. J. Strauss, "Fast, small-space algorithms for approximate histogram maintenance," in *ACM Symposium on Theoretical Computer Science*, 2002.

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

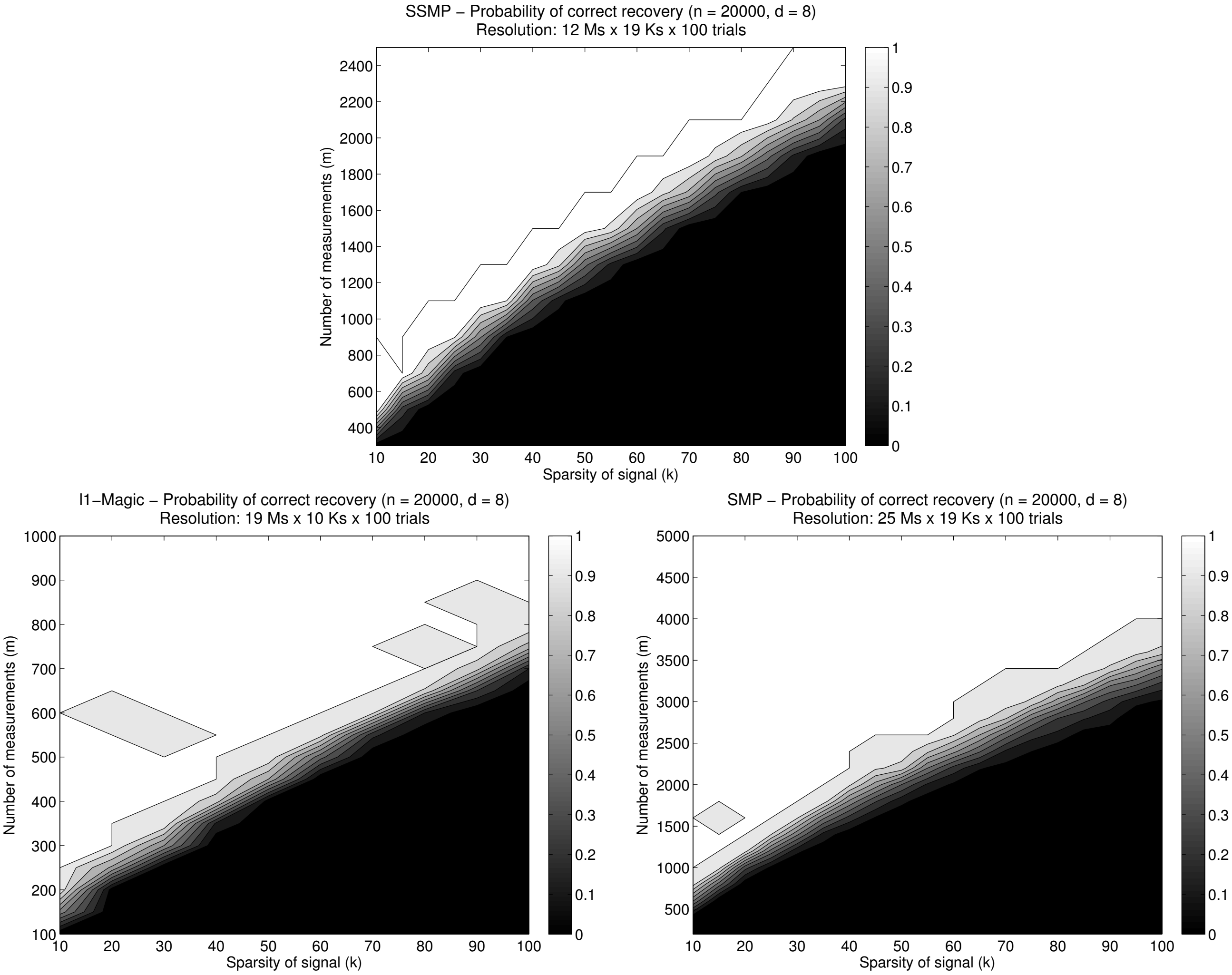

Fig. 2. Results of exact recovery experiments for SMP, SSMP, and ℓ_1-MAGIC. All plots are for the same signal length $n = 20000$, matrix sparsity $d = 8$ and signal sparsities $k \in [10, 100]$. The resolution of each plot is shown: the number of vertical axis divisions ("Ms"), the number of horizontal axis divisions ("Ks"), and the number of trials per point. SMP uses 10 iterations; SSMP uses $S = 4k$ inner steps and $T = 1$ iteration. Note that higher parameter settings for SMP and SSMP yield similar results.

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

[14] A. Gilbert and P. Indyk, "Sparse recovery using sparse matrices," *Manuscript*, 2009.

[15] D. Donoho, A. Maleki, and A. Montanari, "Message passing algorithms for compressed sensing," *PNAS*, 2009.

[16] W. Xu and B. Hassibi, "Efficient compressive sensing with determinstic guarantees using expander graphs," *IEEE Information Theory Workshop*, 2007.

[17] P. Indyk, "Explicit constructions for compressed sensing of sparse signals," *SODA*, 2008.

[18] S. Jafarpour, W. Xu, B. Hassibi, and A. R. Calderbank, "Efficient and robust compressed sensing using high-quality expander graphs," *Manuscript*, 2008.

[19] S. Sarvotham, D. Baron, and R. G. Baraniuk, "Compressed sensing reconstruction via belief propagation," *Technical Report ECE-0601, Electrical and Computer Engineering Department, Rice University*, 2006.

[20] Y. Lu, A. Montanari, B. Prabhakar, S. Dharmapurikar, and A. Kabbani, "Counter braids: A novel counter architecture for per-flow measurement," *SIGMETRICS*, 2008.

[21] M. Akçakaya, J. Park, and V. Tarokh, "Compressive sensing using low density frames," *Submitted to IEEE Trans. on Signal Processing*, 2009.

[22] V. Guruswami, C. Umans, and S. P. Vadhan, "Unbalanced expanders and randomness extractors from Parvaresh-Vardy codes," in *IEEE Conference on Computational Complexity (CCC 2007)*, 2007, pp. 96–108.

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

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

[25] E. J. Candès and J. Romberg, *ℓ_1-MAGIC: Recovery of Sparse Signals via Convex Programming*, 2005, available at http://www.acm.caltech.edu/l1magic.

[26] M. Figueiredo, R. Nowak, and S. Wright, "Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems," *IEEE Journal of Selected Topics in Signal Processing*, vol. 1, 2007.

[27] R. Berinde and P. Indyk, "Sparse recovery using sparse random matrices," *MIT-CSAIL Technical Report*, 2008.

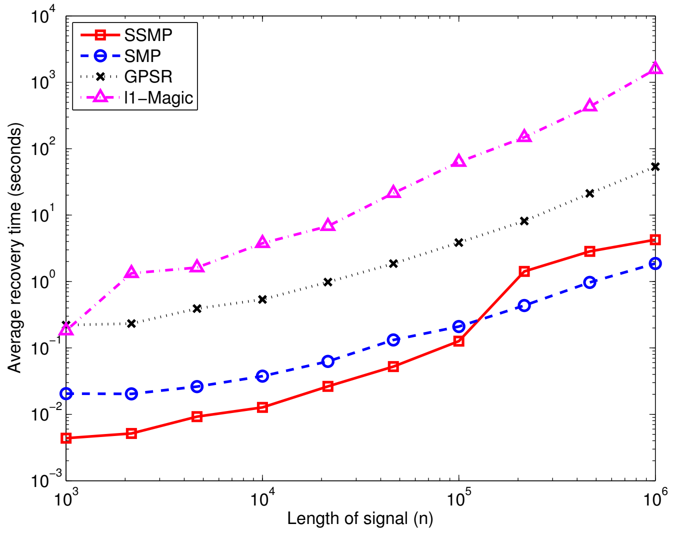

Fig. 3. Comparison of recovery times for sparse signals. We used the following parameters: the signal sparsity $k = 0.002n$, the number of measurements $m = 0.1n$, and the matrix sparsity parameter $d = 8$. Note that, for $n = 20,000$, the parameters fall into the regions depicted in figure 2. SSMP is ran with $S = 4K$ inner steps and $T = 1$ iteratios; the SMP algorithm is run for 10 iterations. The settings roughly correspond to the parameters in figure 2.

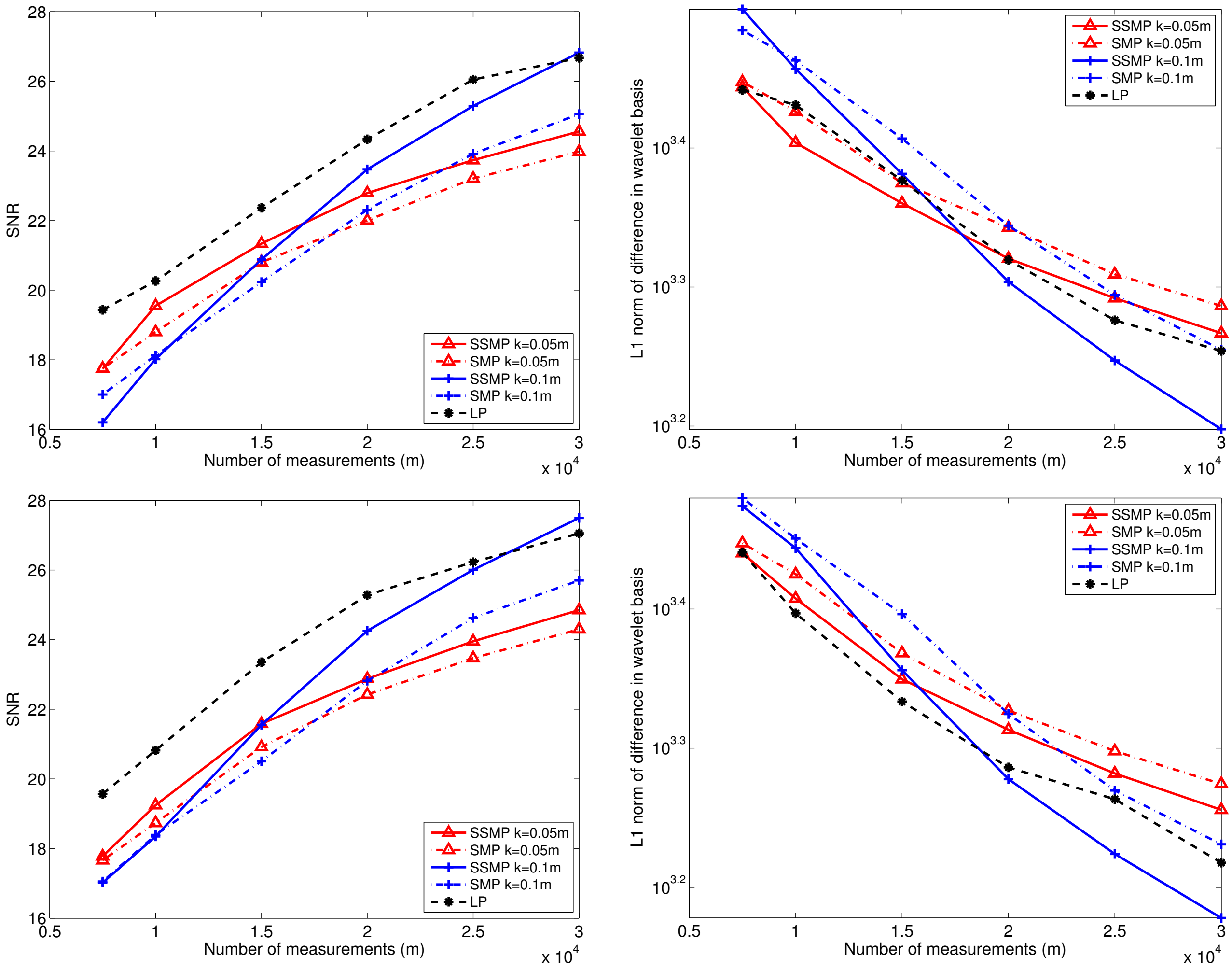

Fig. 4. Approximate recovery experiments with the boat image (top) and peppers image (bottom). SSMP is ran with $S = 10000, T = 20$. SMP is ran for 100 iterations with convergence control parameter 0.5. Matrix sparsity is $d = 8$.

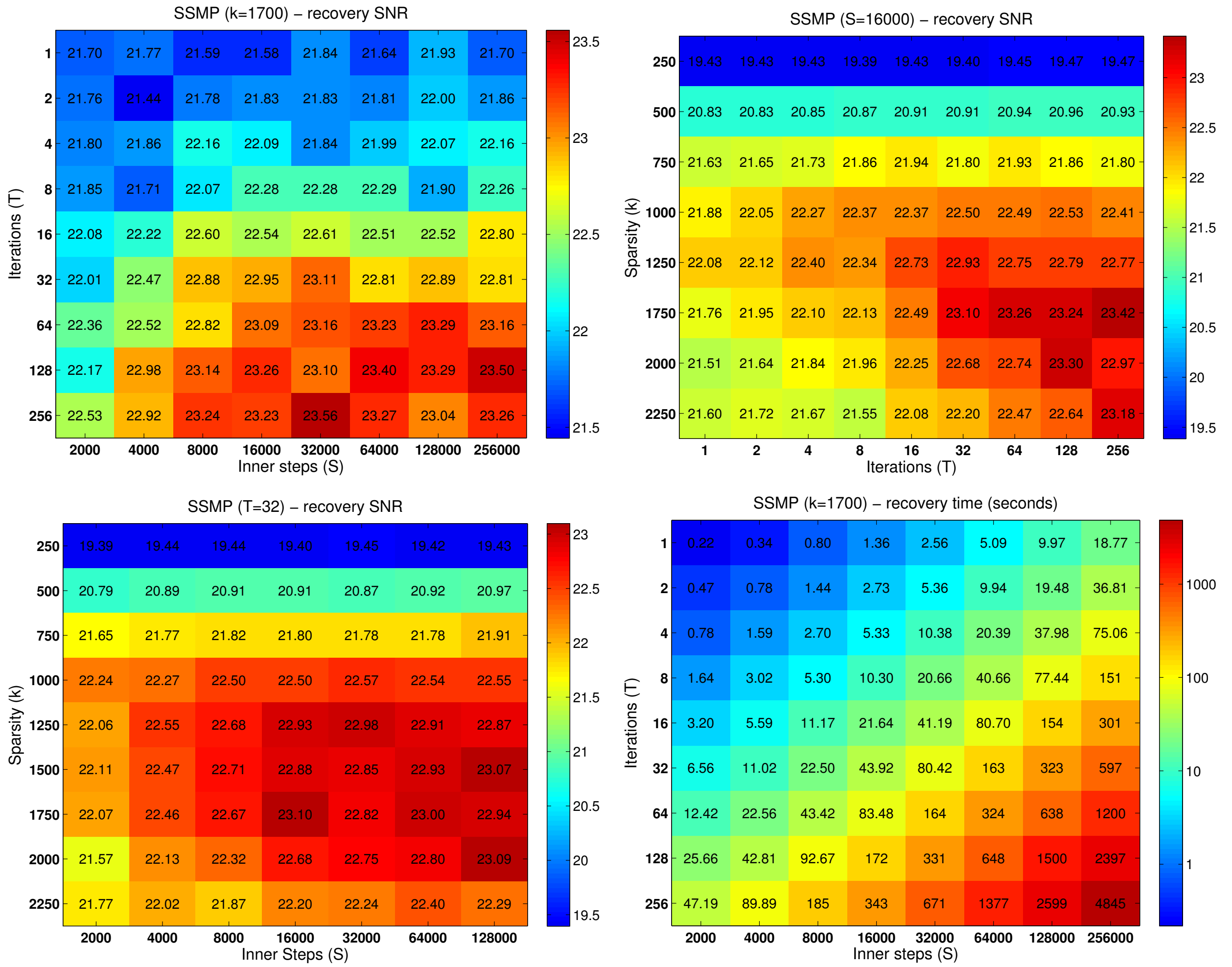

Fig. 5. Experimental characterisation of SSMP (peppers image, $m = 17000$, $d = 8$).