On Fast Decoding of Noiseless Signals from One-Bit or Phaseless Measurements

Vasileios Nakos

March 30, 2016

Abstract

In the problem of one-bit compressed sensing, the goal is to find an \( \epsilon \)-close estimation of a \( k \)-sparse vector \( x \in \mathbb{R}^n \) given the signs of the entries of \( y = \Phi x \), where \( \Phi \) is called the measurement matrix. Similarly, in the problem of compressed sensing from phaseless measurements, the goal is to estimate a \( k \)-sparse vector \( x \in \mathbb{C}^n \) given only the magnitudes of the measurements \( y = |\Phi x| \). For the one-bit compressed sensing problem, previous work [PV13b, GNJN13] achieved \( \Theta(\epsilon^{-6}k \log(n/k)) \) and \( \tilde{O}(\frac{1}{\epsilon}k \log(n/k)) \) measurements, respectively, in the for-all model, but the decoding time was \( \Omega(nk \log(n/k)) \). In this paper, using tools and techniques developed in the context of two-stage group testing, we contribute towards the direction of very fast decoding time. First, we give a simple scheme with \( O(k \log n + \epsilon^{-5}k + \log n \log \log n \log k) \) measurements and decoding time \( \text{poly}(k, \log n) \) in the for-each model. In addition, for the for-all version of the problem we give a scheme with \( O(k^2 \log(n/k) \log \log k \log \log n + \epsilon^{-5}k \log(n/k)) \) measurements, that allows decoding of the signal in \( \text{poly}(k, \log n) \) time. For the phaseless measurements version in the for-all model, we give a recovery scheme with \( O(k^2 \log \frac{k}{\epsilon} \log \log k) \) measurements with the same running time. Furthermore, we show how to get an exponential improvement (in terms of \( n \)) in the decoding time of the support recovery algorithm in [GNJN13]. To the best of our knowledge, these are the first schemes that allow such a fast decoding time with a number of measurements that have such a small polynomial dependence on \( k \).

1 Introduction

1.1 Standard Compressed Sensing

The compressed sensing framework describes how to reconstruct a vector (signal) \( x \in \mathbb{R}^n \) given the linear measurements \( y = \Phi x \) where \( \Phi \in \mathbb{R}^{m \times n} \) for some \( m \ll n \). This is an undetermined system with \( n \) variables and \( m \) equations. In many applications, however, such as images, we know that the vector \( x \) can be approximated by a \( k \)-sparse one. In this case, the matrix \( \Phi \) contains a sufficient amount of information to roughly recover \( x \) if \( m \) is large enough; in particular, as shown in [CT06], [CRTV05], the signal can be reconstructed exactly from \( \Theta(k \log(n/k)) \) measurements when \( \Phi \) is a Gaussian matrix. In order to do this, one has to solve the non-convex program

\[
\min \|x\|_0 \; \text{s.t.} \; y = \Phi x
\]

*Harvard University. vasileiosnakos@g.harvard.edu. Supported in part by ONR grant N00014-15-1-2388
In order to avoid solving this problem, \cite{Chen01, Cai06} show that we can use Basis Pursuit (BP), which changes the objective to $\min \|x\|_1$. This can be achieved using linear programming.

Compressed sensing, or sparse recovery, has appeared to be a very useful tool in many areas such as analog-to-digital conversion \cite{Joh13b}, threshold group testing \cite{Ahl06}, Discrete Signal Processing \cite{Don06}, streaming algorithms \cite{Mut05} and even bioinformatics \cite{Kou10}. Depending on the application, different parameters are needed to optimize (measurements, decoding time, encoding time, low error probability). Often we discriminate between the for-all model (or universal recovery) and the for-each model (non-universal recovery). In the for-all model, a single matrix is picked, which allows reconstruction of all $k$-sparse vectors, whereas in the for-each model the measurements are chosen at random such that, for some error probability $\delta$, they will contain plenty of information to reconstruct a single vector $x$ with probability at least $1 - \delta$. We note that all aforementioned papers refer to the for-all model.

Moreover, we need to keep the number of measurements low and achieve sublinear decoding time. The first paper that achieved $\text{poly}(k, \log n, \epsilon^{-1})$ decoding time with $\Theta(k \log(n/k))$ measurements in the for-each model is \cite{Gir12}; the error probability was improved later in \cite{Gir13} using a much more complicated scheme. In the for-all model, Porat and Strauss devised a scheme with $\mathcal{O}(\epsilon^{-3}k \log(n/k))$ measurements accompanied with the first sublinear decoding procedure running in time $\mathcal{O}(k^{1-\alpha}n^\alpha)$, for any constant $\alpha \in \mathbb{N}$. Later, in \cite{Gir14}, the authors manage to bring the dependence of $\epsilon$ down to the right order of $\epsilon^{-2}$ and achieve runtime $\text{poly}(k, \log n)$, when $\epsilon \leq \left(\frac{\log k}{\log n}\right)^\gamma$, for any constant $\gamma > 0$.

### 1.2 One-Bit Compressed Sensing

In applications many times compressed sensing measurements must be quantized, since the requirement of infinite precision is not realistic: any measurement must be mapped to a small finite value in some universe. In hardware implementations, for example, where quantizers are implemented using comparators to zero \cite{Bar08}, there is need of quantization to one-bit measurements. Comparators are indeed fast, but they are expensive, so there minimizing their usage is really important. Moreover, dynamic range issues are a smaller problem in the case of one-bit quantizers. Details and motivation can be found in \cite{Bar08}.

It is clear that quantization, increases the complexity of the decoding procedure and, additionally, is abiding: given $y = \text{sign}(\Phi x)$ it is impossible to get the exact vector back. Previous results inquired the case in which the quantization maps each coordinate to $\{-1, +1\}$, which means that we learn only the sign of each coordinate. First, it is not obvious whether there is sufficient information to reconstruct a signal given its one-bit measurements. Of course, since we cannot know the length of the signal, nor the exact signal (even if its length were given), but the following question remains: assuming that the length of the signal is 1, can we find another signal that it is close to it in the $\ell_2$ norm?

The problem was first studied in the work of Boufounos and Baraniuk \cite{Bar08}, where the authors suggest recovering the signal $x$ by solving the optimization problem

$$
\min_x \|x\|_1 \text{ s.t.: } y \odot Ax \geq 0, \|x\|_2 = 1,
$$

where $\odot$ stands for the element-wise product between two vectors. The goal is to find a vector $y$ on the unit sphere such that $\|y - \frac{\bar{x}}{\|\bar{x}\|_2}\| \leq \epsilon$. It is clear that this relaxation requires solving a non-convex program, something which Laska et al. \cite{Las11} tried to remedy by giving an optimization
algorithm that finds a stationary point of the aforementioned program; both papers, however, do not provide provable guarantees for the number of measurements needed. An alternative formulation was studied in [JLBB13a], which showed that the number of measurements could be brought down to $O(\epsilon^{-1}k \log n)$, but the main obstacle of the non-convex formulation remained. In [PV13a] Vershynin and Plan gave the first computationally tractable algorithm for the problem of one-bit compressed sensing by designing a compressed sensing scheme that approximately recovers a $k$-sparse vector from $O(\epsilon^{-5}k \log^2(\frac{n}{k}))$ one-bit measurements via a linear programming relaxation in time $O(\epsilon^{-5}kn^4 \log(n/k))$. Their techniques were based on random hyperplane tessellations; the main geometric lemma they needed was that $O(k \log(n/k))$ random hyperplanes partition the set of $k$-sparse vectors with unit norm into cells, each one having small diameter. In [PV13b] they improve the number of measurements to $O(\epsilon^{-6}k \log(\frac{n}{k}))$ and the decoding time to $O(\epsilon^{-6}nk \log(n/k))$ by analyzing a simple convex program. Surprisingly, their analysis result still holds even if each measurement bit is flipped with probability close to $\frac{1}{2}$. Their results can also be generalized to other sparsity structures, where the crucial quantity that determines the number of measurements is the gaussian mean-width of the set of all unit vectors having a specific sparsity pattern. Last but not least, they manage to handle adversarial noise. In [GNNJ13] a two-stage algorithm with $O(\epsilon^{-6}k \log(n/k))$ measurements and $O(nk \log(n/k) + \epsilon^{-6}(k \log(n/k))^5)$ decoding time was proposed. Apart from recovering the vector, other algorithms that recover only the support of the signal have been proposed; see for example [GNNJ13] [GNR10].

In [GVT98] it is suggested that even if the support of the vector is known, the dependence on $\epsilon$ must be at least $\frac{1}{\epsilon}$. In order to circumvent this, alternative quantization schemes were proposed, with the most common being Sigma-Delta quantization [GLP+10] [KSY14]. In [BFN+14] Baraniuk, Foucart, Needell, Plan and Wooters manage to bring the dependence on $\epsilon$ down to $\log(\frac{1}{\epsilon})$ if the quantizer is allowed to be adaptive and the measurements take a special form of threshold signs.

### 1.3 Compressed Sensing From Phaseless Measurements

In several fields of science, such as crystallography, electron microscopy, astronomy and optical imaging, there is a need to recover a function given the magnitude of the values of its Fourier Transform. In optical settings, for example, when the light emanates from a laser, the measurement of the phase of a light wave from detection devices is impossible [SEC+15]. Moreover, a very common technique is Coherent Diffractive Imaging (CDI), which is used for reconstructing the image of nanoscale structures such as nanotubes [ZVG+03], nanocrystals [VRO+05], defects [PWV+06], proteins, and generally any biological macromolecule [MCHR+03]. When this techniques is used, however, the gathered data lack phase information.

Phase retrieval is also significant in astronomy [FD87] [Rod04]. In this case the objects are usually distant stars and the optical signal that is being measured is the intensity of the light. Atmospheric turbulence or limitations in the technology of the optical system, such as the diameter of the telescope, forced researchers to develop a new family of techniques to record long-exposure images [Gons2] [LBL02] [CCN+14]: this family is called interferometric imaging. However, all these methods have a common drawback: it is almost impossible to get the phase of the image and only the intensity of the measurements is known.

Much work has been done on cases in which the measurements acquired are Fourier measurements. Since there is no unique solution, the aim is to find the solution up to a global phase. However, even under this assumption the problem does not always have a unique solution. Furthermore, even with the existence of a unique solution, there is no guarantee that there exists a fast algorithm
to find this solution. However, over the last decade, new technological advances in optics and new imaging techniques have provided algorithm designers with more flexibility to develop algorithms that solve phase-retrieval problems \cite{TY12, EAA10, SPR07, CDT09, SAT11, SSW08, BvPB12}. For example, like in classical Compressed Sensing, tools from optimization theory have been used \cite{CESV15, CSV13}. More recent work has begun exploring connections between phase retrieval for optical imaging and sparse recovery \cite{SESS11, BR15, MRB07, BCMN14, BE13}. However, since the phase-retrieval problem is nonlinear, employing sparsity-based concepts in phase retrieval requires the modification of known algorithms that are used in the field of compressed sensing; a recent survey on reconstructing a sparse signal from its Fourier measurements gives a nice overview of the current techniques \cite{JEH15}.

The problem was first studied from an algorithm perspective in \cite{SESS11} under the name of quadratic compressed sensing, where the authors considered a semidefinite of the problem and suggest an algorithm based on iterative rank minimization. Like in classical compressed sensing schemes, another iterative algorithm was proposed in \cite{JOH12a}, while the greedy method of \cite{BE13} and \cite{SBE12} is more efficient and equivalently accurate. In \cite{LV13} the authors show that $O(k^2 \log n)$ gaussian measurements suffice to estimate the vector, using convex programming. \cite{SBE14, JOH12b} investigate the case when the sensing matrix is restricted to be a Fourier matrix; in \cite{OJF15} an SDP-approach is given that recoves vectors with spars ity up to $\sqrt{n}$. When the sensing matrix is allowed to be designed, the authors in \cite{JOH13} indicate that using $O(k \log n)$ intensity measurements a $k$ sparse signal can be estimated in time $O(nk \log n)$. The limits of the problem in the for-all model was investigated in a series of papers \cite{OE14, AT13} it was shown that $4k - 2$ equations suffice for full recovery; if the measurements are restricted to be Fourier measurements then they can achieve full recovery from $2(k^2 - k + 1)$. We remark, however, that the above proofs are injective and hence by nature non-algorithmic; as a result they do not provide tractable algorithms, something that makes them impractical. Eldar and Mendelson in \cite{EM14} show that $O(k \log(\frac{n}{k} \log k))$ noisy measurements suffice for stable recovery. As far as the for-each model is concerned, a series of papers \cite{PLR14, YPB15, CBJC14} investigate the possibility of fast decoding with a small number of measurements, where it is shown that for the noiseless case $O(k)$ decoding time with $O(k)$ measurements is possible.

1.4 Our Results

For one-bit compressed sensing, we give a simple scheme for noiseless signals (namely when $x$ is $k$-sparse), that has an almost optimal number of measurements and a very fast decoding time ($poly(k, \log n)$) in the for-each model for one-bit compressed sensing with $O(k \log n + \epsilon^{-5} k + \log n \frac{\log \log n}{\log k}$) and a recovery scheme with $O(k^2 \log(n/k) \log \log n + \epsilon^{-5} k \log(n/k))$ measurements in the for-all model (universal recovery). We should note that the scheme corresponding to the for-each version of the problem, uses a number of measurements that depends additively on $\epsilon^{-5}$ and $\log n$; these two factors where multiplied together in previous work.s Furthermore, for the phaseless measurements version we give a scheme with $O(k^2 \log(n/k) \log \log n)$ measurements and decoding time $poly(k, \log n)$ for signals with real coordinates in the for-all model. All of our algorithms are based on the simple observation that a superset of the support of the signal can be recovered using a group-testing matrix coming from a two-stage group testing scheme. Using ideas from \cite{GLPS14}, we are able to get the decoding time down to $poly(k, \log n)$. Last but not least, we describe how our techniques give an algorithm for support recovery in the one-bit compressed sensing setting, with $O(k^3 \log n)$ measurements and $poly(k, \log n)$ decoding time, which is an exponential improvement -
in terms of the dependence of the decoding time in $n$- over one of the main algorithms in [GNJN13].

As far as we know, our work is the first that looks at sublinear decoding time in the one-bit compressed sensing framework and the first one that achieves a fast recovery time for the phaseless case in the for-all setting, which are considerable improvements over previous works; we believe that a strong point of our schemes is their neatness. We remark, however, that our algorithms do not handle noise.

2 Main Part

2.1 Basic Background

We define the sign function as $\text{sign}(z) = +1$, for $z \geq 0$ and $\text{sign}(z) = -1$ for $z < 0$. For a vector $x$, we define $\text{sign}(x)_i = \text{sign}(x_i)$, for all $i \in [n]$.

Any one-bit compressed sensing scheme is defined by a pair $(\mathcal{D}, \text{Dec})$ where $\mathcal{D}$ is a distribution over $\mathbb{R}^{m \times n}$ and $\text{Dec}$ is an algorithm that takes input $\text{sign}(\Phi x)$ for some $x \in \mathbb{R}^n$ with $\|x\|_0 \leq k$ and gives back a vector $\hat{x}$. We will refer to $\text{Dec}$ either the “decoder” or “decoding procedure”.

When we say number of measurements we refer to $m$ and the decoding time is the running time of $\text{Dec}$. We also define $\Sigma_k = \{x : \|x\|_0 \leq k, \|x\|_2 = 1\}$ to be the set of all $k$-sparse vectors and $\Sigma_{\leq k} = \{x : \|x\|_2 = 1, \|x\|_0 \leq k\}$ the set of unit norm vectors with at most $k$ non-zero coordinates. We will pick $m$ random hyperplanes chosen uniformly at random from the Haar measure. This leads to a partition of $\Sigma_{\leq k}$, which we will call a tessellation. The resulting partition of $\Sigma_{\leq k}$ by this collection of hyperplanes is called a random tessellation of $\Sigma_{\leq k}^1$. The cells of the tessellation are formed by intersection of $\Sigma_{\leq k}^1$ and the $m$ random half-spaces with particular orientations. In what follows, we will make use of the following result from [PV13a], concerning the maximum diameter in any partition in a random tessellation:

**Theorem 1.** [PV13a, Theorem 4.3]. Let $m \leq n$ be positive integers. Consider the tessellation of the set $\Sigma_{\leq k}^1$ by $m$ random hyperplanes in $\mathbb{R}^n$ chosen independently and uniformly from the Haar measure. Let $\epsilon < 1$ and assume that $m = \Omega(\epsilon^{-5}k\log(n/k))$. Then with probability at least $1 - 2\exp(-em)$, all cells of the tessellation of $\Sigma_{\leq k}^1$ have diameter at most $\epsilon$.

For $x \in \mathbb{R}^n$ we denote its support set by $\text{supp}(x)$. For each $S \subset n$, let $x_S \in \mathbb{R}^{|S|}$ denote the signal $x$ restricted to coordinates in $S$. Similarly, for a matrix $M \in \mathbb{R}^{r \times n}$ and each $S \subset n$ let $M_S \in \mathbb{R}^{r \times |S|}$ be the matrix $M$ restricted to columns in $S$. Given two matrices $A, B$ we denote the Hadamard or entrywise product by $A \odot B$; so the $(i, j)$ entry of $AB$ is $A_{ij} \cdot B_{ij}$. The row-direct sum $A \uplus B \in \mathbb{R}^{(r_1 + r_2) \times n}$ of $A, B$ is defined as the vertical concatenation of $A$ and $B$. For a vector $x$, we define $|x|$ to be the vector where each entry of $x$ is replaced by its absolute value. For a real number $x$ we define $\text{val}(x) = 1$ if $x \neq 0$ and $\text{val}(x) = 0$ if $x = 0$.

We also give the definition of the tensor product of two matrices.

**Definition 1.** Let $A \in \{0, 1\}^m \times \{0, 1\}^N$ and $A' \in \{0, 1\}^{m'} \times \{0, 1\}^N$. The tensor product $A \otimes A'$ is an $m m' \times N$ binary matrix with rows indexed by the elements of $[m] \times [m']$ such that for $i \in [m]$ and $i' \in [m']$, the rows of $A \uplus A'$ indexed by $(i, i')$ is the coordinate-wise product of the $i$-th row of $A$ and $i'$-th row of $A'$.

In order to proceed, we have to explain the difference between the for-all and the for-each model. Let $P_x$ be the predicate that the sparse recovery scheme returns a vector $\hat{x}$ such that
\[ \| \frac{\tilde{x}}{\| \tilde{x} \|_2} - \frac{\hat{x}}{\| \hat{x} \|_2} \|_2 > \epsilon, \] when the matrix \( \Phi \) is chosen from the distribution \( \mathcal{D} \). Let \( \delta \) be some target probability. In the for-each model the guarantee is that \( \forall x \in \Sigma^1_k, P[P_x] \leq \delta \). In the for-all model the guarantee is that \( P[\exists x \in \Sigma^1_k : P_x] \leq \delta \). The randomness of the scheme is over the distribution \( \mathcal{D} \).

### 2.2 Group Testing

In the group testing problem, we have a large population, which consists of “items”, with a known number of defectives. The goal is to find the defectives using as few tests as possible, where a test is just a query whether a certain subset of items contains at least one defective. The group testing problem was first studied by Dorfman in [Dor43]. There are two types of algorithms for this problem, namely adaptive and non-adaptive. In the first case, the outcome of previous tests can be used to determine future tests, whereas in non-adaptive algorithms all tests are performed at the same time. Group testing has many applications in DNA library screening and detection of patterns in data; more can be found in [CCH11], [CH08].

Any solution for the group testing problem corresponds to a binary matrix, where the number of rows equals the number of tests and the number of columns equals the cardinality of the population. Given such a matrix \( M \) and a vector \( x \) indicating the positions of the defectives, we should be able to identify \( x \) from \( Mx \), where the addition here corresponds to the addition operation of Boolean algebra. Since decoding time is important, the brute-force algorithm that iterates over each possible subset in order to recognise the defective set does not suffice. However, one can design matrices such that the \( O(nk) \) decoding algorithm, which eliminates items belonging to negative tests and returns all the other items, correctly identifies all defective items [DH99]. In literature these matrices are known as \( k \)-disjunct matrices.

In the present case, we are interested in the so-called two-stage group testing problem, where two stages are allowed: the first stage recognises a superset of the defectives, and the second stage, which is performed after seeing the results of the first stage, recognizes the exact set of the defectives by querying separately for each one. We refer to \((k, l)\) two-stage group testing as the case when there are \( k \) defectives and the superset is allowed to have up to \( k + l \) elements. In fact, this is equivalent to the existence of a matrix \( M \) such that given \( Mx \) one can find a set \( S \) with \( k + l \) elements such that all defectives are included in \( S \). The same naive algorithm, which eliminates all items that belong to a negative test and returns all other items, will be used here. A matrix is called list-disjunct if this algorithm finds a superset of the support with at most \( k + l \) elements. The term ‘list-disjunct’ appeared in [INR10], although it was also studied before in [DBGV05], under the name of super-imposed codes, and in [Ras90], under the name of list-decoding super-imposed codes. In [NPR11] Ngo, Porat and Rudra give efficient and strongly explicit constructions of matrices that allow two-stage group testing, which are also error-tolerant, in the sense that they can correct \( e_0 \) false positives and \( e_1 \) false negatives in sub-linear time and additional \( \Theta(e_0 + k \cdot e_1) \) tests. They also prove matching lower bounds for several cases, including the case that \( k = \Theta(l) \).

A \((k, l)\) group testing scheme is a tuple \((M, R)\), where \( M \) is a matrix in \( \{0, 1\}^{m \times n} \) and \( R \) a procedure that takes as input \( Mx \) and outputs a vector \( y \) with the following guarantee: if \( x \) has at most \( k \) non-zeros (in other words, if at most \( k \) items are defective out of \( n \)) then \( y \) has at most \( k + l \) non-zeros and \( \text{supp}(x) \subseteq \text{supp}(y) \). The worst-case running time of the procedure \( R \) corresponds to the decoding time of the scheme.
2.3 For-each Sparse Recovery from One-Bit Measurements

We begin by modifying known group-testing recovery schemes such that they can be applied to the for-each model, using less measurements. We first proceed by reminding the definition of a \((k,k)\)-disjunct matrix. Given a matrix \(M\) and a vector \(x\), we will say that the naive decoding algorithm succeeds if and only if it returns a set \(\hat{S}\) with \(\text{supp}(x) \subseteq \hat{S}\) and \(|\hat{S}| \leq 2k\). The next definition and the lemma following it, appear in [NR10]; for our purposes, however, we phrase them slightly differently.

**Definition 2.** A matrix \(M\) is called \((k,k)\)-disjunct if and only if, for all sets \(S\) the naive decoding algorithm succeeds on \(S\).

**Lemma 1.** A matrix \(M\) is \((k,k)\)-disjunct if and only if for any two disjoint sets \(S,T\) of size \(k\), there exists a row of \(M\) in which all columns in \(T\) have 0, but at least one column in \(S\) has a 1.

We start by solving a for-each version of the two-stage group testing problem. This means, we do not want our matrix to work for all possible \(k\)-sparse vector is estimated. We proceed by showing the construction of this matrix, let it be \(A'\). Let \(q = 4k\) and \(d = 5 \log n + \frac{\log \log n}{k}\). Let \(C_{id} : [q] \to \{0,1\}^q\) be the identity code, namely the collection of all \(4k\) standard basis vectors in \(\mathbb{R}^q\). Let \(C_{r} : [n] \to [q]^d\) be a random code. Consider the concatenation of the aforementioned codes, \(C_{id} \circ C_{r}\). Let \(A_n\) be the matrix that has columns the codewords of the concatenated code \(C_{id} \circ C_{r}\). For the construction, we closely follow [NPRRT], using recursion.

Let us give the high level idea first. We may assume that our signal is organized in a rectangular grid \(G\) of dimensions roughly \(\sqrt{n} \times \sqrt{n}\). We define two other matrices \(A_{\sqrt{n}}^{(1)}\) and \(A_{\sqrt{n}}^{(2)}\), such that \(A_{\sqrt{n}}^{(1)}\) considers all nodes in the same row as a super-node and \(A_{\sqrt{n}}^{(2)}\) considers all nodes in the same column as a supernode. Hence, from the point of view of these matrices the universe size is \(\sqrt{n}\) and the sparsity parameter is still \(k\). In other words, \(A_{\sqrt{n}}^{(1)}\) is the matrix where the \(i\)th column is identical to the \(j\)th column of \(A\) such that the first \(\frac{1}{2} \log n\) bits of \(i\) is \(j\). For \(A_{\sqrt{n}}^{(2)}\) the same thing holds, except that the last \(\frac{1}{2} \log n\) bits of \(i\) is \(j\). Let \(A' = A_{\sqrt{n}}^{(1)} \cup A_{\sqrt{n}}^{(2)} \cup A_n\). By running the naive decoder on \(A_{\sqrt{n}}^{(1)}\) and \(A_{\sqrt{n}}^{(2)}\), we get two sets \(S_1,S_2\) of at most \(2k\) elements each one. Note that these sets correspond to columns and rows of the initial signal. Then all elements in the support of our vector lie in both an element of \(S_1\) and an elements of \(S_2\), hence there are at most \(4k^2\) of these. By running the naive decoder on \(A_n\) and checking these \(4k^2\) elements we get a set \(\hat{S}\) that is guaranteed to have at most \(2k\) elements and contain all defective items. This procedure reduces the dependence on \(n\) only to \(\sqrt{n}\), and not to the desired \(\log(n/k)\). We will apply recursion a couple of times to drive this dependence down to logarithmic.

The construction goes as follows:

We proceed as in [NPRRT]. For any \(m \leq n\), let \(A_{\sqrt{m}}^{(1)}\) be the \(s(k,\sqrt{m}) \times m\) matrix where the \(i\)-th column is identical to the \(j\)-th column of \(A_{\sqrt{m}}\) such that the first \(\frac{1}{2} \log m\) bits of \(i\) is \(j\). Similarly, \(M_R\) is the \(s(k,\sqrt{n}) \times m\) where the last \(\frac{1}{2} \log n\) bits of \(i\) is \(j\). Now, we set \(A' = \ldots\)
Let \( A_n \) be a matrix with \( m \) rows, constructed at some point in the recursion procedure. Let \( S \) be any subset of the columns of \( A_m \), with at most \( 4k^2 \) elements. Then \( A_n^{\prime} \) (\( A_m \) restricted on the columns of \( S \)) is \((k, k)\)-disjunct with probability at least \( 1 - 4^{-k \log k + \log \log k} m \).

**Lemma 2.** Let \( A_m \) be a matrix with \( m \) rows, constructed at some point in the recursion procedure. Let \( S \) be any subset of the columns of \( A_m \), with at most \( 4k^2 \) elements. Then \( A_n^{\prime} \) (\( A_m \) restricted on the columns of \( S \)) is \((k, k)\)-disjunct with probability at least \( 1 - 4^{-k \log k + \log \log k} m \).

**Proof.**

Let \( S, T \) be two sets of columns of matrix \( A_m \), such that \( T \cap S = \emptyset \) and \( |S| = |T| = k \). The columns in \( S, T \) correspond to two sets of codewords. For each position \( i \), let \( T_i \) and \( S_i \) denote the set of symbols which the codewords in \( S \) and \( T \) have at that position, respectively. We restrict ourselves to a block of consecutive \( q \) rows, which corresponds to a symbol of the outer code. Then, when restricted to this block of rows, the union of columns in \( S \) is contained in the union of columns in \( T \) with probability \((\frac{k}{2k})^{5k \log k + \log \log k} \leq (\frac{1}{4})^{5k \log k + \log \log k} \). Now, using Lemma 1, we get that \( A_n^{\prime} \) is \((k, k)\)-disjunct with error probability at most \((4k^2 - k) \leq 4^{-5k \log k - \log \log k} \leq 2^{8k \log k - 10k \log k - 2 \log \log k} \leq 4^{-k \log k - \log \log k} \).

**Lemma 3.** Given \( A_n^{\prime} x \), we can find in time \( O(k^3 (\log k + \log \log k) n \log \log k) \) a superset of the defected items with at most \( 2k \) items. The number of rows of \( A \) is \( O(k \log n + \log n \frac{\log \log n}{\log k}) \).

**Proof.**

We assume that \( n = k^2 T \) for some positive integer \( T \). We will drop this assumption later. Observe that the recursion corresponds to a binary tree of height \( T = \log \log k \), because at each time we take the square root of \( k \), till we reach \( k^2 \). At each node of the tree, we run the naive decoding algorithm on the matrix \( A_m \) of that node, restricted on \( 4k^2 \) elements. By Lemma 2, the naive algorithm succeeds with probability \( 4^{-k \log k + \log \log k} \). Since we have \( 2^T = \log \log k \) nodes in the tree, by a union bound, at each node of the tree the naive algorithm will succeed. Hence, at the end of the execution of decoding algorithm, we are going to get a set with the desired guarantees. The decoding time is easy, for at its node of the tree, once we have the result from its children, we spend \( O(k^3 \log k + k^3 \log \log k) \) to check for all \( 4k^2 \) elements if they are defective. By summing over all nodes of the tree, we need \( O(k^3 (\log k + \log \log k) n \log \log k) \) time in total. Since we have \( \log k \) nodes in the tree, we have that the total number of rows of \( A \) is \( O(k \log k + \log \log k) \cdot \log k n = O(k \log n + \log n \frac{\log \log n}{\log k}) \).

To remove the assumption \( n = k^2 T \), take \( i^* \) such that \( k^{2i^*-1} < n \leq k^{2i^*} \). It is easy to check that removing any \( k^{2i^*} - n \) arbitrary columns from \( A_k^{\prime} \), gives a matrix with the desired guarantees, if we treat the removed columns as corresponding to negative items.
Let $G_1 \in \mathbb{R}^{r \times n}, G_2 \in \mathbb{R}^{r' \times n}$ be two gaussian matrices, i.e. each entry is a random variable coming from the normal distribution, where $r'$ is going to be specified later. Our measurement matrix is going to be $\Phi = (A'_n \odot G_1) \cup (-A'_n \odot G_1) \cup G_2$. This matrix has $m = 2r + r'$ rows. The following theorem holds:

**Theorem 2.** Let $r' = \Omega(e^{-5}k)$. Given $y = \text{sign}(\Phi x)$, the decoder can find a $2k$ sparse vector $\hat{x}$ such that $\|\frac{x}{\|x\|_2} - \frac{\hat{x}}{\|\hat{x}\|_2}\|_2 < \epsilon$ with error probability at most $2 \epsilon^5$ in time $O(k^3 (\log k + \log \log k n) \log \log k n) + \text{poly}(r')$.

**Proof.**

Let $T = \text{supp}(x)$ and fix any index $i$ that corresponds to a row of $A'_n$. If $\text{supp}(e_i^T A'_n) \cap T = \emptyset$ then $\text{sign}(\langle e_i^T (A'_n \odot G_1), x \rangle) = 1$ and $\text{sign}(\langle e_i^T (-A'_n \odot G_1), x \rangle) = 1$. If $\text{supp}(e_i^T A'_n) \cap T \neq \emptyset$ we have with probability $1$ that $\langle e_i^T (A'_n \odot G_1), x \rangle \neq 0$. This means that $\text{sign}(\langle e_i^T (A'_n \odot G_1), x \rangle) \cdot \text{sign}(\langle e_i^T (-A'_n \odot G_1), x \rangle) = -1$, so we can recognize that $T$ and the support of that row have a non-empty intersection. This is translated in a combinatorial group testing framework: We know for the given queries if a given query has a common intersection with $T$. By the group testing decoding algorithm we can find a set $S \subseteq [n]$ with at most $2k$ elements such that $T \subseteq S$ in time $t$. Observe now that $G_2^S x_S = G_2 x$. Since we have the sign values of $G_2 x$, this corresponds to a linear system with $2k$ variables and $r'$ constraints. Using any linear programming solver we get in $\text{poly}(r')$ time a solution $\hat{x}$ such that the normalized vector $\frac{\hat{x}}{\|\hat{x}\|_2}$ lies in the same region as $x$, which region has diameter at most $\epsilon$, because of Theorem 1. This completes the proof.

By setting $r$ to be $O(e^{-5}k)$ we get the following corollary:

**Corollary 1.** There exists a matrix $\Phi$ with $O(k \log n + e^{-5}k + \log n \frac{\log \log n}{\log k})$ rows such that given $y = \text{sign}(\Phi x)$, the decoder can find a vector $\hat{x}$ such that $\|\frac{x}{\|x\|_2} - \frac{\hat{x}}{\|\hat{x}\|_2}\|_2 < \epsilon$ with error probability at most $4^{-k \log k} + 4^{-C' \epsilon^{-4}k}$ in time $\text{poly}(k, \log n, \frac{1}{\epsilon})$, for some universal constant $C$.

The factor of $k$ hiding inside $\text{poly}(k, \log n)$ is $k^3$. If one is willing to spend $O(e^{-6}k)$ measurements instead of $O(e^{-5}k)$, then the scheme in [PV13b] can bring the dependence of the decoding time on $k$, down to $k^3$. For a concise review of these running times check the table in [GNJN13]. Thus, we get the following corollary.

**Corollary 2.** There exists a matrix $\Phi$ with $O(k \log n + e^{-6}k + \log n \frac{\log \log n}{\log k})$ rows such that given $y = \text{sign}(\Phi x)$, the decoder can find a vector $\hat{x}$ such that $\|\frac{x}{\|x\|_2} - \frac{\hat{x}}{\|\hat{x}\|_2}\|_2 < \epsilon$ with error probability at most $4^{-k \log k} + 4^{-C' \epsilon^{-4}k}$ in time $O(\epsilon^{-6}k^2 \log n + k^3(\log k + \log \log k n) \log \log k n)$, for some universal constant $C$.

### 2.4 For-all Sparse Recovery from One-Bit Measurements

Our approach is again the same: first we find a superset of the support, and then we run a linear programming algorithm restricted on this set. The crucial idea is again the recovery of this superset from $O(k^2 \log(n/k) \log \log k n)$ measurements and $\text{poly}(k, \log n)$ decoding time. For the construction, let $A'_n$ be any $(k, k)$-disjunct matrix that allows $\text{poly}(k, \log n)$ decoding time. Such a matrix is guaranteed by [GLPS13].
Let $G$ be a $k \times n$ a random gaussian matrix, and consider the matrix $A'' = A'_n \otimes G$. We claim that the following modification of the decoding algorithm recognises a superset of the support of $x \in \Sigma_k$, given $Gx$: Suppose that at some step the the decoding algorithm checks the result of the $j$-th row of $A'_n$ (let it be $r$) dotted with $x$ and checks if it is 0 or 1. The modified decoding algorithm checks if $(r \otimes G)x$ is the zero vector. If this is the case, then it proceeds as the initial decoding algorithm would do if the answer was 0, otherwise it proceeds as if the answer was 1. We will refer to this algorithm as the modified decoding algorithm.

**Lemma 4.** Given val($A'x$) for some $x \in \Sigma_k$, the decoder can find a set $S$, such that supp($x$) $\subseteq S$ and $|S|$ $\leq 2k$.

**Proof.** If we were using $A'_n$ as our sensing matrix and all values of $x$ were 0 or 1, then the decoding algorithm would give the desired result. But now, if we used $A'_n$ as our sensing matrix, there is a chance that we get $Ax_i = 0$ zero, whereas there are at least 2 elements in the intersection of supp($x$) and the $i$-th row of $A$. Taking the tensor product with $G$, we can circumvent this. Let $r_i$ be the $i$-th row of $A$. Let $a_i$ be the support of the $i$-th row of $A$ and let $G'$ be the restriction of $G$ on the columns indexed by any $k$ element subset that contains $a_i \cap$ supp($x$). Every $k \times k$ submatrix of $G$ is invertible since its determinant is non-zero with probability 1. Since we look at $k$-sparse vectors, if all entries of $r_i \otimes G$ were zero, this would mean that $G'$ has a non-trivial kernal, which is a contradiction. Hence, we can correctly perform each that the initial algorithm performs, which implies that the modified decoding algorithm is correct.

**Theorem 3.** Let $\Phi = A'' \mathbin{\biguplus} A'' \mathbin{\bigcup} G_2$, where $G_2$ is a $O(\epsilon^{-5}k \log(n/k)) \times n$ random gaussian matrix. Then for all $k$-sparse $x \in \mathbb{R}^n$, given $y = \Phi x$, one can find $\hat{x} \in \mathbb{R}^n$ such that $\|\frac{x}{\|x\|_2} - \frac{\hat{x}}{\|\hat{x}\|_2}\|_2 \leq \delta$ in poly($k, \log n$) time. The matrix $\Phi$ has $O(k^2 \log(n/k) \log \log k + \epsilon^{-5}k \log(n/k))$ rows.

**Proof.** Similarly, from $A'' \mathbin{\biguplus} A''$ the decoder can find a superset $S$ of the support of $x$ in time poly($k, \log n$). Then, using Theorem 1 he can find the desired $\hat{x}$ in time poly($k$), by restricting to the columns of $G_2$ indexed by $S$.

We also explain how to improve the main algorithm that finds for support identification in [GNJN13]. They authors give an algorithm with $O(nk \log n)$ decoding time and $O(k^3 \log n)$ measurements. Although it is possible to modify their algorithm achieving $O(k^3 \log n \log \log k \log n)$ measurements and poly($k, \log n$) decoding time by applying recursion, we follow another route and get $O(k^3 \log n)$ measurements.

Let $M$ be any $k$-disjunct matrix that has poly($k, \log n$) decoding time and $O(k^2 \log n)$ rows. Such a construction is given in [INR10]. Our construction is again $M' = M \otimes G$, for a $k \times n$ gaussian matrix. The following theorem should be immediate, by the discussion we've done in this section:

**Theorem 4.** For any vector $x \in \Sigma_k$, we can find supp($x$) in poly($k, \log n$) time from $A'x$. The number of rows of $A'$ is $O(k^3 \log n)$.

A conjecture in the Group Testing community is that there exists a $k$-disjunct matrix with $O(\frac{k^2}{\log k} \log n)$ rows. If this conjecture holds, we can also get an algorithm that finds the support in time $O(n \frac{k^2}{\log k} \log(n/k))$, with less measurements than the one in [GNJN13]. Let $N$ any $k$-disjunct matrix with $O(\frac{k^2}{\log k} \log n)$ rows. It is straightforward to check that the following statement holds:
Let \( N' = N \otimes G \), where \( G \) is a \( k \times n \) gaussian matrix. Then, for any vector \( x \in \Sigma_k \), we can find \( \text{supp}(x) \) in \( O(n \frac{k^3}{\log k} \log(n/k)) \) time from \( N'x \). The number of rows of \( N' \) is \( O(k^3 \log(n/k) \log(k)) \).

2.5 For-all Sparse Recovery from Phaseless Measurements

In this section we consider sparse recovery from phaseless measurements, and we design a universal scheme, that recovers \( x \) or \(-x\) from \( y = |\Phi x| \), when \( x \in \Sigma_k \). This result is of the same flavor as one of the results in \cite{JOH13}: their result, however, refers to the for-each model and they do not get sublinear decoding time, mainly because of their support recovery algorithm. For our construction, we will use the same matrix as before to recognise the support. We only present the case that \( x \in \mathbb{R}^n \). The case of complex entries is completely similar, albeit slightly more messy; the interested reader can check the Appendix of \cite{JOH13} to understand how complex entries are handled. Our measurement matrix this time is \( \Phi = A'' \biguplus L \biguplus E \), where \( L \) is a matrix that is used to recognise the magnitude of the entries of the signal, and \( E \) a matrix that is used to recognise their signs. Matrix \( L \) is the vertical concatenation of matrices \( L_1, \ldots, L_{\log n} \); each \( L_i \) has \( \sqrt{k} + 1 \) rows with exactly one 1 per column and all the other entries are zero. Matrix \( E \) consists of \( O(k^2 \log \frac{n}{k}) \) rows and each entry of \( E \) is equal to 1 with probability \( \frac{2}{k} \), otherwise it is zero.

We move with the following lemmas, concerning the matrices \( E \) and \( L \).

**Definition 3.** Let \( C \) be any matrix. Then define a graph \( G(V, E) \) where \( V \) is the set of columns of \( C \) and the edges satisfy the following property: \( (i, j) \in E \) if there exists a row \( r \) such that \( \text{supp}(r) = \{i, j\} \). We refer to the graph \( G \) as the connectivity graph of \( C \).

Let \( \hat{I} \) be the maximum set of disjoint \( \sqrt{k} \) intervals (except, possibly, the last one), that cover \([n]\), namely

\[
I = \{\{\sqrt{k} \ast l + 1, \sqrt{k} \ast l + 2, \ldots, \min\{\sqrt{k} \ast (l + 1)\}, n\}, l \in [0, \left\lceil \frac{n}{\sqrt{k}} \right\rceil]\}
\]

**Lemma 5.** For each interval \( I \in \mathcal{I} \), there exists an index \( j(I) \) such that all elements in \( I \) are perfectly hashed into \( L_{j(S)} \).

**Proof.** With probability at least \( \frac{1}{2} \), a specific interval \( I \) with \( k \) elements is perfectly hashed to \( M_i \). The probability that there exists no index \( j(S) \) such that the claim of the lemma holds is at most \( (\frac{1}{2})^{\log n} = \frac{1}{n} \). Taking a union bound over all intervals we get the desired result.

We give a definition that will help us to analyze and design our algorithm.

**Lemma 6.** For each subset \( S \) of \( k \) elements, there exists a sub-matrix \( E' \) of \( E \) restricted to these \( k \) columns such that the connectivity graph of \( E' \) is the full graph.

**Proof.** Fix a subset \( S \) of \( k \) elements. The expected intersection of the support of a row of the matrix \( E \) with \( S \) is 2 and will be 2 with constant probability. We will call such a row good, otherwise it is bad. We consider the submatrix \( E' \), which consists of the matrix \( E \) restricted on columns of \( S \) and has the rows \( r \) for which \( |\text{supp}(r) \cap S| = 2 \). Let \( G \) be the connectivity graph of this matrix and let \( T \) be an arbitrary spanning tree of it. By a Chernoff bound, with probability at least
1 − O((n \choose k)^{-2})( the exponent can be made to be any arbitrarily big constant) a constant fraction of the rows in E will be good. Conditioned on this fact, the expected number of rows we have to consider until we see all k − 1 edges of T is O(k log k) by the coupon collector problem. Using the concentration results of the same problem, the probability that after considering O(k^2 \log 2^k) we have not seen all edges of G, is at most kO(k^2 \log(2^k) (\log k)^{-1}) = O((n \choose k)^{-2}). Putting everything together, with probability 1 − O((n \choose k)^{-2}) we conclude that the connectivity graph of E’ is connected. Taking a union-bound over all subsets of size k finishes the proof of the lemma.

\[ \text{Theorem 5.} \] For Φ = A'' ∪ L ∪ E it holds that for all x ∈ Σk given y = |Φx| we can find x* such that either x* = x or x* = −x in time poly(k, log n). The number of rows of Φ is O(k^2 \log(n/k) \log \log k n).

Proof.

Using |A''x|, we can find a set S’ that contains supp(x), in time poly(k, log n). Let S be the lexicographically smallest set such that |S| = 2k, S’ ⊆ S. From |Lx| and Lemma 2 we can find the values of x: this happens because there is at least one matrix L_i such that the values of supp(x) are perfectly hashed into it. To find such, we iterate over all matrices L_i and find such a matrix by checking if every column of L_i indexed in the set S has at most 1 non-zero element. Let j(S) be an index such that the aforementioned property holds. Then from |L_{j(S)}x| we get the magnitudes of the entries of x. What remains is to find the relative signs of the entries. We can do this using the matrix E. First, of all we start by a graph G(V, E) with vertex set the set S and set of edges initially zero. We iterate over all rows of E and keep the rows r such that |supp(r) \cap S| = 2; then we add an edge in G between the elements of |supp(r) \cap S|. Lemma 3 guarantees that the graph G will connected after the procedure ends. The only thing we have to do is check for every pair (i, j) ∈ S × S, i ≠ j if |x_i + x_j| = |x_i| + |x_j|. This will indicate if the sign of x_i is different than the sign of x_j. This can be done since we have the magnitudes of x_i, x_j and, moreover, we know |x_i + x_j| by the construction of the graph G: the row that led to the addition of edge between i and j corresponds to the measurement |x_i + x_j|. Since we have the magnitudes of the entries and the relative signs, any x* that agrees with the above gives the desired result.

To complete the proof, observe that the identification of a superset of the support requires poly(k, log n) time and the other two sub routines operate on matrices with O(k^2 \log(n/k)) rows and 2k columns.

Discussion One obvious future direction would be to reduce the exponent of k in the number of measurements of the one-bit compressed sensing and sparse recovery from phaseless measurements in the for-all case. Another interesting direction would be to understand how these algorithms can be made robust to adversarial bits flips (in the case of one-bit sparse recovery) and gaussian noise. It would also be worthwhile to see if these techniques can be applied to speed up the algorithms in [BFN+14], in order to get a very fast decoding time and an exponentially small reconstruction error.

Acknowledgements The author would like to thank Jelani Nelson for numerous useful discussions, as well as Ely Porat for pointing to him [NPR11].
References


