Efficiently Decodable Compressed Sensing by List-Recoverable Codes and Recursion

Hung Q. Ngo[1], Ely Porat[2], and Atri Rudra[1]

1 **Department of Computer Science and Engineering**
University at Buffalo, SUNY,
Buffalo, NY, 14260.
{hungngo,atri}@buffalo.edu

2 **Department of Computer Science,**
Bar-Ilan University,
Ramat Gan 52900, Israel.
porately@cs.biu.ac.il

Abstract

We present two recursive techniques to construct compressed sensing schemes that can be "decoded" in *sub-linear* time. The first technique is based on the well studied code composition method called *code concatenation* where the "outer" code has strong *list recoverability* properties. This technique uses only one level of recursion and critically uses the power of list recovery. The second recursive technique is conceptually similar, and has multiple recursion levels. The following compressed sensing results are obtained using these techniques:

- (*Strongly explicit efficiently decodable ℓ_1/ℓ_1 compressed sensing matrices*) We present a *strongly explicit* ("for all") compressed sensing measurement matrix with $O(d^2 \log^2 n)$ measurements that can output near-optimal d-sparse approximations in time $\text{poly}(d \log n)$.
- (*Near-optimal efficiently decodable ℓ_1/ℓ_1 compressed sensing matrices for non-negative signals*) We present two randomized constructions of ("for all") compressed sensing matrices with near optimal number of measurements: $O(d \log n \log \log_d n)$ and $O_{m,s}(d^{1+1/s} \log n (\log^{(m)} n)^s)$, respectively, for any integer parameters $s, m \geq 1$. Both of these constructions can output near optimal d-sparse approximations for *non-negative* signals in time $\text{poly}(d \log n)$.

To the best of our knowledge, none of the results are dominated by existing results in the literature.

1998 ACM Subject Classification F.2.1 Computations on Matrices, E.4. Coding and Information Theory

Keywords and phrases Compressed Sensing, Sub-Linear Time Decoding, List-Recoverable Codes, Recursion, Strongly Explicit Construction

Digital Object Identifier 10.4230/LIPIcs.xxx.yyy.p

1 Introduction

Compressed sensing [4,6] is a sparse recovery problem that has seen a surge in recent research activity due to a wide variety of practical applications [8]. Compressed sensing (CS) has two components. The combinatorial part is to design a $t \times N$ *measurement matrix* M (where typically $t \ll N$) such that, given the "measurements" Mx of any signal $x \in \mathbb{R}^N$one needs to recover a sparse approximation of x. More precisely the algorithmic task is as follows. Given the measurements $y = Mx + \nu$ which was contaminated with a noise vector ν, compute a vector $\hat{x} \in \mathbb{R}^N$ (ideally $\hat{x}$ is d-sparse or $O(d)$-sparse, i.e. having at most d or

Conference title on which this volume is based on.
Editors: Billy Editor, Bill Editors; pp. 1–16

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

$O(d)$ non-zero entries for some parameter $1 \leq d \ll N$) such that the following conditions holds: $\|x - \hat{x}\|_p \leq C \cdot \|x - x_d^*\|_p + C' \cdot \|\nu\|_p$, where x_d^* is the vector x with all but its d highest-magnitude components zeroed out. In the above $C \geq 1$ is the approximation factor. Ideally, we would like to achieve $C = 1 + \epsilon$ for any $\epsilon > 0$. The noise dependency C' should also be as small as possible. Typically, we consider $p = 1$ in the "for all" case (i.e. the same matrix has to work for every signal) and $p = 2$ in the "for each" case (i.e. a distribution on matrices that works with high probability for each signal). This paper will concentrate on the ℓ_1/ℓ_1 for all problem.

The primary objective in compressed sensing is to minimize the number of measurements t. It is known that $t = \Theta(d \log(N/d))$ measurements are both necessary and sufficient [3]. The second objective is to "decode" (i.e. compute $\hat{x}$ given $y = Mx$) efficiently. It was shown recently that for the ℓ_1/ℓ_1 for all problem, decoding can be done in time $O(N \log N)$ while still attaining the optimal $O(d \log(N/d))$ number of measurements [14].

While near linear time decoding is fairly efficient, it is natural to wonder whether one could decode in *sub-linear* time. In particular, can we achieve decoding with time $\text{poly}(d, \log N)$? Note that when d is small, as is the case in many applications of compressed sensing, then $\text{poly}(d, \log N)$ could be an exponential improvement over $\tilde{O}(N)$. Given the wide applicability of compressed sensing, this is an interesting theoretical question in itself. In particular, compressed sensing is directly related to data streaming [17], where $\text{poly}(d, \log N)$-time decoding is crucial.

For the ℓ_1/ℓ_1 for all problem, sublinear time decoding for compressed sensing has been considered by Gilbert et al. [9]. They achieve a $\text{poly}(t)$ time decoding with a sub-optimal $t = O(d \log^c N)$ number of measurements (where $c \geq 4$ is some absolute constant). Their result has a few more shortcomings: (i) their measurement matrix M is randomized; and (ii) measurement noise was not handled. Indyk and Ruzic [14] overcome these drawbacks but they can only obtain near-linear time decoding. Very recently, Porat and Strauss [20] obtain the optimal number of measurements with sub-linear decoding time. However, the decoding time is always polynomial in N.

Our Results. We have three results for the sub-linear time decodable ℓ_1/ℓ_1 for all problem. The first result is a CS matrix that uses $t = O(d^2 \log^2 N)$ measurements. This is an improvement over [9] only for $d = o(\log^2 N)$. However, our scheme has a couple of advantages: (i) the matrix M is strongly explicit and (ii) it can handle measurement noise. (A matrix is *strongly explicit* if any entry in the matrix can be computed in $\text{poly}(\log N)$ time.) Our construction and decoding schemes are arguably much simpler too: the matrix is the classic d-disjunct matrix of Kautz-Singleton matrix based on Reed-Solomon codes [15].

Our next two results only work for the case when the original signal x is non-negative (the first result works for arbitrary signals). While the constraint is restrictive, it does hold in some applications, such as when x is an image pixel vector, or when we address super-imposed coding problems under multiple-access *adder channels* [1, 7, 16]. The second result is a randomized CS scheme for non-negative signals with $t = O(d \log N \log \log_d N)$ measurements along with $\text{poly}(t)$ decoding time. However, this result cannot handle measurement noise. The third result is a randomized CS scheme for non-negative signals with $t = O(d^{1+1/s} \log N (\log^{(m)} N)^s)$, for any integer parameters $s, m \geq 1$ where we have suppressed some terms that depend only on s and m and $\log^{(m)}(\cdot)$ denotes the $\log(\cdot)$ operator applied m times. Though the number of measurements is worse, this scheme can handle measurement noise and is efficiently decodable.

All of our CS results obtain an approximation ratio of $C = 1 + \epsilon$ for arbitrary $\epsilon > 0$, with the dependence of t on ϵ matching the best known quadratic guarantee of [14].

Our Techniques. The strongest point of our paper is probably its conceptual simplicity.

There is a generic decoding framework that has been used many times before: e.g. in [5, 9, 14] for CS and in [13, 18] for group testing. The decoding has two main steps. The first step, called *filtering*, approximately identifies the locations of the "heavy hitters" in x, whose omission from the output would likely result in a large approximation factor. The second step, called *estimation*, assigns values to these coordinates to obtain $\hat{x}$. The final CS matrix is obtained by vertically stacking the filtering and the estimation matries.

The main insight in sublinear time decodable schemes in this paper (and in [13, 18]) is two-fold. The first observation is that many existing estimation algorithms (e.g., [9, 14, 20]) will work in (near) linear time in $|S|$ if given a set $S \subseteq [N]$ of coordinates which do not miss too much heavy hitter mass. Thus, to get a sublinear time decoding algorithm, it would be sufficient to compute S in the filtering stage with $|S| = \text{poly}(d, \log N)$ in time $\text{poly}(d, \log N)$. The second observation is that, by taking advantage of the fact that $|S|$ can be larger than d, we can design the filtering matrix with about $O(d \log N)$ measurements.

The main technical contribution of this paper is in the filtering step.

Let's start with the $O(d^{1+1/s} \log N (\log^{(m)} N)^s)$ ℓ_1/ℓ_1 for each result. In this case we use only one level of recursion. In particular, we use a $(n, k)_q$ code[1] to "hash" each of the N coordinates of x into nq "buckets" n times. (In particular, if the ith codeword in the jth position has a symbol β, where $i \in [N], j \in [n], \beta \in [q]$, then the jth copy of x_i goes to the (j, β) bucket.) Then we use another filtering scheme on each of the n chunks of q buckets corresponding to each position in the codeword. Three things are in our favor: (i) We can pick $q \ll N$, which means we can use a "wasteful" filtering scheme, such as the identity matrix, on each of n chunks of buckets; (ii) Since x is non-negative it is not too hard to show that a heavy hitter in x is likely "contained" in a heavy hitter of the hashed-down domain of size q; (iii) A simple Markov argument shows that if we pick a large enough number of heavy hitters in the domain of size q, then a lot of non-heavy hitters in x will not suddenly become heavy hitters in the smaller domain due to collisions with other non-heavy hitters in x. This implies that in sub-linear time, we can get for each of the n chunks of buckets, a small list of possible bucket locations that the heavy hitters in x will reside. (A similar construction was used for group testing by the authors in [18].)

In coding terminology, the remaining problem is the following: for every $i \in [n]$, given a small subset $S_i \subseteq [q]$, we want to output all codewords $(c_1, \ldots, c_n)$ such that $c_i \in S_i$ for every $i \in [n]$. Using a simple Markov argument one can get a similar result with measurement noise except we'll need to work with the weaker condition that $c_i \in S_i$ for at least $n/2$ values of $i \in [n]$. It turns out that this problem is *exactly* the problem of list recovery (cf. [10]). The recent work of Parvaresh and Vardy [19] leads to excellent list recoverable codes that can perform the task above algorithmically in time $\text{poly}(n)$ (which in our setting of parameters is $\text{poly}(t)$). However, these codes have too large a q for our purposes.

Fortunately, if we recursively combine several families of Parvaresh-Vardy codes (via a well-known code composition technique called "code concatenation"), then we get codes over acceptably small q that still have acceptable list recoverability. Typically, code concatenation is done with two *different* families of codes while ours does it with the same family.

The $O(d \log N \log \log_d N)$ result follows by a different recursive construction which has multiple recursive levels. (Similar construction was again used for group testing by the authors in [18].) Here is the main idea behind the construction. Let us consider the simple

[1] I.e., we use a code with $N = q^k$ codewords each of which is a vector in $[q]^n$. See Section 2 for more details on coding terminology/definitions.

case where by a simple hashing we map the N coordinates of x into two domains of size $\sqrt{N}$ each. The way we do this is hashing all coordinate indices that agree in the first $\log N/2$ bits into one bucket (and we similarly do this for the last $\log N/2$ bits). Now recursively we obtain sub-linear time decodable filtering schemes that work on domains of size $\sqrt{N}$. In other words, we will get two lists S_1 and S_2 which will contain the first and second $\log N/2$ bits of the indices of the heavy hitters of x respectively. Note that all the indices of the heavy hitters are contained in the set $S_1 \times S_2$. To complete the recursive step, we use a filtering scheme that can exploit the fact that all the heavy hitters are contained in $S_1 \times S_2$. (This is similar to special property of the estimation algorithms mentioned earlier.) For the base case of the recursion, we can use pretty much any filtering scheme (including the identity matrix). For the recursive steps, we use a filtering scheme using the ideas outlined in the previous paragraph but with a random code (which has excellent list recoverability) instead of the Parvaresh-Vardy type codes. As mentioned earlier this scheme cannot handle measurement noise. However, this scheme has another nice property: unlike the other construction, this one allows for a tradeoff between the decoding time and the number of measurement. Other than the claimed result, one can also obtain sublinear-time decoding (though not as efficient as poly(t) decoding time) while being within a constant factor of the optimal number of measurements.

Our result for general signals follows the list recoverability paradigm above but with Reed-Solomon codes instead of Parvaresh-Vardy codes. This leads to worse number of measurements but Reed-Solomon codes have better "distance" properties than Parvaresh-Vardy codes, which allows us to use the *same* matrix for both filtering and estimation. This allows us to have a strongly explicit construction, whereas the $O(d^{1+1/s} \log N(\log^{(m)} N)^s)$ result is randomized as the estimation step is randomized (while the filtering step is explicit). The estimation procedure is also extremely simple: just take the median of the measurements (of a filtered heavy hitter). We do not need the "pursuit" steps as in related prior works.

Even though list recoverable codes have been used to construct good group testing matrices [13], to the best of our knowledge, this is the first work to explicitly use list recoverable codes in compressed sensing. Since sufficiently strong list recoverable codes are known to imply good expanders (cf. [12]), the work of [14] (and related papers) have used list recovery implicitly. However, the direct use of list recovery in our work leads to better parameters.

2 Coding Theory Facts

A code of *dimension* k and *block length* n over an *alphabet* Σ is a subset $C \subseteq \Sigma^n$ of *size* $|\Sigma|^k$. The *rate* of such a code equals k/n. Each vector in C is called a *codeword*. The *distance* of C is the minimum number of positions that any two distinct codewords differ in. A code with dimension k, block length n and distance Δ over Σ will be compactly referred to as an $(n, k, \Delta)_{|\Sigma|}$-code (or simply $(n, k)_{|\Sigma|}$-code if we do not care about its distance). A code C over $\mathbb{F}_q$ is called a *linear code* if C is a linear subspace of $\mathbb{F}_q^n$. A linear code with dimension k, block length n and distance Δ over $\mathbb{F}_q$ will be compactly referred to as an $[n, k, \Delta]_q$- (or simply $[n, k]_q$-) code.

A *concatenated binary code* has an *outer* code $C_{\text{out}} : [q]^{k_1} \to [q]^{n_1}$ over a large alphabet of size $q = 2^{k_2}$, and a binary *inner* code $C_{\text{in}} : \{0,1\}^{k_2} \to \{0,1\}^{n_2}$. The encoding of a message in $(\{0,1\}^{k_2})^{k_1}$ is natural. First, it is encoded with C_{out} and then C_{in} is applied to each of the outer codeword symbols. The concatenated code is denoted by $C_{\text{out}} \circ C_{\text{in}}$.

Let $\ell, L \geq 1$ be integers and let $0 \leq \alpha \leq 1$. A q-ary code C of block length n is called

(α, ℓ, L)-*list recoverable* if for every sequence of subsets $S_1, \ldots, S_m$ such that $|S_i| \leq \ell$ for every $i \in [n]$, there exists at most L codewords $(c_1, \ldots, c_n)$ such that for at least αn positions i, $c_i \in S_i$. A $(1, \ell, L)$-list recoverable code will be henceforth referred to as (ℓ, L)-*zero error list recoverable*.

We will need the following powerful result due to Parvaresh and Vardy[2]:

▶ **Theorem 2.1** ([19]). *For all integers $s \geq 1$, for all prime powers r and all powers q of r, every pair of integers $1 < k \leq n \leq q$, there is an explicit $\mathbb{F}_r$-linear map $E : \mathbb{F}_q^k \to \mathbb{F}_{q^s}^n$ such that:*

1. *The image of E, $C \subseteq \mathbb{F}_{q^s}^n$, is a code of minimum distance at least $n - k + 1$.*
2. *Provided*

$$\alpha > (s+1)(k/n)^{s/(s+1)} \ell^{1/(s+1)}, \tag{1}$$

C is an $(\alpha, \ell, O((rs)^s n\ell/k))$-list recoverable code. Further, a list recovery algorithm exists that runs in $\mathrm{poly}((rs)^s, q, \ell)$ time.

▶ **Corollary 2.2.** *Let $0 < \alpha \leq 1$ be a real and $s, \ell \geq 1$ be integers. Then for any prime power q, there is a strongly explicit $\mathbb{F}_p$ linear $\left(q, k \stackrel{def}{=} \frac{q}{s} \cdot \left(\frac{\alpha}{2s}\right)^{1+1/s} \cdot \frac{1}{\sqrt[s]{\ell}}\right)_{q^s}$ that is $\left(\alpha, \ell, s^{O(s)} q\ell/k\right)$-list recoverable in time $\mathrm{poly}(ps^s, q, \ell)$.*

In the above, the s'th "order" Parvaresh-Vardy code will be referred to as the PV^s code. PV^1 is the well-known *Reed-Solomon codes* (RS code for short). For RS codes, the multiplicative factor of $(s+1)$ can be removed from (1). Actually for the RS codes, we will use the following result which has a slightly weaker list recoverability but has a much faster running time:

▶ **Theorem 2.3** ([2]). *An $[n, k]_q$ Reed-Solomon code is an $(\alpha, \ell, \sqrt{2n\ell/k})$-list recoverable provided $\alpha > \sqrt{\frac{2k\ell}{n}}$. Further, there is a $O(n^2\ell^2 \log(n\ell)\mathrm{poly}(\log q))$-time list recovery algorithm.*

The next two results are folklore. (For the sake of completeness we provide the proofs in the Appendix.)

▶ **Theorem 2.4.** *Let $0 < \alpha \leq 1$ be a real and let $\ell \geq 1$ be an integer. Then for integers $q \geq \frac{e^6 \ell}{\alpha^2}$ and large enough n, the following holds. A random[3] $\left(n, k = \frac{\alpha n}{2 \log q}\right)_q$ code is $\left(\alpha, \ell, L \stackrel{def}{=} \left\lceil \frac{2\ell}{\alpha} \right\rceil\right)$-list recoverable with probability at least $1 - 2^{-\Omega(\alpha n L)}$.*

▶ **Lemma 2.5.** *Let C_{out} be an $(n_1, k_1)_{Q \stackrel{def}{=} q^k}$ code that is $(\alpha_1, \ell, f_1(\ell))$-list recoverable and C_{in} be an $(n_2, k_2)_q$ code that is $(\alpha_2, \ell, f_2(\ell))$-list recoverable. Then $C_{\mathrm{out}} \circ C_{\mathrm{in}}$ is an $(n_1 n_2, k_1 k_2)_q$ code that is $(1 - (1-\alpha_1)(1-\alpha_2), \ell, f_1(f_2(\ell)))$-list recoverable. Further if list recovery for C_{out} and C_{in} can be performed in time $g_1(n_1, Q)$ and $g_2(n_2, q)$ time respectively, then $C_{\mathrm{out}} \circ C_{\mathrm{in}}$ can be list recovered in time $g_1(n_1, Q) + O(n_1 \cdot g_2(n_2, q))$.*

Applying Lemma 2.5 to the PV codes in Corollary 2.2 recursively m times leads to:

▶ **Corollary 2.6.** *Let $0 < \alpha \leq 1$ be a real and $s, m, \ell \geq 1$ be integers. Then the following holds for large enough n: there exists a $\left(n, \frac{n}{R}\right)_{q^s}$-code that is $(1 - (1-\alpha)^m, \ell, s^{O(ms)}\ell/R)$-list recoverable where $R = \frac{1}{s} \cdot \left(\frac{\alpha}{2s}\right)^{1+1/s} \cdot \frac{1}{\sqrt[s]{\ell}}$, and $\frac{1}{R} \leq q \leq \frac{1}{R} \cdot \left(\log^{(m)} n + m\right)$.*

---

[2] This statement of the theorem appears in [11].

[3] A codeword in a random $(n, k)_q$ code is chosen by assigning each of the n symbols independently and uniformly at random from $[q]$.

3 Constructions based on one-level code concatenation

We first fix some notations. For any $x \in \mathbb{R}^m$, $S \subseteq [m]$, let x_S denote the restriction of x on to S. For any positive integer $d \leq m$, let $H_d(x)$ denote the set of d largest (or "heaviest") coordinates of x in magnitudes, breaking ties arbitrarily. Note that, when $|S| \geq d$, we have $H_d(x_S) \subseteq S$.

3.1 General signals and Reed-Solomon-based compressed sensing

Fix a $[n, k, n-k+1]_q$-RS code $C = \{c_1, \cdots, c_{q^k}\} \subset [q]^n$, and the identity code $\mathrm{ID}_q : [q] \to \{0,1\}^q$. ($\mathrm{ID}_q(i)$ is the ith standard basis vector.) Let $M = M_{\mathrm{RS}\circ\mathrm{ID}}$ denote the matrix of the concatenated code $\mathrm{RS} \circ \mathrm{ID}$ defined as follows. The matrix M is a binary matrix with $N = q^k$ columns and $t = qn$ rows. We index each row of M by a pair $(p, i) \in [n] \times [q]$. Let M_j denote the jth column of M, then $M_j(p, i) = 1$ iff $c_j(p) = i$. Note that M is n-sparse. We use M to compress signals in $\mathbb{R}^N$. Algorithm 1 is used for decoding. We will choose parameters n, k, q so that the RS code is $(\alpha = 1/2, l = cd, L = \sqrt{2nl/k})$-list recoverable. In particular, the parameters $q \geq n > k$ have to satisfy $1/2 > \sqrt{2kl/n}$, or $n > 8cdk$. Furthermore, we shall also choose $c > 2$.

Algorithm 1 Decoding algorithm for $M = M_{\mathrm{RS}\circ\mathrm{ID}}$

Input: $y \leftarrow Mx + \nu$, where ν is the noise vector
Input: $c > 2$ is a parameter, d is the desired output signal sparsity
// note again that $n > 8cdk$
for each position $p \in [n]$ **do**
    Let $S_p \subset [q]$ denote the set of cd indices i such that the corresponding measurements $y(p, i)$ are the cd heaviest-magnitude measurements in the set $\{y(p, i) \mid i \in [q]\}$.
    // break ties arbitrarily
end for
Use the list-recovery algorithm (with the inputs $S_p, p \in [n]$) for the RS code to recover a set $H \subset [N]$ of $\leq L$ indices.
for each $j \in H$ **do**
    Let $\hat{x}_j = \mathrm{median}\{y(p, i) \mid c_j(p) = i\}$.
end for
Return the top d (in magnitude) $\hat{x}_j, j \in H$.

Let D be the d indices of largest-magnitude components of the input signal x. Briefly, the analysis has two basic steps. First, we show that H retains all of the sufficiently heavy coordinates in D. Second, we show that the median estimates are almost correct. Before realizing the two steps, we need a simple auxiliary lemma.

▶ **Lemma 3.1.** *Let $\delta = k/n$. Consider an arbitrary index $j \in [N]$. There is a subset $P \subset [n]$ of positions satisfying the following: (a) $|P| \geq 7(1-\delta d)n/8 \geq 3n/4$, and (b) for every $p \in P$, we have $|y(p, c_j(p)) - x_j| \leq \frac{10}{n}\|\nu\|_1 + \frac{8}{cd}\|x - x_D\|_1$.*

Proof. The RS code has relative distance $> 1-\delta$. Hence, every two codewords have at most δn positions with the same symbols. In particular, there is a set P' of at least $n - d\delta n = (1-\delta d)n$ positions satisfying the following: for every $p \in P'$, $c_j(p) \neq c_{j'}(p), \forall j' \in D \setminus \{j\}$.

Next, because for every $j' \in [N] - (D \cup \{j\})$ the codeword $c_{j'}$ shares at most δn positions with c_j, the triangle inequality implies $\sum_{p \in P} |y(p, c_j(p)) - x_j| \leq \|\nu\|_1 + \delta n \|x - x_D\|_1$. Since

$|P'| \geq (1-\delta d)n$, by Markov inequality there is a subset $P \subset P'$ of at least $|P| \geq \frac{7}{8}(1-\delta d)n$ positions such that, for every $p \in P$

$$|y(p, c_j(p)) - x_j| \leq \frac{8(\|\nu\|_1 + \delta n\|x - x_D\|_1)}{(1-\delta d)n} \leq \frac{10}{n}\|\nu\|_1 + \frac{8}{cd}\|x - x_D\|_1.$$

The inequality $7(1-\delta d)n/8 \geq 3n/4$ follows from our assumptions that $n > 4ckd$ and $c > 2$. ◀

▶ **Lemma 3.2.** *Consider an arbitrary $j \in D$ such that $|x_j| > \frac{10\|x-x_D\|_1}{cd} + \frac{18\|\nu\|_1}{n}$. Then, $c_j(p) \in S_p$ for at least $\alpha n = n/2$ different positions $p \in [n]$.*

Proof. Let $\delta = k/n$, and let P be the set of positions satisfying the properties stated in Lemma 3.1. Then, for each $p \in P$,

$$\begin{aligned} |y(p, c_j(p))| &\geq |x_j| - |y(p, c_j(p)) - x_j| > \frac{10\|x - x_D\|_1}{cd} + \frac{18\|\nu\|_1}{n} - \frac{10}{n}\|\nu\|_1 - \frac{8}{cd}\|x - x_D\|_1 \\ &\geq \frac{8}{cdn}\|\nu\|_1 + \frac{2}{cd}\|x - x_D\|_1. \end{aligned} \tag{2}$$

Next, for every position $p \in [n]$, let $\nu(p) = [\nu(p,1), \cdots, \nu(p,q)]$ denote the restriction of the noise vector ν on to the q coordinates within position p. Since $\|\nu\|_1 = \sum_{p=1}^{n} \|\nu(p)\|_1$, by Markov inequality there must be a set P' of at least $3n/4$ positions such that $\|\nu(p)\|_1 \leq 4\|\nu\|_1/n$ for every $p \in P'$. Let $\bar{P} = P \cap P'$. Then, $|\bar{P}| \geq n/2 = \alpha n$. We will prove that $c_j(p) \in S_p$ for every $p \in \bar{P}$, which would then complete the proof of the lemma.

Fix a position $p \in \bar{P}$. Let $y(p) = [y(p,1), \cdots, y(p,q)]$ be the subvector of y restricted to position p. For each $i \in [q]$, define $D_i = \{j \in D \mid c_j(p) = i\}$. By the triangle inequality and the fact that $p \in P'$, $\sum_{i \in S_p} |y(p,i) - \sum_{j \in D_i} x_j| \leq \|\nu(p)\|_1 + \|x - x_D\|_1 \leq \frac{4}{n}\|\nu\|_1 + \|x - x_D\|_1$. Noting that $|S_p| = cd$, by Markov inequality again there is a subset $S' \subset S_p$ of size $cd/2$ satisfying

$$|y(p,i) - \sum_{j \in D_i} x_j| \leq \frac{8}{cdn}\|\nu\|_1 + \frac{2}{cd}\|x - x_D\|_1 \qquad \text{for every } i \in S'.$$

Because $\sum_i |D_i| \leq d$ and $cd/2 > d$, there must be at least one $i' \in S'$ for which $D_{i'} = \emptyset$. Hence,

$$\min_{i \in S_p} |y(p,i)| \leq |y(p,i')| = |y(p,i') - \sum_{j \in D_i} x_j| \leq \frac{8}{cdn}\|\nu\|_1 + \frac{2}{cd}\|x - x_D\|_1. \tag{3}$$

From (2) and (3), and the fact that S_p contains the d largest coordinates in magnitudes of $y(p)$, we conclude that $i \in S_p$ as desired. ◀

We next prove that all the median estimates are pretty good.

▶ **Lemma 3.3.** *For any $j \in [N]$, $|x_j - \hat{x}_j| \leq \frac{10}{n}\|\nu\|_1 + \frac{8}{cd}\|x - x_D\|_1$.*

Proof. Let $\delta = k/n$, and let P be the set of positions satisfying the properties stated in Lemma 3.1. This means for $|P| \geq 3n/4 > n/2$ of the positions we know the values $y(p, c_j(p))$ are within $\pm\left(\frac{10}{n}\|\nu\|_1 + \frac{8}{cd}\|x - x_D\|_1\right)$ of x_j. Thus, so is the median $\hat{x}_j$ of the $y(p, c_j(p))$. ◀

We are now ready to prove the main result.

▶ **Theorem 3.4.** *With parameter $c = 18/\epsilon$, Algorithm 1 runs in time $\mathrm{poly}(d \log n)$ and outputs a d-sparse vector $\hat{x}$ satisfying $\|x - \hat{x}\|_1 \leq (1+\epsilon)\|x - x_D\|_1 + (28d/n)\|\nu\|_1$, where D is the set of k highest-magnitude coordinates of x.*

Proof. The total "extra mass" we get, relative to the best $\|x - x_D\|_1$, comes from the estimation error mass and the total mass of the small magnitude coordinates in D. It is not hard to see that

$$\begin{aligned}\|x - \hat{x}\|_1 &\leq \|x - x_D\|_1 + d\left(\frac{18\|\nu\|_1}{n} + \frac{10\|x - x_D\|_1}{cd}\right) + d\left(\frac{10}{n}\|\nu\|_1 + \frac{8}{cd}\|x - x_D\|_1\right) \\ &= (1 + 18/c)\|x - x_D\|_1 + (28d/n)\|\nu\|_1.\end{aligned}$$

◀

We next choose the parameters to minimize the number of measurements t of the matrix M. The following is not best possible (we can reduce it by a $\log\log$ factor). Choose $n = q$, $k = n/(4cd) = \epsilon n/(72d)$. For $N \leq q^k$, we need $\log N \leq \frac{\epsilon}{72d} q \log q$. Thus, we can pick $n = q = O\left(\frac{d}{\epsilon}\log_d N\right)$. The total number of measurements is $t = nq = O(\frac{d^2}{\epsilon^2}(\log_d N)^2)$.

Finally, we analyze the run time of the algorithm. The following steps dominate the running time of the other steps: (i) The first **for** loop, which takes $O(nq\log q) = O(\frac{d^2}{\epsilon^2}\text{poly}(\log_d N))$; (ii) Step 8 (the list recovery step), which by Theorem 2.3 takes $O(l^2n^2\log nl\text{poly}(\log q)) = d^2/\epsilon^2\text{poly}(\log N)$ time and (iii) the second **for** loop which takes $O(Lq) = O(d^2/\epsilon\log_d N)$ time. Thus, the overall running time is $d^2/\epsilon^2\text{poly}(\log N)$, as desired. This completes the proof of Theorem 3.5.

▶ **Theorem 3.5.** *For every $\epsilon > 0$, $M = M_{RS\circ\text{ID}}$ with $c = O(1/\epsilon)$ is a compressed sensing measurement matrix with $t = O(d^2(\log_d N)^2/\epsilon^2)$ measurements which admits a $(d/\epsilon)^2\text{poly}(\log N)$-time decoding algorithm that, given a noisy measurement vector $y = Mx+\nu$, outputs (the non-zero coordinates and values of) a d-sparse approximation $\hat{x} \in \mathbb{R}^N$ with the following approximation guarantee: $\|x - \hat{x}\|_1 \leq (1+\epsilon)\|x - x_{H_d(x)}\|_1 + \frac{0.4\epsilon}{\log N}\|\nu\|_1$.*

3.2 Non-negative signals

Estimation Algorithm. We will use the following estimation result by Porat and Strauss [20].. We could have also used the estimation procedures given in Indyk-Ruzic [14] or the DFT-based matrix in [9] which will give different approximation guarantees. Our contribution is the filtering matrix.

▶ **Theorem 3.6** ([20]). *Let $N \geq d \geq 1$ be integers and $\epsilon > 0$ be a real. Then there exists a random $t \times N$ matrix M with the following properties:*

(i) $t = O\left(\frac{d}{\epsilon^2}\cdot\log(N/d)\right)$*; and*

(ii) Let $S \subseteq [N]$, $x \in \mathbb{R}^N$ and $\nu \in \mathbb{R}^t$. Then there exists a $|S|\cdot(\log(N/d)/\epsilon)^{O(1)}$ time algorithm that given a noisy measurement $Mx + \nu$, outputs a vector $x' \in \mathbb{R}^N$ with at most $O(d)$ non-zero entries such that

$$\|x' - x_{H_d(x)}\|_1 \leq (1+\epsilon)\cdot(\|x - x_{H_d(x)}\|_1 + \|x_{H_d(x)\setminus S}\|_1) + \frac{c\cdot\epsilon}{\log(N/d)}\cdot\|\nu\|_1,$$

where $c \geq 1$ is some absolute constant.

From list recovery to compressed sensing

▶ **Theorem 3.7.** *Let $d \geq 1$ be an integer and $c \geq 1$ be a real. Let $\ell, L \geq 1$ be integers. Then the following holds for any $q \geq \ell$, where q is a power of 2. Let C_{out} be an $(n_1, k_1)_q$-code that is $(1/2, \ell, L)$-list recoverable. Let C_{in} be an $(n_2, k_2 \stackrel{def}{=} \log q)_2$ code with the following property. For any vector $z \in \mathbb{R}^q_{\geq 0}$ and measurement noise $\mu \in \mathbb{R}^{n_2}$, given the measurement*

outcome $M_{C_{\text{in}}}z + \mu$, there is a $T_{in}(n_2, q)$-time algorithm that outputs at most ℓ coordinates of z containing the set

$$\left\{i \in [q] \mid z_i \geq \gamma \cdot \|z - z_{H_d(z)}\|_1 + \delta \cdot \|\mu\|_1\right\},$$

where $\gamma, \delta > 0$. Then the matrix $M \stackrel{def}{=} M_{C_{\text{out}} \circ C_{\text{in}}}$ has the following properties:

(i) M is $t \times N$ matrix, where $t = n_1 n_2$ and $N = q^{k_1}$.

(ii) For any $x \in \mathbb{R}^N_{\geq 0}$ and $\nu \in \mathbb{R}^t$, consider the noisy measurement vector $y = Mx + \nu \in \mathbb{R}^t_{\geq 0}$. There exists a set $H \subseteq [N]$ with $|H| \leq L$ such that

$$\left\{i \in [N] \mid x_i \geq \gamma \cdot \|x_T\|_1 + 2\delta \cdot \frac{\|\nu\|_1}{n_1}\right\} \subseteq H, \tag{4}$$

where $T = [N] \setminus H_d(x)$.

(iii) If C_{out} can be list recovered in time $T_{out}(n_1, q)$, then given y, the set H from part (ii) can be computed in time $n_1 \cdot T_{in}(n_2, q) + T_{out}(n_1, q)$.

Proof. Property (i) follows from the properties of concatenated codes. In the rest of the proof, we will prove (ii) and (iii) together by outlining an algorithm to compute the set H.

For notational convenience, let the codewords in C_{out} be denoted by $\{c_1, \ldots, c_N\}$. We will associate the ith coordinate of x with c_i.For any $j \in [n_1]$, let $\mu_j \in \mathbb{R}^{n_2}$ be the projection of ν to the positio ns in $[t]$ corresponding to the outer codeword position j. Note that $\|\nu\|_= \sum_{j \in [n_1]} \|\mu_j\|_1$. Thus by a Markov argument, there exists a subset $U \subseteq [n_1]$ with $|U| \geq n_1/2$ such that for every $j \in U$,

$$\|\mu_j\|_1 \leq \frac{2\|\nu\|_1}{n_1}. \tag{5}$$

Fix an outer codeword position $j \in U$. For any $i \in [N]$, let $c_i(j) \in [q]$ denote the jth symbol in the codeword c_i. Now define the vector $z = (z_1, \ldots, z_q)$ such that for any $\beta \in [q]$, $z_\beta = \sum_{i \in [N]: c_i(j) = \beta} x_i$. By the assumption in the Theorem statement and (5), we know that in time $T_{in}(n_2, q)$ we can compute a set S_j of size at most ℓ such that

$$\left\{\beta \in [q] \mid z_\beta \geq \gamma \cdot \|z - z_{H_d(z)}\|_1 + 2\delta \cdot \frac{\|\nu\|_1}{n_1}\right\} \subseteq S_j.$$

Before we proceed we claim that

$$\|x_T\|_1 \geq \|z - z_{H_d(z)}\|_1. \tag{6}$$

Indeed, the above follows from the subsequent argument. Let $H' = \{c_i(j) | i \in H_d(x)\}$. Now note that $\|x_T\|_1 \geq \|z - z_{H'}\|_1 \geq \|z - z_{H_d(z)}\|_1$, where the first inequality follows from the definitions of z and H' while the second inequality follows from the definition of $H_d(z)$ and the fact that $|H'| \leq d$.

Thus, (6) (and the fact that x is a non-negative signal) implies that for every $i \in [N]$ such that $x_i \geq \gamma \cdot \|x_T\|_1 + 2\delta\|\nu\|_1/n_1$, $z_{c_i(j)} \geq x_i \geq \gamma \cdot \|z - z_{H_d(z)}\|_1 + 2\delta \cdot \|\nu\|_1/n_1$. In other words, $c_i(j) \in S_j$. As the choice of j was arbitrary, it holds that for every $i \in [N]$ such that $x_i \geq \gamma \cdot \|x_T\|_1 + 2\delta\|\nu\|_1/n_1$, $c_i(j) \in S_j$ for every $j \in U$. Recall that since $|S_j| \leq \ell$ for every $j \in U$, $|U| \geq n_1/2$ and as C_{out} is $(1/2, \ell, L)$-list recoverable in time $T_{out}(n_1, q)$ one can compute a set $H \subseteq [N]$ of size at most L such that every $i \in [N]$ such that $x_i \geq \gamma \cdot \|x_T\|_1 + 2\delta \cdot \|\nu\|_1/n_1$ satisfies $i \in H$. This completes the proof of (ii). Further, (iii) just follows from the description of the algorithm above to compute H. ◀

Specific Instantiations. The "identity" code $\mathrm{ID}(q) : [q] \to \{0,1\}^q$ is often used as an inner code. Here, $\mathrm{ID}(i)$ for any $i \in [q]$ is the q-bit vector that is set to 1 in position i and is zero otherwise. (The omitted proofs are in the Appendix.)

▶ **Lemma 3.8.** *Let $d \geq 1$ be an integer and $c \geq 1$ be a real. Let $q \geq 2(c+1)d$ be an integer. Then for any vector $x \in \mathbb{R}^q_{\geq 0}$ and measurement noise $\mu \in \mathbb{R}^q$, given the outcome $M_{\mathrm{ID}(q)}x + \mu$, there is an $O(q \log(cd))$ time algorithm that outputs $\ell \stackrel{def}{=} 2(c+1)d$ coordinates of x such that it contains the set*

$$T \stackrel{def}{=} \left\{ i \in [q] \mid x_i \geq \frac{1}{cd} \cdot \|x - x_{H_d(x)}\|_1 + \frac{2}{(c+1)d} \cdot \|\mu\|_1 \right\}.$$

Random CS construction. Applying Theorem 3.7 with the outer code from Theorem 2.4 as C_{out} (with q being a power of 2) and the code from Lemma 3.8 as C_{in} implies the following result:

▶ **Corollary 3.9.** *Let $N \geq d \geq 1$ be integers and $\epsilon > 0$ be a real. Then there exists a random $t \times N$ matrix M with the following properties:*

(i) $t = O\left(\frac{d}{\epsilon} \cdot \log N\right)$; *and*

(ii) *Let $S \subseteq [N]$, $x \in \mathbb{R}^N_{\geq 0}$ and $\nu \in \mathbb{R}^t$. Then there exists a $\tilde{O}(|S| \cdot t)$ time algorithm that given a noisy measurement $Mx + \nu$, outputs a subset $H \subseteq [N]$ with $|H| \leq O(d/\epsilon)$ such that*

$$S \cap \left\{ i \in [N] \mid x_i \geq \frac{\epsilon}{d} \cdot \|x - x_{H_d(x)}\|_1 + \frac{\epsilon}{d \cdot \log N} \cdot \|\nu\|_1 \right\} \subseteq H.$$

Explicit CS construction. Applying Theorem 3.7 with the outer code from Corollary 2.6 (with q being a power of 2) and the inner code from Lemma 3.8 implies the following result:

▶ **Corollary 3.10.** *Let $n \geq d \geq 1$ and $s, m \geq 1$ be integers and $\epsilon > 0$ be reals. Then there exists an explicit $t \times N$ matrix M with the following properties:*

- $t \leq (sm)^{O(s)} \cdot \left(\frac{d}{\epsilon}\right)^{1+1/s} \cdot (\log^{(m)} N + m)^s$
- *Let $x \in \mathbb{R}^N_{\geq 0}$ and $\nu \in \mathbb{R}^t$. Then there exists a $\mathrm{poly}(t)$ time algorithm that given a noisy measurement $Mx + \nu$, outputs a subset $H \subseteq [N]$ with $|H| \leq$ such that*

$$\left\{ i \in [N] | x_i \geq \frac{\epsilon}{d} \cdot \|x - x_{H_d(x)}\|_1 + \frac{\epsilon \cdot \log^{(m+1)} N}{d \cdot \log N} \cdot \|\nu\|_1 \right\}.$$

The above (instantiated with $C_{\mathrm{out}} = \mathrm{PV}^s$ and $C_{\mathrm{in}} = \mathrm{ID}$) with Theorem 3.6 implies

▶ **Theorem 3.11.** *For every $\epsilon > 0$, and $s, m \geq 1$ there is a strongly explicit compressed sensing scheme with $t = (sm)^{O(s)} \cdot \left(\frac{d}{\epsilon}\right)^{1+1/s} \cdot (\log^{(m)} N + m)^s$ measurements and $\mathrm{poly}(t)$ decoding time that on input a signal $x \in \mathbb{R}^N_{\geq 0}$ and measurement vector $\nu \in \mathbb{R}^t$ outputs a vector $\hat{x} \in \mathbb{R}^N$ that is d-sparse with the following approximation guarantee:*

$$\|x - \hat{x}\|_1 \leq (1+\epsilon)\|x - x_{H_d(x)}\|_1 + \frac{O(\epsilon) \cdot \log^{(m+1)} N}{\log N} \cdot \|\nu\|_1.$$

4 Recursive construction with multiple recursion levels

Using our second recursive construction technique, we obtain the following results:

▶ **Theorem 4.1.** *Let $n \geq d \geq 1$ be integers. Assume for every $i \geq d$, there is a $t(i) \times i$ matrix $\mathbf{M}_i$ with the following property. For any subset $S \subseteq [i]$, vector $z \in \mathbb{R}^i_{\geq 0}$ and given the outcome vector $\mathbf{M}_i z$, there is a $d(|S|, i)$-time algorithm that outputs at most ℓ coordinates of z containing the set*

$$\{i \in S \mid z_i \geq \gamma \cdot \|z - z_{H_d(z)}\|_1\}, \tag{7}$$

where $\gamma > 0$. Let $1 \leq a \leq \log n$ and $1 \leq b \leq \log n/a$ be integers. Then there exists a $t_{a,b} \times n$ matrix $\mathbf{M}_{a,b}$ that has the following property. Given any $x \in \mathbb{R}^n_{\geq 0}$, from the measurement vector $\mathbf{M}_{a,b}x$, in time $D_{a,b}$, one can compute a set H with $|H| \leq \ell$ such that

$$\{i \in [n] \mid x_i \geq \gamma \cdot \|x - x_{H_d(x)}\|_1\}, \tag{8}$$

where

$$t_{a,b} = \sum_{j=0}^{\lceil \log_b(\frac{\log n}{a}) \rceil - 1} b^j \cdot t\left(\sqrt[b^j]{n}\right) \tag{9}$$

and

$$D_{a,b} = \sum_{j=0}^{\lceil \log_b(\frac{\log n}{a}) \rceil - 2} b^j \cdot d\left(\ell^b, \sqrt[b^j]{n}\right) + \frac{\log n}{a} \cdot d(2^a, 2^a). \tag{10}$$

Finally, if the family of matrices $\{\mathbf{M}_i\}_{i \geq d}$ is (strongly) explicit then so is $\mathbf{M}_{a,b}$.

Near optimal sublinear random CS scheme. Applying Theorem 4.1 to Corollary 3.9, we obtain

▶ **Corollary 4.2.** *Let $N \geq d \geq 1$ be integers and $\epsilon > 0$ be a real. Then there exists a random $t \times N$ matrix M with the following properties:*

(i) $t = O\left(\frac{d}{\epsilon} \cdot \log N \log \log_d N\right)$*; and*

(ii) Let $x \in \mathbb{R}^N_{\geq 0}$. Then there exists a $\mathrm{poly}(t)$ *time algorithm that given a measurement Mx, outputs a subset $H \subseteq [N]$ with $|H| \leq O(d/\epsilon)$ such that*

$$\left\{i \in [N] | x_i \geq \frac{\epsilon}{d} \cdot \|x - x_{H_d(x)}\|_1\right\}.$$

The above (with the $\mathbf{M}_i$ family following from the construction in Theorem 3.7 instantiated with a random code as C_{out} and the identity code as the inner code) with Theorem 3.6 implies Theorem 4.3.

▶ **Theorem 4.3.** *For every $\epsilon > 0$, there is a randomized compressed sensing scheme with $t = O(d/\epsilon^2 \log N \log \log_d N)$ measurements and* $\mathrm{poly}(t)$ *decoding time that on input a signal $x \in \mathbb{R}^N_{\geq 0}$ outputs a vector $\hat{x} \in \mathbb{R}^N$ that is d-sparse with the following approximation guarantee:*

$$\|x - \hat{x}\|_1 \leq (1 + \epsilon)\|x - x_{H_d(x)}\|_1.$$

References

1 Rudolf Ahlswede and Vladimir B. Balakirsky. Construction of uniquely decodable codes for the two-user binary adder channel. *IEEE Trans. Inform. Theory*, 45(1):326–330, 1999.

2 Daniel Augot and Lancelot Pecquet. A hensel lifting to replace factorization in list-decoding of algebraic-geometric and reed-solomon codes. *IEEE Transactions on Information Theory*, 46(7):2605–2614, 2000.

3 Khanh Do Ba, Piotr Indyk, Eric Price, and David P. Woodruff. Lower bounds for sparse recovery. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1190–1197, 2010.

4 Emmanuel J. Candès, Justin K. Romberg, and Terence Tao. Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52(2):489–509, 2006.

5 Graham Cormode and S. Muthukrishnan. Combinatorial algorithms for compressed sensing. In *13th International Colloquium on Structural Information and Communication Complexity (SIROCCO)*, pages 280–294, 2006.

6 David L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, 2006.

7 A. G. D′yachkov and V. V. Rykov. A coding model for a multiple-access adder channel.

8 Anna C. Gilbert and Piotr Indyk. Sparse recovery using sparse matrices. *Proceedings of the IEEE*, 98(6):937–947, 2010.

9 Anna C. Gilbert, Martin J. Strauss, Joel A. Tropp, and Roman Vershynin. One sketch for all: fast algorithms for compressed sensing. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing (STOC)*, pages 237–246, 2007.

10 Venkatesan Guruswami. *List decoding of error-correcting codes.* Number 3282 in Lecture Notes in Computer Science. Springer, 2004.

11 Venkatesan Guruswami and Atri Rudra. Soft decoding, dual BCH codes, and better list-decodable ϵ-biased codes. In *Proceedings of the 23rd Annual IEEE Conference on Computational Complexity (CCC)*, pages 163–174, 2008.

12 Venkatesan Guruswami, Christopher Umans, and Salil P. Vadhan. Unbalanced expanders and randomness extractors from parvaresh-vardy codes. In *Proceedings of the 22nd Annual IEEE Conference on Computational Complexity*, pages 96–108, 2007.

13 Piotr Indyk, Hung Q. Ngo, and Atri Rudra. Efficiently decodable non-adaptive group testing. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1126–1142, 2010.

14 Piotr Indyk and Milan Ruzic. Near-optimal sparse recovery in the l1 norm. In *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 199–207, 2008.

15 W. H. Kautz and R. C. Singleton. Nonrandom binary superimposed codes. *IEEE Trans. Inf. Theory*, 10:363–377, 1964.

16 Lasse Kiviluoto and Patric R. J. Östergård. New uniquely decodable codes for the T-user binary adder channel with $3 \leq T \leq 5$. *IEEE Trans. Inform. Theory*, 53(3):1219–1220, 2007.

17 S. Muthumrishnan. *Data Streams: Algorithms and Applications*, volume 1. NOW, 2005.

18 Hung Q. Ngo, Ely Porat, and Atri Rudra. Efficiently decodable error-correcting list disjunct matrices and applications. In *ICALP (1)*, pages 557–568, 2011.

19 Farzad Parvaresh and Alexander Vardy. Correcting errors beyond the Guruswami-Sudan radius in polynomial time. In *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 285–294, 2005.

20 Ely Porat and Martin Strauss. Sublinear time, measurement-optimal, sparse recovery for all, July 2012. To be presented at SODA 2012. Also arXiv:1012.1886 [cs.DS].

A Missing Proofs from Section 2

A.1 Proof of Theorem 2.4

Proof. The proof follows from a simple application of the probabilistic method. Fix any input $S_1, \ldots, S_n \subseteq [q]$ with $|S_i| \leq \ell$ for every $i \in [n]$. Given this input, the probability that any L codewords satisfy the list recoverability condition is upper bounded by $q^{kL} \cdot \left(\binom{n}{\alpha n}\left(\frac{\ell}{q}\right)^{\alpha n}\right)^L$. Taking the union bound over all the possibilities for the subsets $S_1, \ldots, S_n$, we obtain that the probability that the random code is not (α, ℓ, L)-list recoverable is upper bounded by

$$\begin{aligned}
\binom{q}{\ell}^n \cdot q^{kL} \cdot \left(\binom{n}{\alpha n}\left(\frac{\ell}{q}\right)^{\alpha n}\right)^L &\leq q^{kL} \cdot e^n \cdot \left(\frac{q}{\ell}\right)^{n\ell} \cdot \left(\frac{e}{\alpha}\right)^{\alpha n L} \cdot \left(\frac{\ell}{q}\right)^{\alpha n L} && (11)\\
&\leq q^{kL} \cdot e^n \cdot \left(\frac{e}{\alpha}\right)^{\alpha n L} \cdot \left(\frac{\ell}{q}\right)^{\alpha n L/2} && (12)\\
&= q^{kL} \cdot e^n \cdot \left(\frac{e^2\ell}{\alpha^2 q}\right)^{\alpha n L/2} && (13)\\
&\leq q^{kL} \cdot e^n \cdot e^{-2\alpha n L} && (14)\\
&\leq q^{kL} \cdot e^{-\alpha n L} && (15)\\
&\leq 2^{-\alpha n L/2}, && (16)
\end{aligned}$$

as desired. In the above, (11) follows by using the upper bound $\binom{a}{b} \leq (ea/b)^b$, (12) follows from the fact that $L \geq 2\ell/\alpha$, (14) follows from the lower bound on q, (15) follows for large enough n and (16) follows from our choice of k. ◀

A.2 Proof of Lemma 2.5

Proof. The parameters of $C^* \stackrel{def}{=} C_{\text{out}} \circ C_{\text{in}}$ follows from the definition of code concatenation. We prove the claims on the list recoverability of C^* by presenting the algorithm to list recover C^*.Given an input $S_{i,j} \subseteq [q]$, for every $i \in [n_1]$ and $j \in [n_2]$ such that $|S_{i,j}| \leq \ell$, first run the list recovery algorithm for C_{in} for the input $S_{i,1}, \ldots, S_{i,n_2}$ for every $i \in [n_1]$ to obtain subsets $T_i \subseteq [Q]$ for every $i \in [n_1]$. (Note that $|T_i| \leq f_2(\ell)$.) Finally, run the list recovery algorithm for C_{out} for the input $T_1, \ldots, T_{n_1}$ and return it's at most $f_1(f_2(\ell))$ output C_{out} codewords (which are in one to one correspondence with the message vectors) as the final output. Note that this algorithm has the claimed running time.

Next, we argue that the algorithm above is correct. Note that a codeword will not be output if for strictly more than $(1-\alpha_1)n_1$ positions $i \in [n_1]$, the corresponding codeword symbols are not in T_i. Further, the latter can happen only if for at least $(1-\alpha_2)n_2$ positions $j \in [n_1]$, the corresponding symbols in the C_{in} codeword do not belong to $S_{i,j}$. Note that these two events cannot happen if the codeword in C^* has at least $(1-(1-\alpha_1)(1-\alpha_2))n_1n_2$ pairs $(i,j) \in [n_1] \times [n_2]$ where the corresponding C^* codeword symbols is in $S_{i,j}$. This proves the claimed parameters of the list recoverability of C^*. ◀

A.3 Proof of Corollary 2.6

Proof. We concatenate m codes $(q_i, k_i \stackrel{def}{=} q_i/R)_{q_i^s}$ codes ($i \in [m]$) from Corollary 2.2, where for every $i \in [m-1]$, we have $(q_i^s)^{k_{i+1}} = q_i^s$, which along with the the fact that $k_{i+1} = R \cdot q_{i+1}$, implies that $q_{i+1} = \frac{1}{R} \cdot \frac{\log q_i}{\log q_{i+1}} \leq \frac{1}{R\log(1/R)} \cdot \log q_i$, where the inequality follows from the fact

that for every i, $q_i \geq 1/R$. (The latter also proves the claimed lower bound on $q = q_m$.) Unraveling the above recursive relation, one can verify that

$$\begin{aligned} q_m &\leq \frac{1}{R\log(1/R)} \cdot \left(\log^{(m)} q_1 + \log(1/R) + \log^{(2)}(1/R) + \cdots + \log^{(m-1)}(1/R)\right) \\ &\leq \frac{1}{R} \cdot \left(\log^{(m)} q_1 + m\right). \end{aligned}$$

The claims on the list recoverability of the final code follows from Lemma 2.5 and the bound on q follows from the inequality above and the fact that $q = q_m$. ◀

B Missing proofs from Section 3.2

B.1 Proof of Lemma 3.8

Proof. We simply output the top ℓ coordinates of $M_{\mathrm{ID}(q)}x + \mu = x + \mu$. To prove that the set S of the top ℓ coordinates of $x + \mu$ contains T, it is sufficient to show that

$$\min_{j\in S}(x_j + \mu_j) \leq \frac{1}{cd} \cdot \|x - x_{H_d(x)}\|_1 + \frac{2}{(c+1)d} \cdot \|\mu\|_1.$$

First, there are at most $(c+1)d$ positions $j \in S$ such that $|x_j| \geq \frac{1}{cd} \cdot \|x - x_{H_d(x)}\|_1$. (At most d indices can come from $H_d(x)$ while at most another cd indices can come from the tail: $[q] \setminus H_d(x)$.) In other words, there is a set $P \subseteq S$ of at least $(c+1)d$ indices for which $|x_j| < \frac{1}{cd} \cdot \|x - x_{H_d(x)}\|_1$ for all $j \in P$. Second, by a Markov argument there is a subset $Q \subseteq S$ of size $|Q| \geq \frac{3}{2}(c+1)d$ for which $|\mu_j| < \frac{2}{(c+1)d} \cdot \|\mu\|_1$ for every $j \in Q$. Note that $P \cap Q \neq \emptyset$, and that for any $j \in P \cap Q \subset S$ we have $(x_j + \mu_j) \leq \frac{1}{cd} \cdot \|x - x_{H_d(x)}\|_1 + \frac{2}{(c+1)d} \cdot \|\mu\|_1$. The proof is thus completed. ◀

B.2 Proof of Corollary 3.9

Proof. Let $c \geq 1$ be a real to be fixed later. Define $\ell = 2(c+1)d$. Let C_{out} be the $\left(n, k = \frac{n}{4\log q}\right)_q$ (with q being the smallest integer larger than $4e^6\ell$) that is $(1/2, \ell, 4\ell)$-list recoverable from Theorem 2.4. Let C_{in} be the $\mathrm{ID}(q)$ code from Lemma 3.8. (Note that the condition on q in Theorem 2.4 implies the condition on q in Lemma 3.8.) By Lemma 3.8, C_{in} satisfies the condition (4) with $\gamma = \frac{1}{cd}$ and $\delta = \frac{2}{(c+1)d}$. Note that $N = q^k$, which implies that $k = \log N/\log q$. Thus we have $t = n \cdot q = 4kq\log q = 4\log Nq \leq O(cd\log N)$. Finally, we use the naive algorithm for list recovering C_{out} (i.e. go through all the codewords in S and retain those that satisfy the "intermediate" input for C_{out} computed by the algorithm outlined in Theorem 3.7). By Theorem 3.7 (and our choices of C_{out} and C_{in}), we obtain in time $O(|S| \cdot n) + n \cdot O(q\log(cd)) = \tilde{O}(|S| \cdot t)$ a set H of size at most 4ℓ such that it contains the set $\left\{i \in S \mid x_i \geq \frac{1}{cd} \cdot \|x - x_{H_d(x)}\|_1 + \frac{4}{(c+1)d\cdot n} \cdot \|\nu\|_1\right\}$. Noting that $n = 4k\log q = 4\log N$, the above implies (ii) if we pick $c = \Theta(1/\epsilon)$. The latter choice also proves (i). ◀

B.3 Proof of Corollary 3.10

Proof. Let $c \geq 1$ be a real to be determined. Let C_{out} be the $(n, k = n/R)_{q^s}$ be the code guaranteed by Corollary 2.6, where $\alpha = \Theta(1/m)$ so that C_{out} is $(1/2, \ell, s^{O(ms)}\ell/R$ and R and q are as in Corollary 2.6. We pick $\ell = 2(c+1)d$. Let C_{in} be the $\mathrm{ID}(q^s)$ from Lemma 3.8. Let M be the matrix given by Theorem 3.7 with these choices of C_{out} and C_{in}.

Recall that $N = q^{ks}$, which implies that $k \leq \log N/s$, which in turn implies that

$$t = n \cdot q^s = \frac{k}{R} \cdot q^s \leq \frac{\log N}{sR} \cdot \frac{1}{R^s} \cdot \left(\log^{(m)} N + m\right)^s \leq (sm)^{O(s)} \cdot \ell^{1+1/s} \cdot (\log^{(m)} N + m)^s$$

By Theorem 3.7, we get that there exists a $\text{poly}(t)$ time algorithm that outputs a set H with $|H| \leq (sm)^{O(sm)} \cdot \ell^{1+1/s}$ that contains the set $\left\{i \in [N] \mid x_i \geq \frac{1}{cd} \cdot \|x - x_{H_d(x)}\|_1 + \frac{4}{(c+1)d \cdot n} \cdot \|\nu\|_1\right\}$. From the fact that $N = q^{ks}$ and $k = nR$, we get that

$$n = \frac{\log N}{sR \log q} \geq \frac{\log N}{sR(\log(1/R) + s \log(\log^{(m)} N + m))} \geq \frac{\log N}{s^2 mR \log(1/R) \log^{(m+1)} N} \geq \frac{\log N}{s^2 m \log^{(m+1)} N},$$

where the first lower bound follows from the upper bound on q. This implies that H contains the set $\left\{i \in [N] \mid x_i \geq \frac{1}{cd} \cdot \|x - x_{H_d(x)}\|_1 + \frac{4s^2 m \log^{(m+1)} N}{(c+1)d \cdot \log N} \cdot \|\nu\|_1\right\}$. The proof is complete by picking $c = \Theta(s^2 m/\epsilon)$. ◀

C Proof of Theorem 4.1

Proof. We will construct the final matrix $\mathbf{M}_{a,b}$ recursively. In particular, let such a matrix in the recursion with N columns be denoted by $\mathbf{M}_{a,b}(N)$. Note that the final matrix is $\mathbf{M}_{a,b} = \mathbf{M}_{a,b}(n)$. (For notational convenience, we will define $D_{a,b}(N)$ and $t_{a,b}(N)$ to be the decoding time for and the number of rows in $\mathbf{M}_{a,b}(N)$respectively). Next, we define the recursion. If $N \leq 2^a$, then set $\mathbf{M}_{a,b}(N) = \mathbf{M}_N$. Note that in this case, $t_{a,b}(N) = t(N)$. Further, we will use the given algorithm in the base case, which implies that $D_{a,b}(N) = d(N, N)$. It is easy to check that both (9) and (10) are satisfied. Finally since $\mathbf{M}_N$ satisfies (7), $\mathbf{M}_{a,b}(N)$ satisfies (8).

Now consider the case when $N > 2^a$. For $i \in [b]$, define $\mathbf{M}^{(i)}$ to be the $t_{a,b}(\sqrt[b]{N}) \times N$ matrix whose kth column (for $k \in [N]$) is identical to the mth column in $\mathbf{M}_{a,b}(\sqrt[b]{N})$ where m is the ith chunk of $\frac{1}{b} \log n$ bits in k (e think of k and m as their respective binary representations). Define $\mathbf{M}_{a,b}(N)$ to be the stacking of $\mathbf{M}^{(1)}, \mathbf{M}^{(2)}, \ldots, \mathbf{M}^{(b)}$ and $\mathbf{M}_N$. First, we verify that (9) holds. To this end note that

$$t_{a,b}(N) = b \cdot t_{a,b}(\sqrt[b]{N}) + t(N). \tag{17}$$

In particular, (by induction) all the $\mathbf{M}^{(i)}$ contribute

$$b \cdot \sum_{j=0}^{\left\lceil \log_b\left(\frac{\log \sqrt[b]{N}}{a}\right)\right\rceil - 1} b^j \cdot t\left(\sqrt[b^j]{\sqrt[b]{N}}\right) = \sum_{j=1}^{\lceil \log_b(\frac{\log N}{a})\rceil - 1} b^j \cdot t\left(\sqrt[b^j]{N}\right)$$

rows. Since $\mathbf{M}_N$ adds another $t(N)$ rows, $\mathbf{M}_{a,b}(N)$ indeed satisfies (9).

Finally, we consider the decoding of $\mathbf{M}_{a,b}(N)$. The decoding algorithm is natural: we run the decoding algorithm for $\mathbf{M}_{a,b}(\sqrt[b]{N})$ (that is guaranteed by induction) on the part of the outcome vector corresponding to each of the $\mathbf{M}^{(i)}$ ($i \in [b]$) to compute sets S_i with the following guarantee.

Let $z^{(i)}$ (for $i \in [b]$) be defined as follows. For any $0 \leq j \leq \sqrt[b]{N} - 1$, the jth entry of $z^{(i)}$ is the sum of the x_k's where the ith chunk of $\log n/b$ bits in k is the same as j (where we think of k and j as $\log n$-bit and $\log n/b$-bit vectors respectively). By induction, when the decoding algorithm for $\mathbf{M}_{a,b}(\sqrt[b]{N})$ is run on $\mathbf{M}^{(i)} z^{(i)}$, then it outputs a set S_i with $|S_i| \leq \ell$ such that

$$\{j \in [\sqrt[b]{N}] \mid z^{(i)}_j \geq \gamma \cdot \|z^{(i)} - z^{(i)}_{H_d(z^{(i)})}\|_1\} \subseteq S_i.$$

Finally, we run the algorithm for $\mathbf{M}_N$ on $\mathbf{M}_N x$ given the set $S \stackrel{def}{=} S_1 \times S_2 \times \cdots \times S_b$ to obtain a set H with $|H| \leq \ell$. To show that H satsifies (8), we need to show that $\{j \in [N] \mid x_j \geq \gamma \cdot \|x - x_{H_d(x)}\|_1\} \subseteq S$. In particular, we need to show that for any $j \in [N]$ with $x_j \geq \gamma \cdot \|x - x_{H_d(x)}\|_1$, $j_i \in S_i$ for every $i \in [\sqrt[b]{N}]$, where j_i denotes the ith chunk of $\log n/b$ bits in j (where we think of j as a $\log N$-bit vector). To this end, we first note that using the same argument as in Theorem 3.7, we have $\|x - x_{H_d(x)}\|_1 \geq \|z^{(i)} - z^{(i)}_{H_d(z^{(i)})}\|_1$. Since x is a non-negative signal (and the definition of $z^{(i)}$), it is easy to see that if $x_j \geq \gamma \cdot \|x - x_{H_d(x)}\|_1$, then $z^{(i)}_{j_i} \geq \gamma \cdot \|z^{(i)} - z^{(i)}_{H_d(z^{(i)})}\|_1$, which in turn implies $j_i \in S_i$, as desired.

To complete the proof, we need to verify that this algorithm takes time asclaimed in (10). Note that $D_{a,b}(N) = b \cdot D_{a,b}(\sqrt[b]{N}) + t(\ell^b, N)$. The rest of the proof of (10) is similar to that of (9).

Finally, the claim on explicitness follows from the construction. ◀