Fast and Robust Support Recovery for Compressive Sensing with Continuous Alphabet

Dong Yin, Ramtin Pedarsani, Xiao Li, Kannan Ramchandran

March 21, 2016

Abstract

In this paper, we consider the compressive sensing (CS) problem in the presence of noise. The problem is to recover a $K$-sparse signal $s \in \mathbb{R}^n$ from noisy linear measurements $y = As + w$. We propose the Fast-Continuous algorithm with large fraction recovery guarantee and full recovery guarantee. Specifically, with high probability, to recover an arbitrarily large fraction of the support of the sparse signal, the algorithm uses $\Theta(K \log(n) \log \log(n))$ samples and $\Theta(K \log^{1+r}(n))$ computational cost, where $r > 0$ is an arbitrarily small constant. The sample and time complexities are almost order optimal. To recover the full support, the algorithm uses sample and time complexities $\Theta(K \log(K) \log(n) \log(n))$ and $\Theta(K \log(K) \log^{1+r}(n))$, respectively. With a mild technical assumption, our algorithm can have $\Theta(K \log(n))$ sample and time complexities for the large fraction recovery and $\Theta(K \log(K) \log(n))$ sample and time complexities for the full support recovery. The design of measurements and the recovery algorithm are based on sparse graph codes. The main contribution of our algorithm compared with previous CS algorithms based on sparse graph codes is that our algorithm relax the assumption that the non-zero elements of the sparse signal lie in finite constellation points, i.e., we allow the non-zero elements taking continuous-valued real numbers. The algorithm that we propose also provides a general framework for the sparse-graph-code based algorithms in other problems such as compressive phase retrieval and sparse covariance estimation to work with continuous-valued signals.

1 Introduction

Sparse signal processing has become an important area in both theory and applications \cite{1, 2}. Compressive sensing (CS) \cite{3} is one of the major problems in sparse signal processing. The goal of compressive sensing is to recover a sparse signal from linear measurements using small number of samples, and in addition, the computational cost of the algorithm should also be as small as possible. Most of the common approaches to compressive sensing, such as $\ell_1$ norm minimization \cite{3}, Iterative Hard Thresholding \cite{4}, and the OMP algorithm \cite{5}, require $m = \mathcal{O}(K \log(n))$ samples to guarantee successful recovery, where $K$ and $n$ denote the sparsity and dimension of the signal, respectively. However, the time complexities of these algorithms are usually $\Omega(n)$, and this computational cost can be prohibitive when processing signals with extremely high dimensions. Therefore, it is becoming increasingly important to study fast algorithms for compressive sensing. In recent years, some fast algorithms, such as \cite{6, 7, 8}, for compressive sensing have been proposed. Among these algorithms, the peeling style algorithm based on sparse graph codes \cite{9} is shown to have almost order optimal sample and time complexity $\Theta(K \log(n))$ for noisy compressive sensing. However, the algorithm is based on the assumption that the non-zero elements in the signal lie in finite constellation points, which is impractical in some applications. In this paper, we propose an
algorithm, called the Fast-Continuous algorithm, which can relax this assumption and deal with continuous-valued signals while maintaining the small sample and time complexity. Our algorithm also provides a general framework for the signal processing and machine learning algorithms which are based on sparse graph codes to provably work for continuous-valued signals in the presence of noise. Specifically, we point out that the robustified algorithms for compressive phase retrieval \cite{10} and sparse covariance estimation \cite{11} can also work for continuous-valued signals using the approaches that we proposed in this paper.

1.1 Problem Formulation

We formally define our problem as follows. Let \( s \in \mathbb{R}^n \) be a \( K \)-sparse unknown signal. With measurement matrix \( A \in \mathbb{R}^{m \times n} \), we can get the linear measurements \( y = As + w \), where \( w \sim \mathcal{N}(0, \sigma^2 I) \) denotes the i.i.d. Gaussian noise, with known variance \( \sigma^2 \). We define the support of the signal \( s \), denoted by \( \text{supp}(s) \), as the set of indices of the non-zero elements of \( s \), i.e., \( \text{supp}(s) = \{ j : s_j \neq 0 \} \). Consequently, we have \( |\text{supp}(s)| = K \) in our problem. We assume that the sparse signal \( s \) satisfies the following minimum magnitude assumption:

**Assumption 1.** For each \( j \in \text{supp}(s) \), \(|s_j| \geq \beta \) for some constant \( \beta > 0 \).

We consider the asymptotic regime where \( n \) and \( K \) approaches infinity, and we do not have other assumptions on the relationship between \( n \) and \( K \), except the fact that \( K \leq n \). Our goal is to design the measurement matrix \( A \) and the decoding algorithm such that we can decode \( s \) with small sample and time complexity.

1.2 Notations

In this paper, for any positive integer \( k \), \([k]\) denotes the set \( \{1, 2, \ldots, k\} \). For a vector \( x \), \( x_j \) denotes the \( j \)th element of \( x \). For \( x \in \mathbb{R}^n \), and a subset \( \Gamma \subset [n] \), \( x_\Gamma \) is a vector consisting of the elements of \( x \) indexed by \( \Gamma \). We use standard BIG-O notations \( \mathcal{O}(\cdot) \), \( \Theta(\cdot) \), and \( \Omega(\cdot) \). The notation \( \text{poly}(n) \) denotes a polynomial with an arbitrarily large constant degree.

2 Main Results

We state our main results in this section. Our first result gives the guarantee for recovering an arbitrarily large fraction \( 1 - \delta \) of the support (we call this kind of recovery guarantee the large fraction recovery).

**Theorem 1.** Let \( s \in \mathbb{R}^n \) be a \( K \)-sparse signal with support \( \Gamma \), satisfying Assumption 1. Let \( \epsilon > 0 \) be a preassigned accuracy parameter, and suppose that there exist universal constants \( c \), \( c_3 \), and \( c_4 \) such that \( \beta > \max\{c\epsilon, (c_3\sigma + c_4\epsilon)^2\} \). Let \( \hat{s} \) be the signal recovered by the Fast-Continuous algorithm and \( \hat{\Gamma} = \text{supp}(\hat{s}) \). Then, for an arbitrarily small \( \delta > 0 \), with proper parameters and \( m = \Theta(\log(\frac{1}{\delta})K \log(n)\log\log(n) + \max\{\frac{\sigma^2}{c^2}, 1\}) \) samples, the Fast-Continuous algorithm satisfies \( \hat{\Gamma} \subset \Gamma \) and

\[
P \left\{ \frac{|\hat{\Gamma}|}{|\Gamma|} \geq 1 - \delta \text{ and } \|\hat{s}_\Gamma - s_\Gamma\|_\infty \leq C\epsilon \right\} \geq 1 - \mathcal{O}(\exp\{-c_1(\delta)K^{c_2(\delta)}\} + \frac{1}{\text{poly}(n)}),
\]
where $C$ is a universal constant and $c_1(\delta)$ and $c_2(\delta)$ are determined only by $\delta$. The time complexity is $\Theta((\log(\frac{1}{\delta})K \log(n)\log^2(n) + \max\{\frac{\sigma^2}{c^2}, 1\}))$, where $r > 0$ is an arbitrarily small constant.

Here, the probability statement comes from the randomness in the Fast-Continuous algorithm. We can see that if $\sigma$, $\epsilon$, and $\delta$ are considered as constants, then the sample and time complexities of the Fast-Continuous algorithm are $\Theta(K \log(n) \log(n))$ and $\Theta(K \log^{1+r}(n))$, respectively, and the probability of accurate large fraction recovery (in terms of $\ell_\infty$ norm) approaches 1 as $n$ and $K$ approaches infinity. We need several steps to prove Theorem 1. These steps are given in Sections 4 and 5. The final proof will be given in Section 7.

In addition, since one single trial of the Fast-Continuous algorithm can recover an arbitrarily large fraction of the support, we can make $\Theta(\log(K))$ trials of the Fast-Continuous algorithm with independent randomness to recover the full support of the sparse signal. We call this algorithm the Repeated-Fast-Continuous algorithm, and formally state the result here.

**Theorem 2.** Under the same conditions as in Theorem 1. Then, with proper parameters and $m = \Theta(K \log(K) \log(n) \log^2(n) + \max\{\frac{\sigma^2}{c^2}, 1\})$ samples, the Repeated-Fast-Continuous algorithm satisfies

$$
P\left\{ \hat{\Gamma} = \Gamma \text{ and } \|\hat{s} - s\|_\infty \leq C\epsilon \right\} \geq 1 - \mathcal{O}(\exp\{-c_1K^{c_2}\} \log(K) + \frac{1}{\text{poly}(n)}),$$

where $c_1$ and $c_2$ are universal constants. The time complexity is $\Theta(K \log(K) \log(n) \log^r(n) + \max\{\frac{\sigma^2}{c^2}, 1\})$.

We can see that if $\sigma$ and $\epsilon$ are considered as constants, then the sample and time complexities of the Fast-Continuous algorithm are $\Theta(K \log(K) \log(n))$ and $\Theta(K \log(K) \log(n))$, respectively, and the probability of accurate full recovery (in terms of $\ell_\infty$ norm) approaches 1 as $n$ and $K$ approaches infinity.

We will see in Section 4 that one part of the recovery algorithm can be formulated as a decoding algorithm in a binary symmetric channel (BSC) with symbols $\pm 1$ and that we only know an upper bound of the bit flip probability. We construct a concatenated code for this BSC, and the encoding and decoding algorithm only need the knowledge of an upper bound of the bit flip probability. The block length of the codeword is $\Theta(\log(n) \log(n))$ and the decoding complexity is $\Theta(\log^{1+r}(n))$. The decoding algorithm of the concatenated code provably succeed with probability $\mathcal{O}(\frac{1}{\text{poly}(n)})$. This is the reason why the log log(n) and log^r(n) factors appear in the sample and time complexities. More details will be provided in Section 4. Here, we mention that this code is used only for theoretical purpose. To the best of our knowledge, the decoding algorithms of the existing codes with decoding complexity linear in the block length of the codewords, such as the LDPC codes, all need the exact knowledge of the bit flip probability of the BSC channel, rather than an upper bound, to provably succeed, and therefore, we need this concatenated code. However, in practice, we believe that the existing state-of-the-art codes can work well in BSC with an upper bound of the bit flip probability. We make the following conjecture.

**Conjecture 1.** There exists a code $f : [n] \to \{-1, 1\}^{R_1}$ with block length of the codewords $R_1 = \Theta(\log(n))$ and decoding complexity $\Theta(\log(n))$, such that when the bits are transmitted through a memoryless BSC, the decoding algorithm can succeed with probability $\mathcal{O}(\frac{1}{\text{poly}(n)})$. The encoding and decoding of this code only need an upper bound of the bit flip probability of the BSC.

\footnote{We use the terminology BSC even though the conventional definition of BSC involves symbols \{0, 1\}.}
Then, by replacing the concatenated code with this conjectured code, we have the following two corollaries.

**Corollary 1.** Under the same conditions as in Theorem 2 and using the code described in Conjecture 1, with \( m = \Theta(\log(\frac{1}{\delta}) \max\{\sigma^2, 1\} K \log(n)) \) samples, the Fast-Continuous algorithm satisfies \( \hat{\Gamma} \subset \Gamma \) and

\[
P\left\{ \frac{|\hat{\Gamma}|}{|\Gamma|} \geq 1 - \delta \text{ and } \|\hat{s}_{\hat{\Gamma}} - s_{\hat{\Gamma}}\|_\infty \leq C\epsilon \right\} \geq 1 - O(\exp\{-c_1(\delta)K^{c_2(\delta)}\}) + \frac{1}{\text{poly}(n)}.
\]

The time complexity is \( \Theta(\log(\frac{1}{\delta}) \max\{\sigma^2, 1\} K \log(n)) \).

**Corollary 2.** Under the same conditions as in Theorem 2 and using the code described in Conjecture 1, with \( m = \Theta(\max\{\sigma^2, 1\} K \log(K) \log(n)) \) samples, the Repeated-Fast-Continuous algorithm satisfies

\[
P\left\{ \hat{\Gamma} = \Gamma \text{ and } \|\hat{s} - s\|_\infty \leq C\epsilon \right\} \geq 1 - O(\exp\{-c_1 K^{c_2}\} \log(K) + \frac{1}{\text{poly}(n)}).
\]

The time complexity is \( \Theta(\max\{\sigma^2, 1\} K \log(K) \log(n)) \).

Again, if \( \sigma, \epsilon, \) and \( \delta \) are considered as constants, then the sample and time complexities of the Fast-Continuous algorithm are \( \Theta(K \log(n)) \) and \( \Theta(K \log(n)) \), respectively; the sample and time complexities of the Repeated-Fast-Continuous algorithm are \( \Theta(K \log(K) \log(n)) \) and \( \Theta(K \log(K) \log(n)) \), respectively.

We summarize our results in Table 1. Here, all the recovery guarantees have success probabilities approaching 1 as \( n \) and \( K \) approaches infinity.

<table>
<thead>
<tr>
<th>Recovery</th>
<th>Sample</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fraction</td>
<td>( \Theta(K \log(n) \log \log(n)) )</td>
<td>( \Theta(n \log^{1+r}(n)) )</td>
</tr>
<tr>
<td>Fraction with conjecture</td>
<td>( \Theta(K \log(n)) )</td>
<td>( \Theta(K \log(n)) )</td>
</tr>
<tr>
<td>Full</td>
<td>( \Theta(K \log(K) \log(n) \log \log(n)) )</td>
<td>( \Theta(K \log(K) \log^{1+r}(n)) )</td>
</tr>
<tr>
<td>Full with conjecture</td>
<td>( \Theta(K \log(K) \log(n)) )</td>
<td>( \Theta(K \log(K) \log(n)) )</td>
</tr>
</tbody>
</table>

### 3 Related Work

Compressive sensing has been extensively studied in recent years and various algorithms has been proposed for different problem settings. We do not attempt to provide a thorough review of all the works related to compressive sensing. We refer the readers to [9] for a summary of the main classes of compressive sensing. Since our proposed Fast-Continuous algorithm is an extension of the sublinear time complexity algorithm for compressive sensing proposed in [9], we provide a brief review of the peeling style algorithm in [9], which is based on sparse graph codes, and the basic idea of the algorithm is divide-and-conquer.

We use a bipartite graph to model the problem. We divide the \( m \) measurements into \( M \) sets, and call each set of measurements a *bin*. By careful design of the measurement matrix, we can
Figure 1: Peeling style algorithm. Left nodes correspond to the signal elements; white, blue, and green nodes represent the zero elements, unrecovered non-zero elements, and recovered non-zero elements, respectively. (a) The bipartite graph representation of the design of measurements. The signal $s \in \mathbb{R}^7$ supports $\{1, 3, 5, 6, 7\}$. The connection between the zero elements and the bins are shown in dashed edges, since the zero elements do not have actual influence on the measurements. (b) The bins 1 and 7 are singletons, so the singleton balls $s_1$ and $s_7$ are recovered. (c) Peel $s_1$ and $s_7$ from other bins. Then bins 3 and 4 become singletons with singleton balls $s_3$ and $s_6$, which can be recovered. (d) Peel $s_3$ and $s_6$ from other bins and $s_5$ becomes a singleton ball. Then, all the non-zero elements are recovered.

make the measurements in each bin only influenced by a small number of signal elements. For example, for a 7 dimensional signal, if we design the $i$th row of the measurement matrix to be $a_i^T = [1 \ 0 \ 1 \ 0 \ 0 \ 0 \ 1]$, then the measurement $y_i$ is only influenced by the first, third, and the seventh signal elements. Then, we can draw a bipartite graph with $n$ left nodes representing the elements of the sparse signal, $M$ right nodes representing the bins, and the edges representing the influence of the signal elements on the bins.

Further, we know that only the non-zero elements can have actual influence on the measurements. Therefore, we prune the graph and only focus on the $K$ left nodes which correspond to the non-zero signal elements. We call these $K$ left nodes balls. Then, we can classify the bins according to the number of balls connected to them (or equivalently, the number of non-zero elements which have influence on the bins). We say that the bins connected to zero, one, and more than one balls are zerotons, singletons, and multitons, respectively. The single signal element connected to a singleton is called a singleton ball.

In the peeling style algorithm, the measurements in a bin are carefully designed such that one can detect whether a bin is a singleton or not, and if a bin is a singleton, one can find the location index and the value of the singleton ball. Then we can subtract (or peel) the influence of this element from other bins. By doing this process iteratively, one can show that if the bipartite graph

---

There are various ways to find the location index and the value. For example, in the noiseless case, one can use ratio tests. We do not provide details of the singleton detection process of the peeling style algorithm in [9] here. In later parts of this paper, we will provide the methods that we use in our Fast-Continuous algorithm.
is properly designed, we can recover all the non-zero elements. We provide an example in Figure 1 and in Section 5, we will revisit this example with a slightly different peeling strategy.

In the following sections, we will provide details of the design of the measurement matrix, the peeling algorithm, and the singleton detection algorithm of the proposed Fast-Continuous algorithm.

4 Design of Measurements

The measurement matrix $\mathbf{A}$ of the Fast-Continuous algorithm is a row tensor of a modulation matrix $\mathbf{T} \in \{-1, 1\}^{R \times n}$ and a code matrix $\mathbf{H} \in \{0, 1\}^{M \times n}$ with rows $\mathbf{h}_i$ and entries $h_{ij}$, $i \in [M]$, $j \in [n]$, denoted by $\mathbf{A} = \mathbf{T} \otimes \mathbf{H}$. Specifically, the row tensor means that $\mathbf{A} = [\mathbf{A}_1 \mathbf{T} \mathbf{A}_2 \mathbf{T} \cdots \mathbf{A}_M \mathbf{T}]^\mathbf{T}$, where $\mathbf{A}_i = \mathbf{T} \text{diag}(\mathbf{h}_i)$ and $\text{diag}(\mathbf{h}_i)$ is a square diagonal matrix with the elements of $\mathbf{h}_i$ on the main diagonal. Consequently, the total number of measurements is $m = MR$.

4.1 Code Matrix

The code matrix $\mathbf{H}$ is the biadjacency matrix of a $d$-left regular random bipartite graph, where $d$ is a parameter of the algorithm. Specifically, we construct a random bipartite graph with $n$ left nodes and $M$ right nodes, and each left node is connected to $d$ right nodes uniformly at random. The left nodes correspond to the $n$ signal elements and the $M$ right nodes correspond to the $M$ sets of measurements generated by sub-matrices $\mathbf{A}_1, \ldots, \mathbf{A}_M$. The $i$-th row of $\mathbf{H}$ represents the connection between the $i$-th right node and all the left nodes. For example, in Figure 1(a), the third right node is connected to the third and fourth left nodes, and therefore, the third row of $\mathbf{H}$ is $\mathbf{h}_3 = [1 \ 0 \ 1 \ 1 \ 0 \ 0 \ 0]$.

We still use the terminologies in Section 3. We call the left nodes which correspond to the $K$ non-zero signal elements balls, and call the $M$ right nodes bins, and we still classify the bins according to the number of balls in them, using the terminologies zeroons, singletons, singleton balls, and multitons. We also define the measurements of bin $i$ as $\mathbf{y}_i = \mathbf{A}_i \mathbf{s} + \mathbf{w}_i$, where $\mathbf{w}_i$ is the noise. Consequently, $\mathbf{y}_i = [\mathbf{y}_i^\mathbf{T} \mathbf{y}_i^\mathbf{T} \cdots \mathbf{y}_i^\mathbf{T}]^\mathbf{T}$ and $\mathbf{w} = [\mathbf{w}_1^\mathbf{T} \mathbf{w}_2^\mathbf{T} \cdots \mathbf{w}_M^\mathbf{T}]^\mathbf{T}$. Since $\mathbf{A}_i = \mathbf{T} \text{diag}(\mathbf{h}_i)$, we have $\mathbf{y}_i = \mathbf{T} \mathbf{x}_i + \mathbf{w}_i$, where $\mathbf{x}_i = \text{diag}(\mathbf{h}_i) \mathbf{s}$ is called the true signal of bin $i$. The Fast-Continuous algorithm is the process of finding all the non-zero signal elements iteratively, or equivalently, finding the locations and values of the $K$ balls, and one should notice that the type of a bin can change during the decoding process due to the iterative peeling. For instance, if a bin contains two balls and one ball is peeled in a certain iteration, the bin becomes a singleton.

4.2 Modulation Matrix

Now we describe how we design the modulation matrix $\mathbf{T}$. The matrix $\mathbf{T}$ consists of two parts, the location matrix $\mathbf{L} \in \{-1, 1\}^{R_1 \times n}$ and the verification matrix $\mathbf{V} \in \{-1, 1\}^{R_2 \times n}$, i.e., $\mathbf{T} = (\mathbf{L}^\mathbf{T} \mathbf{V}^\mathbf{T})^\mathbf{T}$, and thus, we have $R = R_1 + R_2$. We let $\mathbf{T}_j$, $\mathbf{L}_j$, and $\mathbf{V}_j$ be the columns of $\mathbf{T}$, $\mathbf{L}$, and $\mathbf{V}$, $j \in [n]$.

Before we demonstrate the design of the location matrix and the verification matrix, we briefly describe the functions of the two matrices. The modulation matrix $\mathbf{T}$ is used to check whether a bin is a singleton, and if a bin is a singleton, the modulation matrix finds the location index and estimate the value of the singleton ball. Suppose bin $i$ is a singleton and the singleton ball is located at $j$, $j \in [n]$, i.e., $\mathbf{x}_i = s_j \mathbf{e}_j$, where $\mathbf{e}_j$ is the $j$-th vector of the standard basis. Then, the
measures of this bin is \( y_i = s_jT_j + w_i \). We divide the measurements into two parts, location measurements \( y_i^f \) and verification measurements \( y_i^v \), which correspond to the location matrix and verification matrix, respectively. Then, we have \( y_i^f = s_jL_j + w_i^f \) and \( y_i^v = s_jV_j + w_i^v \), where \( w_i^f \) and \( w_i^v \) are the corresponding noise vectors. The location matrix is used to find the location index \( j \), and the verification matrix is used to check whether this bin is indeed a singleton and estimate the value of \( s_j \).

The design of the verification matrix is relatively simple. The entries of the verification matrix \( V \) are i.i.d. Rademacher distributed, i.e., all the entries are independent and equally likely to be either 1 or \(-1\). The detailed procedures of singleton verification and value estimation are shown in Section 3. The design of the location matrix \( L \) is more complicated. Here, we demonstrate the basic idea. As we can see, if bin \( i \) is indeed a singleton, then the location measurements \( y_i^f \) is a scaled version of \( L_j \) with additive noise \( w_i^f \). Let \( \zeta_i = \Phi(-|s_j|/\sigma) \), where \( \Phi(\cdot) \) is the CDF of standard Gaussian distribution. Taking the sign of all the location measurements and considering the randomness of the Gaussian noise, we can see that for each element \( y_{i,k}^f \) in the location measurements, \( k \in [R_1] \), we have

\[
\text{sgn}(y_{i,k}^f) = \begin{cases} 
\text{sgn}(s_j)L_{j,k} & \text{with probability } 1 - \zeta_i \\
-\text{sgn}(s_j)L_{j,k} & \text{with probability } \zeta_i.
\end{cases}
\]

Now the problem becomes a channel coding problem in a symmetric channel with symbols \(+1, -1\). The channel is similar to the binary symmetric channel (BSC) except the fact that we are using \{+1, −1\} rather than \{0, 1\}. For simplicity we will still call this channel a BSC in the following context. Consider the \( n \) possible locations of the singleton balls as \( n \) messages. We encode the \( n \) messages by \( R_1 \)-bit codewords with symbols \(+1, -1\), or equivalently, we design a map \( f : [n] \rightarrow \{+1, -1\}^{R_1} \), and the columns of the location matrix are the codewords of all the messages, i.e., \( L_j = f(j), j \in [n] \). If \( s_j < 0 \), the codeword gets a global sign flip and then we get the modified codeword \( \text{sgn}(s_j)L_j \). Transmitting this modified codeword through a BSC with bit flip probability \( \zeta_i \), we get the received sequence, \( \text{sgn}(y_i^f) \). Then we need a decoding algorithm to decode the original codeword \( L_j \), up to a global sign flip, and then, there are at most two possible locations at which the singleton ball can be. The verification measurements can check whether the bin is indeed a singleton, find the correct location among the two possible choices, and estimate the value of the singleton ball.

Now we describe the encoding and decoding scheme of the location matrix. The code should satisfy four properties:

(i) The block length of the codewords should be as small as possible. Since we need at least \( \Theta(\log(n)) \) bits to encode \( n \) messages, \( R_1 \) should be as close to \( \Theta(\log(n)) \) as possible.

(ii) The decoding complexity should be as close to \( \Theta(\log(n)) \) as possible.

(iii) The decoding algorithm succeeds with high probability; specifically, when there are \( \Theta(1) \) bits flipped, we need the probability of successful decoding to be \( O\left(\frac{1}{\text{poly}(n)}\right)\).

(iv) The decoding algorithm should be universal, i.e., it should not rely on the exact knowledge of the bit flipping probability.

Many of the state-of-the-art capacity achieving codes, such as LDPC codes and Polar codes, satisfy the first three properties. The decoding algorithms in these codes need exact knowledge of the channel, meaning that these algorithms need the flip probability \( \zeta_i \) as a known input parameter. However, in our problem, \( \zeta_i = \Phi(-|s_j|/\sigma) \), where \(|s_j|\) is unknown. This is the reason that we
need universal decoding algorithm. In practice, since we have an upper bound of the bit flip probability, $\zeta_i \leq \Phi(-\beta/\sigma)$, it is reasonable to believe that if we use the upper bound as the bit flip probability, the state-of-the-art capacity achieving codes still work well, although there is no theoretical guarantee. For theoretical interests, here we propose a concatenated code which satisfies all the four properties provably. The results are given in Lemma 1.

**Lemma 1.** There exists a concatenated code $f_c : [n] \rightarrow \{1, -1\}^{R_1}$ for BSC with block length $R_1 = \Theta(\log(n) \log \log(n))$ and universal decoding algorithm, which can successfully decode with probability $O\left(\frac{1}{\text{poly}(n)}\right)$. The decoding complexity is $\Theta(\log(1+r(n)))$, where $r > 0$ is an arbitrarily small constant.

We refer the readers to the Appendix A for the proof of Lemma 1. With this concatenated code, we can construct the location matrix $L$ by setting the $j$-th column as the codeword of $j$, i.e., $L_j = f_c(j)$.

### 5 Peeling Decoder with Truncation

The decoding algorithm of Fast-Continuous is based on the peeling style decoder as in [9]. The basic idea is to use the location matrix and verification matrix to find the location and estimate the value of singleton balls. After identifying a singleton, we peel this singleton ball from the bins that it is connected to. Then, more bins become singletons. We keep the peeling process iteratively until no singletons can be found. The major difference between our work and the previous works is that the signal elements are on a continuous alphabet, which implies that we cannot obtain the exact value of the singleton balls. Then, the error propagation in the peeling process is inevitable. In this paper, we propose a truncation peeling strategy to control the error propagation.

Now we revisit the example in Figure 1 using the truncation peeling strategy, and demonstrate the algorithm in Figure 2. The key in the truncated peeling strategy is that we fix the maximal number of balls that can be peeled from a bin. Denote this maximal number by $D$, which is an input constant parameter of the algorithm. This means that if at least $D$ balls have been peeled from a particular bin, we stop use this bin in following iterations, i.e., we “truncate” large multiton with more than $D$ balls. We set $D = 2$ in the example in Figure 2.

We first assume that by the location measurements and verification measurements, we can perfectly identify whether a bin is a singleton and find the location of the singleton ball. As we can see, in Figure 2, the bins 1 and 7 are singletons and the corresponding singleton balls are balls 1 and 7. In the first iteration, the two singleton balls are found and we let $\hat{s}_1$ and $\hat{s}_7$ be the estimated values. Then, we do peeling, meaning that we subtract the measurements contributed by the two singleton balls from the measurements in other bins. We get the remaining measurements of bins 2, 3, 4, 5, and 6 after the first iteration:

$$
y^{(1)}_2 = y_2 - \hat{s}_1 T_1 - \hat{s}_7 T_7
$$
$$
y^{(1)}_3 = y_3 - \hat{s}_1 T_1
$$
$$
y^{(1)}_4 = y_4 - \hat{s}_7 T_7
$$
$$
y^{(1)}_5 = y_5
$$
$$
y^{(1)}_6 = y_6.
$$
Then we can see that bins 3 and 4 become singletons and the corresponding singleton balls are 3 and 6. We should also notice that since two balls have been peeled from bin 2, according to the truncated peeling strategy, no balls can be peeled from bin 2, and consequently bin 2 is not used in the following iterations. Let \( \hat{s}_3 \) and \( \hat{s}_6 \) be the estimated values of the two singleton balls. Then, the remaining measurements of bins 5 and 6 after the second iteration are:

\[
\begin{align*}
    y^{(2)}_5 &= y^{(1)}_5 - \hat{s}_3 T_3 \\
    y^{(2)}_6 &= y^{(1)}_6 - \hat{s}_6 T_6.
\end{align*}
\]

Then, bins 5 and 6 become singletons and the singleton ball is ball 5. We can estimate the value of ball 5 and get \( \hat{s}_5 \). So far, all the balls have been found, meaning that all the non-zero elements are found.

We summarize the peeling algorithm in Algorithm 1. We have the following result for the peeling algorithm, which guarantees that after the peeling process stops, an arbitrarily large fraction of balls are found.

**Lemma 2.** Assume that we can always find the correct location indices of singleton balls. For any \( \delta > 0 \), when \( K \) is large enough, there exist proper parameters \( d \) and \( M = \Theta(\log(\frac{1}{\delta})K) \), such that after \( N(\delta) \) iterations of truncation peeling, the fraction of non-zero signal elements which are not detected is less than \( \delta \), with probability \( \mathcal{O}(\exp\{-c_1(\delta)K^{c_2(\delta)}\}) \). Here, \( c_1(\delta), c_2(\delta) > 0 \) are two quantities determined by \( \delta \).

We refer the readers to Appendix B for the proof of Lemma 2.

### 6 Singleton Detection and Signal Estimation

In Section 5 we have shown that if the singleton balls are always perfectly detected, an arbitrarily large fraction of non-zero signal elements can be found. Then, an important issue is to guarantee the correct singleton detection and accurate value estimation.
Algorithm 1 Peeling decoder with truncation strategy

**Input:** \( y, T, H, D \)

**Output:** Estimated signal \( \hat{s} \)

\[
\hat{s} \leftarrow 0, \text{ number of peeled balls in each bin: } B_i \leftarrow 0, \ i \in [M], \\
\text{Indicator of utilizability of bins: } U_i \leftarrow \text{true}, \ i \in [M], \\
y_i^{(0)} \leftarrow y_i, \ i \in [M], \text{ stop } \leftarrow \text{false}, \ t \leftarrow 1
\]

while stop = \text{false} do

Find singleton balls. \( I_t \leftarrow \{\text{indices of all singletons found in the } t\text{-th iteration}\} \).

\( U_i \leftarrow \text{false}, \ \text{for all } i \in I_t \).

\( J_t \leftarrow \{\text{location indices of singleton balls found in } t\text{-th iteration}\} \).

\( y_i^{(t)} \leftarrow y_i^{(t-1)}, \ i \in [M] \).

if \( J_t \neq \emptyset \) then

for all \( j \in J_t \) do

Estimate \( \hat{s}_j \).

for all \( i \in [M] \) such that \( U_i = \text{true} \) and \( h_{i,j} = 1 \) do

\[ y_i^{(t)} \leftarrow y_i^{(t)} - \hat{s}_j T_j, \]

\( B_i \leftarrow B_i + 1. \)

if \( B_i = D \) then

\( U_i = \text{false} \)

end if

end for

end for

else

stop \leftarrow \text{true}

end if

\( t \leftarrow t + 1 \)

end while

return \( \hat{s} \)

First, recall that we have shown in Section 4, in the first iteration, if a bin is indeed a singleton, from the location measurements, one can decode the modified codeword corresponding to the location index of the singleton ball. Due to the sign ambiguity, there may be two possible locations and the true location is guaranteed to be one of them with high probability. We still need to find the correct location and estimate the value of the singleton ball. On the other hand, if the bin is not a singleton, the decoding algorithm of the concatenated code still returns at most two possible locations and we have to make sure that these bins are not considered as singletons. These problems are addressed by energy tests using the verification measurements, based on the same idea as in [10].

### 6.1 Signal Value Estimation

Consider the \( i\)-th bin in the \((t + 1)\)-th iteration, and let \( j \) be a possible location of the singleton ball that the decoding algorithm of the concatenated code suggests. We assume that the bin is indeed a singleton with the singleton ball located at \( j \), and estimate \( s_j \) by the remaining verification
measurements, i.e.,
\[ \hat{s}_j = \frac{1}{R_2} \sum_{k=1}^{R_2} V_{k,j} y_{i,k}^{(t),v}. \] (1)

Here, \( y_{i,k}^{(t),v} \) is the \( k \)-th remaining verification measurement of the \( i \)-th bin after the \( t \)-th iteration. Intuitively, this estimation method is simply averaging over the measurements with corrected sign, meaning that we flip the sign if the corresponding entry in the verification matrix is \(-1\). The theoretical guarantee of singleton detection and estimation is presented in Lemma 3.

**Lemma 3.** In Fast-Continuous algorithm, there exists a constant \( c > 0 \) such that for an accuracy level \( \epsilon > 0 \), if \( \beta \geq c \epsilon \), then, for a singleton in any constant iteration, we have:

(i) the location measurements can find the correct location of the singleton ball with probability \( 1 - O(1/\text{poly}(n)) \).

(ii) by equation (1), the estimated value of singleton ball \( \hat{s}_j \) satisfies \( |\hat{s}_j - s_j| \leq C_j \epsilon \) for some constant \( C_j > 0 \) with probability \( 1 - O(1/\text{poly}(n)) \) given the number of verification measurements in a bin is \( R_2 = \Theta(\frac{\sigma^2}{\epsilon^2 \log(n)}) \).

We should note that result (i) is a simple extension of the conclusion that we get in Section 4, since in Section 4 we focused on the first iteration. Result (ii) shows that for the preassigned accuracy level \( \epsilon > 0 \), if the number of verification measurements is \( R_2 = \Theta(\frac{\sigma^2}{\epsilon^2 \log(n)}) \), then we can estimate the signal value within constant factor of \( \epsilon \) with high probability. The detailed proof of Lemma 3 is given in Appendix C.

### 6.2 Energy Test

So far, we have seen that if a bin is indeed a singleton, the location measurements can find the correct location of the singleton ball and the verification measurements can give accurate estimation of the value. However, there are still several things left. As we have mentioned, we need to clarify sign ambiguity, and rule out non-singleton bins. These operations can be done by energy tests.

Consider the \( i \)-th bin in the \( t \)-th iteration. Let \( B \) be the set of location indices of balls that have been found before this iteration. Before using the location measurements to find the location index of a singleton ball, we use an energy test to check if this bin is a zeroton, i.e., check if \( \text{supp}(x_i) = B \). If it is, there is no need to run the decoding algorithm of the concatenated code. Specifically, we construct \( \hat{y}_1 = \sum_{g \in B} \hat{s}_g V_g \) and conduct the zeroton energy test with threshold \( \tau > 0 \):

\[
\text{if } \frac{1}{R_2} \| y_{i}^v - \hat{y}_1 \|_2^2 < \tau, \text{ bin } i \text{ is a zeroton};
\]

otherwise bin \( i \) is not a zeroton.

If the bin is not a zeroton, we use the location measurements to find a possible singleton location \( j \) and get the estimated the value \( \hat{s}_j \). We need to verify if there is indeed \( \text{supp}(x_i) = B \cup \{j\} \). Similar to the zeroton test, we construct \( \hat{y}_2 = \sum_{g \in B \cup \{j\}} \hat{s}_g V_g \) and conduct the singleton energy test with threshold \( \tau > 0 \):

\[
\text{if } \frac{1}{R_2} \| y_{i}^v - \hat{y}_2 \|_2^2 < \tau, \text{ bin } i \text{ is a singleton with singleton ball at } j;
\]

otherwise bin \( i \) is not a singleton with singleton ball at \( j \).
The intuition of both of the two kinds of energy tests is simple. We actually make a hypothesis that the true signal of a bin is $\hat{x}$ and construct the corresponding measurements $\hat{y} = V\hat{x}$. If the support of $\hat{x}$ and $x_i$ are the same and the values are accurately estimated, i.e., $\|\hat{x} - x_i\|_{\infty} < C_0\epsilon$, for some constant $C_0 > 0$. Then the energy of the difference between the actual measurements and the constructed measurements should be small, otherwise the energy should be large. For zero-to-energy test, the hypothesis signal consists of the balls that are found before the $t$-th iteration, and for the singleton energy test, the hypothesis signal consists of the balls which are found before and including the $t$-th iteration. The theoretical guarantee of both kinds of energy tests are given in Lemma 4.

**Lemma 4.** There exist constants $c_3 > 0$ and $c_4 > 0$ such that when $\beta > (c_3\sigma + c_4\epsilon)^2$, there exists proper threshold $\tau > 0$ such that a particular energy test succeeds with probability $1 - O(\frac{1}{\text{poly}(n)})$, when $R_2 = \Theta(\max\{\frac{\sigma^2}{\tau^2}, 1\} \log(n))$.

We refer the readers to Appendix D for the proof of Lemma 4.

### 7 Proof of Theorem 1

In this section, we give the final proof of our main result. First, we analyze the error probability. There are three possible error events,

(i) $E_1$: the peeling algorithm does not find an arbitrarily large fraction of non-zero elements.

(ii) $E_2$: error in decoding algorithm of concatenated code (location decoding).

(iii) $E_3$: error in value estimation or energy test.

We have shown that $P\{E_1|E_2^C, E_3^C\} = O(\exp\{-c_1(\delta)K^{-c_2(\delta)}\})$. Similar to the analysis in [10], since we need to conduct $\Theta(K)$ times of location decoding and energy tests, using union bound, $P\{E_2\} = O(\frac{1}{\text{poly}(n)})$ and $P\{E_3\} = O(\frac{1}{\text{poly}(n)})$. Then by union bound and law of total probability, we get the error probability $O(\exp\{-c_1(\delta)K^{-c_2(\delta)}\} + \frac{1}{\text{poly}(n)})$. Then computational complexity can be analyzed by the same method as in [10]. Combining Lemma 1, 2, 3, and 4 our main result is proved.

### References


Appendices

A Proof of Lemma 1

In this section, we give a brief proof of Lemma 1. The construction of the concatenated code in Lemma 1 is based on Justesen’s concatenation scheme [12] and similar method is also analyzed in [13]. The concatenated code consists of an outer code \( f_{\text{out}} \) and an ensemble of inner codes \( \mathcal{I} \). For the outer codes, we use an expander-based code proposed in [14]. The outer code maps the message complexity of testing each possible message is \( \Theta(q) \) which can have arbitrarily large degree. Consider the decoding complexity. For the inner code, the algorithm of the outer code does not rely on the knowledge of the channel. Now let \( (\mathcal{I}, f_{\text{out}}) \) be the codeword that we obtained from the outer code, and we call it the outer codeword. As we have mentioned, we use an ensemble of inner codes \( \mathcal{I} \), which means that \( \mathcal{I} = \{g_1, \ldots, g_p\} \) is a collection of \( p \) codes which encode the symbols in the outer codeword as a new \( q \)-bit codeword with alphabet \( \{1, -1\} \). Specifically, each code \( g_i \) in \( \mathcal{I} \) is a map \( g_i : [2^k] \rightarrow \{1, -1\}^q \), and we encode the \( i \)-th symbol in the outer codeword by the \( i \)-th code in \( \mathcal{I} \). This gives us the final codeword \( (g_1(c_1), g_2(c_2), \ldots, g_p(c_p)) \in \{1, -1\}^q \), which also implies that the block length of the concatenated code is \( R_1 = qp \).

Then we show the details of the inner code ensemble. We choose the inner code ensemble to be the Wozencraft’s ensemble [15]. The Wozencraft’s ensemble satisfies the property that all but a \( o(1) \) fraction of the codes in the ensemble are capacity achieving, where the asymptotic is with respect to the block length \( q \). Specifically, for the capacity achieving codes in the ensemble, the probability of decoding error is exponentially small in the block length \( q \), i.e., \( e^{-\alpha q} \) for some constant \( \alpha > 0 \), as long as the rate of the codes \( R_{\text{in}} = k/q \) is below the capacity of the BSC. Here, we should notice that we do need an upper bound of the bit flip probability in the design of the inner code since we need to get a lower bound of the capacity of the BSC, however, we do not need the exact value of the bit flip probability. Then, it is shown in [13] that using brute force maximum likelihood decoder for the inner code and the decoding algorithm of the expander-based outer code, the error probability is exponentially small in the block length of the concatenated code, i.e., \( e^{-\alpha' R_1} \) for some constant \( \alpha' > 0 \).

Now we analyze the block length and decoding complexity of the concatenated code. The number of codes in the Wozencraft’s ensemble is \( 2^q \), meaning that \( p = 2^q \). Since rate of the outer code is a constant \( R_{\text{out}} = [\log(n)]/p \) which can be arbitrarily close to 1, we know that \( p = \Theta(\log(n)) \). Then \( q = \Theta(\log \log(n)) \) and the block length of the concatenated code is \( R_1 = qp = \Theta(\log(n) \log \log(n)) \). Consequently the error probability is \( e^{-\alpha R_1} = O(\frac{1}{\text{poly}(n)}) \), where \( \text{poly}(n) \) is a polynomial of \( n \) which can have arbitrarily large degree. Consider the decoding complexity. For the inner code, the complexity of testing each possible message is \( \Theta(q) \) and there are \( 2^k = 2^{R_{\text{in}}} \) messages. Therefore, for each inner code, the computational complexity of the brute force maximum likelihood decoding is \( \Theta(2^{R_{\text{in}}} q) \). Since there are \( p \) inner codes, the complexity of decoding all the inner codes is
\( \Theta(2^{q_R \ln q}) = \Theta(p^{1+R_{in}}) = \Theta(\log^{1+R_{in}}(n) \log \log(n)) \). Since we do not require the inner code to be capacity achieving, \( R_{in} \) can be arbitrarily close to 0, we can conclude that complexity of decoding all the inner codes is \( \Theta(\log^{1+r}(n)) \), where \( r > 0 \) can be arbitrarily small. Since the complexity of decoding the outer code is linear in its block length, which is \( \Theta(p) = \Theta(\log(n)) \), we know that the decoding complexity of the concatenated code is \( \Theta(\log^{1+r}(n)) \).

### B Proof of Lemma 2

The proof of Lemma 2 is based on density evolution, which is a common and powerful tool in modern coding theory. Similar ideas have been applied in [9] [16] [17]. The basic idea is to get a recursive equation to analyze the fraction of singleton balls that are not recovered in a particular iteration. We will only give brief proof here and focus on the truncation peeling strategy, which is main difference between our algorithm and previous works. We refer the readers to [9] [16] for more details.

From now on, we do not consider the connection between the zero signal elements and the bins, meaning that we only focus on the \( d \)-left regular random bipartite graph with \( K \) balls on the left and \( M \) bins on the right. We let \( M = \eta K \) for some constant \( \eta > 0 \). Using the same Poisson approximation as in [16], we get the expected fraction of edges which are connected to right nodes with degree \( i \) is

\[
\rho_i \approx \frac{(d/\eta)^i e^{-d/\eta}}{(i-1)!}.
\]

Now we consider the peeling process as a message passing process on the bipartite graph. According to our peeling decoding algorithm, a singleton can send a “peeling” message to the ball that is connected to it, and a peeled singleton ball sends “peeled” message to all the bins that are connected to it. In a particular iteration, a bin sends a “peeling” message to a ball through an edge if other edges connected to this bin all send “peeled” messages in the previous iteration and a ball sends a “peeled” message to a bin through an edge if if at least one of the bins that is connected to it sends a “peeling” message to it. We should also notice that the bins with degree greater than \( D \) never send “peeling” message to the balls due to the truncation strategy.

As in [9] [16], we still need to first assume that the neighborhood of each edge with a constant depth is a tree (tree-like assumption). Let \( p_j \) be the probability that in the \( j \)th iteration, a randomly chosen edge is not peeled, i.e., sending a “not peeled” message. Then, under the tree-like assumption, we have the density evolution equation:

\[
p_{j+1} = F(p_j) = \left(1 - \sum_{i=1}^{D} \rho_i (1 - p_j)^{i-1}\right)^{d-1}.
\]

Similar to the analysis in [16], we need to consider the fix point of \( F(t) \), i.e., the point such that \( F(t) = t \), and show that the fix point can be arbitrarily small by choosing proper parameters. We
have

\[
F(t) = \left( 1 - \sum_{i=0}^{D-1} \frac{(d/\eta(1-t))^i e^{-d/\eta}}{i!} \right)^{d-1}
\]

Then we know that the fix point of \( F(t) \) should be upper bounded by that of \( G(t) \). Further, if we keep \( d/\eta \) to be a constant and enlarge \( d \), the fix point of \( G(t) \) can be arbitrarily small, and consequently, the fix point of \( F(t) \) can be arbitrarily small. Specifically, let \( p^* \in (0,1) \) be the fix point of \( F(t) \), then for any \( \delta > 0 \), there exist parameters \( d \) and \( \eta \) such that \( p^* < \delta \). Here, we briefly analyze the relationship between \( \eta \) and the fix point of \( G(t) \), denoted by \( t^* \). Since \( t^* = G(t^*) = (1 - \frac{1}{2} e^{-dt/\eta})^{d-1} \), and \( t^* \) is close to 0, we have \( e^{-dt/\eta} \approx 1 \) and thus \( t^* \approx (\frac{1}{2})^{d-1} \). Therefore, \( d = \Theta(\log(1/t^*)) \), and further, since we keep \( d/\eta \) as a constant, \( \eta = \Theta(\log(1/p^*)) \). We can choose parameters such that \( p^* = \Theta(\delta) \) and then, \( \eta = \Theta(\log(1/\delta)) \). Using the same argument as in [16], we can show that for any \( \delta > 0 \), there exist a constant \( N(\delta) \) and proper parameters \( d \) and \( \eta \) such that \( p_N < \delta \).

By the same martingale argument as in [16] and taking the event that the tree-like assumption does not hold into consideration, we can show that the fraction of balls which are not peeled is highly concentrated around \( p_N \). Let \( Z \) be the fraction of balls which are not peeled after \( N \)-th iteration, when \( K \) is large enough, we have for any \( \delta > 0 \),

\[
\mathbb{P} \{ |Z - p_N| > \delta \} < 2 \exp\{-C\delta^2 K^{1/(4N+1)}\},
\]

where \( C > 0 \) is a universal constant. The proof of Lemma 2 is completed by choosing \( N \) such that \( p_N < \delta \).

C Proof of Lemma 3

Suppose that in a certain constant iteration \( t \), there is a singleton bin with the singleton ball located at \( j \). We simplify the notations before showing the details. Without loss of generality, we omit the bin index \( i \) and the number of iterations \( t \), use \( z^\ell = y^\ell_i \) and \( z^v = y^{(v)}_i \) to denote the remaining location and verification measurements of the \( i \)-th bin after the \( t \)-th iteration, respectively, and use \( x = x_i \) to denote the true signal of bin \( i \). We also denote the noise in the location and verification measurements of bin \( i \) as \( w^\ell \) and \( w^v \), respectively. Thus, we have \( z^\ell = Lx + w^\ell \) and \( z^v = Vx + w^v \).

First, consider \( t = 1 \). In the first iteration, we know that the remaining measurements are the original measurements, i.e., \( z^\ell = y^\ell_i \) and \( z^v = y^{(v)}_i \). In Section 4 we have shown that if a singleton
ball is located at $j$, the concatenated code designed in the location matrix can find the location $j$ with probability $1 - O(\frac{1}{\text{poly}(n)})$. According to the estimation method in [1], we have

$$\hat{s}_j = \frac{1}{R_2} \sum_{k=1}^{R_2} V_{k,j} z_k^j = \frac{1}{R_2} \sum_{k=1}^{R_2} s_j + \tilde{w}_k,$$

where $\tilde{w}_k = V_{k,j} w_k^j$. Then we have $\tilde{w} = \{V_{k,j} w_k^j\}_{k=1}^{R_2} \sim \mathcal{N}(0, \sigma^2 I)$. By Chernoff bound, we have for any $\epsilon > 0$,

$$\mathbb{P}\{|\hat{s}_j - s_j| \geq \epsilon\} \leq \exp\{-\frac{R_2 \epsilon^2}{2\sigma^2}\},$$

therefore, by choosing $R_2 = \Theta(\frac{\sigma^2}{\epsilon^2} \log(n))$, we can get $\hat{s}_j$ such that $|\hat{s}_j - s_j| < \epsilon$ with probability $1 - O(\frac{1}{\text{poly}(n)})$.

Consider the $t$-th iteration, $t > 1$. Since $t > 1$, bin $i$ is not a singleton in the first iteration, and thus, $|\text{supp}(x)| > 1$. Let $\mathcal{B} = \text{supp}(x) \setminus \{j\}$, i.e., $\mathcal{B}$ is the set of location indices of the balls which are peeled off from bin $i$. According to the truncation peeling strategy, we have $|\mathcal{B}| \leq D - 1$. Assume that for any $g \in \mathcal{B}$, we have $|\hat{s}_g - s_g| < C_g \epsilon$ with probability $1 - O(\frac{1}{\text{poly}(n)})$ for some constant $C_g > 0$. We let $C_B = \sum_{g \in \mathcal{B}} C_g$.

Now we show that if there exists appropriate constant $C$ such that $\beta \geq C \epsilon$, then the decoding algorithm can find the location of the singleton ball $j$ with probability $1 - O(\frac{1}{\text{poly}(n)})$. Recalling that to conduct the decoding algorithm of the concatenated code, we need to take the sign of $\sum_{g \in \mathcal{B}} (s_g - \hat{s}_g) L_g + s_j L_j + w_j$. Let $\bar{s} = \sum_{g \in \mathcal{B}} (s_g - \hat{s}_g) L_g + s_j L_j$. We assume that $\beta$ is large enough such that in any constant iteration $\beta \geq 2C_B \epsilon$. Then, for each entry $\text{sgn}(\hat{s}_k) = \text{sgn}(s_j L_{k,j})$, and we can think of $\text{sgn}(z_k^j)$ as a received symbol by transmitting $\text{sgn}(s_j L_{k,j})$ through a BSC with bit flip probability upper bounded by $\Phi(-\frac{\beta}{2\sigma})$. Then, the decoding algorithm of the concatenated code still works since we have a constant upper bound of the bit flip probability.

Then, we show that the value estimation method still works in the $t$-th iteration. Since $V_{k,j} z_k^j = s_j + \sum_{g \in \mathcal{B}} (s_g - \hat{s}_g) V_{k,j} L_g + V_{k,j} w_k^j$, and $\hat{s}_j = \frac{1}{R_2} \sum_{k=1}^{R_2} V_{k,j} z_k^j$, we know that conditioned on $\mathcal{V}$, $\hat{s}_j$ and the event that $|\hat{s}_g - s_g| < C_g \epsilon$ for all $g \in \mathcal{B}$, $\hat{s}_j \sim \mathcal{N}(s_j + \bar{s}, \frac{\sigma^2}{R_2})$, where $\bar{s} = \frac{1}{R_2} \sum_{k=1}^{R_2} \sum_{g \in \mathcal{B}} (s_g - \hat{s}_g) V_{k,j} L_g$. We can see that $|\bar{s}| < C_B \epsilon$, and by Chernoff bound,

$$\mathbb{P}\{|\hat{s}_j - s_j - \bar{s}| \geq \epsilon \mid \mathcal{V}, |\hat{s}_g - s_g| < C_g \epsilon\} \leq \exp\{-\frac{R_2 \epsilon^2}{2\sigma^2}\}. \tag{2}$$

Since (2) is true for all $\mathcal{V}$, we can get rid of the condition on $\mathcal{V}$. Considering the fact that $|\bar{s}| < C_B \epsilon$, we get

$$\mathbb{P}\{|\hat{s}_j - s_j| \geq (C_B + 1) \epsilon \mid |\hat{s}_g - s_g| < C_g \epsilon\} \leq \exp\{-\frac{R_2 \epsilon^2}{2\sigma^2}\}.$$

Then, by law of total probability and union bound, we get

$$\mathbb{P}\{|\hat{s}_j - s_j| \geq (C_B + 1) \epsilon\} \leq \exp\{-\frac{R_2 \epsilon^2}{2\sigma^2}\} + \sum_{g \in \mathcal{B}} \mathbb{P}\{|\hat{s}_g - s_g| \geq C_g \epsilon\} \leq O(\frac{1}{\text{poly}(n)}),$$

if $R_2 = \Theta(\frac{\sigma^2}{\epsilon^2} \log(n))$, which completes the proof.
D Proof of Lemma 4

We make essential use of the results in [18], which is also stated in [19].

Lemma 5. For any $\theta \in (0,1)$ and any $\mathbf{u} \in \mathbb{R}^n$,
\[
\mathbb{P}\left\{ \frac{1}{R_2} \left| \mathbf{V} \mathbf{u} \right|_2^2 - \left| \mathbf{u} \right|_2^2 \geq \theta \left| \mathbf{u} \right|_2^2 \right\} \leq 2 \exp\{-R_2(\frac{\theta^2}{4} - \frac{\theta^3}{6})\}.
\]

Similar to Appendix C, we also omit the bin index and iteration counter here, and let $\mathbf{y}^v$ be the actual verification measurements of bin $i$ and $\mathbf{w}^v$ be the corresponding noise, i.e., $\mathbf{y}^v = \mathbf{y}_i^{(i),v}$ and $\mathbf{w}^v = \mathbf{w}_i^v$. Let $\mathbf{x}$ be the true signal of the bin, i.e., $\mathbf{x} = \mathbf{x}_i$, and $\hat{\mathbf{x}}$ be the hypothesis signal. We have $\mathbf{y}^v = \mathbf{V} \mathbf{x} + \mathbf{w}^v$, and then $\mathbf{y}^v - \mathbf{V} \hat{\mathbf{x}} = \mathbf{V} \mathbf{x} + \mathbf{w}^v - \mathbf{V} \hat{\mathbf{x}}$, where $\hat{\mathbf{x}} = \mathbf{x} - \hat{\mathbf{x}}$. By Lemma 5 we have
\[
\mathbb{P}\left\{ \sqrt{1-\theta} \left| \mathbf{x} \right|_2 \leq \frac{1}{\sqrt{R_2}} \left| \mathbf{V} \hat{\mathbf{x}} \right|_2 \leq \sqrt{1+\theta} \left| \mathbf{\hat{x}} \right|_2 \right\} \geq 1 - 2 \exp\{-R_2(\frac{\theta^2}{4} - \frac{\theta^3}{6})\}.
\]

By triangle inequality, $\left| \mathbf{V} \hat{\mathbf{x}} \right|_2 - \left| \mathbf{w}^v \right|_2 \leq \left| \mathbf{y}^v - \mathbf{V} \hat{\mathbf{x}} \right|_2 \leq \left| \mathbf{V} \hat{\mathbf{x}} \right|_2 + \left| \mathbf{w}^v \right|_2$.

Then, on the one hand, we have
\[
\mathbb{P}\left\{ \frac{1}{\sqrt{R_2}} \left| \mathbf{y}^v - \mathbf{V} \hat{\mathbf{x}} \right|_2 \geq \sqrt{1-\theta} \left| \mathbf{\hat{x}} \right|_2 - \frac{1}{\sqrt{R_2}} \left| \mathbf{w}^v \right|_2 \right\} \geq 1 - 2 \exp\{-R_2(\frac{\theta^2}{4} - \frac{\theta^3}{6})\}.
\]

By the concentration inequality of $\chi^2$ distribution, for any $\phi \in (0,3)$, we have
\[
\mathbb{P}\left\{ \frac{1}{R_2} \left| \mathbf{w}^v \right|_2 \geq \sigma^2(1+\phi) \right\} \leq \exp\{-R_2\frac{\phi^2}{18}\}.
\]

By union bound, we get
\[
\mathbb{P}\left\{ \frac{1}{\sqrt{R_2}} \left| \mathbf{y}^v - \mathbf{V} \hat{\mathbf{x}} \right|_2 \geq \sqrt{1-\theta} \left| \mathbf{\hat{x}} \right|_2 - \sigma \sqrt{1+\phi} \right\} \geq 1 - 2 \exp\{-R_2(\frac{\theta^2}{4} - \frac{\theta^3}{6})\} - \exp\{-R_2\frac{\phi^2}{18}\}.
\]

Suppose that the supports of the hypothesis signal and the true signal are different, i.e., supp($\hat{\mathbf{x}}$) $\neq$ supp($\mathbf{x}$), then by our assumption of the signal, $\left| \mathbf{\hat{x}} \right|_2 \geq \sqrt{\beta}$. If $\sqrt{1-\theta} \sqrt{\beta} - \sigma \sqrt{1+\phi} > 0$, we can get a valid threshold, which means that if $\beta > \sigma^2(\frac{1+\phi}{\sqrt{2}^2})$, when $R_2 = \Theta(\log(n))$,
\[
\mathbb{P}\left\{ \frac{1}{R_2} \left| \mathbf{y}^v - \mathbf{V} \mathbf{\hat{x}} \right|_2^2 \geq \tau \right\} \geq 1 - \mathcal{O}\left(\frac{1}{\text{poly}(n)}\right),
\]
for any $\tau \in (0, (\sqrt{1-\theta} \sqrt{\beta} - \sigma \sqrt{1+\phi})^2)$.

On the other hand, we also have
\[
\mathbb{P}\left\{ \frac{1}{\sqrt{R_2}} \left| \mathbf{y}^v - \mathbf{V} \mathbf{x} \right|_2 \leq \sqrt{1+\theta} \left| \mathbf{\hat{x}} \right|_2 + \frac{1}{\sqrt{R_2}} \left| \mathbf{w}^v \right|_2 \right\} \geq 1 - 2 \exp\{-R_2(\frac{\theta^2}{4} - \frac{\theta^3}{6})\}.
\]

Consider the case when supp($\hat{\mathbf{x}}$) = supp($\mathbf{x}$). In this case, we have found the correct support, or equivalently, all the locations of the singleton balls are found. By Lemma 3, we know that $\left| \mathbf{x} \right|_\infty < \hat{C} \epsilon$ for some constant $\hat{C}$ with probability $1 - \mathcal{O}(\frac{1}{\text{poly}(n)})$, when $R_2 = \Theta(\frac{\epsilon^2}{2} \log(n))$. According
to the truncation strategy, we also have $|\tilde{x}| \leq D$, and thus $\|\tilde{x}\|_2 < \sqrt{D}\tilde{C}\epsilon := C'\epsilon$. Using this fact and union bound, we get

$$P \left\{ \frac{1}{\sqrt{R_2}} \|y^v - V\tilde{x}\|_2 \leq \sqrt{1 + \theta C'\epsilon + \sigma \sqrt{1 + \phi}} \right\} \geq 1 - O\left( \frac{1}{\text{poly}(n)} \right),$$

and thus, for any $\tau > (\sqrt{1 + \theta C'\epsilon + \sigma \sqrt{1 + \phi}})^2$,

$$P \left\{ \frac{1}{R_2} \|y^v - V\tilde{x}\|_2 \leq \tau \right\} \geq 1 - O\left( \frac{1}{\text{poly}(n)} \right). \quad (4)$$

We can see that to get a valid threshold for both tests (3) and (4), we need

$$\sqrt{1 - \theta \sqrt{\beta} - \sigma \sqrt{1 + \phi}} > \sqrt{1 + \theta C'\epsilon + \sigma \sqrt{1 + \phi}},$$

and since $\theta$ and $\phi$ are constants, the proof is completed.