Sub-string/Pattern Matching in Sub-Linear Time Using a Sparse Fourier Transform Approach

Nagaraj T. Janakiraman, Avinash Vem, Krishna R. Narayanan
Department of Electrical & Comp. Engg., Texas A&M University, College Station, TX, U.S.A
{tjnagaraj,vemavinash,krn}@tamu.edu

Abstract

We consider the problem of querying a string (or, a database) of length N bits to determine all the locations where a substring (query) of length M appears either exactly or is within a Hamming distance of K from the query. We assume that sketches of the original signal can be computed off line and stored. Using the sparse Fourier transform computation based approach introduced by Pawar and Ramchandran, we show that all such matches can be determined with high probability in sub-linear time. Specifically, if the query length $M = N^{\mu}$ and the number of matches is $L = N^{\lambda}$, as $N \to \infty$, we show that for $\lambda < 1 - \mu$ all the matching positions can be determined with a probability that approaches 1 for $K \leq \frac{1}{6}M$. More importantly our scheme has a worst-case computational complexity that is only $O\left(N^{\max(\alpha,\lambda+1-\alpha)}\log^2 N\right)$ where $\alpha = \max(\mu, 1-\mu)$ which means we can recover all the matching positions in *sub-linear* time for $\lambda < 1-\mu$. Further, the number of Fourier transform coefficients that need to be computed, stored and accessed, i.e., the sketching complexity of this algorithm is only $O\left(N^{\max(\mu,1-\mu)}\right)$. Several extensions of the main theme are also discussed.

I. Introduction and Problem Statement

We consider the substring/pattern matching problem, which is a problem that has been studied extensively in theoretical computer science. In this problem, a signal $\vec{x} := (x[0], x[1], \ldots, x[N-1])$ of length N symbols representing a string, library or database is available and the objective is to answer queries regarding whether a given string $\vec{y}$ of length M is a substring of $\vec{x}$. We consider the case where a sketch of $\vec{x}$ can be computed offline and stored, and the one-time computational complexity of creating the sketch can be ignored. In this paper, we consider the random setting in which $x[i]$'s are a sequence of independent and identically distributed (i.i.d) random variables taking values in $\mathscr{A} \subset \mathbb{R}$. We begin by considering the binary case where $\mathscr{A} := \{\pm 1\}$ [1]. Within this context, we consider the following two problems:

Exact Pattern Matching: In the exact pattern matching problem, a substring of length M of $\vec{x}$, namely $\vec{y} := \vec{x}[\tau : \tau + M - 1]$ is obtained by taking M consecutive symbols from $\vec{x}$ and is presented as a query. This pattern may appear within $\vec{x}$ in upto L different locations $\tau_1, \tau_2, \ldots, \tau_L$ and the objective is to determine all the locations $\tau_i, \forall i \in [1 : L]$. We consider the probabilistic version where our objective is to recover the matching locations with a probability that approaches 1 as $M, N \to \infty$.

Approximate Pattern Matching: In the approximate pattern matching version, $\vec{y}$ is a noisy version of a substring, i.e., $\vec{y} = \vec{x}[\tau : \tau + L - 1] \odot \vec{b}$ where $\vec{b}$ is a noise sequence with $b[i] \in \pm 1$ such that $d_H(\vec{y}, \vec{x}[\tau : \tau + M - 1]) \leq K$. Here d_H denotes the Hamming distance, and $\odot$ represents component-wise multiplication. The objective is to determine all locations τ_i such that $d_H(\vec{y}, \vec{x}[\tau_i : \tau_i + M - 1]) \leq K$ with a probability that approaches 1 as K, M and $N \to \infty$.

Broadly speaking, these problems are relevant to many applications including text matching, audio/image matching, DNA matching in omics, searching for signatures of events within large databases and is particularly relevant now due to the interest in applications involving huge volumes of data. The presented results are most useful in the following situations - (i) the string $\vec{x}$ is available before querying and one time computations such as computing a sketch of $\vec{x}$ can be performed off line and stored, and the complexity of computing the sketch

This work was supported by the National Science Foundation under grants CCF-1320924 and EARS-1547447.

[1] Extensions to other alphabets $\mathscr{A} \subset \mathbb{R}$ are straightforward. Extension to the non-i.i.d case is also possible

of $\vec{x}$ can be ignored. Then, when queries in the form of $\vec{y}$ appear, one would like to decrease the computational complexity in searching for the string $\vec{y}$. (ii) The string $\vec{x}$ is not centrally available but parts of the string are sensed by different data collecting nodes distributively and communicated to a central server. A search query is presented to the server and the server needs to decide if the string appeared in the data sensed by one or more of the distributed nodes and the time instants where the query appeared. In this case, we would like to minimize the amount of data communicated by the nodes to the server and the computational complexity in searching for the string. Finally, the proposed approach is most useful when the query $\vec{y}$ is not a pattern that can be predicted and, therefore, creating a look-up table to quickly identify commonly-occurring patterns will not be effective.

A naive approach to searching for a substring $\vec{y}$ in $\vec{x}$ is to compute the cross correlation between $\vec{x}$ and $\vec{y}$ denoted by $\vec{r} = [r[0], r[1], \ldots, r[N-1]]$

$$r[m] = (\vec{x} * \vec{y})[m] \triangleq \sum_{i=1}^{M} x[m+i-1]y[i] \tag{1}$$

and choosing the index m that maximizes $r[m]$. This approach uses all the N samples of $\vec{x}$ and has a super-linear computational complexity of $MN = O(N^{1+\mu})$. A computationally more efficient approach uses the fact that $\vec{r}$ can be computed by taking the inverse Fourier transform of the product of the Fourier transforms of $\vec{x}$ and $\vec{y}^*[-n]$, where $\vec{y}^*[-n]$ is the conjugate of time reversed version of $\vec{y}$. Such an approach still uses all the N samples of $\vec{x}$, but reduces the computational complexity to $O(N \log N)$. Note that even though the Fourier transform of $\vec{x}$ can be precomputed, the N-point Fourier transform of $\vec{y}$ still needs to be computed resulting in the $O(N \log N)$ computational complexity.

Both the exact pattern matching problem and the approximate pattern matching problem have been extensively studied in computer science and two recent papers [1] and [2] provide a brief summary of the results. For the exact matching problem, the Rabin-Karp algorithm solves a more general problem of finding the occurrence of a set of query strings; however, the algorithm has a computational complexity that is at least linear in N. Boyer and Moore presented an algorithm in [3] for finding the first occurrence of the match (only τ_1) that has an expected complexity of $O(N/M \log N) = O(N^{1-\mu} \log N)$, whereas the worst case complexity (depending on τ_1's) can be $O(N \log N)$. For large M, the algorithm indeed has an average complexity that is sub-linear in N. More recently, it has been shown that techniques based on the Burrows-Wheeler transform can be used to solve the exact matching problem with sub-linear time complexity []. For small $|\mathscr{A}|$, this algorithm remains fairly competitive; however, the complexity increases with $|\mathscr{A}|$. Further, it cannot be extended to the approximate setting. The Boyer and Moore algorithm has been generalized to the approximate pattern matching problem in [4] with an average case complexity of $O(NK/M \log N)$, which provides a sub-linear time algorithm only when $K \ll M$. In [1], Andoni, Hassanieh, Indyk and Katabi have given the first sub-linear time algorithm with a complexity of $O(N/M^{0.359})$ even for $K = O(M)$.

II. Our Main Results and Relation to Prior Work

Theorem 1. *Assume that a sketch of $\vec{x}$ of size $O(N/M)$ can be computed and stored. Then for the exact pattern matching and approximate pattern matching (with $K = \eta M$) problems, with the number of matches L scaling as $O(N^{\lambda})$, our algorithm has*

- *a sketching function for $\vec{y}$ that computes $O(\max(M, \frac{N}{M}) \log N) = O(N^{\max(\mu, 1-\mu)} \log N)$ samples*
- *a decoder that recovers all the L matching positions with a failure probability that approaches zero asymptotically in N*
- *a computational complexity of*
  - $O(N^{\max(1-\mu, \mu+\lambda)} \log^2 N)$ *for* $\mu < 0.5$
  - $O(N^{\max(\mu, 1-\mu+\lambda)} \log^2 N)$ *for* $\mu > 0.5$

When $L < \frac{N}{M}$ (i.e. $\lambda < 1-\mu$), which is typically the interesting case, our algorithm has a sub-linear time complexity.

Proof. We present the error performance analysis of our scheme and the proof of decaying error probability in Sec. V after we describe our scheme and decoding algorithm in Sec. IV. In Sec. VI we present the sample and complexity analyses. □

Our paper is inspired by and builds on two recent works by Hassanieh *et al.* in [5] and Pawar and Ramchandran [6]. In [5], Hassanieh et al., considered the correlation function computation problem for a Global Positioning System (GPS) receiver and exploited the fact that the cross-correlation vector $\vec{\mathrm{r}}$ is a very sparse signal in the time domain and, hence, the Fourier transform of $\vec{\mathrm{r}}$ need not be evaluated at all the N points. In the GPS application, which was the focus of [5], the query $\vec{\mathrm{y}}$ corresponds to the received signal from the satellites and, hence, the length of the query was at least N. As a result, the computational complexity is still $O(N\log N)$ (still linear in N) and the only the constants were improved in relation to the approach of computing the entire Fourier transform. In a later paper by Andoni *et al.,* [1], a sub-linear time algorithm for shift finding in GPS is presented; however, this algorithm is completely combinatorial and eschews algebraic techniques such as FFT-based techniques.

There are important differences between our paper and [5], [1], [3], [2]. First and foremost, the algorithms used for pattern matching are entirely different. While their algorithms are combinatorial in nature, our algorithm is algebraic and uses signal processing and coding theoretic primitives. Secondly, the system model considered in our paper is different from the model in [5], [1], [3], [2] in that we allow for preprocessing or creating a sketch of the data $\vec{\mathrm{x}}$. Our algorithm exploits this fact and results in a computational complexity $O(N/M)$ which is better than that in [1] for the approximate pattern matching problem. Finally, we also consider the problem of finding all matches of the pattern $\vec{\mathrm{y}}$ instead of looking for only one match. In [6], Pawar and Ramchandran present an algorithm based on aliasing and the peeling decoder for computing the Fourier transform of a signal with noisy observations for the case when the Fourier transform is known to be sparse. This algorithm has a complexity of $O(N\log N)$ and they do not consider the pattern matching problem. Our algorithm is similar to that of Pawar and Ramchandran's algorithm with one very important distinction. We modify their algorithm to exploit the fact that the peak of the correlation function of interest is always positive. This modification is key in computing the Sparse Inverse Discrete Fourier Transform (SIDFT) with sub-linear time complexity of $O(N^{1-\alpha}\log N)$, $0<\alpha\leq 1$. Thus, one of the main contributions of this paper is to show that signal processing and coding theoretic primitives, i.e., Pawar and Ramchandran's algorithm with some modifications can be used to solve the pattern matching algorithm in sub-linear time.

III. Notations

This table below will introduce the notations we use in this paper. We denote signals and vectors using the standard vector notation of arrow over the letter, time domain signals using lowercase letters and the frequency domain signals using uppercase letter. For example $\vec{\mathrm{x}} = (x[0], x[1], \ldots, x[N])$ denotes a time domain signal with i^{th} time component denoted by $x[i]$, and $\vec{X} = \mathscr{F}_N\{\vec{\mathrm{x}}\}$ denotes the N-point Fourier coefficients of $\vec{\mathrm{x}}$. We denote matrices using boldface upper case letters. We denote the set $\{0,1,2\cdots,N-1\}$ by $[N]$. Note that in Table. III we suppose

TABLE I
PARAMETERS AND VARIOUS QUANTITIES INVOLVED IN DESCRIBING THE ALGORITHM

Symbol	*Meaning*
N	Size of the string or database in symbols
$M = N^{\mu}$	Length of the query in symbols
$L = N^{\lambda}$	Number of matches
K	$\max_{\tau} d_H(\vec{\mathrm{x}}[\tau:\tau+M-1], \vec{\mathrm{y}})$
η	$\frac{K}{M}$
$G = N^{\gamma}$	Number of blocks
$\tilde{N} = N^{1-\gamma}$	Length of one block
$f_i = N^{\alpha}$	Length of smaller point IDFT at each branch
$n_i = N/f_i$	Sub-sampling parameter
B	Number of shifts also referred to as branches
d	Number of stages in the FFAST algorithm

the values of M and L to be N^μ and N^λ respectively but it is not necessary that these parameters have this form. All our results can be extended straight forwardly to any M and L as long as $M = O(N^\mu)$ and $L = O(N^\lambda)$.

IV. Description Of The Algorithm

In this section, given the input string $\vec{\mathrm{x}}$ and the query string $\vec{\mathrm{y}}$, we describe our algorithm that finds the matching positions $\mathscr{T} := \{\tau_1, \tau_2, \cdots \tau_L\}$ with sample and time complexities that are sub-linear in N. The main idea here is that the correlation vector $\vec{\mathrm{r}}$ is sparse (upto some noise) with dominant peaks at L matching positions denoted by $\mathscr{T}$ and noise components at $N-L$ positions where the strings do not match.

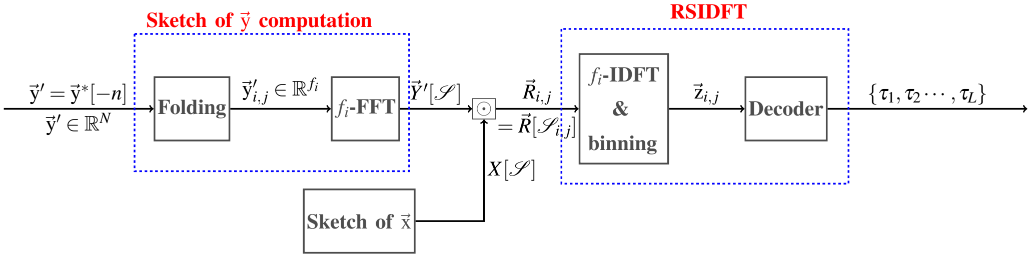
Sketch of $\vec{\mathrm{y}}$ computation
RSIDFT
$\vec{\mathrm{y}}' = \vec{\mathrm{y}}^*[-n]$
$\vec{\mathrm{y}}' \in \mathbb{R}^N$
Folding
$\vec{\mathrm{y}}'_{i,j} \in \mathbb{R}^{f_i}$
f_i-FFT
$\vec{Y}'[\mathscr{S}]$
$\odot$
$X[\mathscr{S}]$
Sketch of $\vec{\mathrm{x}}$
$\vec{R}_{i,j}$
$= \vec{R}[\mathscr{S}_{i,j}]$
f_i-IDFT & binning
$\vec{\mathrm{z}}_{i,j}$
Decoder
$\{\tau_1, \tau_2 \cdots, \tau_L\}$

Fig. 1.

Consider the correlation signal $\vec{\mathrm{r}}$ in the case of exact matching:

$$r[m] = \begin{cases} M, & \text{if } m \in \mathscr{T} \\ n_m, & m \in [N] - \mathscr{T} \end{cases} \tag{2}$$

where n_m is the noise component that is induced due to correlation of two i.i.d. sequence of random variables each taking values from $\mathscr{A} := \{+1, -1\}$. The sparse vector $\vec{\mathrm{r}}$ can be computed indirectly using Fourier transform approach as shown below:

$$\vec{\mathrm{r}} = \underset{\text{I}}{\mathscr{F}_N^{-1}} \; \{\underset{\text{II}}{\mathscr{F}_N\{\vec{\mathrm{x}}\}} \odot \; \underset{\text{III}}{\mathscr{F}_N\{\vec{\mathrm{y}}'\}}\} \tag{3}$$

where $\mathscr{F}_N\{\cdot\}$ and $\mathscr{F}_N^{-1}\{\cdot\}$ refer to N-point discrete Fourier transform and its inverse respectively, $\odot$ is the point-wise multiplication operation and $y'[n] = y^*[-n]$. Fig. 1 presents a notional schematic of our Algorithm. As evident from Eq. (3), our algorithm for computing $\vec{\mathrm{r}}$ consists of three stages:

- Computing the sketch(FFT) $\vec{X}$ of $\vec{\mathrm{x}}$
- Computing the sketch(FFT) $\vec{Y}$ of $\vec{\mathrm{y}}$
- Computing the IDFT of $\vec{R}$ given $\vec{X}$ and $\vec{Y}$

A. *Sparse Inverse Discrete Fourier Transform*

In this section we present Robust Sparse Inverse Discrete Fourier Transform(RSIDFT) scheme that exploits sparsity in the cross-correlation signal $\vec{\mathrm{r}}$ and efficiently recovers it's L dominant coefficients. The architecture of RSIDFT is similar to that of the FFAST scheme proposed in [6], but the decoding algorithm has some key modifications to handle the noise model induced in this problem. We will see in Sec. IV-B how the sketches $\vec{X}$ and $\vec{Y}$ are computed efficiently but for this section we will focus only on the recovery of the sparse coefficients in $\vec{\mathrm{r}}$ given $\vec{X}$ and $\vec{Y}$.

Consider the RSIDFT framework shown in Fig. 2. Let $\vec{R} = \vec{X} \odot \vec{Y}'$ be the DFT of the cross-correlation signal of $\vec{\mathrm{x}}$ and $\vec{\mathrm{y}}$. The framework consists of *d-stages* with each stage correpsonding to a sub-sampling factor of $\frac{N}{f_i}$. The design scheme for choosing parameters $\{f_1, f_2, \ldots, f_d\}$ for various values of μ such that $f_i|N$ and $f_i = N^\alpha + O(1) \forall\ i \in [d]$ are given in IV-A2. In each stage, there are B *branches* with shifts from the set $\{s_1, s_2, \cdots s_B\}$, where $s_1 = 0$ in the first branch and the rest are chosen randomly from $[N]$.

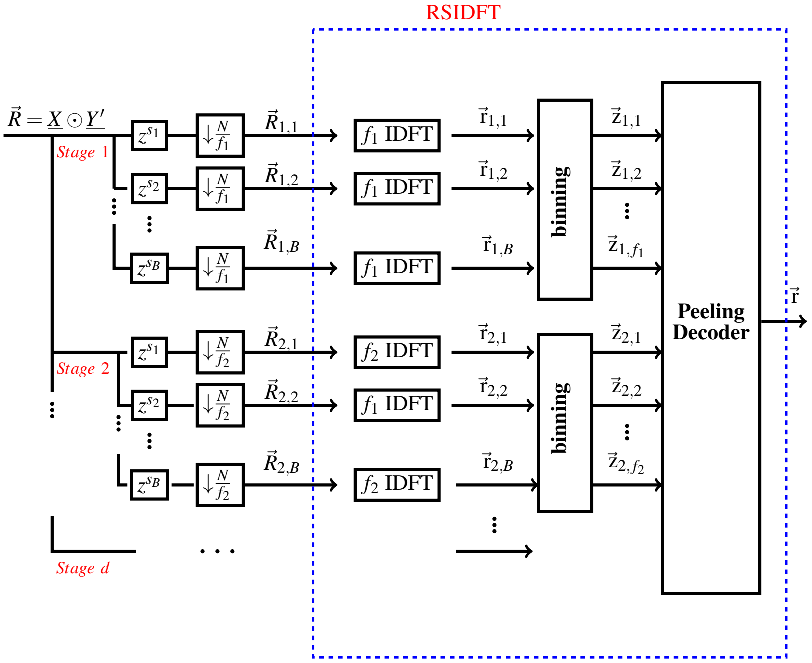

Fig. 2. RSIDFT Framework to compute inverse Fourier Transform of a signal $\vec{R}$ that is sparse in time domain.

Given the input $\vec{R}$, in branch j of i^{th} stage of RSDIFT, referred to as *branch* (i,j) for simplicity, RSIDFT sub-samples the signal $\vec{R}$ at

$$\mathscr{S}_{i,j} := \{s_j,\ s_j + g_i,\ s_j + 2g_i,\ \cdots s_j + (f_i - 1)g_i\}, \quad i \in [d], j \in [B] \tag{4}$$

where $g_i := \frac{N}{f_i}$ to obtain $\vec{R}_{i,j} := \vec{R}[\mathscr{S}_{i,j}]$. The sub-sampling operation is followed by a f_i-point IDFT in each branch of stage-i to obtain $\vec{\mathrm{r}}_{i,j}$. Notice that $\vec{\mathrm{r}}_{i,j}$ is an aliased version of $\vec{\mathrm{r}}$ due to the property that sub-sampling in Fourier domain is equivalent to aliasing in time domain.

Let $\vec{\mathrm{z}}_{i,k} \in \mathbb{R}^B$, for $k \in [f_i]$, be the kth *binned* observation vector of stage-i formed by stacking $\vec{\mathrm{r}}_{i,j}[k], j \in [B]$, together as a vector i.e.

$$\vec{\mathrm{z}}_{i,k} = \begin{bmatrix} r_{i,1}[k] \\ r_{i,2}[k] \\ \vdots \\ r_{i,B}[k] \end{bmatrix}$$

Using the properties of Fourier transfrom, we can write the relationship between the observation vectors $\vec{\mathrm{z}}_{i,j}$ at bin (i,j) and sparse vector to be estimated $\vec{\mathrm{r}}$ as:

$$\vec{\mathrm{z}}_{i,k} = \mathbf{W}_{i,k} \times \begin{bmatrix} r[k + (0)f_i] \\ r[k + (1)f_i] \\ \vdots \\ r[k + (g_i - 1)f_i] \end{bmatrix} \tag{5}$$

where we refer to $\mathbf{W}_{i,k}$ as the sensing matrix at bin (i,k) and is defined as

$$\mathbf{W}_{i,k} = \left[\vec{\mathrm{w}}^{k}, \vec{\mathrm{w}}^{k+f_i}, \ldots, \vec{\mathrm{w}}^{k+(g_i-1)f_i}\right] \text{ where } \vec{\mathrm{w}}^{k} = \begin{bmatrix} e^{\frac{j2\pi k s_1}{N}} \\ e^{\frac{j2\pi k s_2}{N}} \\ \vdots \\ e^{\frac{j2\pi k s_B}{N}} \end{bmatrix} \tag{6}$$

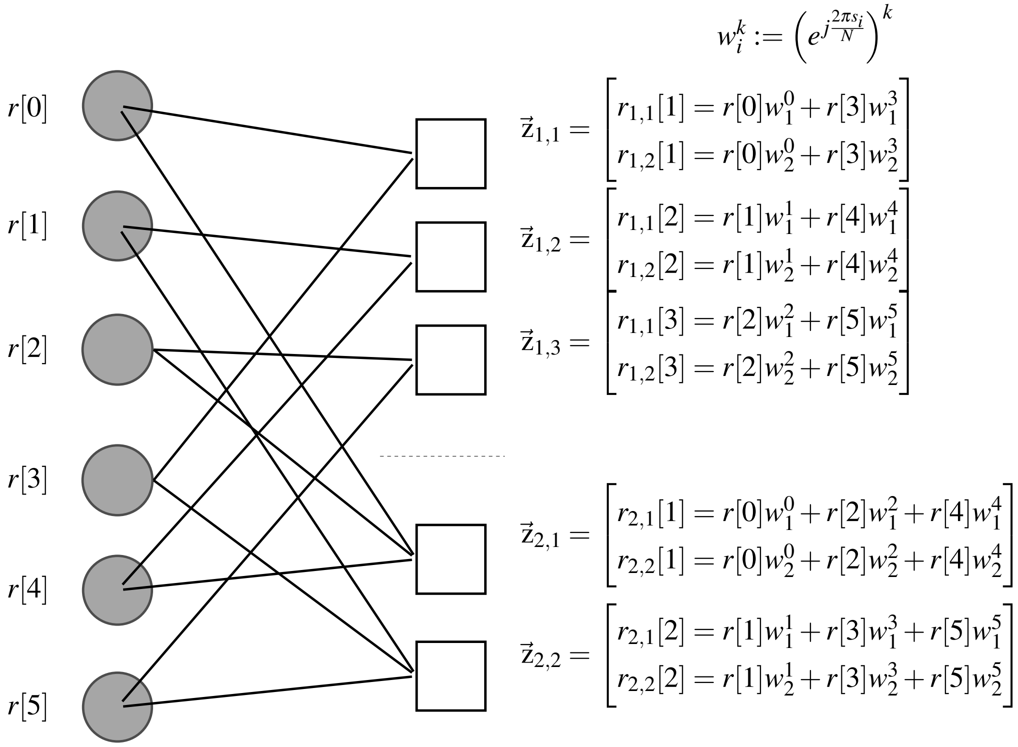
$w_i^k := \left(e^{j\frac{2\pi s_i}{N}}\right)^k$
$r[0]$
$r[1]$
$r[2]$
$r[3]$
$r[4]$
$r[5]$
$\vec{\mathrm{z}}_{1,1} = \begin{bmatrix} r_{1,1}[1] = r[0]w_1^0 + r[3]w_1^3 \\ r_{1,2}[1] = r[0]w_2^0 + r[3]w_2^3 \end{bmatrix}$
$\vec{\mathrm{z}}_{1,2} = \begin{bmatrix} r_{1,1}[2] = r[1]w_1^1 + r[4]w_1^4 \\ r_{1,2}[2] = r[1]w_2^1 + r[4]w_2^4 \end{bmatrix}$
$\vec{\mathrm{z}}_{1,3} = \begin{bmatrix} r_{1,1}[3] = r[2]w_1^2 + r[5]w_1^5 \\ r_{1,2}[3] = r[2]w_2^2 + r[5]w_2^5 \end{bmatrix}$
$\vec{\mathrm{z}}_{2,1} = \begin{bmatrix} r_{2,1}[1] = r[0]w_1^0 + r[2]w_1^2 + r[4]w_1^4 \\ r_{2,2}[1] = r[0]w_2^0 + r[2]w_2^2 + r[4]w_2^4 \end{bmatrix}$
$\vec{\mathrm{z}}_{2,2} = \begin{bmatrix} r_{2,1}[2] = r[1]w_1^1 + r[3]w_1^3 + r[5]w_1^5 \\ r_{2,2}[2] = r[1]w_2^1 + r[3]w_2^3 + r[5]w_2^5 \end{bmatrix}$

Fig. 3. Example of a Tanner graph formed in a RSIDFT framework with system parameters being $d = 2$, $B = 2$, $N = 6$, $f_1 = 2$ and $f_2 = 3$. The variable nodes (colored gray circles) represent the cross-correlation vector $\vec{\mathrm{r}}$ and the bin nodes (uncolored white boxes) represent the binned observation vector $\vec{\mathrm{z}}_{i,k}$. The figure also illustrates the relationship between $\vec{\mathrm{z}}_{i,k}$ and $\vec{\mathrm{r}}$.

In Fig. 3 we represent the relation between the set of observation vectors $\{\vec{\mathrm{z}}_{i,k}, i \in [1:d], k \in [f_i]\}$ and $\vec{\mathrm{r}}$ via an example using Tanner graph. The nodes on the left, which we refer to as *variable nodes*, represent the N elements of vector $\vec{\mathrm{r}}$. Similarly the nodes on the right, which we refer to as *bin nodes*, represent the $\sum_{i \le d} f_i$ sub-sensing signals. We will now describe the decoding algorithm which takes the set of observation vectors $\{\vec{\mathrm{z}}_{i,k}, i \in [1:d], k \in [f_i]\}$ at df_i bins as input and estimates the L-sparse $\vec{\mathrm{r}}$.

1) Decoder: Observe from the Tanner graph that the degree of each variable node is d and that of each bin node at stage i is n_i. A variable node is referred to as non-zero if it corresponds to a matching position and as zero if it corresponds to a non-matching position. Note that even though the cross-correlation vector value corresponding to a non-matching position is not exactly zero but some negligible noise value we refer to them as zero variable nodes for simplicity. We refer to a bin node as *zero-ton* (or $\mathcal{H}_z$) if the number of non-zero variable nodes connected to the bin node is zero. The *singleton ($\mathcal{H}_s$), double-ton ($\mathcal{H}_d$) and multi-ton ($\mathcal{H}_m$)* bin nodes are defined similarly where the number of non-zero variable nodes connected are one, two and greater than two, respectively. The peeling decoder has the following three steps in the decoding process.

Bin Classification: In this step a bin node is classified as a zero-ton or a singleton or a multi-ton. At bin (i,j) the classification is done based on comparing the first observation $z_{i,j}[1]$, which corresponds to zero shift, with a predefined threshold. For $z_{i,j}[1] = z$, the classification at bin (i, j), $\widehat{\mathcal{H}}_{i,j}$, can be written as follows:

$$\widehat{\mathcal{H}}_{i,j} = \begin{cases} \mathcal{H}_z & z/M < \gamma_1 \\ \mathcal{H}_s & \gamma_1 < z/M < \gamma_2 \\ \mathcal{H}_d & \gamma_2 < z/M < \gamma_3 \\ \mathcal{H}_m & z/M > \gamma_3 \end{cases} \tag{7}$$

where $(\gamma_1, \gamma_2, \gamma_3) = (\frac{1-2\eta}{2}, \frac{3-4\eta}{2}, \frac{5-6\eta}{2})$. Note that for the case of exact matching $\eta = 0$.

Singleton decoding: If a bin node (i, j) is classified as a singleton, we need to identify the position of the non-zero variable node connected to it. This is done by correlating the observation vector $\vec{\mathrm{z}}_{i,j}$ with each column of the sensing matrix $\mathbf{W}_{i,j} := [\vec{\mathrm{w}}^{j}, \vec{\mathrm{w}}^{j+g_i}, \ldots, \vec{\mathrm{w}}^{j+(f_i-1)g_i}]$ and choose the index that maximizes the correlation value.

$$\hat{k} = \underset{k \in \{j+lg_i\}}{\arg\max}\ \vec{\mathrm{z}}_{i,j}^{\dagger} \vec{\mathrm{w}}^{k}$$

where † denotes the conjugate transpose. The value of the variable node connected to the singleton bin is estimated as follows:

$$\hat{r}[\hat{k}] = M(1-\eta).$$

Note that for the case of exact matching we know the value to be exactly equal to M which is our estimate in the case of exact matching $(\eta = 0)$. But in the case of approximate matching the actual value of $r[k] \in \{M(1-2\eta), \ldots, M-1, M\}$ and our estimate $\hat{r}[\hat{k}] = M(1-\eta)$, the mid-point of the range, although is only a loose approximate it would suffice for our purposes as we are mainly interested in recovering only the positions of matches i.e. the indices of the sparse coefficients in $\vec{\mathrm{r}}$.

Peeling Process: The peeling based decoder we employ consists of finding a singleton bin, then identify the single non-zero variable node connected to the bin, decoding it's value and removing (*peeling off*) it's contribution from all the bin nodes connected to that variable node. The main idea behind this decoding scheme is that hopefully the peeling off operation induces at least one more singleton bin and thus the process of peeling off can be repeated iteratively. Although the main idea is similar for exact matching and the approximate matching scenarios, there are some subtle differences in their implementation.
Exact Matching: In the case of exact matching, we remove the decoded variable node's contribution from all the bin nodes it is connected to.
Approximate Matching: In this case we remove the decoded variable node's contribution only from bins that are originally a singleton or a double-ton. We do not alter the bins which are classified to be multi-tons with degree more than two.

Note: The differences in peeling implementation for exact and approximate matching cases is because unlike exact matching the approximate matching may result in non-zero error (e_1) between the actual correlation value $r[k]$ and the estimate $\hat{r}[\hat{k}]$, i.e. $e_1 = |r[k] - \hat{r}[\hat{k}]|$ with $0 \leq e_1 \leq \eta M$. This may lead to error propagation if we use the same decoding scheme as in exact matching. Hence to overcome this problem we impose a constraint on the type of bin nodes participating in the peeling process.

We present the overall recovery algorithm, which comprises of *bin classification, singleton decoding* and *peeling process*, in the Algorithm. 1 pseudo-code. Note that $\mathcal{N}(k)$ for variable node k denote it's neighborhood i.e. the set of bins connected to k^{th} variable node.

2) Choosing f_i and α for various μ: For a given value of μ, we will describe how to choose the parameters d and f_i.
Exact Matching Find $d \in \mathbb{N}\backslash\{1,2\}$ such that $\mu \in (\frac{1}{d}, \frac{1}{d-1}] \cup [1-\frac{1}{d-1}, 1-\frac{1}{d})$
Approximate Matching: If $\mu \in (\frac{1}{8}, \frac{7}{8})$, choose $d = 8$. Else find $d \geq 8$ such that $\mu \in (\frac{1}{d}, \frac{1}{d-1}) \cup (1-\frac{1}{d-1}, 1-\frac{1}{d})$

- Find a factorization for signal length $N = \prod_{i=0}^{d-1} \mathcal{P}_i$ such that the set of integers $\{\mathcal{P}_0, \mathcal{P}_1, \ldots, \mathcal{P}_{d-1}\}$ are pairwise co-prime and all the $\mathcal{P}_i$ are approximately equal
- More precisely, let $\mathcal{P}_i = \mathbf{F} + O(1)\ \forall i$ for some value $\mathbf{F}$and choose $f_i = N/\mathcal{P}_i$
- Note that our choice of design parameters results in $\alpha = 1 - \frac{1}{d}$ (recall $f_i \approx N^{\alpha}$)

Algorithm 1 Peeling based recovery algorithm

```
Compute $\widehat{\mathcal{H}}_{i,j}$ for $i \in [d], j \in [f_i]$. See Eqn. (7)          Bin Classification
while $\exists\ i,j : \widehat{\mathcal{H}}_{i,j} = \mathcal{H}_s$, do
    $(\hat{k}, \hat{r}[\hat{k}])$ =Singleton-Decoder($\vec{z}_{i,j}$)
    Assign $\hat{r}[\hat{k}]$ to $\hat{k}^{\text{th}}$ variable node
    for $(i_0, j_0) \in \mathcal{N}(\hat{k})$ do
        $\vec{z}_{i_0,j_0} \leftarrow \vec{z}_{i_0,j_0} - \hat{r}[\hat{k}]\vec{w}^{\hat{k}}$                                   Exact Matching
        $\vec{z}_{i_0,j_0} \leftarrow \vec{z}_{i_0,j_0} - \hat{r}[\hat{k}]\vec{w}^{\hat{k}}$    only if $\widehat{\mathcal{H}}_{i_0,j_0} = \mathcal{H}_s$ or $\mathcal{H}_d$    Approximate Matching
        Re-do the bin classification for $(i_0, j_0)$ and compute $\widehat{\mathcal{H}}_{i_0,j_0}$
```

Algorithm 2 Singleton-Decoder

```
Input: $\vec{z}_{i,j}$
Output: $(\hat{k}, \hat{r}[\hat{k}])$
Estimate singleton index to be $\hat{k} = \arg\max_{k \in \{j + l n_i\}} \vec{z}_{i,j}^{\dagger} \vec{w}^{k}$
Estimate the value to be:
    $\hat{r}[\hat{k}] = \begin{cases} M & \text{Exact Matching case} \\ M - K & \text{Approximate Matching case} \end{cases}$
```

To summarize our choice of design parameters: In both the exact and approximate matching cases, for any $0 < \mu < 1$, we choose the down-sampling factors f_i to be approximately equal to N^{α} in all the d-stages where $\alpha = \frac{d-1}{d}$ satisfies:

$$\alpha > \max(\mu, 1 - \mu). \tag{8}$$

B. Sketches of $\vec{X}$ and $\vec{Y}$

As we have already seen in Sec. IV-A the RSDIFT framework requires the values of $\vec{R}(= \vec{X} \odot \vec{Y})$ only at indices $\mathscr{S}$ or in other words we need $\vec{X}$ and $\vec{Y}$ only at the indices $\mathscr{S}$. We assume that the sketch of $\vec{x}$, $\vec{X}[\mathscr{S}] = \{X[i], i \in \mathscr{S}\}$ is pre-computed and stored in a database.

Computing the sketch of $\vec{y}$*:* : For every new query $\vec{y}$, only $\{\vec{Y}'[\mathscr{S}_{i,j}], i \in [d], j \in [B]\}$ needs to be computed where the subsets $\mathscr{S}_{i,j}$ are defined in Eq. (4). Naively, the FFT algorithm can be used to compute $\vec{Y}'$ and the required subset of coefficients can be taken but this is inefficient and would be of $O(N \log N)$ complexity. Instead, we observe that $\vec{Y}'[\mathscr{S}_{i,j}]$ is $\vec{Y}'$ shifted by s_j and sub-sampled by a factor of g_i. Thus for a given (i, j) this corresponds to, in time domain, a point-wise multiplication by $[1, w_{s_j}, w_{s_j}^2, \ldots, w_{s_j}^{N-1}]$ followed by *folding* the signal into f_i number of length $\frac{N}{f_i}$ signals and adding them up resulting in a single length-n_i signal denoted by $\vec{y}'_{i,j}$. Formally the *folding* operation can be described as follows:

$$\vec{y}'_{i,j} = \sum_{m=0}^{g_i - 1} \vec{y}'[mf_i : (m+1)f_i - 1] \odot [w_{s_j}^{mf_i}, w_{s_j}^{mf_i+1}, \ldots, w_{s_j}^{(m+1)f_i - 1}], \qquad w_{s_j} = e^{-\frac{j2\pi s_j}{N}}. \tag{9}$$

Taking n_i-point DFT of $\vec{y}'_{i,j}$ produces $\vec{Y}'[\mathscr{S}_{i,j}]$ i.e. $\vec{Y}'$ sampled only at $\mathscr{S}_{i,j}$ indices which is what we need in branch (i, j). To obtain all the samples in $\mathscr{S}$ required for the RSDIFT framework, the *folding* technique followed by a IDFT needs to be carried out for each (i, j), for $i \in [d], j \in [B]$, a total of dB times $N^{1-\alpha}$-point DFT.

V. Performance Analysis

In this section, we will analyze the overall probability of error involved in finding the correct matching positions. This can be done by analyzing the following three error events independently and then using a union bound to bound the total probability of error.

- $\mathscr{E}_1$-*Bin Classification*: Event that a bin is wrongly classified
- $\mathscr{E}_2$-*Position Identification*: Given a bin is correctly identified as a singleton, event that the position of singleton is identified incorrectly
- $\mathscr{E}_3$-*Peeling Process*: Given the classification of all the bins and the position identification of singletons in each iteration is accurate, event that the peeling process fails to recover the L significant correlation coefficients

A. Bin Classification

Lemma 2. *The probability of bin classification error at any bin (i, j) can be upper bounded by*

$$\mathbb{P}[\mathscr{E}_1] \leq 6e^{-\frac{N^{\mu+\alpha-1}(1-6\eta)^2}{16}}$$

Proof.

$$\begin{aligned}\mathbb{P}[\mathscr{E}_1] &= \mathbb{P}[\mathcal{H}_z]\mathbb{P}[\mathscr{E}_1|\widehat{\mathcal{H}}_{i,j} = \mathcal{H}_z] \ + \quad \mathbb{P}[\mathcal{H}_s]\mathbb{P}[\mathscr{E}_1|\widehat{\mathcal{H}}_{i,j} = \mathcal{H}_s] \ + \quad \mathbb{P}[\mathcal{H}_d]\mathbb{P}[\mathscr{E}_1|\widehat{\mathcal{H}}_{i,j} = \mathcal{H}_d \cup \mathcal{H}_m] \\ &\leq \mathbb{P}[\mathscr{E}_1|\widehat{\mathcal{H}}_{i,j} = \mathcal{H}_z] \ + \quad \mathbb{P}[\mathscr{E}_1|\widehat{\mathcal{H}}_{i,j} = \mathcal{H}_s] \ + \quad \mathbb{P}[\mathscr{E}_1|\widehat{\mathcal{H}}_{i,j} = \mathcal{H}_d \cup \mathcal{H}_m] \\ &\leq e^{-\frac{N^{\mu-\alpha}(1-2\eta)^2}{8}} + 2e^{-\frac{N^{\mu-\alpha}(1-4\eta)^2}{16}} + 2e^{-\frac{N^{\mu+\alpha-1}(1-6\eta)^2}{16}} + e^{-\frac{N^{\mu+\alpha-1}(1-6\eta)^2}{16}} \\ &\leq 6e^{-\frac{N^{\mu+\alpha-1}(1-6\eta)^2}{16}}\end{aligned}$$

where the inequalities in the third line are due to Lemmas 8, 9, 10 and 11 respectively provided in Appendix B. □

B. Position Identification

Lemma 3. *Given that a bin (i, j) is correctly classified as a singleton, the probability of error in identifying the position of the non-zero variable node can be upper bounded by*

$$\mathbb{P}[\mathscr{E}_2|\overline{\mathscr{E}}_1] \leq \exp\left\{-\frac{N^{\mu+\alpha-1}(1-2\eta)^2(c_1^2-1)}{8(c_1^2+1)}\right\} + \exp\left\{-\frac{N^{\mu+\alpha-1}\ (c_1(1-2\eta)-1)^2}{8(1+c_1^2)}\right\}$$

Proof. The detailed proof is provided in Appendix C. □

C. Peeling Process

d	2	3	4	5	6	7	8	9
η	1.000	0.4073	0.3237	0.2850	0.2616	0.2456	0.2336	0.2244
$d\eta$	2.000	1.2219	1.2948	1.4250	1.5696	1.7192	1.8688	2.0196

TABLE II
CONSTANTS FOR VARIOUS ERROR FLOOR VALUES

To analyze the peeling process alone independently, we refer to a *oracle based peeling decoder* which has the accurate classification of all the bins and can accurately identify the position of the singleton given a singleton bin at any iteration. In other words, oracle based peeling decoder is the peeling part of our decoding scheme but with the assumption that the bin classification and position identification are carried out without any error.

Lemma 4 (Exact Matching)**.** *For the exact matching case, choose $F^{d-1} = \eta N^{\alpha}$ where η is chosen as given in Table. II. Then the oracle based peeling decoder:*

- *successfully uncovers all the L matching positions if $L = \Omega(N^{\alpha})$ and $L \leq N^{\alpha}$, with probability at least $1 - O(1/N^{\frac{1}{d}})$*

- *successfully uncovers all the L matching positions, if $L = o(N^{\alpha})$, with probability at least $1 - e^{-\beta \varepsilon_1^2 N^{\alpha/(4l+1)}}$ for some constants $\beta, \varepsilon_1 > 0$ and $l > 0$.*

Proof. We borrow this result from Pawar and Ramchandran's [6]. Although our RSDIFT framework and their robust-FFAST scheme have three main differences:

- We are computing smaller IDFT's to recover a sparse bigger IDFT whereas in [6] the same is true for DFT instead of IDFT
- In the decoding scheme: Our problem model is such that the sparse components of the signal space has only positive amplitude and thus our bin processing part (bin classification and position identification) is different when compared to [6]
- The sparsity of the signal L to be recovered is exactly known in the case of [6] whereas we have no information about L not even the order with which the quantity scales in N

Irrespective of these two differences, the Tanner graph representation of the framework and the peeling part of the decoder are identical to that of the robust-FFAST scheme. And thus the limits of the *oracle based peeling decoder* for our scheme, which assumes the bin processing is accurate thus making the peeling decoder analysis independent of the bin decoder and hence will be identical to that of the peeling part of the decoding scheme in robust-FFAST. With respect to the third difference, in robust-FFAST scheme the authors choose $F^{d-1} = \eta k$ where k is the sparsity of the signal (which is assumed to be known) and show the first assertion of the lemma. And it was also shown that upto a constant fraction $(1-\varepsilon)$ of k-variables node can be recovered with exponential probability. But in our framework, if $L = o(N^{\alpha})$, then this result translates to recovering all the L non-zero variable nodes with an exponentially decaying failure probability. □

In any iteration, given a singleton bin, the peeling process, in the case of approximate matching, runs the Singleton-Decoder algorithm on the bin only if it was either originally a singleton or originally a double-ton with one of the variable nodes being peeled off already. This is in contrast to the exact matching case where the peeling decoder runs the Singleton-Decoder on the bin irrespective of it's original degree. Hence we need to analyze the oracle based peeling decoder for the approximate matching case separately compared to the exact matching case.

Lemma 5 (Approximate Matching)**.** *For the approximate matching case, choose parameter $d \geq 8$ as described in Sec. IV-A2 and $F^{d-1} = 0.7663N^{\alpha}$. Then the oracle based peeling decoder:*

- *successfully uncovers all but a small fraction $\varepsilon = 10^{-3}$ of the L matching positions, if $L = \Omega(N^{\alpha})$ and $L \leq N^{\alpha}$ with exponentially decaying failure probability*
- *successfully uncovers all the L matching positions, if $L = o(N^{\alpha})$, with probability at least $1 - e^{-\beta \varepsilon_1^2 N^{\alpha/(4l+1)}}$ for some constants $\beta, \varepsilon_1 > 0$ and $l > 0$.*

Proof. As mentioned earlier the key difference in the approximate matching case is peeling off variable nodes from only singleton and double-tons. An identical peeling decoder is used and analyzed in the problem of group testing [7] by Lee, Pedarsani and Ramchandran which the authors refer to as SAFFRON scheme. In SAFFRON the authors claim that for a graph ensemble which has a regular degree of d on the variable nodes and a Poisson degree distribution on the bins, this peeling decoder with a left degree of $d = 8$ and a total number of bins atleast equal to $6.13k$ recovers atleast $(1-\varepsilon)$ fraction of the k non-zero variable nodes with exponentially decaying probability. Note that $\frac{6.13}{8} \approx 0.7663$ is approximately the number of bins per stage for $d = 8$. We also leverage the result from [6] that the Tanner graph representation of the robust-FFAST (or equivaently RSDIFT framework) has a Poisson degree distribution on the bins. Combining these two results gives us the required results. □

VI. Sample and Computational Complexity

In this section, we will analyze the sample complexity which is the number of samples we access from the sketch of the signal $\vec{x}$ stored in the database and the computational complexity as a function of the system parameters.

A. Sample Complexity

In each branch of the RSDIFT framework we down-sample the N samples by a factor of $\frac{N}{f_i}$ to get $f_i \approx N^{\alpha}$ samples. As mentioned we repeat this for a random shift in each branch for $B = O(\log N)$ branches in each stage thus resulting in a total of $O(N^{\alpha} \log N)$ samples per block per stage. We repeat this for $d = \frac{1}{1-\alpha}$ such stages resulting in a total of $dN^{\alpha} \log N$ samples per block. So, the total number of samples is given by

$$\begin{aligned}\text{Total \# of samples required (S)} &= O\left(dN^{\alpha} \log N\right) \\ &= O(N^{\max(\mu, 1-\mu)} \log N)\end{aligned}$$

B. Computational Complexity

As described in Eq. 3, the computation of $\vec{\mathrm{r}}$ involves three steps:

1) Operation - I: Computing $\mathscr{F}_N^{-1}\{\vec{\mathrm{x}}\}$ involves two parts: RSIDFT framework and the decoder. The RSIDFT framework involves computing smaller f_i point IDFTs, which takes approximately $O(N^{\alpha} \log N^{\alpha})$ computations in each branch. For a total of dB branches, we get a complexity of $O(dBN^{\alpha} \log N^{\alpha})$. In the decoding process, the dominant computation is from position identification. Each position identification process involves correlating the observation vector of length B with $\frac{N}{f_i} \approx N^{1-\alpha}$ column vectors, which amounts to $BN^{1-\alpha}$ computations. There will a maximum of dL such position identifications, which gives a complexity of $O(dLBN^{1-\alpha})$. So, the total number of computations involved in this step, C_I, is given by

$$\begin{aligned}C_I &= d\ B \left(\underbrace{O(N^{\alpha}\ \log N^{\alpha})}_{\text{Shorter IFFTs /block/stage}} + L\ N^{1-\alpha} \right) \\ &= \begin{cases} O(\max(N^{1-\mu} \log^2 N, N^{\mu+\lambda} \log N)), & \mu < 0.5 \\ O(\max(N^{\mu} \log^2 N, N^{1-\mu+\lambda} \log N)), & \mu > 0.5 \end{cases}\end{aligned}$$

2) Operation - II: Since we assume that the sketch of database $\vec{\mathrm{x}}$, $\mathscr{F}_N\{\vec{\mathrm{x}}\}$, is pre-computed, we do not include this in computational complexity.
3) Operation - III:
As described in Sec. IV-B, in each branch (i, j), we use a folding based technique to compute the sketch of $\vec{\mathrm{y}}$, $\mathscr{F}_N\{\vec{\mathrm{y}}\}$ at points in the set $\mathscr{S}_{i,j}$. The folding technique involves two steps: folding and adding (aliasing) which has a complexity of $O(M)$ computations , and computing f_i-point IDFTs that takes $O(N^{\alpha} \log N^{\alpha})$ computations. So, for a total of dB branches the number of computations in this step is given by

$$\begin{aligned}C_{III} &= dB\ (\underbrace{N^{\mu}}_{\text{\# of additions (aliasing)}} + \underbrace{N^{\alpha}\ \log N^{\alpha}}_{\text{Shorter FFTs}}) \\ &= O(N^{\max(\mu, 1-\mu)} \log^2 N))\end{aligned}$$

Note: Folding and adding, for each shift, involves adding $N^{1-\alpha}$ vectors of length N^{α}. We know that the length of the query is $M = N^{\mu}$, i.e., the number of non-zero elements in $\vec{\mathrm{y}}$ (zero-padded version of the query) is M and hence we only need to compute M additions instead of length of the vector N.

Total number of computations, $C = C_I + C_{III}$, is given by

$$C = \begin{cases} O(\max(N^{1-\mu} \log^2 N, N^{\mu+\lambda} \log N)), & \mu < 0.5 \\ O(\max(N^{\mu} \log^2 N, N^{1-\mu+\lambda} \log N)), & \mu > 0.5 \end{cases}$$

APPENDIX A
CHERNOFF BOUNDS

Lemma 6 (Hoeffding inequality for bounded random variables)**.** *Let $X_1, X_2, \ldots, X_n$ be a sequence of independent random variables such that $X_i \in [a_i, b_i]$ and $E[X_i] = \mu_i$ for all i. Then for any $\delta > 0$:*

$$\textbf{\textit{Upper Tail}}: \ \mathbb{P}\left[\sum (X_i - \mu_i) \geq \delta\right] \leq \exp\left\{-\frac{2\delta^2}{\sum (b_i - a_i)^2}\right\}$$
$$\textbf{\textit{Lower Tail}}: \ \mathbb{P}\left[\sum (X_i - \mu_i) \leq -\delta\right] \leq \exp\left\{-\frac{2\delta^2}{\sum (b_i - a_i)^2}\right\}$$

Lemma 7 (Tail bounds for noise terms)**.** *Let us consider $r[\theta_0], r[\theta_1], \ldots, r[\theta_{g_i-1}]$ where $\theta_j = \theta_0 + jf_i$ and $\theta_j \notin \{\tau_1, \ldots, \tau_L\}$ is not one of the matching positions for any j. Then for any $\delta > 0$:*

$$\textbf{\textit{Upper Tail}}: \ \mathbb{P}\left[\left(\frac{1}{M}\sum_{j=0}^{g_i-1}\sum_{k=0}^{M-1} x[\theta_j + k]y[k]\right) \geq \delta\right] \leq \exp\left\{-\frac{M\delta^2}{2g_i}\right\}$$
$$\textbf{\textit{Lower Tail}}: \ \mathbb{P}\left[\left(\frac{1}{M}\sum_{j=0}^{g_i-1}\sum_{k=0}^{M-1} x[\theta_j + k]y[k]\right) \geq \delta\right] \leq \exp\left\{-\frac{M\delta^2}{2g_i}\right\}$$

Proof. Since θ_j is not one of the matching positions for any j, $x[\theta_j + k] \neq y[k]$ and more importantly $x[\theta_j + k] \perp\!\!\!\perp y[k] \forall k$. This implies that $x[\theta_i + k]y[k] = \pm 1$ with equal probability and $\mathbb{E}[x[\theta_i + k]y[k]] = 0$. Now to show the independence of the sets of random variables $\{x[\theta_i + k]y[k], k \in [0 : M-1]\}$ and $\{x[\theta_j + k]y[k], k \in [0 : M-1]\}$ for $i \neq j$ we will use the fact that $\mu < \alpha$ from Eqn. (8) and thus $M < f_i$, resulting in non-overlapping parts of $\vec{\mathrm{x}}$ participating in the correlation coefficients $r[\theta_i]$ and $r[\theta_j]$. Thus the Mg_i random variables involved are shown to be independent and hence we can apply the bounds from Lem. 6 to achieve the required result. □

APPENDIX B
BIN CLASSIFICATION ERRORS

We employ classification rules based only on the first element of the measurement vector at bin (i, j) which can be given by

$$Z[1] = \begin{cases} \sum\limits_{\ell=0}^{g_i-1}\sum\limits_{k=0}^{M-1} n_{l,k} & \text{if} \quad \mathcal{H} = \mathcal{H}_z \\ M_1 + \sum\limits_{\ell=0}^{g_i-2}\sum\limits_{k=0}^{M-1} n_{l,k} & \text{if} \quad \mathcal{H} = \mathcal{H}_s \\ M_1 + M_2 + \sum\limits_{\ell=0}^{g_i-3}\sum\limits_{k=0}^{M-1} n_{l,k} & \text{if} \quad \mathcal{H} = \mathcal{H}_d \end{cases} \tag{10}$$

where $n_{l,k} = x[\theta_\ell + k]y[k]$ and $\theta_\ell \notin \{\tau_1, \tau_2, \ldots, \tau_L\}$. Also for the case of exact matching $M_1 = M_2 = M$ whereas in the case of approximate matching the values of $M_1, M_2 \in [M(1-2\eta) : M]$.

Lemma 8 (zero-ton)**.** *Given that the bin (i, j) is a zero-ton, the classification error can be bounded by*

$$\mathbb{P}[\mathscr{E}_1 | \mathcal{H}_z] \leq e^{-\frac{N^{\mu+\alpha-1}(1-2\eta)^2}{8}}$$

Proof. The above expression can be derived by observing that a bin is not classified as zero-ton if $\frac{Z[1]}{M} \geq \frac{1-2\eta}{2}$. Let us denote the probability of this event as p_{z1} which can be bounded as:

$$\begin{aligned} p_{z1} =& \mathbb{P}\left[\frac{Z[1]}{M} \geq \frac{1-2\eta}{2}\right] \\ \leq& \, e^{-\frac{Mg_i(1-2\eta)^2}{8g_i^2}} \\ \approx& \, e^{-\frac{N^{\mu+\alpha-1}(1-2\eta)^2}{8}} \end{aligned}$$

where the second bound is due to Eqn. (10) and Lemma. 7. The approximation in the third line is from our design that all the g_i are chosen such that $g_i \approx N^{1-\alpha}$ and $M = N^{\mu}$. □

Lemma 9 (singleton)**.** *Given that the bin (i,j) is a singleton, the classification error can be bounded by*

$$\mathbb{P}[\mathscr{E}_1|\mathcal{H}_s] \leq 2e^{-\frac{N^{\mu+\alpha-1}(1-4\eta)^2}{16}}$$

Proof. We observe that a bin is not classified as singleton if $\frac{Z[1]}{M} \leq \frac{1-2\eta}{2}$ or $\frac{Z[1]}{M} \geq \frac{3-4\eta}{2}$. Let us denote the probability of the two events as p_{s1} and p_{s2} respectively which can be bounded as:

$$\begin{aligned}
p_{s1} &= \mathbb{P}\left[\frac{1}{M}\sum_{\ell=0}^{g_i-2}\sum_{k=0}^{M-1} n_{l,k} \leq \frac{1-2\eta}{2} - \frac{M_1}{M}\right] \\
&\leq \mathbb{P}\left[\frac{1}{M}\sum_{\ell=0}^{g_i-2}\sum_{k=0}^{M-1} n_{l,k} \leq -\frac{1-2\eta}{2}\right] \\
&\leq e^{-\frac{Mg_i(1-2\eta)^2}{16g_i^2}} \\
&\approx e^{-\frac{N^{\mu+\alpha-1}(1-2\eta)^2}{16}}
\end{aligned}$$

where we used *lower tail* of Lemma 6 and $g_i \approx N^{1-\alpha}$ and the lower bound on $\frac{M_1}{M} \geq (1-2\eta)$. Similarly p_{s2} can be upper bounded by:

$$\begin{aligned}
p_{s2} &= \mathbb{P}\left[\frac{1}{M}\sum_{\ell=0}^{g_i-2}\sum_{k=0}^{M-1} n_{l,k} \geq \frac{3-4\eta}{2} - \frac{M_1}{M}\right] \\
&\leq \mathbb{P}\left[\frac{1}{Mg_i}\sum_{\ell=0}^{g_i-2}\sum_{k=0}^{M-1} n_{l,k} \geq -\frac{1-4\eta}{2g_i}\right] \\
&\approx e^{-\frac{N^{\mu+\alpha-1}(1-4\eta)^2}{8}}
\end{aligned}$$

Thus the overall probability of error for classifying a singleton can be obtained by combining p_{s1} and p_{s2}. □

Lemma 10 (double-ton)**.** *Given that the bin (i,j) is a double-ton, the classification error can be bounded by*

$$\mathbb{P}[\mathscr{E}_1|\mathcal{H}_d] \leq 2e^{-\frac{N^{\mu+\alpha-1}(1-6\eta)^2}{16}}$$

Proof. We observe that a bin is not classified as double-ton if $\frac{Z[1]}{M} \leq \frac{3-4\eta}{2}$ or $\frac{Z[1]}{M} \geq \frac{5-6\eta}{2}$. Let us denote the probability of these two events as p_{d1} and p_{d2} respectively which can be bounded similar to Lemma 9.

$$\begin{aligned}
p_{d1} &= \mathbb{P}\left[\frac{1}{M}\sum_{\ell=0}^{g_i-3}\sum_{k=0}^{M-1} n_{l,k} \leq \frac{3-4\eta}{2} - \frac{M_1+M_2}{M}\right] \\
&\leq \mathbb{P}\left[\frac{1}{M}\sum_{\ell=0}^{g_i-3}\sum_{k=0}^{M-1} n_{l,k} \leq -\frac{1-4\eta}{2}\right] \\
&\leq e^{-\frac{M(g_i-2)(1-4\eta)^2}{16(g_i-2)^2}} \\
&\approx e^{-\frac{N^{\mu+\alpha-1}(1-4\eta)^2}{16}}.
\end{aligned}$$

Similarly p_{d2} can be bounded as

$$\begin{aligned}
p_{d2} &= \mathbb{P}\left[\frac{1}{M}\sum_{\ell=0}^{g_i-3}\sum_{k=0}^{M-1} n_{l,k} \geq \frac{5-6\eta}{2} - \frac{M_1+M_2}{M}\right] \\
&\leq \mathbb{P}\left[\frac{1}{M}\sum_{\ell=0}^{g_i-3}\sum_{k=0}^{M-1} n_{l,k} \geq \frac{1-6\eta}{2}\right] \\
&\leq e^{-\frac{M(g_i-2)(1-6\eta)^2}{8(g_i-2)^2}} \\
&\approx e^{-\frac{N^{\mu+\alpha-1}(1-6\eta)^2}{8}}
\end{aligned}$$

where we use the lower bounds $M_1, M_2 \leq M$. □

Lemma 11 (multi-ton)**.** *Given that the bin (i,j) is a multi-ton, the classification error can be bounded by*

$$\mathbb{P}[\mathscr{E}_1|\mathcal{H}_m] \leq e^{-\frac{N^{\mu+\alpha-1}(1-6\eta)^2}{16}}$$

Proof. We observe that a bin is not classified as multi-ton if $\frac{Z[1]}{M} \leq \frac{5-6\eta}{2}$. Let us denote the probability of this event as p_{m1} which can be bounded as:

$$\begin{aligned}
p_{m1} &= \mathbb{P}\left[\frac{1}{M}\sum_{\ell=0}^{g_i-3}\sum_{k=0}^{M-1} n_{l,k} \leq \frac{5-6\eta}{2} - \frac{M_m}{M}\right] \\
&\leq \mathbb{P}\left[\frac{1}{M}\sum_{\ell=0}^{g_i-m}\sum_{k=0}^{M-1} n_{l,k} \leq -\frac{1-6\eta}{2}\right] \\
&\leq e^{-\frac{M(g_i-m)(1-4\eta)^2}{16(g_i-m)^2}} \\
&\leq e^{-\frac{M(1-6\eta)^2}{16n_i}} \\
&\approx e^{-\frac{N^{\mu+\alpha-1}(1-6\eta)^2}{16}}.
\end{aligned}$$

□

Appendix C
Position Identification

We will analyze the singleton identification in two separate cases:

- $\mathscr{E}_{21}$: Event where the position is identified incorrectly when the bin is classified correctly a singleton
- $\mathscr{E}_{22}$: In the case of approximate matching, event where the position is identified incorrectly when the bin is originally a double-ton and one of the non-zero variable nodes has already been peeled off

Definition 12 (Mutual Incoherence)**.** *The mutual incoherence $\mu_{max}(\mathbf{W})$ of a matrix $\mathbf{W} = [\vec{\mathrm{w}}_1\ \vec{\mathrm{w}}_2\ \cdots\vec{\mathrm{w}}_i\cdots\vec{\mathrm{w}}_N]$ is defined as*

$$\mu_{max}(\mathbf{W}) \triangleq \max_{\forall i\neq j} \frac{|\vec{\mathrm{w}}_i^{\dagger}\vec{\mathrm{w}}_j|}{||\vec{\mathrm{w}}_i||.||\vec{\mathrm{w}}_j||}$$

Lemma 13 (Mutual Incoherence Bound for sub-sampled IDFT matrix [[6],Proposition A.1)**.** *The mutual incoherence $\mu_{max}(\mathbf{W_{i,k}})$ of the sensing matrix $\mathbf{W}_{i,k}$ (defined in Eq. 6), with B shifts, is upper bounded by*

$$\mu_{max} < 2\sqrt{\frac{\log(5N)}{B}}$$

Proof. The proof follows similar lines as the proof for Lemma V.3. in [6]. □

Lemma 14. *For some constant $c_1 \in \mathbb{R}$ and the choice of $B = 4c_1^2 \log 5N$, the probability of error in identifying the position of a singleton at any bin (i, j) can be upper bounded by*

$$\mathbb{P}[\mathscr{E}_{21}] \leq \exp\left\{ -\frac{N^{\mu+\alpha-1}(1-2\eta)^2(c_1^2-1)}{8(c_1^2+1)} \right\}$$

Proof. Let j_p be the variable node participating in the singleton (i, j). Then the observation vector $\vec{z}_{i,j}$ is given by

$$\begin{aligned}
\underset{\sim}{z}_{i,j} &= \begin{bmatrix} \vec{w}_{j_1}, \vec{w}_{j_2}, & \cdots & \vec{w}_{j_p}, & \cdots & \vec{w}_{j_{g_i}} \end{bmatrix} \times \begin{bmatrix} n_1 \\ \vdots \\ r[j_p] \\ \vdots \\ n_j \\ \vdots \\ n_{g_i} \end{bmatrix} \\
&= r[j_p]\ \vec{w}_{j_p} + \sum_{k \neq p} n_k \vec{w}_{j_k}
\end{aligned}$$

where for convenience we use a simpler notation $j_k = j + (k-1)\frac{N}{f_i}, \vec{w}_{j_k} = \vec{w}^{j_k}$ as defined in Eq. and $n_l = \sum_{k=0}^{M-1} x[\theta_\ell + k]y[k]$ as defined in Eq. (10).

The estimated position $\hat{p}$ is given by

$$\hat{p} = \arg\max_{l} \ \frac{\vec{w}_{j_l}^{\dagger} \underset{\sim}{z}_{i,j}}{B} \tag{11}$$

where $\dagger$ denotes the conjugate transpose of the vector. Also note that $||\vec{w}_{j_k}|| = B$ for any j and k. From Eq. (11) we observe that the position is wrongly identified when $\exists p'$ such that

$$\begin{aligned}
r[j_p] + \frac{1}{B} \sum_{k \neq p} n_k \vec{w}_{j_p}^{\dagger} \vec{w}_{j_k} &\leq \frac{r[j_p]}{B}\ \vec{w}_{j_{p'}}^{\dagger} \vec{w}_{j_p} + n_{p'} + \frac{1}{B} \sum_{k \neq p, p'} n_k \vec{w}_{j_{p'}}^{\dagger} \vec{w}_{j_k} \\
\leftrightarrow \sum_{k \neq p, p'} \alpha_k n_k + \beta n_{p'} &\geq r[j_p] \left(1 - \frac{\vec{w}_{j_{p'}}^{\dagger} \vec{w}_{j_p}}{B} \right) \geq M(1-2\eta)(1-\mu_{\max})
\end{aligned}$$

where α_k and β are constants and can be shown to be in the range $\alpha_k \in [-2\mu_{\max}, 2\mu_{\max}]$ and $\beta \in [1-\mu_{\max}, 1+\mu_{\max}]$. Now using the bound given Chernoff Lemma in Lem. **??** we obtain

$$\begin{aligned}
\mathbb{P}[\mathscr{E}_{21}] &\leq \exp\left\{ -\frac{2M(1-2\eta)^2(1-\mu_{\max})^2}{16 g_i \mu_{\max}^2 + 4(1+\mu_{\max})^2} \right\} \\
&\leq \exp\left\{ -\frac{2M(1-2\eta)^2(1-\mu_{\max})^2}{16(g_i \mu_{\max}^2 + 1)} \right\} \\
&\leq \exp\left\{ -\frac{2M(1-2\eta)^2(c_1-1)^2}{16(g_i + c_1^2)} \right\} \\
&\approx \exp\left\{ -\frac{N^{\mu+\alpha-1}(1-2\eta)^2(c_1^2-1)}{8(c_1^2+1)} \right\}
\end{aligned}$$

The second inequality follows by the definition of $\mu_{\max} \leq 1$. We choose $B = 4c_1^2 \log 5N$, and substituting $\mu_{\max} \leq 2\sqrt{\frac{\log 5N}{B}} = 1/c_1$ (Lemma 13) we get the third inequality.

□

Lemma 15. *For some constant $c_1 \in \mathbb{R}$ and the choice of $B = 4c_1^2 \log 5N$, the probability of error in identifying the position of second non-zero variable node at a double-ton at any bin (i, j), given that the first position identification is correct, can be upper bounded by*

$$\mathbb{P}[\mathscr{E}_{22}] \leq \exp\left\{ -\frac{N^{\mu+\alpha-1}\ (c_1(1-2\eta)-1)^2}{8(1+c_1^2)} \right\}$$

Proof. $\mathscr{E}_{22}$**:**

Let j_p and $j_{\tilde{p}}$ be the two variable nodes participating in the doubleton (i, j). Then the observation vector $\vec{\mathrm{z}}_{i,j}$ is given by

$$\begin{aligned}
\underline{z}_{i,j} &= \begin{bmatrix} \vec{\mathrm{w}}_{j_1}, \vec{\mathrm{w}}_{j_2}, & \cdots & \vec{\mathrm{w}}_{j_p}, & \cdots & \vec{\mathrm{w}}_{j_{g_i}} \end{bmatrix} \times \begin{bmatrix} n_1 \\ \vdots \\ r[j_p] \\ \vdots \\ n_j \\ \vdots \\ r[j_{\tilde{p}}] \\ \vdots \\ n_{g_i} \end{bmatrix} \\
&= r[j_p]\ \vec{\mathrm{w}}_{j_p} + r[j_{\tilde{p}}]\ \vec{\mathrm{w}}_{j_{\tilde{p}}} + \sum_{k \neq p} n_k \vec{\mathrm{w}}_{j_k}
\end{aligned}$$

Let the contribution from $j_{\tilde{p}}$ be peeled off from the doubleton at some iteration, then we get

$$\vec{\mathrm{z}}_{i,j} = r[j_p]\ \vec{\mathrm{w}}_{j_p} + \frac{e_1}{B}\ \vec{\mathrm{w}}_{j_{\tilde{p}}} + \sum_{k \neq p} n_k \vec{\mathrm{w}}_{j_k}$$

where $e_1 \in [-\eta M, \eta M]$ is an extra error term induced due to peeling off.

Now the estimated second position $\hat{p}$ is calculated using Eq. (11). We can observe that the position is wrongly identified when $\exists p'$ such that

$$\frac{\vec{\mathrm{w}}_{j_p}^{\dagger} \underline{z}_{i,j}}{B} \leq \frac{\vec{\mathrm{w}}_{j_{p'}}^{\dagger} \underline{z}_{i,j}}{B}$$

$$\begin{aligned}
\Longrightarrow\ r[j_p] + \frac{1}{B} \sum_{k \neq p, \tilde{p}} n_k \vec{\mathrm{w}}_{j_p}^{\dagger} \vec{\mathrm{w}}_{j_k} + \frac{e_1}{B} \vec{\mathrm{w}}_{j_p}^{\dagger} \vec{\mathrm{w}}_{j_{\tilde{p}}} &\leq \frac{r[j_p]}{B}\ \vec{\mathrm{w}}_{j_{p'}}^{\dagger} \vec{\mathrm{w}}_{j_p} + n_{p'} + \frac{1}{B} \sum_{k \neq p, p', \tilde{p}} n_k \vec{\mathrm{w}}_{j_{p'}}^{\dagger} \vec{\mathrm{w}}_{j_k} + \frac{e_1}{B} \vec{\mathrm{w}}_{j_{p'}}^{\dagger} \vec{\mathrm{w}}_{j_{\tilde{p}}} \\
\leftrightarrow \sum_{k \neq p, p', \tilde{p}} \alpha_k n_k + \beta n_{p'} &\geq r[j_p] \left(1 - \frac{\vec{\mathrm{w}}_{j_{p'}}^{\dagger} \vec{\mathrm{w}}_{j_p}}{B} \right) - \frac{2\eta M}{B} \vec{\mathrm{w}}_{j_{p'}}^{\dagger} \vec{\mathrm{w}}_{j_{\tilde{p}}} \\
&\geq M(1-2\eta)(1-\mu_{\max}) - 2\eta M \mu_{\max} = M(1 - 2\eta - \mu_{\max})
\end{aligned}$$

where α_k and β are constants and can be shown to be in the range $\alpha_k \in [-2\mu_{\max}, 2\mu_{\max}]$ and $\beta \in [1-\mu_{\max}, 1+\mu_{\max}]$. Now using the bound given by Chernoff Lemma in Lem. **??** we obtain

$$\begin{aligned}\mathbb{P}[\mathscr{E}_{22}] &\le \exp\left\{-\frac{2M(1-2\eta-\mu_{\max})^2}{16g_i\mu_{\max}^2+4(1+\mu_{\max})^2}\right\}\\ &\le \exp\left\{-\frac{2M(1-2\eta-\mu_{\max})^2}{16(g_i\mu_{\max}^2+1)}\right\}\\ &\le \exp\left\{-\frac{M(c_1(1-2\eta)-1)^2}{8(g_i+c_1^2)}\right\}\\ &\le \exp\left\{-\frac{N^{\mu+\alpha-1}\ (c_1(1-2\eta)-1)^2}{8(1+c_1^2)}\right\}\end{aligned}$$

where for the choice of $B = 4c_1^2\log 5N$, $\mu_{\max} \le 2\sqrt{\frac{\log 5N}{B}} = 1/c_1$.

□

REFERENCES

[1] A. Andoni, H. Hassanieh, P. Indyk, and D. Katabi, "Shift finding in sub-linear time," in *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 457–465, Society for Industrial and Applied Mathematics, 2013.

[2] A. Amir, M. Lewenstein, and E. Porat, "Faster algorithms for string matching with k mismatches," *Journal of Algorithms*, vol. 50, no. 2, pp. 257–275, 2004.

[3] R. S. Boyer and J. S. Moore, "A fast string searching algorithm," *Communications of the ACM*, vol. 20, no. 10, pp. 762–772, 1977.

[4] W. I. Chang and T. G. Marr, "Approximate string matching and local similarity," in *Annual Symposium on Combinatorial Pattern Matching*, pp. 259–273, Springer, 1994.

[5] H. Hassanieh, F. Adib, D. Katabi, and P. Indyk, "Faster gps via the sparse fourier transform," in *mobicom*, pp. 353–364, ACM, 2012.

[6] S. Pawar and K. Ramchandran, "A robust R-FFAST framework for computing a k-sparse n-length DFT in O($k\log\ n$) sample complexity using sparse-graph codes," in *Proc. IEEE Int. Symp. Information Theory*, pp. 1852–1856, 2014.

[7] K. Lee, R. Pedarsani, and K. Ramchandran, "Saffron: A fast, efficient, and robust framework for group testing based on sparse-graph codes," *arXiv preprint arXiv:1508.04485*, 2015.