Sublinear Time Algorithms for the Sparse Recovery Problem

by

Yi Li

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Computer Science and Engineering)
in The University of Michigan
2013

Doctoral Committee:

Professor Martin J. Strauss, Chair
Associate Professor Kevin J. Compton
Professor Anna C. Gilbert
Professor Alfred O. Hero III
Associate Professor Yaoyun Shi

ACKNOWLEDGEMENTS

Foremost, I would like to express my deepest appreciation to my adviser, Prof. Martin Strauss, for his continuous support and guidance of my Ph.D. study and research. I am also indebted to Prof. Anna Gilbert deeply for introducing me to various workshop opportunities in addition to her guidance of my research.

My sincere gratitude also goes to my collaborators, Prof. Ely Porat at Bar Ilan University and Dr David Woodruff at IBM Almaden Research Center, both of whom possess numerous sparkling ideas. I learnt a lot from them, and I must also thank them for their patience. In particular, I am very grateful to David Woodruff for the many long phone discussions.

I would also like to thank my thesis committee members, Assoc. Prof. Kevin Compton, Prof. Alfred Hero III and Assoc. Prof. Yaoyun Shi, for their reviews and helpful comments.

As a non-driver in Ann Arbor, I am obliged to Caoxie Zhang and Hsin-hao Su, who offered me several important rides in my hour of need. I also want to thank Qian Li for offering me the rides while we were on our internship at IBM Almaden Research Center. Special thanks to Hsin-hao Su for acquainting me with a number of restaurants in the Ann Arbor area. Many thanks to other friends and colleagues who have made made my life in Ann Arbor more enjoyable, including but not limited to, Ye Du, Ran Duan, Lujun Fang, Yudong Gao, Junxian Huang, Zhaoguang Wang, Xiaodi Wu, Jie Yu, Xiaohan Yu.

Finally, and above all, I must thank my parents for their unconditional love and support. This thesis is dedicated to them. I wish my mother were here to see it completed, as she has always been desiring to visit me in America.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ALGORITHMS

ABSTRACT

Sublinear Time Algorithms for the Sparse Recovery Problem
by
Yi Li

Chair: Martin Strauss

In the *sparse recovery problem*, we have a signal $\mathbf{x} \in \mathbb{R}^N$ that is sparse; i.e., it consists of k significant entries ('heavy hitters') while the rest of the entries are essentially negligible. Let $\mathbf{x}_{[k]} \in \mathbb{R}^N$ consist of the k largest coefficients (in magnitude, i.e., absolute value) of $\mathbf{x}$, zeroing out all other entries. We want to recover $\mathbf{x}_{[k]}$, the positions and values of only the heavy hitters, as the rest of the signal is not of interest. Mathematically, we wish to design an m-by-N measurement matrix $\mathbf{\Phi}$ and a recovery algorithm $\mathcal{R}$, such that for signal $\mathbf{x} \in \mathbb{R}^N$ with which we acquire measurements $\mathbf{y} = \mathbf{\Phi x}$, the recovery algorithm produces an approximation $\widehat{\mathbf{x}} = \mathcal{R}(\mathbf{y})$, which satisfies that

$$\|\mathbf{x} - \widehat{\mathbf{x}}\|_p \leq (1+\epsilon) \left\|\mathbf{x} - \mathbf{x}_{[k]}\right\|_p . \tag{0.1}$$

for some norm $\|\cdot\|_p$ in $\mathbb{R}^N$. We would also usually require that $|\operatorname{supp} \widehat{\mathbf{x}}| = O(k)$. Our key goal is to minimize m and achieve a sub-linear runtime for $\mathcal{R}$, ideally, $O(k \operatorname{polylog} N)$ time. Two classical choices of p are $p = 1$ and 2. The problem with $p = 1$ is called the ℓ_1/ℓ_1 problem and the problem with $p = 2$ the ℓ_2/ℓ_2 problem. In general, we would like to have the measurement matrix $\mathbf{\Phi}$ and the decoding algorithm such that (0.1) holds for all $\mathbf{x}$. This guarantee will be called the *for-all* guarantee. Unfortunately, it is known that the ℓ_2/ℓ_2 problem would require $m = \Omega(N)$, which extinguishes our hope for anything substantially better than the trivial algorithm by tracking the whole vector $\mathbf{x}$. Nevertheless, in many applications, a weaker *for-each* guarantee works also well: suppose that $\mathbf{\Phi}$ is drawn from some probability distribution $\mathscr{D}$ such that

$$\Pr_{\Phi \sim \mathscr{D}} \left\{ \|\mathbf{x} - \widehat{\mathbf{x}}\|_p \leq C \left\|\mathbf{x} - \mathbf{x}_{[k]}\right\|_q \right\} \geq \frac{3}{4},$$

that is, the guarantee (0.1) holds with probability at least $3/4$ for each $\mathbf{x}$. This weaker guarantee allows us to use significantly fewer measurements, $m \ll N$. We shall consider the ℓ_2/ℓ_2 problem in for-each setting and the ℓ_1/ℓ_1 problem in for-all setting.

In Chapter 2, we shall show an algorithm for the for-each ℓ_2/ℓ_2 problem which uses the optimal number of measurements $O(k/\epsilon \log(N/k))$ and runs in time $O(k/\epsilon \operatorname{polylog} N)$.

In Chapter 3 we shall show an algorithm for the for-all ℓ_1/ℓ_1 problem which uses $O(k/\epsilon^2 \log(N/k))$ measurements, matching the best measurement complexity among all superlinear time algorithms, and runs in time $O(k^{1+\beta} \operatorname{poly}(\log N, 1/\epsilon))$ for any $\beta > 0$ under a mild assumption $\epsilon \lesssim (\log k/\log N)^\gamma$ for any $\gamma > 0$. This is the first sublinear time algorithm whose runtime is not a polynomial of N.

It is known that the same ideas and techniques apply to the discrete Fourier case where the frequencies lie on a grid $\left\{0, \frac{1}{N}, \frac{2}{N}, \ldots, 1 - \frac{1}{N}\right\}$ and the problem is to recover the signal by taking samples (in the time domain) much fewer than prescribed by the Nyquist rate. In Chapter 4, we shall the sublinear-time techniques to the off-grid case, i.e., the frequencies are real numbers in $[0, 1]$.

CHAPTER 1

Introduction[1]

1.1 Problem Description

Tracking heavy hitters in high-volume, high-speed data streams [CCFC02], monitoring changes in data streams [CM03], designing pooling schemes for biological tests [ECG$^+$09] (e.g., high throughput sequencing, testing for genetic markers), localizing sources in sensor networks [ZBSG05, ZPB06], and combinatorial pattern matching [CP07] are all quite different technological challenges, yet they can all be expressed in the same mathematical formulation, called the *sparse recovery problem*. This problem has further application to telecommunications [PAW07] and medical imaging processing [DDT$^+$08, LDP07]. See more at the extensive web-page [Ric] of the compressive sensing group at Rice University.

In the sparse recovery problem, we have a signal $\mathbf{x}$ of length N that is sparse or highly compressible; i.e., it consists of k significant entries ("heavy hitters") while the rest of the entries are essentially negligible. We wish to acquire a small amount of information (approximately commensurate with the sparsity) about this signal in a linear, non-adaptive fashion, and then use that information to recover the significant entries quickly. In a data stream setting, our signal is the distribution of items seen, while in biological group testing, the signal is proportional to the binding affinity of each drug compound (or the expression level of a gene in a particular organism). We want to recover the positions and values of only the heavy hitters, as the rest of the signal is not of interest. Mathematically, let $\mathbf{x}_{[k]} \in \mathbb{R}^N$ consist of the k largest coefficients (in magnitude, i.e., absolute value) of $\mathbf{x}$, zeroing out all other entries. We wish to design an m-by-N measurement matrix $\mathbf{\Phi}$ and a recovery algorithm $\mathcal{R}$, such that for signal $\mathbf{x} \in \mathbb{R}^N$ with which we acquire measurements $\mathbf{y} = \mathbf{\Phi}\mathbf{x}$, the recovery

[1]This chapter contains part of [GLPS12] and [BCG$^+$12], in compliance with the copyright policy of SIAM and Springer.

algorithm produces an approximation $\widehat{\mathbf{x}} = \mathcal{R}(\mathbf{y})$, which satisfies that

$$\|\mathbf{x} - \widehat{\mathbf{x}}\|_p \leq C \left\|\mathbf{x} - \mathbf{x}_{[k]}\right\|_q . \tag{1.1}$$

for some norm $\|\cdot\|_p$ and $\|\cdot\|_q$ in $\mathbb{R}^N$. A key goal is to minimize m, i.e., to use least possible measurements, because measurements correspond to physical resources (e.g., memory in data stream monitoring devices, number of screens in biological applications) or, more seriously, in medical imaging, the radiation that a patient receives in a CT scan, thus reducing the number of necessary measurements is critical for these problems. It is also natural to minimize the runtime of the recovery algorithm, which is crucial to network traffic monitoring and data streaming applications. Ideally we want $O(k \operatorname{polylog} N)$ time, which is sublinear in N.

There are three typical settings of p, q and C as follows.

1. $p = q = 2$, $C = 1 + \epsilon$;
2. $p = 2$, $q = 1$, $C = \epsilon/\sqrt{k}$;
3. $p = q = 1$, $C = 1 + \epsilon$,

where $\epsilon > 0$ is a parameter of the problem. The problem with parameters p and q will be referred to as the ℓ_p/ℓ_q problem. When p, q belongs to one of the three cases above, the C should be automatically understood as described above in the corresponding case unless otherwise specified. The ℓ_2/ℓ_1 problem is also called the *mixed-norm* problem, which is widely considered in signal processing.

It is known that the ℓ_2/ℓ_2 problem is harder than the ℓ_2/ℓ_1 problem, in the sense that if we can solve the ℓ_2/ℓ_2 problem we can use the recovery system (consisting of the measurement matrix $\mathbf{\Phi}$ and the recovery algorithm $\mathcal{R}$) to construct a recovery system to the ℓ_2/ℓ_1 problem (with ϵ and k different by at most a constant factor). It is also known that ℓ_2/ℓ_1 problem is harder than the ℓ_1/ℓ_1 problem in the same sense.

In general, we would like to have the measurement matrix $\mathbf{\Phi}$ and the decoding algorithm such that (1.1) holds for all $\mathbf{x}$. This guarantee will be called the *for-all* guarantee. Unfortunately, it is proved in [CDD09] that the ℓ_2/ℓ_2 problem would require $m = \Omega(N)$, which extinguishes our hope for anything substantially better than the trivial algorithm which tracks the whole vector $\mathbf{x}$. Nevertheless, in many applications, a weaker *for-each* guarantee works also well: suppose that $\mathbf{\Phi}$ is drawn from some probability distribution $\mathscr{D}$ such that

$$\Pr_{\Phi\sim\mathscr{D}}\left\{\|\mathbf{x} - \widehat{\mathbf{x}}\|_p \leq C \left\|\mathbf{x} - \mathbf{x}_{[k]}\right\|_q\right\} \geq 1 - \delta$$

for some $\delta > 0$, that is, the guarantee (1.1) holds with probability $\geq 1 - \delta$ for each $\mathbf{x}$. This weaker guarantee allows us to use significantly fewer measurements, $m \ll N$. Our aim is to solve the ℓ_2/ℓ_2 problem in for-each setting and the ℓ_2/ℓ_1 problem in for-all setting. For the ℓ_2/ℓ_2 problem, the ideal solution is to have $m = O(k/\epsilon \log(N/k))$ and runtime $O(k/\epsilon \operatorname{polylog} N)$; for the ℓ_2/ℓ_1 problem, the ideal solution is to have $m = O(k/\epsilon^2 \log(N/k))$ and runtime $O(k/\epsilon^2 \operatorname{poly}\log N)$.

In several of the applications, such as high throughput screening and other physical measurement systems, it is also important that the result be robust to the corruption of the measurements by an arbitrary noise vector ν_2. (It is less critical for digital measurement systems that monitor data streams in which measurement corruption is less likely.) In this case, the measurements $\mathbf{y} = \mathbf{\Phi x} + \nu_2$, and the error guarantee would be dependent on ν_2 naturally.

Our problem assumes that k is a given parameter; i.e., the value of k is known or estimated *a priori*. If the parameter k is set to be smaller than the actual number of heavy hitters, the algorithm will miss some of them. However, the error guarantee will remain satisfied, as the right hand of (1.1) depends on the inevitable error $\|\mathbf{x} - \mathbf{x}_{[k]}\|$, which will contain some heavy hitters and thus be large if the heavy hitters are. If all the heavy hitters have about the same magnitude, then finding arbitrary k of them suffices. If the heavy hitters have varying magnitudes, of which the larger magnitudes (if there are at most k of them) are at least $(1+\epsilon)$ times larger than the smaller ones, then the heavy ones will be recovered for the error guarantee to be satisfied.

1.2 Algorithm Overview

Table 1.1 summarizes the known sparse recovery algorithms. The algorithms of sparse recovery generally fall into two categories: geometric algorithms and combinatorial algorithms.

Geometric algorithms. These algorithms usually solve an optimization problem and thus run in poly(N) time, e.g., the notable ℓ_1 minimization algorithm [CRT06, Don06]. Specifically, when there is no post-measurement noise, the solution

$$\widehat{\mathbf{x}} = \arg\min_{\mathbf{x}'} \|\mathbf{x}'\|_1 \quad \text{s.t.} \quad \mathbf{\Phi x}' = \mathbf{y}$$

Paper	A/E	Number of Measurements	Column sparsity/ Update time	Decode time	Approx. error	Noise
[CCFC02]	E	$k \log^c N$	$\log^c N$	$N \log^c N$	$\ell_2 \leq C\ell_2$	
[CM05]	E	$k \log^c N$	$\log^c N$	$k \log^c N$	$\ell_1 \leq C\ell_1$	
[CM06]	E	$k \log^c N$	$\log^c N$	$k \log^c N$	$\ell_2 \leq C\ell_2$	
[Don06, CRT06]	A	$k \log(N/k)$	$k \log(N/k)$	LP	$\ell_2 \leq (C/\sqrt{k})\ell_1$	Y
[GSTV06]	A	$k \log^c N$	$\log^c N$	$k \log^c N$	$\ell_1 \leq (C \log N)\ell_1$	Y
[GSTV07]	A	$k \log^c N$	$k \log^c N$	$k^2 \log^c N$	$\ell_2 \leq (\epsilon/\sqrt{k})\ell_1$	Y
[BGI$^+$08]	A	$k \log(N/k)$	$\log(N/k)$	LP	$\ell_2 \leq (C/\sqrt{k})\ell_1$	Y
[IR08]	A	$k \log(N/k)$	$\log(N/k)$	$N \log(N/k)$	$\ell_1 \leq (1+\epsilon)\ell_1$	Y
[NT09, DM09]	A	$k \log(N/k)$	$\log(N/k)$	$Tnk \log(N/k)$	$\ell_2 \leq (C/\sqrt{k})\ell_1$	Y
[GLPS12] (Chapter 2)	E	$k \log(N/k)$	$\log^c N$	$k \log^c N$	$\ell_2 \leq (1+\epsilon)\ell_2$	Y
[PS12] (any integer ℓ)	A	$\ell^c k \log(N/k)$	$\ell^c \log(N/k) \log k$	$\ell^c k (N/k)^{1/\ell}$	$\ell_1 \leq (1+\epsilon)\ell_1$	Y
Chapter 3 (any $\beta > 0$) (restrictions on ϵ apply)	A	$k \log(N/k)$	$\log(N/k)$	$k^{1+\beta} \log^c N$	$\ell_1 \leq (1+\epsilon)\ell_1$	Y
Lower bound 'A'	A	$k \log(N/k)$	$\log(N/k)$	$k \log(N/k)$	$\ell_2 \leq (\epsilon/\sqrt{k})\ell_1$	Y
Lower bound 'E'	E	$k \log(N/k)$	$\log(N/k)$	$k \log(N/k)$	$\ell_2 \leq (1+\epsilon)\ell_2$	Y

Table 1.1: Summary of known sparse recovery results. Some constant factors are omitted for clarity. "LP" denotes (at least) the time to do a linear program of size at least N. The column "A/E" indicates whether the algorithm works in the for-all (A) model or the for-each (E) model. The column "noise" indicates whether the algorithm tolerates noisy measurements. Measurement and decode time dependence on ϵ, where applicable, is polynomial. Some decoding times depend on a parameter $T = \log(\|\mathbf{x}\|_2/\|\mathbf{x} - \mathbf{x}_{[k]}\|_2)$. The constants c in different occurrences can be different. The lower bound on number of measurements in table above is, in fact, the best upper bound attained by super-linear algorithms.

solves the ℓ_2/ℓ_1 problem. In particular, the recovered signal $\widehat{\mathbf{x}}$ satisfies

$$\|\mathbf{x}-\widehat{\mathbf{x}}\|_2 \leq \frac{c}{\sqrt{k}}\|\mathbf{x}-\mathbf{x}_{[k]}\|_1$$

for some constant $c > 0$, provided that $\mathbf{\Phi}$ is an $O(k)$-RIP matrix (defined below) with restricted isometry constant small enough. The constant c above depends on the restricted isometry constant only. Replacing k with k/ϵ^2 solves our formulation of ℓ_2/ℓ_1 problem with dependence on ϵ.

Definition 1.1 (Restricted isometry property)**.** An $m \times n$ matrix $\boldsymbol{A}$ is said to satisfy the *s-restricted isometry property* (*s-RIP*) with restricted isometry constant δ_s, if it holds that

$$(1-\delta_s)\|\mathbf{y}\|_2 \leq \|\boldsymbol{A}\mathbf{y}\|_2 \leq (1+\delta_s)\|\mathbf{y}\|_2$$

for all n-dimensional vector $\mathbf{y}$ such that $|\operatorname{supp}\mathbf{y}| \leq s$.

Various constructions of RIP matrices have been found. Some important examples are

1. Gaussian/Bernoulli random matrix: Entries of $\boldsymbol{A}$ are independently identically distributed (i.i.d.) $N(0,1)$ or ± 1 for $m = O(k\log(N/k))$, then $\frac{1}{\sqrt{m}}\boldsymbol{A}$ satisfies $O(k)$-RIP property with high probability. This is also an optimal construction, as it is proved in [CDD09] that $m = \Omega(k\log(N/k))$ is the lower bound.

2. Fourier random matrix: $\boldsymbol{A}$ consists of m random rows of the discrete Fourier transform matrix, where $m = O(k\log^3 k\log N)$, then $\frac{1}{\sqrt{m}}\boldsymbol{A}$ satisfies $O(k)$-RIP property with high probability. See [RV08, CGV13]. There is ongoing effort to reduce m further down to $O(k\log N)$.

3. Deterministic constructions: Current best result is $m = O(k^{2-\gamma})$ for $\gamma > 0$ sufficiently small, see [BDF$^+$11]. There is ongoing effort towards $m = O(k\operatorname{polylog} N)$.

In the presence of post-measurement noise ν_2 that is bounded, say, $\|\nu_2\|_2 \leq \eta$, the solution to

$$\widehat{\mathbf{x}} = \arg\min_{\mathbf{x}'} \|\mathbf{x}'\|_1 \quad \text{s.t.} \quad \|\mathbf{\Phi}\mathbf{x}'-\mathbf{y}\|_2 \leq \eta$$

obeys

$$\|\mathbf{x}-\widehat{\mathbf{x}}\|_2 \leq \frac{C_1}{\sqrt{k}}\|\mathbf{x}-\mathbf{x}_{[k]}\|_1 + C_2\eta,$$

for all $\mathbf{x}$, where $C_1, C_2 > 0$ are constants that depend on the restricted isometry constant of $\mathbf{\Phi}$ only.

Combinatorial algorithms. All results in Table 1.1 except for [Don06, CRT06, BGI$^+$08] fall in this category. These algorithms are usually faster and typically iterative. In each round, a combinatorial algorithm would first identify a set $I \subseteq [N] := \{1, \ldots, N\}$, called *candidate set*, which is expected to contain many heavy hitters (though not necessarily all), and then estimate the values of the signal at each index in I with a good accuracy. According to the values recovered, together with the old recovered signal, the algorithm produces a new recovered signal. It then subtracts off the recovered signal and enters the next round, until some halting criterion is satisfied. See Algorithm 1.1 for the general framework.

Algorithm 1.1 General framework of the sparse recovery problem

```
Input: $\mathbf{x}, \mathbf{\Phi}, \mathbf{\Phi x}, k$
Output: $\widehat{\mathbf{x}}$, the appropriate representation of $\mathbf{x}$
  $\mathbf{y} \leftarrow \mathbf{\Phi x}$
  $t \leftarrow 0$
  $\boldsymbol{a}^{(0)} \leftarrow 0$
  while the halting criterion is not satisfied do
      $I \leftarrow \textsc{Identify}(\mathbf{\Phi}, \mathbf{y}, \boldsymbol{a}^{(t)})$
      $\boldsymbol{b} \leftarrow \textsc{Estimate}(\mathbf{\Phi}, \mathbf{y}, I)$
      $\boldsymbol{a}^{(t+1)} \leftarrow \textsc{Merge}(\boldsymbol{a}^{(t)}, \boldsymbol{b})$
      $\mathbf{y} \leftarrow \mathbf{y} - \mathbf{\Phi}\boldsymbol{a}^{(t+1)}$
      $t \leftarrow t + 1$
  end while
  return $\widehat{\mathbf{x}} \leftarrow \boldsymbol{a}^{(t)}$
```

We remark that the candidate sets may contain the recovered heavy hitters positions in all previous rounds so each (previously recovered) heavy hitter would be re-estimated to a better accuracy. The halting criterion varies among the algorithms and is closely related to the *loop invariant* which the algorithm maintains. Two typical loop invariants are

1. the number of remaining heavy hitters: The algorithm reduces the number of remaining heavy hitters by half in each round, hence it needs only $\log k$ rounds, and the halting criterion would simply be $t > \log k$;

2. the norm of the residual $\|\mathbf{x} - \boldsymbol{a}^{(t)}\|$: A typical loop invariant of this kind is to reduce the norm of residual by half in each round, in which case, the number of rounds, and thus the runtime, will depend on the norm of the input signal, $\|\mathbf{x}\|$, and the halting criterion would be $t > \log \|\mathbf{x}\|$.

There is plenty of room to play with the second kind of invariant, see the analysis of the CoSaMP algorithm [NT09], for example. Despite the fact that combinatorial algorithms can be designed to run faster, it is difficult to compress the number of measurements to be as close to optimal as in the geometric algorithms (which run in superlinear time).

1.3 Basic Techniques

We review some basic tricks used in sublinear algorithms in this section, primarily from COUNT-SKETCH [CCFC02], one of the earliest sublinear time algorithms, which solves the for-each ℓ_2/ℓ_2 problem. First we consider the case $k = 1$ and then reduce the general k to the case $k = 1$.

1.3.1 One-sparse signals

In this subsection, we assume that the signal consists of a single heavy hitter at index i_0. Following the outline in Algorithm 1.1, we need to solve two problems: find the value of i_0 (identification) and then estimate $\mathbf{x}_{i_0}$, the value of the heavy hitter. Let ν denote the noise, i.e., $\nu_i = \mathbf{x}_i$ for all $i \neq i_0$ and $\nu_{i_0} = 0$.

We first describe the estimation, assuming that we have obtained the value of I. The easiest way is to take the inner product of the signal with an all-one vector $(1, 1, \ldots, 1)^T$, equivalent to summing all entries in the vector. An alternative way, which is more sophisticated and necessary for the ℓ_2/ℓ_q problem, is to take the inner product of the signal with a random ± 1 vector (each coordinate is a random ± 1 variable and the coordinates are pairwise independent) to cancel the noise. Formally, let $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_M$ be independent random ± 1 vectors defined above. Our estimator is

$$\xi = \operatorname*{median}_{1 \leq j \leq M} \boldsymbol{s}_{j,i_0} \langle \boldsymbol{s}_j, \mathbf{x} \rangle. \tag{1.2}$$

Each $\boldsymbol{s}_{j,i_0} \langle \boldsymbol{s}_j, \mathbf{x} \rangle$ is an unbiased estimator, since the expectation

$$\mathbb{E}(\boldsymbol{s}_{j,i_0} \langle \boldsymbol{s}_j, \mathbf{x} \rangle) = \mathbb{E}\left(\boldsymbol{x}_{i_0} + \sum_{i \neq i_0} \boldsymbol{s}_{j,i_0} \boldsymbol{s}_{j,i} \mathbf{x}_i \right) = \mathbf{x}_{i_0}.$$

The second moment

$$\mathbb{E}(\boldsymbol{s}_{j,i_0} \langle \boldsymbol{s}_j, \mathbf{x} \rangle)^2 = \mathbb{E}\langle \boldsymbol{s}_j, \mathbf{x} \rangle^2 = \|\mathbf{x}\|_2^2 + \sum_{i_1 \neq i_2} \mathbb{E}(\boldsymbol{s}_{j,i_1} \boldsymbol{s}_{j,i_2}) \mathbf{x}_i \mathbf{x}_j = \|\mathbf{x}\|_2^2$$

and thus the variance

$$\mathrm{Var}(\boldsymbol{s}_{j,i_0}\langle \boldsymbol{s}_j, \mathbf{x}\rangle) \leq \|\mathbf{x}\|_2^2 - \mathbf{x}_{i_0}^2 = \|\nu\|_2^2.$$

Therefore by Chebyshev inequality

$$\Pr\left\{|\boldsymbol{s}_{j,i_0}\langle \boldsymbol{s}_j, \mathbf{x}\rangle - \mathbf{x}_{i_0}| \geq 2\|\nu\|_2\right\} \leq \frac{1}{4}$$

and then by Chernoff bound

$$\Pr\left\{|\xi - \mathbf{x}_{i_0}| \geq 2\|\nu\|_2\right\} \leq e^{-cM}$$

for some absolute constant $c > 0$. We obtain a fairly accurate estimate of $\mathbf{x}_{i_0}$ in this way.

Now we consider identification, i.e., finding the value of i_0. Consider the following matrix $\boldsymbol{M}$, called a bit-tester, assuming that N is a power of 2. The columns of the matrix are the binary representations of the numbers 0 to $N-1$. The following is an example of $N = 8$.

$$\boldsymbol{M}\mathbf{x} = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0 \\ 7 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 7 \\ 7 \end{pmatrix}.$$

We can then convert the measurements $\boldsymbol{M}\mathbf{x} = (0,7,7)^T$ back to a $\{0,1\}$-vector $(0,1,1)^T$, and we see that it is exactly the binary representation of the position of the heavy hitter, which equals 3 in our case assuming that the index starts from 0. The same trick works in the presence of noise provided that the noise is small enough so it does not affect the conversion to a $\{0,1\}$-vector. In the ℓ_2/ℓ_2 problem, we also multiply each row of the bit tester by a random ± 1 vector entrywise, hoping to cancel the noise in each measurement. Using a similar estimator to (1.2),

$$\xi' = \operatorname*{median}_{1\leq j\leq M}\langle \boldsymbol{s}_j, \mathbf{x}\rangle,$$

we are able to recover $|\mathbf{x}_{i_0}|$ with accuracy $2\|\nu\|_2$ (note that we do not recover the sign of $\mathbf{x}_{i_0}$ because we do not know the actual value of i_0). Hence we can first obtain an estimate ξ' of $|\mathbf{x}_{i_0}|$ as above, then use $|\xi'|/2$ as the threshold to convert $\boldsymbol{M}\mathbf{x}$ to a $\{0,1\}$-vector, namely by letting

$$\mathbf{v}_i = \begin{cases} 1, & |(\boldsymbol{M}\mathbf{x})_i| \geq |\xi'|/2; \\ 0, & |(\boldsymbol{M}\mathbf{x})_i| < |\xi'|/2. \end{cases}$$

It is clear that if $\|\nu\|_2 < |\mathbf{x}_{i_0}|/4$, with high probability, $\mathbf{v}$ will be the binary representation of i_0 and we shall locate the heavy hitter correctly.

1.3.2 General sparse signals

Now we consider the general case where $k > 1$. The trick is to hash all N positions of the signal into B buckets at random, forming B subsignals. Choosing $B = \Theta(k)$, we hope that many of the subsignals contain only one of the k heavy hitters, reducing the problem to the case of one-sparse signal. Also, the noise in each subsignal is reduced to $\|\mathbf{x} - \mathbf{x}_k\|_2/\sqrt{B} = \Theta(\|\mathbf{x} - \mathbf{x}_{[k]}\|_2/\sqrt{k})$, in expectation. From the discussion in the preceding subsection, we expect to recover a heavy hitter of magnitude at least $\Omega(\|\mathbf{x} - \mathbf{x}_{[k]}\|_2/\sqrt{k})$ up to accuracy $O(\|\nu\|_2/\sqrt{k})$. In the ideal case, $\widehat{\mathbf{x}}$ contains only the k heavy hitters, each is accurate up to $O(\|\mathbf{x} - \mathbf{x}_{[k]}\|_2/\sqrt{k})$ and thus

$$\|\mathbf{x} - \widehat{\mathbf{x}}\|_2 \leq \sqrt{k \cdot O\left(\frac{\|\mathbf{x} - \mathbf{x}_{[k]}\|_2^2}{k}\right)} = O(\|\mathbf{x} - \mathbf{x}_{[k]}\|_2^2).$$

Based upon these ideas, we shall present an optimal algorithm [GLPS12] for the (for-each) ℓ_2/ℓ_2 problem in Chapter 2. More sophisticated techniques are necessary for the (for-all) ℓ_1/ℓ_1 problem, which we shall present in Chapter 3.

1.4 Case of Fourier Basis

So far we have been considering the situation in which the signal $\mathbf{x}$ is sparse under the canonical basis of Euclidean space $\mathbb{R}^N$ and we take the measurements in the same domain. However, in many real applications, the signal has a sparse representation in a different domain and we cannot take samples in that domain, for instance, AM, FM, and other communication signals could be sparse in frequency domain whilst we can only take samples in the space domain. Mathematically, a signal $\mathbf{y} \in \mathbb{R}^N$ has a sparse representation as $\mathbf{y} = \mathbf{\Psi x}$ in a basis $\mathbf{\Psi} \in \mathbb{R}^{N\times N}$ when $k \ll N$ coefficients of $\mathbf{x}$ can represent the signal $\mathbf{y}$. We can pose the same problem as before: design a measurement matrix $\mathbf{\Phi}$, with fewest possible rows, so that we can recover $\mathbf{y}$ from the sketch $\mathbf{\Phi\Psi x}$ in sublinear time.

There has been considerable effort to develop sublinear algorithms within the theoretical computer science community for recovering signals with a few significant discrete Fourier components, beginning with Kushilevitz and Mansour [KM93], including [GGI+02, GMS05, Iwe09], and continuing through in the recent work of Hassanieh, et al. [HIKP12b, HIKP12a]. All of these algorithms are predicated upon treating the vector $\mathbf{y}$ as periodic and the discrete Fourier transform of a vector $\mathbf{x}$ being approximately k-sparse [BCW10, GI10].

The idea in the off-grid case is similar. The discrete Fourier transform $\mathbf{x} = \mathscr{F}\mathbf{y}$ is sparse and we want to realize identification and estimation in the Fourier domain. First we want to hash N possible frequencies (indexed from 0 to $N-1$) into B buckets, for which we choose random a and b such that $\gcd(a, N) = 1$ and take the B buckets to be

$$\mathcal{B}_j = \{a\ell + b\}_{\ell=jN/B-N/(2B)}^{jN/B+N/(2B)}, \quad j = 0, \ldots, B-1.$$

The next question is how to obtain the bucket value. For each $\mathcal{B}_j$, the bucket value analogous to the case of the canonical basis would be

$$\boldsymbol{b}_j = \sum_{i\in\mathcal{B}_j} \mathbf{x}_i = \sum_{\ell=\frac{jN}{B}-\frac{N}{2B}}^{\frac{jN}{B}+\frac{N}{2B}} \mathbf{x}_{a\ell+b} = \sum_{\ell=\frac{jN}{B}-\frac{N}{2B}}^{\frac{jN}{B}+\frac{N}{2B}} (\mathscr{F}\mathbf{y})_{a\ell+b} = \sum_{\ell=\frac{jN}{B}-\frac{N}{2B}}^{\frac{jN}{B}+\frac{N}{2B}} (\mathscr{F}\mathbf{z})_\ell, \tag{1.3}$$

where $\mathbf{z}_j = e^{-2\pi i a^{-1} bj}\mathbf{y}_{a^{-1}j}$. We rewrite (1.3) as

$$\boldsymbol{b}_j = (\mathscr{F}\mathbf{z} * \chi_S)\left(j\frac{N}{B}\right),$$

where χ_S is the characteristic function of set $S = \{-N/(2B), \ldots, N/(2B)\}$ (we identify the notation of a function $f : \mathbb{Z}_N \to \mathbb{C}$ and a vector $f \in \mathbb{C}^N$). The question is how to compute $\boldsymbol{b}_j$ by sampling $\mathbf{y}$. Take a kernel K such that $\mathscr{F}K = \chi_S$, then the bucket value can be written as

$$\boldsymbol{b}_j = (\mathscr{F}\mathbf{z} * \mathscr{F}K)\left(j\frac{N}{B}\right) = (\mathscr{F}(\mathbf{z}\odot K))\left(j\frac{N}{B}\right),$$

where $\mathbf{x}\odot\mathbf{y}$ denotes the componentwise product of two vectors $\mathbf{x}$ and $\mathbf{y}$. Now we observe that

$$\boldsymbol{b} = \begin{pmatrix} \boldsymbol{b}_0 & \boldsymbol{b}_1 & \cdots & \boldsymbol{b}_{B-1} \end{pmatrix}$$

is exactly the discrete Fourier transform of

$$\begin{pmatrix} (\mathbf{z} \odot K)(0) & (\mathbf{z} \odot K)(1) & \cdots & (\mathbf{z} \odot K)(B-1) \end{pmatrix},$$

which can be computed in $O(B \log B)$ time using the Fast Fourier Transform if $\mathbf{z} \odot K$ is known. The trouble is that our choice K has $|\operatorname{supp} K| = N$ so it would take $\Omega(N)$ time to compute $\boldsymbol{b}$, which is prohibitive. The remedy is to use a kernel K with small support, say, of size $\Theta(B)$ or $\Theta(B \log N)$, such that $\mathscr{F} K$ is approximately χ_S, thus we can obtain an approximate $\boldsymbol{b}$ in time $O(B \log N + B \log B) = O(k \log N)$ for $B = \Theta(k)$ by sampling $\mathbf{z}$ and thus $\mathbf{y}$ at $O(k \log N)$ positions. This would be enough provided that $\mathscr{F} K$ is a sufficiently good approximation to χ_S. The idea for trivial bit-testing is similar, using different kernels to read off different bits. A more sophisticated approach is adopted by [HIKP12a] to bring down the number of measurements and the runtime.

The problem can be reformulated as follows. We have a signal of k major frequencies,

$$\sum_{j=1}^{k} \mathbf{y}_{n_j} \mathrm{e}^{2\pi i \omega_j} + \text{noise},$$

where $\omega_j = n_j / N$ and $\{n_j\}$ are the indices of the k largest coordinates in $\mathbf{x} = \mathscr{F}\mathbf{y}$. The problem is to find ω_j and the associated coefficient. Here $\omega_j \in [0, 1)$ and is an integer multiple of $1/N$. Extending ω_j's to real values in $[0, 1]$ will be the topic in Chapter 4.

CHAPTER 2

Sparse Recovery in ℓ_2/ℓ_2 Error Metric

2.1 Problem Description

In this chapter, we consider the ℓ_2/ℓ_2 problem with post-measurement noise.[1] We give a sublinear time recovery algorithm and a distribution over normalized measurement matrices that meet the lower bound (up to constant factors) in terms of the number of measurements and are within $\log^{O(1)} N$ factors of optimal in the running time and $\log^2 k$ in the sparsity of the measurement matrix. In this chapter, we write the signal $\mathbf{x} = \mathbf{x}_{[k]} + \nu_1$ and the measurements $\mathbf{y} = \mathbf{\Phi}\mathbf{x} + \nu_2$, where ν_1 is called pre-measurement noise and ν_2 is the post-measurement noise.

Theorem 2.1. *There is an algorithm and distribution on matrices $\mathbf{\Phi}$ satisfying*

$$\sup_{\|\mathbf{x}\|_2=1} \mathbb{E} \|\mathbf{\Phi}\mathbf{x}\|_2 = 1$$

such that, given $\mathbf{\Phi}\mathbf{x}+\nu_2$, the parameters, and a concise description of $\mathbf{\Phi}$, the algorithm returns $\widehat{\mathbf{x}}$ with approximation error

$$\|\mathbf{x} - \widehat{\mathbf{x}}\|_2^2 \leq (1+\epsilon) \|\nu_1\|_2^2 + \epsilon \|\nu_2\|_2^2$$

with probability at least $3/4$. The algorithm runs in time $O\big(k/\epsilon \log^{O(1)} N\big)$ and $\mathbf{\Phi}$ has $O\big(k/\epsilon \log(N/k)\big)$ rows. In expectation, there are $O(\log^2(k)\log(N/k))$ non-zeros in each column of $\mathbf{\Phi}$.

The approximation $\widehat{\mathbf{x}}$ may have more than k terms. It is known from previous

[1]This chapter is mostly reproduced from [GLPS12], in compliance with the copyright policy of SIAM.

work (e.g. [GSTV07]) that, if

$$\|\mathbf{x} - \widehat{\mathbf{x}}\|_2^2 \leq (1 + \epsilon^2) \left\|\mathbf{x} - \mathbf{x}_{[k]}\right\|_2^2 + \epsilon^2 \|\nu_2\|_2^2,$$

then the truncation $\widehat{\mathbf{x}}_k$ of $\widehat{\mathbf{x}}$ to k terms satisfies

$$\|\mathbf{x} - \widehat{\mathbf{x}}_k\|_2^2 \leq (1 + \Theta(\epsilon)) \left\|\mathbf{x} - \mathbf{x}_{[k]}\right\|_2^2 + \epsilon \|\nu_2\|_2^2.$$

So an approximation with exactly k terms is possible, but with cost $1/\epsilon^2$ versus $1/\epsilon$ for the general case.

2.1.1 Related work

Previous sublinear time algorithms, whether in the for-each model [CCFC02, CM06] or in the for-all model [GSTV07], however, used several additional factors of $\log(N)$ measurements. We summarize some previous algorithms in Table 2.1. The column sparsity denotes how many 1s there are per column of the measurement matrix and determines both the decoding and measurement update time and, for readability, we suppress $O(\cdot)$. The noise column denotes whether the algorithm tolerates post-measurement noise ν_2. The approximation error signifies the metric we use to evaluate the output; $\ell_p \leq C\ell_q(+\ell_r)$ is shorthand for $\|\mathbf{x} - \widehat{\mathbf{x}}\|_p \leq C \left\|\mathbf{x} - \mathbf{x}_{[k]}\right\|_q (+C \|\nu_2\|_r)$. (Some previous results that did not directly claim stability with respect to ν_2 can be modified easily to accommodate non-zero ν_2.) For the (for-each) ℓ_2/ℓ_2 problem, the optimality of the number of measurements in our result—$O(k/\epsilon \log(N/k))$ measurements—is proved by Price and Woodruff [PW11].

2.1.2 Techniques

We build upon the Count-Sketch design but incorporate the following algorithmic innovations to ensure an optimal number of measurements:

- With a random assignment of N signal positions to $O(k)$ subsignals, we need to encode only $O(N/k)$ positions, rather than N as in the previous approaches. Thus we can reduce the domain size which we encode.
- We use a good error-correcting code (rather than the trivial identity code of the bit tester).
- Our algorithm is an iterative algorithm but maintains a *compound* invariant: in our algorithm, the number of undiscovered heavy hitters decreases at each iter-

Paper	For-all/ For-each	Numer of Measurements	Column sparsity/ Update time	Decode time	Approx. error	Noise
[Don06, CRT06]	A	$k \log(N/k)$	$k \log(N/k)$	LP	$\ell_2 \leq (1/\sqrt{k})\ell_1 + \ell_2$	Y
[CCFC02, CM06]	E	$k \log^c N$	$\log^c N$	$k \log^c N$	$\ell_2 \leq C\ell_2$	
[CM05]	E	$k \log^c N$	$\log^c N$	$k \log^c N$	$\ell_1 \leq C\ell_1$	
[GSTV07]	A	$k \log^c N$	$\log^c N$	$k^2 \log^c N$	$\ell_2 \leq (1/\sqrt{k})\ell_1$	
[BGI+08]	A	$k \log(N/k)$	$\log(N/k)$	LP	$\ell_2 \leq (C/\sqrt{k})\ell_1 + \ell_2$	Y
[IR08]	A	$k \log(N/k)$	$\log(N/k)$	$k \log(N/k)$	$\ell_1 \leq C\ell_1 + \ell_1$	Y
[NT09]	A	$k \log(N/k)$	$\log(N/k)$	$Tnk \log(N/k)$	$\ell_2 \leq (C/\sqrt{k})\ell_1 + \ell_2$	Y
Our result	E	$k \log(N/k)$	$\log^c N$	$k \log^c N$	$\ell_2 \leq C\ell_2 + \ell_2$	Y
Lower bound	A	$k \log(N/k)$	$\log(N/k)$	$k \log(N/k)$	$\ell_2 \leq C\ell_2 + \ell_2$	Y

Table 2.1: Summary of known sparse recovery results with post-measurement noise. The sketch type 'A' refers to for-all model: for certain probability, a random measurement matrix works for all signal; and type 'E' for-each model: for each signal, a random measurement matrix works for certain probability. Some decoding times depend on a parameter $T = \log(\|\mathbf{x}\|_2/\|\mathbf{x} - \mathbf{x}_{[k]}\|_2)$. LP denotes the time complexity of solving a linear program. The constants c in different rows can be different.

ation while, simultaneously, the required error tolerance and failure probability become more stringent. Because there are fewer heavy hitters to find at each stage, we can use more measurements to meet more stringent guarantees.

We believe we are the first to consider a for-each algorithm with post-measurement noise, ν_2. As we discuss below, we need to give a new definition of the appropriate metric under which to normalize $\mathbf{\Phi}$.

In Section 2.2 we detail the matrix algebra we use to describe the measurement matrix distribution which we cover in Section 2.3, along with the decoding algorithm. In Section 2.4, we analyse the foregoing recovery system.

2.2 Notations

In order to construct the overall measurement matrix, we form a number of different types of combinations of constituent matrices and to facilitate our description, we summarize our matrix operations in Table 2.2. The matrices that result from all of our matrix operations have N columns and, with the exception of the semi-direct product of two matrices $\ltimes_{\mathrm{r}}$, all operations are performed on matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ with N columns. The full description of the matrix algebra defined in Table 2.2 is as follows.

- **Row direct sum.** The row direct sum $\boldsymbol{A} \oplus_{\mathrm{r}} \boldsymbol{B}$ is a matrix with N columns that is the vertical concatenation of $\boldsymbol{A}$ and $\boldsymbol{B}$.
- **Element-wise product.** If $\boldsymbol{A}$ and $\boldsymbol{B}$ are both $r \times N$ matrices, then $\boldsymbol{A} \odot \boldsymbol{B}$ is also an $r \times N$ matrix whose (i, j) entry is given by the product of the (i, j) entries in $\boldsymbol{A}$ and $\boldsymbol{B}$.
- **Semi-direct product.** Suppose $\boldsymbol{A}$ is a matrix of r_1 rows (and N columns) in which each row has exactly h non-zeros and $\boldsymbol{B}$ is a matrix of r_2 rows and h columns. Then $\boldsymbol{B} \ltimes_{\mathrm{r}} \boldsymbol{A}$ is the matrix with $r_1 r_2$ rows, in which each non-zero entry a of $\boldsymbol{A}$ is replaced by a times the j'th column of $\boldsymbol{B}$, where a is the j'th non-zero in its row.

  This definition can be modified for our purposes in a straightforward fashion when $\boldsymbol{A}$ has fewer than h non-zeros per row.

operator	name	input	output dimensions and construction
$\oplus_r$	row direct sum	$\boldsymbol{A}\colon r_1 \times N$ $\boldsymbol{B}\colon r_2 \times N$	$\boldsymbol{M}\colon (r_1 + r_2) \times N$ $\boldsymbol{M}_{i,j} = \begin{cases} \boldsymbol{A}_{i,j}, & 1 \le i \le r_1 \\ \boldsymbol{B}_{i-r_1,j}, & 1 + r_1 \le i \le r_2 \end{cases}$
$\odot$	element-wise product	$\boldsymbol{A}\colon r \times N$ $\boldsymbol{B}\colon r \times N$	$\boldsymbol{M}\colon r \times N$ $\boldsymbol{M}_{i,j} = \boldsymbol{A}_{i,j}\boldsymbol{B}_{i,j}$
$\ltimes_{\mathrm{r}}$	semi-direct product	$\boldsymbol{A}\colon r_1 \times N$ $\boldsymbol{B}\colon r_2 \times h$	$\boldsymbol{M}\colon (r_1 r_2) \times N$ $\boldsymbol{M}_{i+(k-1)r_2,\ell} = \begin{cases} 0, & \boldsymbol{A}_{k,\ell} = 0 \\ \boldsymbol{A}_{k,\ell}\boldsymbol{B}_{i,j}, & \boldsymbol{A}_{k,\ell} = j\text{th nonzero in row } \ell \end{cases}$

Table 2.2: Notations of matrix algebra. These notations are used in constructing an overall measurement matrix. The last column contains both the output dimensions of the matrix operation and its construction formula.

2.3 Sparse Recovery System

In this section, we specify the measurement matrix and detail the decoding algorithm.

2.3.1 Measurement matrix

The overall measurement matrix, $\boldsymbol{\Phi}$, is multi-layered. At the highest level, $\boldsymbol{\Phi}$ consists of a random permutation matrix $\boldsymbol{P}$ left-multiplying the row direct sum of $O(\log(k))$ summands, $\boldsymbol{\Phi}^{(j)}$, each of which is used in a separate iteration of the decoding algorithm. Each summand $\boldsymbol{\Phi}^{(j)}$ is the row direct sum of two separate matrices, an *identification* matrix, $\boldsymbol{D}^{(j)}$, and an *estimation* matrix, $\boldsymbol{E}^{(j)}$.

$$\boldsymbol{\Phi} = \boldsymbol{P}\begin{bmatrix} \boldsymbol{\Phi}^{(1)} \\ \hline \boldsymbol{\Phi}^{(2)} \\ \hline \vdots \\ \hline \boldsymbol{\Phi}^{(\log(k))} \end{bmatrix} \qquad \text{where } \boldsymbol{\Phi}^{(j)} = \boldsymbol{E}^{(j)} \oplus_{\mathrm{r}} \boldsymbol{D}^{(j)}.$$

In iteration j, the identification matrix $\boldsymbol{D}^{(j)}$ consists of the row direct sum of $O(j)$ matrices, all chosen independently from the same distribution. We construct that distribution,

$$\frac{2^{-\Theta(j)}}{\sqrt{\log(N/k)}}(\boldsymbol{C}^{(j)} \ltimes_{\mathrm{r}} \boldsymbol{H}^{(j)}) \odot \boldsymbol{S}^{(j)},$$

as follows.

- For $j = 1, 2, \ldots, \log k$, the matrix $\boldsymbol{H}^{(j)}$ is a hashing matrix with dimensions $kc^j \times N$, where c in the range $1/2 < c < 1$ will be specified later. Each column has exactly one nonzero, a one, in a uniformly random row. The columns are pairwise independent.

- The matrix $\boldsymbol{C}^{(j)}$ is an encoding of positions by an error-correcting code with constant rate and relative distance, together with several 1s. That is, fix an error-correcting code and encoding and decoding algorithms that encode messages of $\Theta(\log\log N)$ bits into longer codewords, also of length $\Theta(\log\log N)$, and can correct a constant fraction of errors. Let $E(\cdot)$ be its encoding function. The i'th column of $\boldsymbol{C}^{(j)}$ is the direct sum of $\Theta(\log\log N)$ copies of 1 with the direct sum of $E(i_1)$, $E(i_2)$, $\ldots$, where $i_1, i_2, \ldots$ are blocks of $O(\log\log N)$ bits each, whose concatenation is the binary expansion of i. The number of columns in $\boldsymbol{C}^{(j)}$ is the same as the maximum number of non-zeros in $\boldsymbol{H}^{(j)}$, which is approximately the expected number, $\Theta\left(c^j N/k\right)$, where $c < 1$. The number of rows in $\boldsymbol{C}^{(j)}$ is the logarithm of the number of columns, since the process of breaking the binary expansion of index i into blocks has rate 1 and encoding by $E(\cdot)$ has constant rate.

The existence of such an error correcting code can be shown by a simple counting argument. Given a codeword of length $c_2 n$ and a fraction $r < 1/4$, there is a ball of radius $2rc_2 n$ about it with volume $\binom{c_2 n}{2rc_2 n}2^{2rc_2 n}$. If no other codeword is in that ball, nearest-neighbor decoding will recover the correct codeword. Assuming that q codewords have disjoint balls about them, the size of their union is at most $q\binom{c_2 n}{2rc_2 n}2^{2rc_2 n}$. As long as this volume is less than the total number of strings of length $c_2 n$ (i.e., $2^{c_2 n}$), there are more potential codewords we can use. If there are $2^{c_1 n}$ messages (each of length $c_1 n$), each of which needs a codeword, it is possible to find enough decodable codewords as long as

$$2^{c_1 n}\binom{c_2 n}{2rc_2 n}2^{2rc_2 n} \leq 2^{c_2 n}.$$

This relationship holds for appropriately chosen c_1, c_2 and large n.

Note that error correcting encoding often is accomplished by a matrix-vector product, but we are *not* encoding a linear error-correcting code by the usual generator matrix process. Rather, our matrix explicitly lists all the codewords. The code may be non-linear.

- The matrix $\boldsymbol{S}^{(j)}$ is a pseudo-random sign-flip matrix. Each row is a pairwise independent family of uniform ± 1-valued random variables. The sequence of seeds for the rows is a fully independent family. The size of $\boldsymbol{S}^{(j)}$ matches the size of $\boldsymbol{C}^{(j)} \ltimes_{\mathrm{r}} \boldsymbol{H}^{(j)}$.

Below, to achieve our claimed runtime, we will construct $\boldsymbol{C}^{(j)}$ and $\boldsymbol{H}^{(j)}$ together. See Figure 2.3.1 and Section 2.4.2.2.

In summary, the identification matrix at iteration j is of the form

$$\boldsymbol{D}^{(j)} = \frac{2^{-\Theta(j)}}{\sqrt{\log(N/k)}} \left[\begin{array}{c} \left[(\boldsymbol{C}^{(j)} \ltimes_{\mathrm{r}} \boldsymbol{H}^{(j)}) \odot \boldsymbol{S}^{(j)}\right]_1 \\ \hline \vdots \\ \hline \left[(\boldsymbol{C}^{(j)} \ltimes_{\mathrm{r}} \boldsymbol{H}^{(j)}) \odot \boldsymbol{S}^{(j)}\right]_{\Theta(j)} \end{array}\right].$$

In iteration j, the estimation matrix $\boldsymbol{E}^{(j)}$ consists of the direct sum of $O(j + \log\frac{1}{\epsilon})$

matrices, all chosen independently from the same distribution, $\frac{2^{-\Theta(j)}}{\sqrt{\log(N/k)}}\boldsymbol{H'}^{(j)}\odot\boldsymbol{S'}^{(j)}$, so that the estimation matrix at iteration j is of the form

$$\boldsymbol{E}^{(j)} = \frac{2^{-\Theta(j)}}{\sqrt{\log(N/k)}}\left[\begin{array}{c} \left[\boldsymbol{H'}^{(j)}\odot\boldsymbol{S'}^{(j)}\right]_1 \\ \hline \vdots \\ \hline \left[\boldsymbol{H'}^{(j)}\odot\boldsymbol{S'}^{(j)}\right]_{\Theta(j+\log(1/\epsilon))} \end{array}\right].$$

The construction of the distribution is similar to that of the identification matrix, but omits the error-correcting code and uses different constant factors for the number of rows, etc., compared with the analogues in the identification matrix.

- The matrix $\boldsymbol{H'}^{(j)}$ is a hashing matrix with dimensions $O(kc^j)\times N$, for appropriate c, $1/2 < c < 1$. Each column has exactly one nonzero, a one, in a uniformly random row. The columns are pairwise independent.
- The matrix $\boldsymbol{S'}^{(j)}$ is a pseudo-random sign-flip matrix of the same dimension as $\boldsymbol{H'}^{(j)}$. Each row of $\boldsymbol{S'}^{(j)}$ is a pairwise independent family of uniform ± 1-valued random variables. The sequence of seeds for the rows is fully independent.

2.3.2 Measurements

The overall form of the measurements mirrors the structure of the measurement matrices. We do not, however, use all of the measurements in the same fashion. Upon receiving $\boldsymbol{\Phi}\mathbf{x} + \nu_2$, the algorithm first applies the permutation $\boldsymbol{P}^{-1}$. In iteration j of the algorithm, we use the measurements $\mathbf{y}^{(j)} = \boldsymbol{\Phi}^{(j)}\mathbf{x} + (\boldsymbol{P}^{-1}\nu_2)^{(j)}$. As the matrix $\boldsymbol{\Phi}^{(j)} = \boldsymbol{E}^{(j)}\oplus_{\mathrm{r}}\boldsymbol{D}^{(j)}$, we have a portion of the measurements $\mathbf{w}^{(j)} = \boldsymbol{D}^{(j)}\mathbf{x}+(\boldsymbol{P}^{-1}\nu_2)^{D(j)}$ that we use for identification and a portion $\mathbf{z}^{(j)} = \boldsymbol{E}^{(j)}\mathbf{x} + (\boldsymbol{P}^{-1}\nu_2)^{E(j)}$ that we use for estimation. The $\mathbf{w}^{(j)}$ portion is further decomposed into measurements $[\mathbf{v}^{(j)}, \mathbf{u}^{(j)}]$ corresponding to the run of $O(\log\log N)$ 1's in $\boldsymbol{C}^{(j)}$ and measurements corresponding to each of the blocks in the error-correcting code. There are $\Theta(j)$ i.i.d. repetitions in the identification part and $\Theta(j + \log(1/\epsilon))$ repetitions in the estimation part.

2.3.3 Decoding

The decoding algorithm is shown in Algorithm 2.1.

$$\boldsymbol{H} = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}, \quad \boldsymbol{C} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 \end{pmatrix}$$

The matrix $\boldsymbol{H}$ is formed from hash function h which maps $\langle 8, 0, 3, 6\rangle$ to $\langle 0, 1, 2, 3\rangle$. If ρ is the top row of $\boldsymbol{H}$ and $\boldsymbol{S}$ arbitrary, then

$$(\boldsymbol{C} \ltimes_{\mathrm{r}} \rho) \odot \boldsymbol{S} = \begin{pmatrix} -1 & 0 & 0 & 1 & 0 & 0 & -1 & 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & -1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Figure 2.1: Example measurement matrix for identification. Here $N = 11$, $k = 3$, and, in the hashing $h : i \mapsto a + bi \mod N$, we have $a = 1$ and $b = 4$, so that the sequence $i = \langle 0, 1, 2, 3,\ 4, 5, 6, 7,\ 8, 9, 10\rangle$ is mapped to $\langle 1, 5, 9, 2,\ 6, 10, 3, 7,\ 0, 4, 8\rangle$. The three buckets are $\{i : 0 \le a + bi < 4\}, \{i : 4 \le a + bi < 8\}$, and $\{i : 8 \le a + bi < 11\}$. We number starting from 0, so $0 \le i < 11$. The first bucket is therefore $\langle 8, 0, 3, 6\rangle$, in the same order as $\langle h^{-1}(0), h^{-1}(1), h^{-1}(2), h^{-1}(3)\rangle$, corresponding to the first rows of $\boldsymbol{H}$. In this example, we use two rows of ones and the double repetition code instead of a good code. To demonstrate how to calculate $\boldsymbol{C} \ltimes_{\mathrm{r}} \rho$, let us look at the fifth row of $\boldsymbol{C} \ltimes_{\mathrm{r}} \rho$. The fifth row of $\boldsymbol{C}$ is (0 1 0 1), so we multiply the first element in the first bucket by 0, the second by 1, the third by 0 and the fourth by 1. Note that the first element in the bucket is at the 8-th coordinate, the second element at the 0-th coordinate, and so on. Hence the fifth row of $\boldsymbol{C} \ltimes_{\mathrm{r}} \rho$ will have 0 at the 8-th coordinate, 1 at the 0-th coordinate, and so on.

Algorithm 2.1 The overall recovery algorithm for the ℓ_2/ℓ_2 problem

```
Output: $\widehat{\mathbf{x}}$ is the approximate representation of $\mathbf{x}$
 1: function MAIN($\mathbf{\Phi}$,$\mathbf{y}$)
 2:     $\mathbf{y} \leftarrow \boldsymbol{P}^{-1}\mathbf{y}$
 3:     $\boldsymbol{a}^{(0)} = 0$
 4:     for $j \leftarrow 0$ to $O(\log k)$ do
 5:         $\mathbf{y} = \mathbf{y} - \boldsymbol{P}^{-1}\mathbf{\Phi}\boldsymbol{a}^{(j)}$
 6:         split $\mathbf{y}^{(j)} = \mathbf{w}^{(j)} \oplus_{\mathrm{r}} \mathbf{z}^{(j)}$      ▷ Recall that $\mathbf{y} = (\mathbf{y}^{(1)}, \dots, \mathbf{y}^{(O(\log k))})$
 7:         $\Lambda \leftarrow$ IDENTIFY($\boldsymbol{D}^{(j)}, \mathbf{w}^{(j)}$)
 8:         $\boldsymbol{b}^{(j)} \leftarrow$ ESTIMATE($\boldsymbol{E}^{(j)}, \mathbf{z}^{(j)}, \Lambda$)
 9:         $\boldsymbol{a}^{(j+1)} \leftarrow \boldsymbol{a}^{(j)} + \boldsymbol{b}^{(j)}$
10:     end for
11:     return $\widehat{\mathbf{x}} = \boldsymbol{a}^{(j)}$
12: end function
```

```
Output: $\Lambda$ is the list of candidate positions
 1: function IDENTIFY($\boldsymbol{D}^{(j)}$, $\mathbf{w}^{(j)}$)
 2:     $\Lambda \leftarrow \emptyset$
 3:     Divide $\mathbf{w}^{(j)}$ into sections $[\mathbf{v}, \mathbf{u}]$ of size $O(\log(c^j(N/k)))$
 4:     for each section do
 5:         $u \leftarrow \mathrm{median}(|\mathbf{v}_\ell|)$
 6:         for each $\ell$ do                              ▷ threshold measurements
 7:             $\mathbf{u}_\ell = H(|\mathbf{u}_\ell| - u/2)$      ▷ $H(u) = 1$ if $u > 0$, $H(u) = 0$ otherwise
 8:         Divide $\mathbf{u}$ into blocks $b_i$ of size $O(\log\log N)$
 9:         for each $b_i$ do
10:             $\beta_i \leftarrow$ DECODE($b_i$)                    ▷ using error-correcting code
11:         $\lambda \leftarrow$ INTEGER($\beta_1, \beta_2, \dots$)      ▷ integer represented by bits $\beta_1, \beta_2, \dots$
12:         $\lambda \leftarrow$ CONVERT($\lambda$)                ▷ convert bucket index to signal index
13:         $\Lambda \leftarrow \Lambda \cup \{\lambda\}$
14:     end for
15: end function
```

```
Output: $\boldsymbol{b}$ is the vector of positions and values
 1: function ESTIMATE($\boldsymbol{E}^{(j)}$, $\mathbf{z}^{(j)}$, $\Lambda$)
 2:     $\boldsymbol{b} \leftarrow \emptyset$
 3:     for each $\lambda \in \Lambda$ do
 4:         $\boldsymbol{b}_\lambda \leftarrow \mathrm{median}_{\ell \text{ s.t.} \boldsymbol{H}^{(j)}_{\ell,\lambda}=1}(\mathbf{z}^{(j)}_\ell \boldsymbol{S}^{(j)}_{\ell,\lambda})$
 5:     for each $\lambda \in \Lambda$ do
 6:         if $|\boldsymbol{b}_\lambda|$ is not among top $\Theta(k/2^j)$ then
 7:             $\boldsymbol{b}_\lambda \leftarrow 0$
 8: end function
```

2.4 Analysis

The overall structure of our algorithm is greedy, similar to other algorithms in the literature. At each iteration, the algorithm recovers some of the signal, but introduces errors both through many coefficient estimates that are approximately but not perfectly correct, and also through a small number of terms that can be arbitrarily bad. The result is called a *residual signal*.

The measurement and runtime costs of first iteration dominates the combined cost of all the others. In it, we reduce a bound on the number of heavy hitters to recover from k to $k/2$, while increasing the noise energy from 1 to $1 + \epsilon/4$, using $O((k/\epsilon)\log(N/k))$ measurements. In subsequent iterations, the number of heavy hitters is reduced to $k/2^j$, which reduces the leading cost factor from k/ϵ to $2^{-j}k/\epsilon$. This gives the algorithm 2^j times more resources. In particular, the algorithm can tighten the approximation constant from $1+\epsilon/4$ to $1+(\epsilon/4)c^j$, for appropriate c in the range $1/2 < c < 1$, at cost factor $(1/c)^j < 2^j$, which is more than paid for by the $2^{-j} < 1$ savings in the leading factor. Similarly, the algorithm can simultaneously afford to have a smaller failure probability at iteration j. With the tightened approximation constant, the algorithm can tolerate additional ν_2 noise in later iterations, which, as we show below, saves resources.

To prove our result formally, we state a loop invariant maintained by our algorithm and prove that this invariant holds in the the Loop-Invariant Maintenance Lemma, or LIM lemma. We demonstrate how it characterizes a single iteration of the algorithm: (i) how many measurements are used, (ii) how many non-zeros there are in each column of the measurement matrix, (iii) the runtime, and (iv) the properties of the residual. To prove the LIM lemma, we proceed as follows.

- In Claim 2.4, we explain, structurally, how the conclusions of the lemma are met—what are the sources of errors, etc.
- We then examine the three subroutines in the algorithm: (i) isolating heavy hitters, (ii) identifying them, and (iii) estimating coefficients.
- Finally, we show that the number of measurements used, the sparsity of the measurement matrix, the running time, and the effect of post-measurement noise are all as claimed in the Lemma.

Finally, we discuss normalization of $\mathbf{\Phi}$ and show that it is, indeed, normalized. We conclude by analysing the correctness and efficiency of the overall algorithm, using our results about each iteration.

2.4.1 Correctness

Without loss of generality, assume $\|\nu_1\|_2 = \|\nu_2\|_2 = 1$, since our analysis can scale the signal (the algorithm does not need to know the scaling) and, if ν_1 and ν_2 have different energies, we can increase the weaker of the two. Formally, we maintain the following invariant.

Claim 2.2 (Loop Invariant). *At the beginning of iteration j, the residual signal has the form $\boldsymbol{r}^{(j)} = \sigma^{(j)} + \nu_1^{(j)}$ with*

$$\left\|\sigma^{(j)}\right\|_0 \le \frac{k}{2^j} \text{ and } \left\|\nu_1^{(j)}\right\|_2^2 \le 1 + \epsilon\left(1 - \left(\frac{3}{4}\right)^j\right)$$

except with probability $\frac{1}{4}(1 - (\frac{1}{2})^j)$, where $\|\cdot\|_0$ is the number of non-zero entries. Furthermore, the algorithm has computed (the sparse partial representation) $\widehat{\mathbf{x}}^{(j)} = \mathbf{x} - \boldsymbol{r}^{(j)}$.

Clearly, the invariant holds at the start and maintaining the invariant is sufficient to prove the overall result. In order to show that the algorithm maintains the loop invariant, we demonstrate the following lemma, which, after proper instantiation of the lemma's variables, can be used to show the invariant is maintained.

2.4.1.1 Loop Invariant Maintenance

Lemma 2.3 (Loop Invariant Maintenance). *Fix numerical parameters N, ℓ, δ, and η, with $\delta > 0$, and $\eta > 1/N$. Let $\boldsymbol{a}$ be a vector of length N that can be written as $\boldsymbol{a} = \sigma + \nu_1$, with $\|\sigma\|_0 \le \ell$. Let $\boldsymbol{\Phi}$ be of the form of $O(\log(1/\delta))$ repetitions of $(\boldsymbol{C} \ltimes_{\mathrm{r}} \boldsymbol{H}) \odot \boldsymbol{S}$ in row direct sum with $O(\log 1/(\delta\eta))$ repetitions of $\boldsymbol{H'} \odot \boldsymbol{S'}$ described in Section 2.3.1, where $\boldsymbol{H}$ and $\boldsymbol{H'}$ have $O(\ell/\eta)$ rows. Let ν_2 be a noise vector, where each component has magnitude at most $\frac{4}{\sqrt{m}}\|\nu_2\|_2$, where m is the length of ν_2.*

Then, except with probability δ, given $\boldsymbol{\Phi}$, $\mathbf{y} = \boldsymbol{\Phi}\boldsymbol{a} + \nu_2$, and appropriate parameters, the inner loop of the Recover *algorithm in Algorithm 2.1 recovers $\boldsymbol{b}$ that can be written as $\boldsymbol{b} = \sigma' + \nu_1'$, with $\|\sigma'\|_0 \le \ell/2$ and $\|\nu_1'\|_2^2 \le (1 + \eta)\|\nu_1\|_2^2 + \frac{16\eta}{\gamma}\|\nu_2\|_2^2$, where γ is the common expected number of non-zeros in each column of $\boldsymbol{\Phi}$. Furthermore,*

- *The number of rows in $\boldsymbol{\Phi}$ is $O(\ell/\eta)\log(N/\ell)\log(1/\delta)$.*
- *The computation time is $(\ell/\eta)\log^{O(1)}(N/(\ell\delta\eta))$.*
- *The expected number γ of non-zeros in each column of $\boldsymbol{\Phi}$ is $O((\log N/\ell)\log(1/\delta))$.*

Proof. Much of the algorithm and analysis is similar to previous work (e.g. [CCFC02]), so we sketch the proof, focusing on changes versus previous work. We first address the case $\nu_2 = 0$.

Recall that $\mathbf{\Phi}$ works by giving each element of the signal a random sign flip, hashing each item pairwise independently at random to each measurement, and encoding each index by an error-correcting code. We have:

Claim 2.4. *Except with probability $\delta/3$,*

- *The vector $\boldsymbol{b}$ contains all but at most $\ell/4$ terms of σ, with 'good' estimates.*
- *The vector $\boldsymbol{b}$ contains at most $\ell/4$ terms with 'bad' estimates, i.e., with square error greater than proportional to $\eta/\ell \cdot \|\nu_1\|_2^2$.*
- *The total sum square error over all 'good' estimates is at most $\eta/4$.*

Proof. To simplify notation, let T be the set of terms of $\boldsymbol{a}$ that are both among the top ℓ and have energy at least $\frac{\eta}{8\ell}\|\nu_1\|_2^2$. We know that $|T| \le O(\ell)$. We call the elements in T heavy hitters. The proof proceeds in three steps.

Step 1. Isolate heavy hitters with little noise. Consider the action of a hashing and sign-flip matrix $\boldsymbol{H}\odot\boldsymbol{S}$ with $O(\ell/\eta)$ rows. From previous work [CCFC02, AMS99], it follows that, if constant factors parametrizing the matrices are chosen properly,

Lemma 2.5. *For each $t \in T$, the following holds with probability $1 - O(\delta\eta)$:*

(a) The term t is hashed by at least one row ρ in $\boldsymbol{H}$.

(b) There are $O(\eta N/\ell)$ total positions (out of N) hashed by ρ.

(c) The dot product $(\rho\odot\boldsymbol{s})\boldsymbol{a}$ is $\boldsymbol{s}_t\boldsymbol{a}_t \pm O\left(\sqrt{\frac{\eta}{\ell}}\|\nu_1\|_2\right)$, where $\boldsymbol{s}$ is a sign-flip vector.

(d) Every $t' \in T \setminus \{t\}$ is not hashed by ρ.

Proof. (Sketch) For intuition, note that the estimator $\boldsymbol{s}_t(\rho\odot\boldsymbol{s})\boldsymbol{a}$ is a random variable with mean $\boldsymbol{a}_t$ and variance $\|\nu_1\|_2^2$. Then the claims in the Lemma assert that the expected behavior happens, up to constant factors, with probability $\Omega(1)$. The $O(\log 1/(\delta\eta))$ repetitions of $\boldsymbol{H}\odot\boldsymbol{S}$ bring the failure probability down to $O(\delta\eta)$.

In the favorable case, into each row of $\boldsymbol{H}$ is hashed exactly one term of T that dominates the other $\eta N/\ell$ terms hashed into that row. $\square$

Call a row ρ that satisfies the conditions in Lemma 2.5 a good row.

Step 2. Identify heavy hitters with little noise. Next, we show how to identify the heavy hitter t in a good row. Since there are $\eta N/\ell$ different positions hashed by $\boldsymbol{H}$, we need to learn the $O(\log(\eta N/\ell))$ bits describing t in this context. Previous sublinear algorithms [CM06, GSTV07] used a trivial error correcting code, in which the t'th column was simply the binary expansion of t in direct sum with a single 1, for the matrix $\boldsymbol{C}$ in semi-direct product with $\boldsymbol{H}$. Thus, if the signal $\mathbf{x}$ consists of $\mathbf{x}_t$ in the t'th position and zeros elsewhere, the vector $(\boldsymbol{C} \ltimes_{\mathrm{r}} \boldsymbol{H})\mathbf{x}$ would include $\mathbf{x}_t$ and $\mathbf{x}_t$ times the binary expansion of t (the latter interpreted as a string of 0's and 1's as real numbers). These algorithms require strict control on the failure probability of each measurement in order to use such a trivial encoding. In our case, each measurement succeeds only with probability $\Omega(1)$ and, generally, fails with probability $\Omega(1)$. So we need to use a more powerful error correcting code and a more reliable estimate of $|\mathbf{x}_t|$.

Recall that we have a portion $\mathbf{w}$ of the measurements that are used for identification and that these are further decomposed into the pieces $[\mathbf{v}, \mathbf{u}]$ that correspond to the parallel repetition of $\Theta(\log\log N)$ 1s and to the error-correcting code blocks, respectively. We use the block $\mathbf{v}$ of $b = \Theta(\log\log N)$ independent measurements of $|\mathbf{x}_t|$ to obtain an estimate u of $|\mathbf{x}_t|$ that we use to threshold the subsequent measurements $\mathbf{u}$ to 0/1 values that correspond to the bits in the encoding of t. Let p denote the success probability of each individual measurement in $\mathbf{v}$. We can arrange that $p > 1 - r$ (recall that r is the relative distance of the error correcting code). Then, we expect the fraction p to be approximately correct estimates of $|\mathbf{x}_t|$, we achieve close to this expected fraction, and the median u over the $\Theta(\log\log N)$ estimates is approximately correct with high probability.

Next, we use the median u to threshold the remaining measurements $\mathbf{u}$ to 0/1 values. Let us consider these bit estimates. In a single error-correcting code block of $b = \Theta(\log\log N)$ measurements, we will get close to the expected number, bp, of successful measurements, except with probability $1/\log(N)$, using the Chernoff bound. In the favorable case, we get a number of failures less than the (properly chosen) distance of the error-correcting code and we can recover the block using standard nearest-neighbor decoding. The number of error-correcting code blocks associated with t is $O(\log(\eta N/\ell)/\log\log N) \leq O(\log N)$, so we can take a union bound over all blocks and conclude that we recover t with probability $\Omega(1)$. Because the algorithm takes $O(\log(1/\delta))$ parallel independent repetitions, we guarantee that the failure probability is δ for each $t \in T$ and we expect $\delta|T| = O(\delta\ell)$ failures, overall. The probability of getting more than $\ell/4$ failures is at most $O(\delta)$.

Step 3. Estimate heavy hitters.

Many of the details in this step are similar to those in Lemma 2.5 (as well as to previous work as the function ESTIMATE is essentially the same as COUNT SKETCH), so we give only a brief summary.

The error-correcting code is not necessary for estimating the coefficient values and we use a separate set of measurements $\mathbf{z}$ that do not include the coding overhead. As above, random sign flips and hashing into $O(\ell/\eta)$ buckets suffices to isolate a term t so that the remaining terms hashed to t's bucket have expected energy $O(\eta/\ell)$ and realize energy $O(\eta/\ell)$ with constant probability. Another factor $\log 1/(\delta\eta)$ repetitions suffices to make the failure probability $\delta\eta$, so that, except with probability δ, we have $O(\eta|\Lambda|) = O(\ell)$ failures overall among the $|\Lambda| = \Theta(\ell/\eta)$ candidates whose coefficients we estimate.

This concludes the proof of the claim. □

Number of measurements. We now consider the number of measurements in the matrix. The hashing matrix $\boldsymbol{H}$ contributes $O(\ell/\eta)$ rows. The constant-rate error-correcting code matrix $\boldsymbol{C}$ contributes an additional factor of $O(\log \eta N/\ell)$, to identify one index out of $\eta N/\ell$. The $O(\log 1/\delta)$ repetitions contribute that additional factor to drive down the overall failure probability of identification from $1 - \Omega(1)$ to δ. The $\boldsymbol{S}$ matrix does not contribute to the number of rows. This gives a product of

$$O((\ell/\eta)(\log(\eta N/\ell) \log 1/\delta))$$

for identification.

Similarly, for estimation, we have $O(\ell/\eta)$ rows for hashing. Since we are estimating coefficients for $O(\ell/\eta)$ candidates and can only afford $O(\ell)$ errors except with probability δ, the Markov inequality requires that each estimate fail with probability at most $\eta\delta$, which contributes the factor $O(\log 1/(\eta\delta))$. Thus we get $O((\ell/\eta) \log 1/(\eta\delta))$ for estimation, and

$$O((\ell/\eta)(\log(\eta N/\ell) \log 1/\delta + \log 1/(\eta\delta)),$$

overall. Note that we may assume $\eta > \ell/N$, since, otherwise, we may use $\ell/\eta > N$

measurements to recover trivially. Thus the overall number of measurements is

$$O((\ell/\eta)(\log(N/\ell)\log 1/\delta)).$$

Number of non-zeros. The expected number of non-zeros in each column of the identification part of $\mathbf{\Phi}$ is $O(1)$ from hashing, times the factor $O(\log \eta N/\ell)$ from the general (dense) error-correcting code, times $O(\log 1/\delta)$ for repetition. Analysis of the estimation part is similar. We get $O(\log(N/\ell)\log 1/\delta)$ non-zeros altogether.

Post-measurement noise. Finally, consider the effect of ν_2. Suppose there are m rows in $\mathbf{\Phi}$. A careful inspection of the above proof indicates that ν_1 enters only through the expected energy in each bucket, which is $\Theta((\eta/\ell)\left\|\nu_1\right\|_2^2)$; the error-correcting code and parallel repetitions lead to energy $\Theta((\eta/\ell)\left\|\nu_1\right\|_2^2) = \Theta((\gamma/m)\left\|\nu_1\right\|_2^2)$ in each component of $\mathbf{\Phi}\boldsymbol{a}$. The error $(1+\eta)\left\|\nu_1\right\|_2^2$ represents the 'inevitable' error $\left\|\nu_1\right\|_2^2$ due to terms outside the top $O(\ell)$ that are not recovered by the algorithm, plus 'excess' error $\eta\left\|\nu_1\right\|_2^2$, which is introduced through many small coefficient approximation errors. Since ν_2 does not affect the inevitable error, we can replace $(\gamma/m)\left\|\nu_1\right\|_2^2$ with $(\gamma/m)\left\|\nu_1\right\|_2^2 + (16/m)\left\|\nu_2\right\|_2^2$ when figuring the excess error, giving the claimed result. (Below we will see that γ can be viewed as a normalization factor for $\mathbf{\Phi}$, that makes $\mathbf{\Phi}\nu_1$ and ν_2 comparable.)

The computation time is straightforward. This concludes the proof of Lemma 2.3, the Loop Invariant Maintenance Lemma. $\square$

2.4.1.2 Normalization of the Measurement Matrix

Next we consider the normalization of the overall matrix $\mathbf{\Phi}$ from Section 2.3.1. As has been observed [BIPW10], $\mathbf{\Phi}$ should be normalized in the setting of $\nu_2 \neq 0$. Otherwise, the matrix $\mathbf{\Phi}$ can be scaled up by an arbitrary constant factor $c > 1$ which can be undone by the decoding algorithm: Let $\mathcal{D}'$ be a new decoding algorithm that calls the old decoding algorithm $\mathcal{D}$ as $\mathcal{D}'(\boldsymbol{y}) = \mathcal{D}(\frac{1}{c}\boldsymbol{y})$, so that $\mathcal{D}'(c\mathbf{\Phi}\mathbf{x} + \nu_2) = \mathcal{D}(\mathbf{\Phi}\mathbf{x} + \frac{1}{c}\nu_2)$. Thus we would be able to *reduce* the effect of ν_2 by an arbitrary factor $c > 1$. In our 'for each, $\ell_2 \leq C\ell_2$' model, an appropriate way to normalize $\mathbf{\Phi}$ is as follows.

Definition 2.6. The $\left\|\mathbf{\Phi}\right\|_{2\rightsquigarrow 2}$ norm of a randomly constructed matrix $\mathbf{\Phi}$ is

$$\max_{\boldsymbol{x}\neq 0} \mathbb{E}\left[\frac{\left\|\mathbf{\Phi}\mathbf{x}\right\|_2}{\left\|\mathbf{x}\right\|_2}\right].$$

Note that the usual 2-operator norm,

$$\|\mathbf{\Phi}\|_2 = \max_{\mathbf{x}\neq 0}\left[\frac{\|\mathbf{\Phi x}\|_2}{\|\mathbf{x}\|_2}\right]$$

is typically much larger than $\|\mathbf{\Phi}\|_{2\rightsquigarrow 2}$, which would lead to a much weaker result. But it corresponds to an adversary choosing $\mathbf{x}$ and ν_2 knowing the outcome $\mathbf{\Phi}$, which is counter to the spirit of the 'for each' model in previous work. Here we assume the adversary knows the distribution on $\mathbf{\Phi}$, but not the outcome, when choosing $\mathbf{x}$ and ν_2.

Now we bound $\|\mathbf{\Phi}\|_{2\rightsquigarrow 2}$ for our $\mathbf{\Phi}$. It is straightforward to see that this is the maximum expected column ℓ_2 norm. In the j'th iteration, there are at most $j \log N/k$ non-zero entries, each of magnitude $\sqrt{\frac{c^j}{\log N/k}}$ for some c in the range $1/2 < c < 1$. It follows that

$$\|\mathbf{\Phi}\|^2_{2\rightsquigarrow 2} \leq \sum_j jc^j = O(1),$$

if constants are chosen properly.

2.4.1.3 Invariant

Now we show that the invariant is satisfied, using the LIM lemma. That is all that remains to prove our main theorem:

Theorem 2.1. *There is an algorithm and distribution on matrices* $\mathbf{\Phi}$ *satisfying*

$$\max_{\mathbf{x}\neq 0}\mathbb{E}\left[\frac{\|\mathbf{\Phi x}\|_2}{\|\mathbf{x}\|_2}\right] = 1$$

such that, given $\mathbf{\Phi x}$*, the parameters, and a concise description of* $\mathbf{\Phi}$*, the algorithm returns* $\widehat{x}$ *with approximation error*

$$\|\mathbf{x} - \widehat{\mathbf{x}}\|_2^2 \leq (1+\epsilon)\|\nu_1\|_2^2 + \epsilon\|\nu_2\|_2^2$$

with probability $3/4$. *The algorithm runs in time* $O\big((k/\epsilon)\log^{O(1)} N\big)$ *and* $\mathbf{\Phi}$ *has* $O\big((k/\epsilon)\log(N/k)\big)$ *rows. In expectation, there are* $O(\log^2(k)\log(N/k))$ *non-zeros in each column of* $\mathbf{\Phi}$.

Proof. Note that the matrices described in Section 2.3.1 have two additional features, compared with the matrices in the LIM lemma. First, there is a single random permutation matrix $\boldsymbol{P}$ that multiplies all the error-correcting code, hashing, and sign-flip matrices, and, second, the matrices in iteration j are multiplied by $c^{j/2}$, where c is an appropriate constant in the range $1/2 < c < 1$. Also note that ν_2 is not *a priori* guaranteed to be symmetric, as stipulated by the LIM lemma.

Consider the effect of ν_2. We would like to argue that the noise vector ν_2 is 'distributed at random' by the permutation and each measurement is corrupted by $\|\nu_2\|_2^2/m$, approximately its fair share of $\|\nu_2\|_2^2$, where m is the number of measurements. Unfortunately, the contributions of ν_2 to the various measurements are not independent as ν_2 is permuted, so we cannot use such a simple analysis. Nevertheless, they are negatively correlated and thus the Chernoff bound still applies [DP96].

For a more complete analysis, set $I = \left\{i : (\boldsymbol{P}^{-1}\nu_2)_i \geq \frac{4}{\sqrt{m}}\|\nu_2\|_2\right\}$, so $|I| \leq \frac{m}{16}$. We say row i in the measurement matrix is *heavily corrupted* if $i \in I$. The measurement matrix is decomposed into B blocks (in the sense of error-correcting codes) of rows, these blocks are used to identify a heavy hitter or to estimate a signal position value. For identification, we have a block size of $O(\log\log N)$ and an explicit encoding/decoding procedure, while for estimation, we have a block size of $O(\log N)$ and a trivial encoding/decoding procedure. If some of the blocks are corrupted by the measurement noise, we may still be able to decode accurately. In order to ascertain how many blocks are heavily corrupted and what influence this has on the decoding procedure, we must analyze how the random permutation disperses I over the blocks.

Let $X_i = \mathbb{1}_{\{i\in I\}}$ and $\Lambda_1, \ldots, \Lambda_B$ be the set of indices of the blocks. Define $Y_k = \sum_{i\in\Lambda_k} X_i$ $(1 \leq k \leq B)$, to be the number of corrupted measurements in block k. The most desirable situation is that, as in LIM Lemma, $Y_k \leq \frac{|\Lambda_k|}{16}$ for all k, which is, however, extremely unlikely to happen. We could only expect something weaker. It follows from Chernoff bound that

$$\Pr\left\{|Y_k| \geq \frac{|\Lambda_k|}{6}\right\} \leq e^{-0.05|\Lambda_k|}.$$

Since $|\Lambda_k| = \Omega(\log\log N)$ in the encoding portion of the identification matrix $\boldsymbol{D}$, the probability above is $\frac{1}{\log^{\Omega(1)} N}$. Furthermore, there are $O(\log N)$ rows in a block of the hashing portion of $\boldsymbol{D}$, thus the union bound gives $o(1)$ failure probability of $|Y_k| \leq \frac{|\Lambda_k|}{6}$ for all k corresponding to a specific row in the hashing matrix.

Suppose there are g good hashing rows, $g = \sum g_t$, where $g_t = \Omega(j)$ (recall that the identification matrix $\boldsymbol{D}$ has $\Theta(j)$ layers) is the number of good rows containing heavy hitter t. From the negative association, the probability that $\frac{4}{5}g_t$ good rows are heavily corrupted is at most $o(1)^{g_t} = O(c^{-j})$ for some constant c, in which case we say the heavy hitter t is ruined. By the Markov inequality, only a small fraction of heavy hitters are ruined except with small probability $O(c^{-j})$, which is sufficient for recovery in the j-th iteration. Similar arguments work for the estimation matrix, where heavy hitters and non-heavy hitters are discussed separately. Summing the failure probability over j, we conclude that except with probability $o(1)$, the post-measurement noise ν_2 will be dispersed favorably, *i.e.*, the blocks corresponding to most heavy hitters have at most 1/6 of the measurements being heavily corrupted.

Next, we claim that with the measurement noise dispersed favorably, we only need an increase of a constant factor in the number of measurements to accommodate the noise. Let $\{X_i\}_{i=1}^m$ be i.i.d. Bernoulli random variables with parameter p that denote the failure of measurement i (in which case $X_i = 1$). Let $\lambda > p$ be the thresholding constant. The Chernoff bound tells us that the failure probability of a fraction λm of all the measurements is

$$\Pr\left\{\sum_{i=1}^{m} X_i \geq m\lambda\right\} \leq e^{-C(\delta)mp},$$

where $\delta = \frac{\lambda}{p} - 1$ and $C(\delta)$ is a constant depending on δ. With post-measurement noise, a fraction θ of X_i's are corrupted and not usable, where θ is sufficiently small such that $\theta + p < \lambda$. (For instance, following the above constants, we have that $\theta = \frac{1}{6}$ and we can adjust p and λ in the arguments of the case $\nu_2 = 0$ such that $\theta + p < \lambda$.) The threshold becomes $m(\lambda - \theta)$ instead of $m\lambda$, and thus

$$\Pr\left\{\sum_{i=1}^{m} X_i \geq m(\lambda - \theta)\right\} \leq e^{-C(\zeta)mp},$$

where $\zeta = \frac{\lambda-\theta}{p} - 1$. It is now clear that m needs to increase by only a constant factor, namely $\frac{C(\delta)}{C(\zeta)}$, to keep the probability bound unchanged. Henceforth, we may assume ν_2 corrupts each measurement in $\mathbf{\Phi x}$ by at most $16\|\nu_2\|^2/m$.

We turn now to the complete proof of the invariant (Claim 2.2) with post-measurement noise. With assumed normalization $\|\nu_1\|_2 = \|\nu_2\|_2 = 1$, we have that $\left\|\nu_1^{(0)}\right\|_2 = 1$.

In iteration j, we make ℓ of the LIM lemma equal to $k/2^j$, η of the LIM lemma is $\Theta(\epsilon\beta^j)$, and $\delta = 2^{-j}$, for $\beta < 1$ to be specified below. It is straightforward to confirm that $\left\|\sigma^{(j+1)}\right\|_0 \leq k/2^{j+1}$, provided the invariant held at the previous iteration. We now turn to $\left\|\nu_1^{(j+1)}\right\|_2$.

At the beginning of the j'th iteration,

$$\left\|\nu_1^{(j)}\right\|_2^2 \le 1 + \epsilon\Big(1 - \Big(\frac{3}{4}\Big)^j\Big).$$

This means that $1 \le \left\|\nu_1^{(j)}\right\|_2^2 \le 2$ remains unchanged up to the factor 2. By the LIM lemma and the discussion at the beginning of Section 2.4.1.2, each repetition gives, with high probability, an estimate with

$$\left\|\nu_1^{(j+1)}\right\|_2^2 - \left\|\nu_1^{(j)}\right\|_2^2 \quad \text{at most} \quad \epsilon\beta^j\left(\left\|\nu_1^{(j)}\right\|_2^2 + 16c^{-j}\left\|\nu_2\right\|_2^2\right).$$

It follows that the median over repetitions of this estimate satisfies the same bound with high probability. Since $1 \ll c^{-j}$, it follows that $1 + c^{-j} \approx c^{-j}$. If we put $\beta \approx 5/8$ and $c \approx 5/6$, the invariant is satisfied.

We have proved that the algorithm returns $\widehat{\mathbf{x}}$ with approximation error

$$\left\|\mathbf{x} - \widehat{\mathbf{x}}\right\|_2^2 \le (1+\epsilon)\left\|\nu_1\right\|_2^2 + 16\epsilon\left\|\nu_2\right\|_2^2.$$

Now, replace 16ϵ by ϵ to achieve the desired form of error bound while introducing only a constant to the time cost. □

2.4.2 Efficiency

2.4.2.1 Number of Measurements

The number of measurements in iteration j is computed as follows. There are $\log(1/\delta) = O(j)$ parallel repetitions in iteration j. They each consist of $O(k/(\epsilon\beta^j 2^j)\log(N/k))$ measurements, where $\beta = 5/8$. That is, the number of measurements is

$$\Theta\left(\frac{jk}{\epsilon}\left(\frac{4}{5}\right)^j \log\frac{N}{k}\right) = \frac{k}{\epsilon}\log\frac{N}{k}\left(\frac{4}{5} + o(1)\right)^j.$$

Thus we have a sequence bounded by a geometric sequence with ratio less than 1. The sum, over j, is bounded by $O((k/\epsilon)\log(N/k))$.

Note that the dimension of the random permutation matrix $\boldsymbol{P}$ matches the number of rows, $O((k/\epsilon)\log(N/k))$.

2.4.2.2 Encoding, Decoding, and Update Time

The encoding time is bounded by N times the number of non-zeros in each column of the measurement matrix. This was analysed above in Section 2.4.1; there are $\log(j)\log(N/k)$ non-zeros per column in iteration j, for $j \leq O(\log(k))$, so the total is $\log^2(k)\log(N/k)$ non-zeros per column. This is suboptimal by the factor $\log^2(k)$. By comparison, however, some proposed methods use dense matrices, which are suboptimal by the exponentially-larger factor k.

When constructing the matrix for measuring the original signal or some intermediate representation, our algorithm will need to find, quickly, the bucket to which an index i is hashed and a codeword for i, where i is in the range $1 \leq i \leq N$. Note that it is crucial that we use $O(\log(N/B))$ bits for the codeword to meet the sketch length lower bound $O(k\log(N/k))$ (instead of $O(k\log N)$), where B is the number of buckets, and not $\log(N)$ bits. This means we need to find codewords for just the i's hashed to a particular bucket. Upon decoding, we need to be able to find i from its codeword, quickly.

We can use a pseudorandom number generator that hashes i to a bucket j if $jN/B \leq ai+b \bmod N < (j+1)N/B$ for random a and b. Then encode i by $E(ai+b-jN/B)$, assuming quick encoding for numbers in the contiguous range 0 to $N/B-1$. To decode, knowing j, we first recover $ai+b-jN/B$, whence we easily recover $ai+b$ and subsequently i. Define hash function $f(i) = ai+b$ on $\mathbb{Z}_N$. The non-zeroes at positions $i_1, \ldots, i_h$ in a row of the hashing matrix are ordered such that $f(i_1) < f(i_2) < \cdots < f(i_h)$. An example is given in Figure 2.3.1.

Another issue is the time to find and to encode and to decode the error-correcting code. Observe that the length of the code is $O(\log\log N)$. We can afford time exponential in the length, *i.e.*, time $\log^{O(1)} N$, for finding, encoding, and decoding the code. These tasks are straightforward in that much time. Alternatively, we can use a lookup table of size polylog(N) to decode a code in $O(1)$ time.

CHAPTER 3

Sparse Recovery in ℓ_1/ℓ_1 Error Metric

3.1 Introduction

The first sublinear-time algorithm for the (for-all) ℓ_1/ℓ_1 sparse recovery problem was given in [PS12], though that algorithm had limitations, namely

- The runtime, while sublinear, was $\sqrt{kN}$ or, more generally, of the form $k^{1-\alpha}N^{\alpha}$ for any constant $\alpha > 0$. That algorithm did not achieve runtime polynomial in $k\log(N)/\epsilon$.
- [PS12] required a precomputed table of size $Nk^{0.2}$.
- [PS12] was far from optimal in its dependence of the number of measurements on ϵ.

In this work, we rectify the above limitations, assuming the (modest) restriction that $\epsilon < \log k/\log N$. See the theorems for precise statement, in which the restriction is slightly weaker.

We also make the measurement dependence on ϵ optimal. The best known lower bound for the ℓ_1/ℓ_1 for-all problem is $\Omega(k/\epsilon^2 + (k/\epsilon)\log(\epsilon N/k))$ [NNW12], which is also the best known lower bound for the ℓ_2/ℓ_1 for-all problem. Our algorithm can be viewed as using $O((k/\epsilon)\log(N/k)((\log N/\log k)^{\alpha} + 1/\epsilon))$ measurements (when $\epsilon < (\log k/\log N)^{\alpha}$ it becomes $O(k/\epsilon^2\log(N/k)))$, which is suboptimal only in a logarithmic factor. Note that the dependence on ϵ is ϵ^{-2}, which is optimal.

Overview of Techniques. Our overall approach is as follows, building on [PS12] and [GNP$^+$13]. The matrix Φ implements a two-stage hashing process, in which the original indices $[N]$ are first hashed into B buckets. We then sum the items in each bucket, getting a new signal of length B, in which a heavy hitter $\mathbf{x}_i$ in the original signal is likely to dominate its bucket. Then, in time about B, search the buckets to find the heavy buckets. Finally, determine which element in the found heavy bucket is the original heavy hitter. All this suffices for a *weak* system, that identifies all but $k/2$ of the heavy hitters. We then repeat with smaller (easier) sparsity parameter $k/2 < k$ and smaller (harder) distortion parameter $(3/4)\epsilon < \epsilon$, giving a number of measurements with leading cost factor $(k/2)(4/3\epsilon)^2 = (8/9)k/\epsilon^2 < k/\epsilon^2$. Summing the geometric progression gives the result we need.

The main difficulty is *identification*, i.e., to determine which element in the found heavy bucket is the original heavy hitter. To overcome this, we assign to each position i in $[N]$ a message $\mathbf{m}_i$ that describes its position, such that given $\mathbf{m}_i$ we can identify i. Based on $\mathbf{m}_i$, we build chunks $\{\mathbf{m}_{i,j}\}_j$ such that given a large fraction of the chunks we can recover $\mathbf{m}_i$ via an error correcting code. In particular,

- We show that we can encode a message of $\log(N/k)$ bits for each $i \in [N]$ into the matrix Φ so that, intuitively, the messages corresponding to the large-magnitude entries of $\mathbf{x}$ are decoded properly when $\widehat{\mathbf{x}}$ is produced. In our algorithm, with its two-stage hashing, we will only be able to encode fewer bits at a time. The message we encode as $\mathbf{m}_{i,j}$ in each chunk is some the $\log(N)$ bits of i together with some pointer information that helps us to cluster all the necessary bits of i. This substitutes for the full-space look-up table in [PS12] and leads to better runtime than [GNP$^+$13].

- To decode the message $\mathbf{m}_i$ from the chunks, we use a construction similar to network coding. Our work is not the first to consider list recovery; [INR10] introduces the idea in the context of combinatorial group testing. Like [INR10], we consider weak list recovery, that is, we need only to decode a good fraction of the $\mathbf{m}_i$'s corresponding to heavy hitters. The idea of list recovery is also used in [GNP$^+$13], where the list decoding, however, would affect the hashing and the hashing was thus required to be sufficiently random. In our algorithm, the messages $\{\mathbf{m}_i\}$ are independent of the hashing, which enables us to obtain a better result.

In addition, instead of presenting the algorithm as a hashing algorithm, however, we use randomness to build an unbalanced expander (with additional properties). Then we show that our algorithm works (deterministically) with any unbalanced expander having the appropriate properties.

Our contributions. Our contributions are as follows.

- We give an algorithm for sparse recovery in the for-all setting, under a modest restriction on the distortion factor ϵ, having the number of measurements that matches the best upper bound (which was attained by super-linear algorithms, e.g., [IR08]), and optimal in runtime up to a power.
- We hope our algorithm can be extended from the 1-norm to the mixed norm guarantee and that the restriction on ϵ can be weakened or eliminated. Thus our algorithm may be a stepping stone to the final algorithm.
- Our techniques may be useful in other contexts for soft-decoding or network coding.

Organization. We build the algorithm in a bottom-up approach, first building the weak system and then the overall algorithm based on the weak system. In Section 3.2 we review some properties of expanders, then we set out to build our weak system in the next three sections. In Section 3.3, we show that provided with good identification results, unbalanced expanders with appropriate properties will give a weak system. In Section 3.4, we show that the required properties are satisfied by our two-stage hashing structure. Our construction of weak system culminates in Section 3.5, where we shall show how to achieve good identification via message encoding and decoding. Then we build the overall algorithm on the weak system in Section 3.6. Finally we close with open problems in Section 3.7.

3.2 Preliminaries

Our main algorithm will be built on regular graph expanders and unbalanced bipartite expanders. In this section we review some properties of expanders. Let n, m, d, ℓ be positive integers and ϵ, κ be positive reals. The following two definitions are adapted from [GUV09].

Definition 3.1 (expander)**.** An (n, ℓ, κ)-expander is a graph $G(V, E)$, where $|V| = n$, such that for any set $S \subseteq V$ with $|S| \leq \ell$ it holds that $|\Gamma(S)| \geq \kappa|S|$.

When n is clear from the context, we abbreviate the expander as (ℓ, κ)-expander.

Definition 3.2 (bipartite expander)**.** An $(n, m, d, \ell, \epsilon)$-bipartite expander is a d-left-regular bipartite graph $G(L \cup R, E)$ where $|L| = n$ and $|R| = m$ such that for any

Paper	A/E	Number of Measurements	Column sparsity/ Update time	Decode time	Approx. error	Noise
[CCFC02]	E	$k \log^c N$	$\log^c N$	$N \log^c N$	$\ell_2 \leq C\ell_2$	
[CM06]	E	$k \log^c N$	$\log^c N$	$k \log^c N$	$\ell_2 \leq C\ell_2$	
[GLPS12]	E	$\epsilon^{-1} k \log(N/k)$	$\log^c N$	$\epsilon^{-1} k \log^c N$	$\ell_2 \leq (1+\epsilon)\ell_2$	Y
[Don06, CRT06]	A	$k \log(N/k)$	$k \log(N/k)$	LP	$\ell_2 \leq (C/\sqrt{k})\ell_1$	Y
[GSTV07]	A	$\epsilon^{-2} k \log^c N$	$\epsilon^{-2} k \log^c N$	$\epsilon^{-4} k^2 \log^c N$	$\ell_2 \leq (\epsilon/\sqrt{k})\ell_1$	Y
[GSTV06]	A	$k \log^c N$	$\log^c N$	$k \log^c N$	$\ell_1 \leq (C \log N)\ell_1$	Y
[IR08]	A	$\epsilon^{-2} k \log(N/k)$	$\epsilon^{-1} \log(N/k)$	$N \log(N/k)$	$\ell_1 \leq (1+\epsilon)\ell_1$	Y
[PS12] (any integer ℓ)	A	$\ell^c \epsilon^{-3} k \log(N/k)$	$\ell^c \epsilon^{-3} \log(N/k) \log k$	$\ell^c \epsilon^{-3} k (N/k)^{1/\ell}$	$\ell_1 \leq (1+\epsilon)\ell_1$	Y
Our result (any $\beta > 0$) (restrictions on ϵ apply)	A	$\epsilon^{-2} k \log(N/k)$	$\epsilon^{-1} \log(N/k)$	$k^{1+\beta}(\epsilon^{-1} \log N)^c$	$\ell_1 \leq (1+\epsilon)\ell_1$	Y
Lower bound 'A'	A	$\epsilon^{-2} k \log(N/k)$	$\log(N/k)$	$k \log(N/k)$	$\ell_2 \leq (\epsilon/\sqrt{k})\ell_1$	Y

Table 3.1: Summary of known sparse recovery results with dependence on ϵ. Some constant factors are omitted for clarity. "LP" denotes (at least) the time to do a linear program of size at least N. The column "A/E" indicates whether the algorithm works in the for-all (A) model or the for-each (E) model. The column "noise" indicates whether the algorithm tolerates noisy measurements. Measurement and decode time dependence on ϵ, where applicable, is polynomial. The constants c could be different in different occurrences. The lower bound on number of measurements in table above is, in fact, the best upper bound attained by super-linear algorithms.

$S \subseteq L$ with $|S| \leq \ell$ it holds that $|\Gamma(S)| \geq (1-\epsilon)d|S|$, where $\Gamma(S)$ is the neighbour of S (in R).

When n and m are clear from the context, we abbreviate the expander as (ℓ, d, ϵ)-bipartite expander. When d is also clear from the context, we simply write (ℓ, ϵ)-bipartite expander.

Consider the adjacency matrix A_G of an d-regular expander G. It always holds that the biggest eigenvalue of A_G is d. Let $\lambda(G)$ denote the largest absolute value of any other eigenvalue. The following theorem is now well-known.

Theorem 3.3 ([FKS89])**.** *For all sufficiently large n and even d, there exists a d-regular expander G such that $|V(G)| = n$ and $\lambda(G) \leq C\sqrt{d}$ for some absolute constant $C > 0$.*

Remark 3.4. The Alon-Boppana bound states that $\lambda(G) \geq 2\sqrt{d-1} - o(1)$ [Alo86], which implies that the theorem above is optimal up to a constant factor.

Next we present a result due to Upfal [Upf92], implicitly contained in the proof of Lemma 1 and 2 therein. It states that there exists a expander graph of n nodes and constant degree, such that after removing a constant fraction of nodes the remaining subgraph contains an expander of size $\Omega(n)$.

Theorem 3.5 ([Upf92])**.** *Let G be a δ-regular expander of n nodes such that $\lambda(G) \leq C\sqrt{\delta}$, where δ is a (sufficiently large) constant. There exist absolute constants $\alpha, \zeta > 0$ and $\kappa > 1$ such that after removing an arbitrary set T of nodes with $|T| \leq \zeta n$ from G, the remaining graph contains a subgraph G' such that $|V(G')| \geq \alpha n$ and G' is a $(|V(G')|, n/2, \kappa)$ graph expander.*

Remark 3.6. A closer examination of the proof reveals that the assumption in the theorem could be weakened to $\lambda(G) \leq c\delta$ for some small constant $c > 0$. This would enable the use of graphs G with larger $\lambda(G)$, which are easier to construct.

The following definitions concern hashing, in which the parameters N, B_1, B_2, d_1, d_2 are positive integers. We also adopt the conventional notation that $[m] = \{1, 2, \ldots, m\}$.

Definition 3.7 (one-layer hashing scheme)**.** The (N, B, d) (one layer) hashing scheme is the uniform distribution on the set of all functions $f : [N] \to [B]^d$.

Each instance of such a hashing scheme induces a d-left-regular bipartite graph of Bd right nodes. When N is clear from the context, we simply write (B, d) hashing scheme.

Definition 3.8 (two-layer hashing scheme)**.** An (N, B_1, d_1, B_2, d_2) (two-layer) hashing scheme is a distribution μ on the set of all functions $f : [N] \to [B_2]^{d_1 d_2}$ defined as follows. Let g be a random function subject to the (N, B_1, d_1) hashing scheme and $\{h_{i,j}\}_{i\in[d_1],j\in[d_2]}$ be a family of independent functions subject to the (B_1, B_2, d_2) hashing scheme which are also independent of g. Then μ is defined to be the distribution induced by the mapping

$$x \mapsto (h_{1,1}(g_1(x)), \ldots, h_{1,d_2}(g_1(x)), h_{2,1}(g_2(x)), \ldots, h_{2,d_2}(g_2(x)), \ldots,$$
$$h_{d_1,1}(g_{d_1}(x)), \ldots, h_{d_1,d_2}(g_{d_1}(x)))\,.$$

Each instance of such hashing scheme gives a d_1d_2-left-regular bipartite graph of $B_2d_1d_2$ right nodes. When N is clear from the context, we simply write (B_1, d_1, B_2, d_2) hashing scheme. Conceptually we hash N elements into B_1 buckets and repeat d_1 times, those buckets will be referred to as first-layer buckets; in each of the d_1 repetitions, we hash B_1 elements into B_2 buckets and repeat d_2 times, those buckets will be referred to as second-layer buckets.

Related to hashing, we introduce an isolation property on bipartite graphs.

Definition 3.9. Let $G = (L \cup R, E)$ be a bipartite graph and $S, T \subseteq L$. Define

$$U_S(T) = \{y \in R : (x, y) \in E \text{ for some } x \in T \text{ while } (z, y) \notin E \text{ for all } z \in T, z \neq x\}.$$

Definition 3.10 (isolation property)**.** An $(n, m, d, \ell, \epsilon)$ bipartite expander G is said to satisfy the (L, η, ζ)-isolation property if for any set $S \subset L(G)$, $|S| \leq L$, there exists $S' \subset S$, $|S'| \geq (1 - \eta)|S|$ such that $|U_S(\{x\})| \geq (1 - \zeta)d$ for all $x \in S'$.

3.3 Weak System

Usually we decompose a signal $\mathbf{x}$ into two parts of disjoint support, say, $\mathbf{x} = \mathbf{y}+\mathbf{z}$, where $\mathbf{y}$ has small support and $\mathbf{z}$ has small norm. Loosely speaking, we call $\mathbf{y}$ the *head* and $\mathbf{z}$ the *tail*. To simplify the language we may also use head to refer to $\operatorname{supp}\mathbf{y}$. We aim to recover the head items, i.e., the elements in $\mathbf{y}$. Introduced in [PS12], a *weak system* takes an additional input, some set I of indices (called the candidate set), and tries to estimate $\mathbf{x}_i$ for $i \in I$, hoping to recover some head items with estimate error dependent on $\|\mathbf{z}\|_1$. It is shown in [PS12] that when I contains the entire head, we can always recover a good fraction of the head. In this paper we make a slight modification on the definition of weak system as below. We only need I to contain a good fraction of the head instead of the entire head.

Definition 3.11 (Weak system)**.** A *Weak system* consists of parameters N, s, η, ζ, an m-by-N *measurement matrix* $\mathbf{\Phi}$, and a *decoding algorithm* $\mathcal{D}$, that satisfy the following property:

For any $\mathbf{x} \in \mathbb{R}^N$ that can be written as $\mathbf{x} = \mathbf{y} + \mathbf{z}$, where $|\operatorname{supp} \mathbf{y}| \leq s$ and $\|\mathbf{z}\|_1 \leq 3/2$, given the measurements $\mathbf{\Phi x}$ and a subset $I \subseteq [N]$ such that $|I \cap \operatorname{supp} \mathbf{y}| \geq (1 - \frac{\zeta}{2})|\operatorname{supp} \mathbf{y}|$, the decoding algorithm $\mathcal{D}$ returns $\widehat{\mathbf{x}}$, such that $\mathbf{x}$ admits the following decomposition:

$$\mathbf{x} = \widehat{\mathbf{x}} + \widehat{\mathbf{y}} + \widehat{\mathbf{z}},$$

where $|\operatorname{supp} \widehat{\mathbf{x}}| = O(s)$, $|\operatorname{supp} \widehat{\mathbf{y}}| \leq \zeta s$, and $\|\widehat{\mathbf{z}}\|_1 \leq \|\mathbf{z}\|_1 + \eta$.

Intuitively, $\mathbf{y}$ and $\mathbf{z}$ will be the head and the tail of the residual $\mathbf{x} - \widehat{\mathbf{x}}$, respectively. We shall use the weak system on $\mathbf{x} - \widehat{\mathbf{x}}$ (with a different candidate set I) and iterate, approximating the original $\mathbf{x}$. We shall give the details in Section 3.6. To obtain a weak system, we critically rely on the following two lemmata.

Lemma 3.12 (Noise)**.** *Let $\alpha > 1$ and $t > \alpha k$. Let Φ be the adjacency graph of an $(n, m, d, 2\alpha k, \epsilon)$ expander with $\epsilon < 1/2$. Let $\mathbf{x} \in \mathbb{R}^n$ such that $|\mathbf{x}_1| \geq |\mathbf{x}_2| \geq \cdots \geq |\mathbf{x}_n|$. Let $I = [\alpha k]$, then*

$$\|(\Phi(\mathbf{x} - \mathbf{x}_{[t]}))_{\Gamma(I)}\|_1 \leq 4\epsilon d(\|\mathbf{x} - \mathbf{x}_{[t]}\|_1 + \alpha k |\mathbf{x}_{t+1}|).$$

Proof. Partition $\{1, \ldots, N\}$ into blocks $I \cup H_1 \cup B_1 \cup B_2 \cup \ldots$, where $H_1 = \{\alpha k + 1, \ldots, t\}$ and $B_i = \{t + (i-1)\alpha k + 1, \ldots, t + i\alpha k\}$ for $i \geq 1$. Consider $\mathbf{x}$ restricted to a block B_i.

Case 1. $\mathbf{x}_{B_i}$ is flat, i.e., $|\mathbf{x}_{t+i\alpha k}| \geq |\mathbf{x}_{t+(i-1)\alpha k+1}|/2$. Consider all $d|B_i|$ edges in the expander emanating from B_i. Suppose that Z edges of them are incident to $\Gamma(I)$, then

$$|\Gamma(I) \cup \Gamma(B_i)| \leq \epsilon d(|I| + |B_i|) - Z.$$

On the other hand, by the expansion property,

$$|\Gamma(I) \cup \Gamma(B_i)| \geq (1 - \epsilon) d(|I| + |B_i|),$$

which implies that

$$Z \leq \epsilon d(|I| + |B_i|) \leq 2\epsilon\alpha k d.$$

Each of the Z edges sends a noise of x_i to $\Gamma(I)$, therefore

$$\|(\Phi \mathbf{x}_{B_i})_{\Gamma(I)}\| \leq Z \cdot \max_{i \in B_i} |\mathbf{x}_i| \leq 2\epsilon\alpha k d \cdot |\mathbf{x}_{t+(i-1)\alpha k+1}| \leq 4\epsilon d \|\mathbf{x}_{B_i}\|_1,$$

where the last inequality follows from the fact that $\mathbf{x}_{B_i}$ is flat so that $\alpha k |\mathbf{x}_{t+(i-1)\alpha k+1}| \leq 2\|\mathbf{x}_{B_i}\|_1$.

Case 2. x_{B_i} is not flat, then $|\mathbf{x}_{t+i\alpha k}| < |\mathbf{x}_{t+(i-1)\alpha k+1}|/2$. Let

$$J = \{i \in B_i : |\mathbf{x}_i| < |\mathbf{x}_{t+(i-1)\alpha k+1}|/2\}.$$

Increase $|\mathbf{x}_i|$ for all $i \in J$ so that $|\mathbf{x}_i| = |\mathbf{x}_{t+(i-1)\alpha k+1}|/2$ and $\mathbf{x}_{B_i}$ becomes flat, and this increases $\|\mathbf{x}_{B_i}\|_1$ by at most $\alpha k |\mathbf{x}_{t+(i-1)\alpha k+1}|/2$. Invoking Case 1, we obtain that

$$\|(\Phi \mathbf{x}_{B_i})_{\Gamma(I)}\|_1 \leq 4\epsilon d \left(\|\mathbf{x}_{B_i}\|_1 + \frac{\alpha k \left|\mathbf{x}_{t+(i-1)\alpha k+1}\right|}{2} \right).$$

Now we go back to the entire $\mathbf{x}$. Suppose that $B_{i_1}, \ldots, B_{i_q}$ are not flat, then by triangle inequality we shall have

$$\|(\Phi(\mathbf{x} - \mathbf{x}_t))_{\Gamma(I)}\|_1 \leq 4\epsilon d \|\mathbf{x} - \mathbf{x}_t\|_1 + 4\epsilon d \cdot \frac{\alpha k}{2} \sum_{p=1}^{q} \left|\mathbf{x}_{t+(i_p-1)\alpha k+1}\right|.$$

Observe that $|\mathbf{x}_{t+(i_p-1)\alpha k+1}| \leq |\mathbf{x}_{t+(i_{p-1}-1)\alpha k+1}|$ for $p \geq 2$, we can show inductively that

$$|\mathbf{x}_{t+(i_p-1)\alpha k+1}| \leq \frac{|\mathbf{x}_{t+1}|}{2^{p-1}}, \quad p \geq 1,$$

whence it follows that

$$\|(\Phi(\mathbf{x} - \mathbf{x}_{[t]}))_{\Gamma(I)}\|_1 \leq 4\epsilon d (\|\mathbf{x} - \mathbf{x}_{[t]}\|_1 + \alpha k |\mathbf{x}_{t+1}|). \qquad \square$$

In the usual decomposition, the head contains the entries with large coordinate values, which will be referred to as *heavy hitters*. If a heavy hitter is failed to be recovered, it must have been displaced by another entry, loosely called a decoy, in the recovered signal. The next lemma bounds the number of decoys.

Lemma 3.13 (Decoys). *Suppose that G is a $(4s, \frac{\epsilon}{512})$-bipartite expander which satisfies the $(\frac{9s}{\epsilon}, \beta\epsilon, \zeta)$-isolation property, where $\frac{1}{2} - \zeta > 576\beta$. Let $\mathbf{x} \in \mathbb{R}^n$ be a signal satisfying the assumption in the Weak system, and let $\mathbf{x}' \in \mathbb{R}^n$ be the estimates defined* *as*

$$\mathbf{x}'_i = \operatorname*{median}_{u\in\Gamma(\{i\})} \sum_{(u,v)\in E} \mathbf{x}_u, \quad i \in [N].$$

Define

$$D = \{i \in [N] : |\mathbf{x}_i - \mathbf{x}'_i| \geq \epsilon/(4s)\},$$

then $|D| < s/8$.

Proof. Without loss of generality, assume that $|D| = s/8$, or we replace D with a subset of size exactly $s/8$. Also assume that $|\mathbf{x}_1| \geq |\mathbf{x}_2| \geq \cdots \geq |\mathbf{x}_n|$. Suppose that $|\mathbf{x}_i| \geq \epsilon/(2s)$ for all $i \in H := \operatorname{supp}\mathbf{y}$, otherwise we can place the violated i's into $\mathbf{z}$, causing $\|\mathbf{z}\|_1$ to increase by at most $s \cdot \epsilon/(2s) = \epsilon/2$, so we would have $\|\mathbf{z}\|_1 \leq 2$. Let $T = H \cup D \cup \{i : |\mathbf{x}_i| \geq \epsilon/(4s)\}$, then $t := |T| \leq \|\mathbf{z}\|_1/(\epsilon/(4s)) + |D| + |H| \leq 9s/\epsilon$.

Note that $|\mathbf{x}_{t+1}| \leq \epsilon/(4s)$. Taking $\alpha = 2$ in Lemma 3.12, we know that

$$\|(\Phi(\mathbf{x} - \mathbf{x}_{(t)}))_{\Gamma(H\cup D)}\|_1 \leq 4 \cdot \beta\epsilon d \left(\frac{3}{2} + \frac{\epsilon}{2} + 2s \cdot \frac{\epsilon}{4s}\right) \leq 8\beta\epsilon d.$$

By the isolation property, there are at most $\frac{9s}{\epsilon} \cdot \frac{\epsilon}{144} = \frac{s}{16}$ elements in T which are not isolated in at least ζd nodes from other elements in T. This implies that at least $s/16$ elements in D are isolated in at least ζd nodes from other elements in T.

A decoy at position i receives at least $\epsilon/(4s)$ noise in at least $(1/2 - \zeta)d$ isolated nodes of $\Gamma(\{i\})$, hence in total, a decoy element receives at least $\epsilon(1/2 - \zeta)d/(4s)$ noise. Therefore the $s/16$ decoys overall should receive noise at least

$$\frac{\epsilon(\frac{1}{2} - \zeta)d}{4s} \cdot \frac{s}{16} > 8\beta\epsilon d \geq \|(\Phi(x - x_t))_{\Gamma(H\cup D)}\|_1,$$

which is a contradiction. Therefore $|D| < s/8$. □

Remark 3.14. Despite the fact that we have specified various constants (such as 4, $\frac{1}{512}$, 9, etc) in the lemma above, the constants can be flexibly adjusted such that the number of decoys is at most ζs for any given small $\zeta > 0$ with appropriate choices of other constants.

In [PS12] it is essentially proved that

Theorem 3.15 (Weak)**.** *Suppose that Φ is the adjacency matrix of an $(N, Bd, d, 4s, \eta)$ bipartite expander such that (a) $d = O(\frac{1}{\eta\zeta^2}\log\frac{N}{s})$ and $B = O(\frac{d}{\zeta\eta})$ and (b) it is an instance of a (B, d) hashing scheme. With appropriate instantiations of constants, Φ gives a correct Weak system that runs in time $O(|I|\eta^{-1}\zeta^{-2}\log(N/s))$.*

Algorithm 3.1 Weak system.

```
Input: $N$, $s$, $\mathbf{\Phi}$ (adjacency matrix of a $d$-left-regular expander $G$), $\mathbf{\Phi x}$, and $I$
Output: $\widehat{\mathbf{x}}$
  for $j \leftarrow 1$ to $d$ do
      for each $i \in I$ do
          $\mathbf{x}_i^{(j)} \leftarrow \text{median}_{u \in \Gamma(\{i\})} \sum_{(u,v)\in E} \mathbf{x}_u$        $\triangleright$ each sum is an element of input $\mathbf{\Phi x}$
  for each $i \in I$ do
      $\mathbf{x}'_i \leftarrow \text{median}_{1\le j\le d}\, \mathbf{x}_i^{(j)}$
  $\widehat{\mathbf{x}} \leftarrow$ top $O(s)$ elements of $\mathbf{x}'$
  return $\widehat{\mathbf{x}}$
```

Proof. The proof is essentially the same as [PS12, Lemma 4]. It follows from Lemma 3.13 that with appropriate choices of constants, that there are at most $\zeta s/4$ decoys and at least $(1-\zeta/4)s$ elements i in $\operatorname{supp}\mathbf{y}$ satisfy $|\mathbf{x}_i - \mathbf{x}'_i| \le \eta/(4s)$. Let $I' = I \cap \operatorname{supp}\mathbf{y}$. We describe below the construction of $\widehat{\mathbf{x}}$, $\widehat{\mathbf{y}}$ and $\widehat{\mathbf{z}}$.

- Elements $i \in \operatorname{supp}\widehat{\mathbf{x}}$ with a good estimate (to within $\pm\eta/(4s)$) contribute $\mathbf{x}_i - \widehat{\mathbf{x}}_i$ to $\widehat{\mathbf{z}}$. There are at most s of these, each contributing $\eta/(4s)$, for total contribution $\eta/4$ to $\widehat{\mathbf{z}}$.

- Elements $i \in \operatorname{supp}\widehat{\mathbf{x}}$ with a bad estimate (not to within $\pm\eta/(4s)$) contribute $\mathbf{x}_i - \widehat{\mathbf{x}}_i$ to $\widehat{\mathbf{y}}$. There are at most $\zeta s/4$ of these.

- Elements $i \in \operatorname{supp}\mathbf{z} \setminus \operatorname{supp}\widehat{\mathbf{x}}$ contribute $\mathbf{x}_i$ to $\widehat{\mathbf{z}}$. The ℓ_1 norm of these is at most $\|\mathbf{z}\|_1$.

- Elements $i \in I' \setminus \operatorname{supp}\widehat{\mathbf{x}}$ with a good estimate that are nevertheless displaced by another element $i' \in \operatorname{supp}\widehat{\mathbf{x}} \setminus \operatorname{supp}\mathbf{y}$ with a good estimate contribute to $\widehat{\mathbf{z}}$. There are at most s of these. While the value $\mathbf{x}_i$ may be large and make a large contribution to $\widehat{\mathbf{z}}$, this is offset by $\mathbf{x}_{i'}$ satisfying $|\mathbf{x}_{i'}| \ge |\widehat{\mathbf{x}}_{i'}| - \eta/(4s) \ge |\widehat{\mathbf{x}}_i| - \eta/(4s) \ge |\mathbf{x}_i| - \eta/(2s)$, which contributes to $\mathbf{z}$ but not to $\widehat{\mathbf{z}}$. Thus the net contribution to $\widehat{\mathbf{z}}$ is at most $\eta/(2s)$ for each of the s of these i, for a total $\eta/2$ contribution to $\widehat{\mathbf{z}}$.

- Elements $i \in I' \setminus \operatorname{supp}\widehat{\mathbf{x}}$ that themselves have bad estimates or are displaced by elements with bad estimates contribute $\mathbf{x}_i$ to $\widehat{\mathbf{y}}$. There are at most $\zeta s/4$ bad estimates overall, so there are at most $\zeta s/4$ of these.

- Elements $i \in I \setminus I'$ contribute to $\widehat{\mathbf{y}}$. There are at most $\zeta s/2$ of these.

It is clear that $|\operatorname{supp}\widehat{\mathbf{y}}| \le \zeta s$ and $\|\widehat{\mathbf{z}}\|_1 \le \|\mathbf{z}\|_1 + \eta$, as desired. The runtime is easy to verify. □

3.4 Hashing and Expander

It remains to show that a bipartite expander as required by Theorem 3.15 exists. In fact, by probabilistic methods, we can show that it can be attained by both one-layer and two-layer hashing schemes, with appropriate parameters. The proofs use standard techniques and are postponed to the end of the chapter.

3.4.1 One-layer hashing

Lemma 3.16. *For any $\epsilon \in (0, 1/4)$, $k \geq 1$, $\alpha \geq 1$ and $N = \Omega(\alpha k)$, a random one-layer (B, d) hashing scheme gives an $(\alpha k, \epsilon)$ bipartite expander with probability $\geq 1 - 1/N^c$, where $B = \Omega(\frac{\alpha k}{\epsilon})$ and $d = \Omega(\frac{1}{\epsilon} \log \frac{N}{k})$.*

In addition, hashing gives isolation property as well.

Lemma 3.17. *For any $\epsilon, \zeta \in (0, 1/4)$, $k \geq 1$, $\alpha \geq 1$ and $N = \Omega(k/\epsilon)$, a random one-layer (B, d) hashing scheme gives a bipartite graph with (L, ϵ, ζ)-isolation property with probability $\geq 1 - 1/N^c$, where $B = \Omega(\frac{k}{\zeta\epsilon})$, $d = \Omega(\frac{1}{\zeta\epsilon} \log \frac{N}{k})$, $L = O(k/\epsilon)$.*

Combining Lemma 3.16, Lemma 3.17 and Theorem 3.15, we have completed a clean formulation, in the language of expanders, of the result on weak system in [PS12].

3.4.2 Two-layer Hashing

Now we show that a two-layer hashing scheme, upon which our identification procedure will be built, also gives a desirable bipartite expander.

Lemma 3.18. *Let $\epsilon \in (0, 1/4)$, $k \geq 1$ and $N = \Omega(\max\{k/\epsilon^2, k^2\})$. A random two-layer (B_1, d_1, B_2, d_2) hashing scheme gives an $(N, B_2 d_1 d_2, d_1 d_2, 4k, \epsilon)$ bipartite expander with probability $\geq 1 - 1/N^c$, where $B_1 = \Omega(\frac{k}{\epsilon^2})$, $d_1 = \Omega(\frac{1}{\epsilon} \frac{\log(N/k)}{\log(B_1/k)})$, $B_2 = \Omega(\frac{k}{\epsilon})$ and $d_2 = \Omega(\log \frac{B_1}{k})$ with appropriate choices of constants.*

Remark 3.19. The constraint that $k = O(\sqrt{N})$ could be weakened to $k = O(N^{1-\xi})$ for any $\xi > 0$. The constants hidden in various $\Omega(\cdot)$ notations above will depend on ξ.

We show that this two-layer hashing scheme also gives a good isolation property.

Lemma 3.20. *Let $\alpha > 1$ be an arbitrarily constant and (B_1, d_1, B_2, d_2) be a two-layer hashing scheme with $B_1 = \Omega(\frac{k}{\zeta^\alpha \epsilon^{2\alpha}})$, $d_1 = \Omega(\frac{\alpha}{\alpha-1} \cdot \frac{1}{\zeta\epsilon} \frac{\log(N/k)}{\log(B/k)})$, $B_2 = \Omega(\frac{k}{\zeta\epsilon})$ and*

$d_2 = \Omega(\frac{1}{\zeta}\log\frac{B_1}{k})$. *Then with probability* $\geq 1 - 1/N^c$, *the two-layer hashing scheme with parameters prescribed above gives a bipartite graph with the* (L, ϵ, ζ)*-isolation property, where* $L = O(k/\epsilon)$.

3.5 Identification of Heavy Hitters

In the previous section, we showed how to estimate all candidates in a candidate set I quickly. The main runtime bottleneck is finding a non-trivial set $I \subset [N]$ of candidates. This is the topic of this section.

The overall strategy is as follows. Using the two-layer hashing scheme (B_1, d_1, B_2, d_2), we except that a heavy hitter dominates the first-layer buckets where it lands in $\Omega(d_1)$ repetitions. In each of such repetition, it is a heavy hitter in a signal of length B_1, and expected to be recovered using the Weak algorithm applied to the signal of length B_1 with $I = [B_1]$. After finding the heavy buckets in each repetition, the remaining problem is to extract the position of a heavy hitter i from the $\Omega(d_1)$ repetitions which contain i. To do this, we shall encode the index i in such a way that if we recover the buckets containing i in enough repetitions we shall be able to reconstruct i. Some previous work [GSTV07] uses bit-testing matrices to encode $\log N$ bits of i at the cost of a $\log N$ factor blow-up in the number of measurements. Our goal is to be more efficient in number of measurements. In the for-each setting, [GLPS12] uses error correcting code to tolerate a fraction of corrupted measurements at no additional measurement cost. Unfortunately the method has a relatively large failure probability and thus does not work in the for-all setting.

We introduce the following model of Weak list recovery.

Definition 3.21. Fix parameters m, N, s. The *Sparse Recovery Channel* takes an m-by-N matrix $\mathbf{\Phi}$ as input, chooses a signal $\mathbf{x}$ with decomposition $\mathbf{x} = \mathbf{y} + \mathbf{z}$ with $|\operatorname{supp}\mathbf{y}| \leq s$ and $\|\mathbf{z}\|_1 \leq O(1)$, and outputs $\mathbf{\Phi x}$.

Note that $\mathbf{x}$ may depend on $\mathbf{\Phi}$. Also note that *any* signal may be chosen by the channel and normalized so that $\|\mathbf{z}\|_1 \leq 3/2$. It will be convenient to assign the normalization at this point to match the Weak system (Defintion 3.11). Next, we define the *Weak Recovery Criterion* appropriate for this channel. See Figure 3.1 for detailed explanation.

Definition 3.22 (Weak list Recovery Criterion)**.** Fix parameters m, N, s, ϵ. Let $\mathbf{m}$ be a vector of β-bit messages, for $i \in [N]$, and suppose $\widehat{\mathbf{m}}$ is a list of possible index-

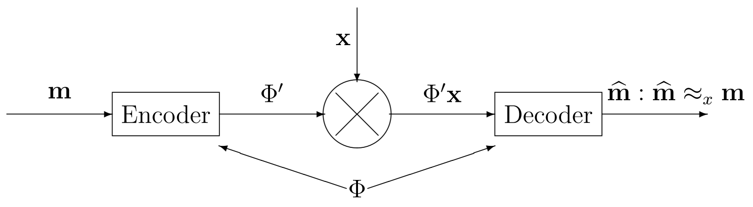

Figure 3.1: Sparse recovery channel. The encoder and decoder agree on some matrix $\mathbf{\Phi}$. The encoder takes messages $\mathbf{m}$ and produces a measurement matrix $\mathbf{\Phi}'$ based on $\mathbf{m}$ and $\mathbf{\Phi}$. The channel is fed with $\mathbf{\Phi}'$ and $\mathbf{x}$ and produces $\mathbf{\Phi}'\mathbf{x}$, from which the decoder tries to recover $\widehat{\mathbf{m}}$ in the sense of weak list recovery.

message pairs. We say that $\widehat{\mathbf{m}}$ is *correct in the List Weak sense* if, for at least $|\operatorname{supp}\mathbf{y}| - s/4$ indices i in $\operatorname{supp}\mathbf{y}$, we have $(i, \mathbf{m}_i) \in \widehat{\mathbf{m}}$.

So we encode N messages but, as we will see, we will want to produce $\widehat{\mathbf{m}}$ in time $k\operatorname{poly}(\log N) \ll N$, so $|\widehat{\mathbf{m}}| \ll N$. A correct recovery must include most messages that are important according to the $\mathbf{x}$. Whether we can code β bits for the m-measurement sparse recovery channel, and how effcient or simple the decoding process is, depends on the values of k, N, m, and β, like many results in coding theory.

The general setup is given in Algorithm 3.2. We break each message $\mathbf{m}_i$ associated with position i into d_1 chunks, $\mathbf{m}_{i,1}, \ldots, \mathbf{m}_{i,d_1}$. Note that $\mathbf{m}_i$ could be much longer than $\log N$ bits in order to guarantee a successful list recovery. Now in the j-th repetition of the d_1 repetitions, we obtain a signal $\widetilde{\mathbf{x}}$ of length B. Each $\widetilde{\mathbf{x}}_\ell$ is associated with a message that can be viewed as a weighted sum of $\mathbf{m}_{i,j}$ for positions i hashed into bucket ℓ. If a heavy hitter i is isolated in bucket ℓ and the noise is mild in this bucket, this weighted sum would be approximately $\mathbf{m}_{i,j}$, and we expect to recover $\mathbf{m}_{i,j}$ from the second-layer hashing, with inner encoding and decoding. Now we assume that we have recovered $\mathbf{m}_{i,j}$ for heavy hitter i in sufficiently many repetitions j. The central difficulty is to match $\mathbf{m}_{i,j}$ with $\mathbf{m}_{i,j'}$ with $j \neq j'$ in order to find enough fraction of $\mathbf{m}_i$ in the end. In order to solve this we shall encode some linking information in the node that will enable us to match $\mathbf{m}_{i,j}$ with $\mathbf{m}_{i,j'}$. This will be the topic of the next subsection, in which we shall use the Parverash-Vardy code to overcome this difficulty.

As a starter we first show how to code $\beta = \log(B/k)$ bits in the length-B Sparse Recovery Channel, and recover the messages associated with $\Omega(k)$ heavy hitters in the length B signal in time approximately B. This simple case illustrates our idea of encoding.

Algorithm 3.2 Back-pointer paradigm.

```
                                        ▷ $(B_1, d_1, B_2, d_2)$ hashing scheme
  for $i = 1$ to $N$ do
      Break: Break the information of $i$ into $d_1$ chunks
      Outer encoding: Encode the chunks with cluster info and against errors, getting $\{\mathbf{m}_{i,j}\}_{j=1}^{d_1}$
  end for
  for $j = 1$ to $d_1$ do
      Inner encoding: Encode $\mathbf{m}_{i,j}$, for $i \in [N]$
                  ▷ ... length $B_1$, $(B_2 d_2)$-measurement Sparse Recovery Channel ...
      Inner decoding: Recover $\widehat{\mathbf{m}}_j$ in the Weak List sense
      Record Side Info: Tag each element of $\widehat{\mathbf{m}}_j$ with $j$
  end for
  Outer decoding: From $\widehat{\mathbf{m}} = \bigcup_j \widehat{\mathbf{m}}_j$'s, find chunk clusters and correct errors; produce $I$
  Call Weak system with candidate set $I$
```

Lemma 3.23. *Fix k, B, β, with $B = \Omega(k)$ and $\beta = O(\log(B/k))$. There is a coding scheme for the length-B m-measurememt Sparse Recovery Channel for $m = O(\frac{k}{\epsilon} \log \frac{B}{k})$ in the weak list recovery sense in which decoding runs in time $O(B \log^3 \frac{B}{k})$. This scheme also uses a look up table of size β.*

Proof. As an outer code, use Reed-Solomon over an alphabet of size $\beta / \log \beta$. This is concatenated with a random code of length $\log \beta$ as an inner code. The inner code can be decoded in constant time from a lookup table of size β and the outer code can be decoded by solving a linear system of size approximately β in time $O(\beta^2)$. To encode the β bits of the inner code, proceed as follows.

To encode a single bit $b \in \{0, 1\}$, replace each row ρ of $\mathbf{\Phi}$ with a 2-by-N submatrix. In column i of ρ, replace each 1 with a height-two column $\binom{\rho_i}{0}$ or $\binom{0}{\rho_i}$ depending on b. For decoding in the presence of noise, consider any $\binom{a}{b}$ to be a *relaxed* encoding equivalent to $\binom{\rho_i}{0}$ if $|a| > |b|$ and $\binom{0}{\rho_i}$ otherwise. Replace each 0 with a height-2 column of zeros.

Overall we use a Weak system (Theorem 3.15) with a $(\Theta(k), O(1))$ bipartite expander that exhibits a $(\Theta(k), d)$ hashing scheme, where $d = \Theta(\log(B/k))$. We know that there exist $\Omega(k)$ heavy hitters, each dominates the buckets where it lands in $\Omega(d)$ repetitions. In each such repetition, our bit encoding scheme ensures that the associated bit can be recovered successfully, hence for each of such heavy hitter, we shall collect $\Omega(d)$ bits, enough to recover the message of β bits.

The runtime is $O(B\beta^2 \log(B/k))$ for exhaustive recovery in the Weak system. $\square$

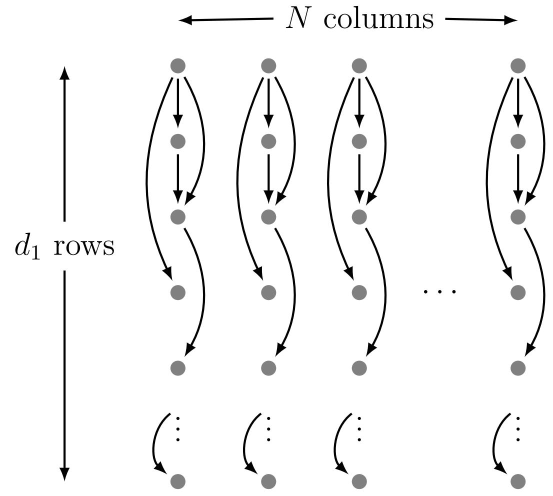

Figure 3.2: Underlying graph G_N. Suppose that $\mathbf{x}_1$ is in tail and that $\mathbf{x}_2$, $\mathbf{x}_3$ and $\mathbf{x}_N$ are heavy hitters.

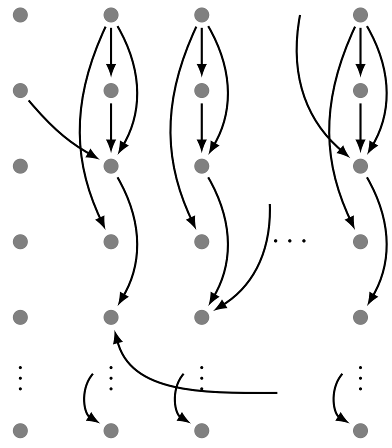

Figure 3.3: Ideally recovered graph $\tilde{G}$, with expander copies aligned in a column. Since the first column corresponds to a tail item, it is almost absent in the recovered graph. There are arcs from non-G_{i_j} nodes to G_{i_j} nodes.

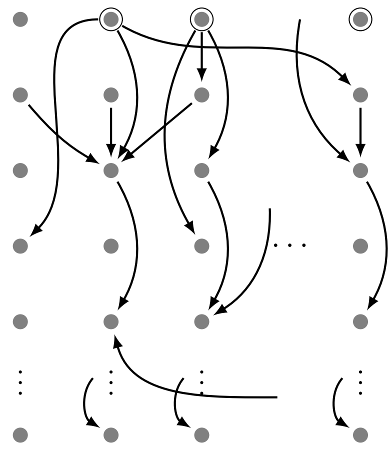

Figure 3.4: Recovered graph $\tilde{G}$, with 'supposed' expander copies aligned in columns. The first column corresponds to a tail item so it is almost absent. The top node in the second column is corrupted so it points to wrong columns but nevertheless the correct rows because the row information is hard-wired. The top node in the third column is correctly recovered but the second node in the column is corrupted. The top node in the last column has a small bucket value in the first repetition so it is absent $\tilde{G}$. If we perform BFS at the top node in the third column, we may include a lot of nodes in the second column.

3.5.1 Expander Encoding

3.5.1.1 Expander-based Coding Description

Parameters. $\beta, \gamma > 0$ are fixed constants. B_1, d_1, B_2, d_2 are as in Lemma 3.20 such that $B_1 = \Omega\big((\frac{k}{\epsilon^2})^{1+\beta}\log\frac{N}{k}\big)$. $c \leq m$ are constant integers, h is an integer, $\epsilon = O\big((\frac{\alpha}{m})^{\frac{m}{m-c}}(\frac{\log(B_1/k)}{\log(N/k)})^{\gamma}\big)$. Let G be a graph of d_1 nodes with constant degree δ that satisfies Theorem 3.3, and α, ζ, κ be constants provided by Theorem 3.5 when applied to G. Without loss generality we can assume that $\alpha \leq 1/2$. Adjust the hidden constants together with c, m and h appropriately (depending on β and γ) such that

(a) $B_1 > d_1$;

(b) $(h-1)m\log_{B_1} N < \alpha d_1$;

(c) $(\alpha d_1 - (h-1)m\log_{B_1} N)\cdot h^m > d_1^c$;

(d) $c \geq \log\delta/\log\kappa$.

We note that an instance of m, h is to choose $m \geq c(1+1/\gamma)$ and $h = \Theta(d_1^{c/m})$.

Encoding. We shall use Reed-Solomon for inner encoding. We now define our outer coding, which is based on Parvaresh-Vardy code [PV05].

Take N disconnected copies of G and call the union G_N, where each node is indexed by a pair $(i,r) \in [N] \times [d_1]$. See Figure 3.2.

Let $\mathbb{F}$ be a field such that $|\mathbb{F}| = \Theta(B_1)$ is a power of 2 and $E(x)$ be an irreducible monic polynomial over $\mathbb{F}$ such that $\deg E(x) = \log_{B_1} N$. View each $i \in [N]$ as a polynomial f over $\mathbb{F}$ with degree $\log_{B_1} N - 1$. For each $(i,r) \in G_N$, associate with it an element $p(i,r) \in \mathbb{F}^{m+1}$ as

$$p(i,r) = (x_{i,r}, f(x_{i,r}), (f^h \bmod E)(x_{i,r}), \ldots, (f^{h^{m-1}} \bmod E)(x_{i,r})),$$

where f is a polynomial associated with $i \in [N]$ and $x_{i,r} \in \mathbb{F}$ so that $x_{i,r}$ are distinct for different r. This is possible because of Property (a).

Attach to a node (i,r) a message $\mathbf{m}_{r,i}$ containing the information of $p(r,i)$ as well as $H(i,r_1(r))$,..., $H(i,r_\delta(r))$, where $r_1(r),\ldots,r_\delta(r)$ are the neighbours of r in G and $H(i,j) \in [B_1]$ gives the bucket index where i lands in the j-th outer hashing repetition. It is clear that $\mathbf{m}_{i,r}$ has $\Theta(\log B_1) = O(d_2)$ bits and therefore we can encode it in d_2 hash repetitions, see Lemma 3.23.

Decoding. In each of the d_1 repetitions, we shall recover $O(k/\epsilon)$ heavy buckets and thus obtain $O(k/\epsilon)$ nodes with their messages. Even when the messages are recovered correctly, we only know that a message corresponds to $\mathbf{m}_{r,i}$ for some i and we do not know which i it is. However, if we can determine that enough messages are associated with the same i, we would have obtained enough $p(i,r)$ for different values of r then we should be able to find f and thus recover the position i.

To determine enough $p(i,r)$ for the same i, we do clustering as follows. Suppose that there are k heavy hitters at position $i_1,\ldots,i_k$.

Let $\widetilde{G}$ be a graph of $d_1 \times O(k/\epsilon)$ nodes, arranged in a $d_1 \times O(k/\epsilon)$ grid. For now we assume that the messages are recovered correctly for heavy hitter i in all d_1 repetitions. Each message has the form $p(i,r), h_1,\ldots,h_\delta$, where $h_j = H(i,r_j(r))$ for $1 \le j \le \delta$. Add an arc $(i,r) \to (h_j, r_j(r))$ for each $1 \le j \le \delta$.

Since the messages are recovered correctly, the graph $\widetilde{G}$ will contain several disjoint copies of the expander graph G, say $G_{i_1},\ldots,G_{i_k}$, though each G_{i_j} is not necessarily aligned within the same column in $\widetilde{G}$. There will be arcs incoming to G_{i_j} from nodes not in any G_{i_j}, but there are no outgoing arcs from G_{i_j}. In this case, we can recover each G_{i_1} perfectly, and collect the full set $\{\mathbf{m}_{i_j,r}\}_{r=1}^{d_1}$ and thus recover i_j.

Rearrange the nodes within each row, we can align each copy of G in the same column to fit these columns in the underlying G_N. In this case, the columns $i_1, \dots, i_k$ are exactly the copies of the expander graph G. See Figure 3.3 for an illustration.

However, the heavy hitters may not be recovered in some repetitions and the messages could be seriously corrupted. When we are adding the arcs, we introduce two kinds of errors, respectively:

(i) We lose a node in G_{i_j}, i.e., the node is not present in $\widetilde{G}$ because the heavy hitter i_j is not recovered in that repetition;

(ii) We connect a node in G_{i_j} to a node in some other $G_{i_{j'}}$ $(j \neq j')$, owing to errorous message.

As before, we align each 'ideal copy' of G in the same column. See Figure 3.4 for an example.

We hope that for a heavy hitter i, only a few messages $\{\mathbf{m}_{i,r}\}_r$ are ruined and the i-th column of G_N will contain a large connected subgraph G' of G, by Theorem 3.5. Hence if we start a breadth-first search from an appropriate node with depth $c \log_\delta d_1$, the whole G' will be visited. In other words, we shall obtain a large set of $\{p(i,r)\}$, only a small number of which will be associated with the same i, but we expect to obtain enough $\{p(i,r)\}$ of the same i, which turns out to be sufficient to extract f associated with i using a good error-correcting code such as Parvaresh-Vardy code.

3.5.1.2 Proofs

Now we show that the system described above meets the aforementioned guarantee.

Lemma 3.24. *Let $\beta, \gamma > 0$. The encoding and decoding strategy of Section 3.5.1.1 are correct in the sense of the guarantee of that section, against the channel described in that section. It uses $O(\epsilon^{-2} s \log(N/s))$ measurements and runs in time $O(s^{1+\beta} \operatorname{poly}(\log N, 1/\epsilon))$, provided that $N = \Omega(\max\{s^2, s/\epsilon^2\})$ and $\epsilon = O\big((\frac{\log s}{\log N})^\gamma\big)$.*

Proof. Combining Lemma 3.18, Lemma 3.16 and Lemma 3.20, one can show that there exists an $(4s, \epsilon)$-expander such that

(a) the expander exhibits a (B_1, d_1, B_2, d_2) hashing structure, where the parameters are as in Lemma 3.20;

(b) the expander satisfies the $(O(s/\epsilon), O(\epsilon), O(1))$-isolation property;

(c) each second layer hashing gives an $(d_2, O(s/\epsilon), O(1))$-expander.

As in the proof of Lemma 3.13, suppose that $|\mathbf{x}_i| \geq \epsilon/s$ for all $i \in \operatorname{supp} \mathbf{y}$, otherwise we can place the violated i's into $\mathbf{z}$, causing $\|z\|_1$ to increase by at most $s \cdot \epsilon/s = \epsilon$, so we would have $\|z\|_1 \leq 2$. Call the elements in $\operatorname{supp} \mathbf{y}$ heavy hitters. If $|\operatorname{supp} \mathbf{y}| \leq s/4$ our goal is automatically achieved, so we assume that $|\operatorname{supp} \mathbf{y}| > s/4$.

Step 1. Overall we know from Remark 3.14 that we have at most $s/8$ decoys, or, we can recover $|\operatorname{supp} \mathbf{y}| - s/8$ heavy hitters from the second-layer bucket values, where successful recovery means that each of them dominates in at least $\alpha_2 d_1 d_2$ second-layer buckets, i.e., the bucket noise is at most $\nu = \epsilon/(2s)$. For each of them, in at least $\beta_1 d_1$ of d_1 outer repetitions, it dominates in at least $\beta_2 d_2$ inner repetitions, where $(1-\beta_1)(1-\beta_2) > 1 - \alpha_2$. Because whenever an element dominates in the second-layer bucket, it must dominate the first-layer bucket incident to that second-layer bucket, we conclude that there exists a set $S \subseteq \operatorname{supp} \mathbf{y}$, $|S| \geq |\operatorname{supp} \mathbf{y}| - s/8$, such that each $i \in S$ dominates at least $\beta_1 d_1$ first-layer buckets among all d_1 repetitions, and in each of such repetitions, it dominates at least $\beta_2 d_2$ second-layer buckets.

We can choose the hidden constants in the expander parameters such that $\beta_1 \geq 1 - \zeta$ and β_2 matches the error tolerance of the coding scheme we described in Lemma 3.23, where ζ is the parameter we set in Section 3.5.1.1.

Step 2. It follows from above that each $i \in S$ will be recovered in at least $\beta_1 d_1$ outer repetitions, since its bucket value is $\geq \epsilon/s - \nu \geq \epsilon/(2s)$. Indeed, in every repetition of outer hashing, we collect top $O(s/\epsilon)$ (first-layer) buckets, so we will include every bucket with value $\geq \epsilon/(2s)$, and thus the heavy hitter i. In this case, the message associated with the heavy hitter will be recovered correctly, as the inner encoding can tolerate $1 - \beta_2$ fraction of error. Therefore we know that for each $i \in S$, the associated messages will be correctly recovered in $\beta_1 d_1$ outer repetitions.

Step 3. As described in the previous section, we shall form a graph $\tilde{G}$. Note that for $i \in S$, $\beta_1 d_1$ nodes in the column are good nodes (i.e., with correct message). For each of them, perform a breadth-first search of $O(\log_\delta d_1)$ steps, collecting at most d_1^c nodes. Since the column contains at most $(1-\beta)d_1 \leq \zeta d_1$ bad nodes, by Theorem 3.5 and Property (d) of our choices of parameters, there exists a good node in the i-th column such that if we perform a breadth-first search of $c \log_\delta d_1$ steps, we shall collect αd_1 good nodes which are all in the i-th column. The Parvaresh-Vardy code with our choice of parameters (Property (b) and (c)) enables us to include it in the list. We shall briefly describe the decoding below. Having collected at most d_1^c points $(x, r(x)) \in \mathbb{F}^{m+1}$, we consider all polynomials $Q(x, y_0, \ldots, y_{m-1})$ of degree at most $d_X = \alpha d_1 - (h-1)m \log_{B_1} N$ in its first variable and at most $h-1$ in each such that $Q(x, r(x)) = 0$ for all i. Our choice of parameters (Property (c), i.e., $d_X h^m > d_1^c$) guarantees that such Q exists. Then, the existence of αd_1 good nodes (in the BFS visited nodes) indicates that the equation

$$Q(x, f_i(x), (f_i^h \bmod E)(x), \dots, (f_i^{h^{m-1}} \bmod E)(x)) = 0$$

has αd_1 roots in $\mathbb{F}$ for f_i corresponding to the coordinate $i \in S$. By our choice of parameters (Property (b)), the univariate polynomial $Q(x)$ has degree less than αd_1 and must be identically zero. This means that $f_i(x)$ is a root of $Q^*(z) = Q(x, z, z^h, \dots, z^{h^{m-1}}) = 0$ over $\mathbb{F}[x]/E(x)$. We can find f_i by factoring Q^* and thus recover the position i of the heavy hitter.

In the end, our candidate list will contain all $i \in S$, that is, we shall have recovered $|\operatorname{supp} \mathbf{y}| - s/8$ heavy hitters.

Number of Measurements. The number of measurements is

$$O(B_2 d_1 d_2) = O(\epsilon^{-2} s \log(N/s)).$$

Size of Look-up Table. The inner decoding uses a look-up table of size $O(\log B_1) = O(\frac{s}{\epsilon} + \log\log \frac{N}{s})$. The algorithm also stores the expander graph G, which takes space $O(d_1)$. Both are smaller than the space cost of the recovered graph $O(sd_1/\epsilon)$, so their contribution to the space complexity can be neglected.

Runtime. For each of d_1 repetition, we shall recover every bucket with value $\geq \epsilon/(2s)$ in $O(B_1 \log^3(B_1/k)) = O(s^{1+\beta} \operatorname{poly}(\log N, 1/\epsilon))$ time. There are $O(s/\epsilon)$ of them in each repetition. Then we form a graph of size $O(sd_1/\epsilon)$. Forming this graph takes time $O(s^{1+\beta} \operatorname{poly}(\log N, 1/\epsilon))$ from the argument above. Then we do breadth-first search of $c \log_\delta d_1$ steps on every node. Each BFS takes $O(d_1^c)$ time. Each decoding of the BFS nodes takes $\operatorname{poly}(d_1, \log |B_1|) = \operatorname{poly}(\log N, 1/\epsilon)$ time, and can be done deterministically (see, e.g., [CEPR09, Theorem 4.3]), since $|\mathbb{F}|$ has a small characteristic. Hence extracting heavy hitters i from the recovered graph $\tilde{G}_N$ takes time $O(s \operatorname{poly}(\log N, 1/\epsilon))$ and therefore, the overall runtime is $O(s^{1+\beta} \operatorname{poly}(\log N, 1/\epsilon))$. In the end, we shall obtain a candidate list of size $O(s \operatorname{poly}(\log N, 1/\epsilon))$. □

3.6 Toplevel System

Now we define Toplevel system as in [PS12]. A Toplevel system is an algorithm that solves our overall problem. The construction here closely follows [GLPS12, PS12].

Definition 3.25. An *approximate sparse recovery system* (briefly, a *Toplevel* system), consists of parameters N, k, ϵ, an m-by-N *measurement matrix* $\mathbf{\Phi}$, and a *decoding algorithm* $\mathcal{D}$ that satisfy the following property: for any vector $\mathbf{x} \in \mathbb{R}^n$, given $\mathbf{\Phi x}$, the system approximates $\mathbf{x}$ by $\widehat{\mathbf{x}} = \mathcal{D}(\mathbf{\Phi x})$, which satisfies

$$\|\widehat{\mathbf{x}} - \mathbf{x}\|_1 \leq (1+\epsilon) \left\|\mathbf{x}_{[k]} - \mathbf{x}\right\|_1 .$$

Algorithm 3.3 Toplevel System

```
Input: $\mathbf{\Phi}$, $\mathbf{\Phi x}$, $N$, $k$, $\epsilon$
Output: $\widehat{\mathbf{x}}$
  $\widehat{\mathbf{x}} \leftarrow 0$
  $\mathbf{y} \leftarrow \mathbf{\Phi x}$
  for $j \leftarrow 0$ to $\log k$ do
      Run Algorithm 3.2 on $\mathbf{y}$ with length $N$, $s \leftarrow k/2^j$, $\eta \leftarrow \frac{\epsilon}{\gamma^j(1-\gamma)}$ and obtain a
  candidate list $I$
      Run Algorithm 3.1 on candiate set $I$ with $s \leftarrow k/2^j$ and $\eta \leftarrow \epsilon\gamma^j$
      Let $\mathbf{x}'$ be the result
      $\widehat{\mathbf{x}} \leftarrow \widehat{\mathbf{x}} + \mathbf{x}'$
      $\mathbf{y} \leftarrow \mathbf{y} - \mathbf{\Phi x}'$
  end for
  return $\widehat{\mathbf{x}}$
```

Theorem 3.26. *Let $\beta, \gamma > 0$. There is a Toplevel system that uses $O(\epsilon^{-2} k \log(N/k))$ measurements and runtime $O(k^{1+\beta} \operatorname{poly}(\log N, 1/\epsilon))$, provided that $N = \Omega(\max\{k^2, k/\epsilon^2\})$ and $\epsilon = O\big((\frac{\log k}{\log N})^\gamma\big)$.*

Proof. Suppose that in Lemma 3.24, the exponent of $1/\epsilon$ in runtime is $c = c(\beta, \gamma) > 2$. Choose $\alpha < 1$ such that $\alpha^c > 1/2$.

Using Lemma 3.24 for identification and Theorem 3.15 for estimation, with appropriate choice of constants, we claim that at the beginning of the j-th step, $\mathbf{x} = \mathbf{y} + \mathbf{z}$, where $|\operatorname{supp} \mathbf{y}| \leq k/2^j$ and

$$\|\mathbf{z}\|_1 \leq 1 + \epsilon \left(1 + \alpha + \alpha^2 + \cdots + \alpha^{j-1}\right).$$

We shall prove this claim by induction. Letting $s = k/2^j$, $\eta = \epsilon(1-\alpha)\alpha^j$ for identification, which introduces at most η into the tail and the tail remains at most $3/2$ by assuming that all head items, i.e., the non-zero elements in $\mathbf{y}$, are all larger than η/s.

The identification procedure returns a candidate I that contains $3/4$ fraction of $\operatorname{supp}\mathbf{y}$ (note that when the head is flat, we can change $\operatorname{supp}\mathbf{y}$ to be a superset that satisfies this condition without changing the norm of $\mathbf{z}$). Then the estimation procedure, with $s = O(k/2^j)$ and $\eta = \epsilon\alpha^{j+1}$ will give us

$$\mathbf{x} = \widehat{\mathbf{x}} + \widehat{\mathbf{y}} + \widehat{\mathbf{z}},$$

where $|\operatorname{supp}\mathbf{x}| = O(s)$, $|\operatorname{supp}\hat{\mathbf{y}}| \leq s/2$ and

$$\|\widehat{\mathbf{z}}\|_1 \leq \|\mathbf{z}\|_1 + \epsilon(1-\alpha)\alpha^j + \alpha^{j+1} = \|\mathbf{z}\|_1 + \alpha^j.$$

It is easy to verify that $\|\hat{\mathbf{z}}\|_1 \leq 1 + \epsilon/(1-\alpha) = O(1)$ and thus Lemma 3.24 for identification and Theorem 3.15 can be applied at the next round and the inductive hypothesis is satisfied. Therefore, in the end we shall obtain that

$$\|\widehat{\mathbf{x}} - \mathbf{x}\|_1 \leq \left(1 + \frac{\epsilon}{1-\alpha}\right)\|\mathbf{x} - \mathbf{x}_k\|_1.$$

The number of measurements used for identification is

$$O\left(\sum_j \frac{1}{\epsilon^2\alpha^{2j}} \cdot \frac{k}{2^j}\log\frac{N}{\frac{k}{2^j}}\right) = O\left(\frac{k}{\epsilon^2}\sum_j \left(\frac{1}{2\alpha^2}\right)^j \left(j + \log\frac{N}{k}\right)\right) = O\left(\frac{k}{\epsilon^2}\log\frac{N}{k}\right)$$

and the number of measurements used for estimation is

$$O\left(\sum_j \frac{1}{\epsilon^2\alpha^{j}} \cdot \frac{k}{2^j}\log\frac{N}{\frac{k}{2^j}}\right) = O\left(\frac{k}{\epsilon^2}\sum_j \left(\frac{1}{2\alpha}\right)^j \left(j + \log\frac{N}{k}\right)\right) = O\left(\frac{k}{\epsilon^2}\log\frac{N}{k}\right)$$

hence the total number of measurements is $O(\epsilon^{-2}k\log(N/k))$ as claimed.

It can be verified in a similar way that total runtime is $O(k^{1+\beta}\operatorname{poly}(\log N, 1/\epsilon))$.

Finally, replacing ϵ with $(1-\alpha)\epsilon$ completes the proof. □

Remark 3.27. Regarding the theorem above,

(a) The constants in big O-notations and the power in $\operatorname{poly}(\log N, 1/\epsilon)$ depend on β and γ.

(b) As in Remark 3.19, The constraint that $k = O(\sqrt{N})$ could be weakened to

$k = O(N^{1-\xi})$ for any $\xi > 0$.

(c) The factor $k^{1+\beta}$ in the runtime is due to our choice of $B_1 = \Omega((k/\epsilon^2)^{1+\beta}\log(N/k))$ such that $\log B_1 = O(\log(B_1/k)) = O(d_2)$. When $k \leq (\log N)^c$ for some $c > 0$, since $B_1 = \Omega(k/\epsilon^{2(1+\beta)})$, choosing $B_1 = \Theta(k\log(N/k)/\epsilon^{2(1+\beta)})$ would suffice. It leads to runtime $O(k\operatorname{poly}(\log N, 1/\epsilon))$.

(d) For large ϵ we can take $d_1 = (\log(N/k)/\log(B_1/k))^{1+\alpha}$ for $\alpha > 0$, which gives an algorithm which uses more measurements $O(k\log^{1+\alpha}(N/k)/\epsilon^2)$ but suboptimal by only a logarithmic factor from the best known lower bound.

3.7 Open Problems

In this section, we present some open problems.

Restriction on ϵ. The algorithm in this chapter restricts ϵ to $(\frac{\log k}{\log N})^\gamma$ for any $\gamma > 0$ because of it applies the Parvaresh-Vardy code. In a sense our construction reduces the problem to a list recovery problem. We ask if it is possible to find an improvement by applying a better list recoverable code. The ultimate goal is to relax the restriction of ϵ to $\epsilon \leq \epsilon_0$ for some constant $\epsilon_0 > 0$.

Sparse Recovery in ℓ_2/ℓ_1 norm. The ultimate problem is the ℓ_2/ℓ_1 problem. We hope that the algorithm in this paper offers new ideas for the mixed-norm problem. Again the difficulty is in identification, as an RIP_2 matrix would be sufficient for estimation.

Post-measurement Noise. In many algorithms on the sparse recovery problem, the input to the decoding algorithm is $\Phi\mathbf{x} + \nu$ instead of $\Phi\mathbf{x}$, where ν is an arbitrary noise vector. It can been seen that our algorithm does tolerate substantial noise in ℓ_1 norm. We leave to future work full analysis and possible improved algorithms.

3.8 Proof of Lemma 3.16

Proof. Let p_s be the probability of a fixed set of s elements hashed into less than $(1-\epsilon)ds$ elements. By symmetry this probability is independent of the s positions and thus is well-defined. Hence the probability

$$\Pr\{\text{hashing does not give an expander}\} = \sum_{s=2}^{4k} \binom{N}{s} p_s. \tag{3.1}$$

Our goal is to show that

$$p_s \leq \exp\left(-cs \ln \frac{eN}{s}\right) \tag{3.2}$$

for some absolute constant $c > 2$, for which it suffices to show that

$$p_s \leq \exp\left(-cs \ln \frac{N}{k} \ln \frac{Ck}{s}\right) \tag{3.3}$$

for some $c, C > 0$. Indeed, it follows from (3.3) that

$$p_s \leq \exp\left(-cs \ln \frac{N}{k} \ln \frac{Ck}{s}\right) \leq \exp\left\{-cs\left(\ln \frac{N}{k} + \ln \frac{Ck}{s}\right)\right\} = \exp\left(-cs \ln \frac{CN}{s}\right)$$

and (3.2) holds. Assume for the moment that (3.2) is proved, then we can bound (3.1) by

$$\begin{aligned}\sum_{s=2}^{\alpha k} \binom{N}{s} p_s &\leq \sum_{s=2}^{\alpha k} \exp\left\{s \ln \frac{eN}{s} - cs \ln \frac{CN}{s}\right\} \\ &\leq \sum_{s=2}^{\alpha k} \exp\left\{-(c-1)s \ln \frac{C'N}{s}\right\} \\ &\leq \sum_{s=2}^{\alpha k} \exp\left(-(c-1)s \log N\right) < \frac{1}{N^{c'}}\end{aligned}$$

as desired.

Now we compute p_s. Fix a set S of s elements. Suppose that they are hashed into X_i ($i = 1, \ldots, d$) buckets in d repetitions, respectively. We have that $1 \leq X_i \leq s$ and $\sum X_i \leq (1-\epsilon)sd$. Define the event

$$E_i(X_i) = \{S \text{ is hashed into } X_i \text{ rows in } i\text{-th reptition}\},$$

and we shall compute $\Pr\{E_i(X_i)\}$.

When E_i happens, there are $s - X_i$ repetitions. Consider hashing the element one by one, choosing $b_1, \ldots, b_d \in \{1, \ldots, B\}$ sequentially. We have a collision when selecting b_i if $b_i \in \{b_1, \ldots, b_{i-1}\}$. The probability that a collision occurs at step i, even conditioned on $b_1, \ldots, b_{i-1}$, is at most $i/B \le s/B$. Therefore,

$$\Pr\{E_i(X_i)\} \le \binom{s}{s - X_i} \left(\frac{s}{B}\right)^{s-X_i}.$$

Hence

$$\begin{aligned} p_s = \sum \Pr\{E_1(X_1), \ldots, E_d(X_d)\} &= \sum \prod_{i=1}^{d} \binom{s}{s-X_i} \left(\frac{s}{B}\right)^{s-X_i} \\ &= \sum \left(\frac{s}{B}\right)^{sd-\sum X_i} \prod_{i=1}^{d} \binom{s}{s-X_i} \end{aligned}$$

where the summation is over all possible configurations of $\{X_i\}$. Invoking the combinatorial identity

$$\sum_{k_1+k_2+\cdots+k_m=n} \binom{r_1}{k_1}\binom{r_2}{k_2}\cdots\binom{r_m}{k_m} = \binom{r_1+r_2+\cdots+r_m}{n} \tag{3.4}$$

and writing $X = \sum X_i$, we see that

$$p_s \le \sum_{X=d}^{(1-\epsilon)sd} \left(\frac{s}{B}\right)^{sd-\sum X_i} \binom{sd}{sd - \sum X_i} \le \sum_{X=\epsilon sd}^{d} \binom{sd}{X}\left(\frac{s}{B}\right)^{X}$$

Now we invoke Chernoff bound

$$\sum_{k=\epsilon n}^{n} \binom{n}{k}\lambda^k \le \left(\frac{e\lambda}{\epsilon}\right)^{\epsilon n}, \quad \lambda < \epsilon \tag{3.5}$$

to obtain that

$$p_s \le \left(\frac{es}{\epsilon B}\right)^{\epsilon sd} \le \exp\left(-cs \log\frac{N}{k} \ln\frac{Ck}{s}\right)$$

as desired, where the constants $c, C > 0$ can be made arbitrarily big. □

3.9 Proof of Lemma 3.17

Proof. Let S be a set of size $s \le L$. We shall bound the probability p_s (which is defined by symmetry) that at least ϵs elements of S collide with each other in at least ζd repetitions. When this happens, there are at least $\epsilon\zeta ds$ colliding element-repetition pairs. As in Lemma 3.16 it suffices to have (3.3) for some $c, C > 0$ that can be made arbitrarily large.

In one repetition, one element of S collides with others with probability $\leq s/B$. By a coupling argument as in [PS12], among all sd element-repetition pairs with expected $\mu = s^2 d/B$ failed pairs, there are at least $\zeta\epsilon sd$ failed pairs with probability

$$\left(\frac{e\mu}{\zeta\epsilon ds}\right)^{\zeta\epsilon sd} = \left(\frac{es}{\zeta\epsilon B}\right)^{\zeta\epsilon sd} \leq \exp\left(-cs\log\frac{N}{k}\ln\frac{Ck}{s}\right)$$

as desired, where the absolute constants $C, c > 0$ can be made arbitrary large. $\square$

3.10 Proof of Lemma 3.18

Proof. Let p_s be the probability of a fixed set of s elements hashed into less than $(1-\epsilon)ds$ elements. By symmetry this probability is independent of the s positions and thus is well-defined. Hence the probability

$$\Pr\{\text{hashing does not give an expander}\} = \sum_{s=2}^{4k}\binom{N}{s}p_s. \tag{3.6}$$

Similarly to Lemma 3.16, it suffices to show that

$$p_s \leq \exp\left(-cs\ln\frac{N}{k}\right) \tag{3.7}$$

Assume for the moment that this is proved, then we can bound (3.6) to be

$$\begin{aligned}\sum_{s=2}^{4k}\binom{N}{s}p_s &\leq \sum_{s=2}^{4k}\exp\left\{s\ln\frac{eN}{s} - cs\ln\frac{N}{k}\right\} \\ &\leq \sum_{s=2}^{4k}\exp\left\{s\ln(eN) - \frac{c}{2}s\ln(eN)\right\} \quad (k \leq \sqrt{N/e}) \\ &\leq \sum_{s=2}^{4k}\exp\left(-\left(\frac{c}{2}-1\right)s\log(eN)\right) < \frac{1}{N^{c'}}\end{aligned}$$

as desired.

Now we prove (3.7). Fix a set S of s elements. The outer layer of hashing has d_1 blocks of size B_1, and let Y_i ($i = 1, \ldots, d_1$) be the number of hashed row of the s elements in i-th block. The inner layer has $d_1 d_2$ blocks, indexed by $(i,j)_{1\leq i\leq d_1, 1\leq j\leq d_2}$ of size B_2, and let X_{ij} be the number of hashed row of the s elements in the (i,j)-th block. Define the events

$$\begin{aligned} E_i(Y_i) &= \{S \text{ is hashed into } Y_i \text{ rows in } i\text{-th outer block}\} \\ E_{ij}(X_{ij}) &= \{S \text{ hashed into } X_{ij} \text{ rows in } (i,j)\text{-th inner block}\} \end{aligned}$$

First we calculate $\Pr\{E_i\}(Y_i)$. Consider picking a row at one time for an element in S in order. When $E_i(Y_i)$ happens there are at least $s - Y_i$ collisions, hence

$$\Pr\{E_i(Y_i)\} \leq \binom{s}{s-Y_i}\left(\frac{s}{B_1}\right)^{s-Y_i}$$

and similarly

$$\Pr\{E_{ij}(X_{ij})|E_i(Y_i)\} \leq \binom{Y_i}{Y_i - X_{ij}}\left(\frac{Y_i}{B_2}\right)^{Y_i - X_{ij}}$$

It follows that

$$\begin{aligned} p_s &= \sum \Pr\{E_{11}(X_{11}), \ldots, E_{d_1 d_2}(X_{d_1 d_2})|E_1(Y_1), \ldots, E_{d_1}(Y_{d_1})\} \Pr\{E_1(Y_1), \ldots, E_{d_1}(Y_{d_1})\} \\ &\leq \sum \prod_i \Pr\{E_i\}(Y_i) \prod_{i,j} \Pr\{E_{ij}(X_{ij})|E_i(Y_i)\} \\ &\leq \sum \prod_i \binom{s}{Y_i}\left(\frac{s}{B_1}\right)^{s-Y_i} \cdot \prod_{i,j}\binom{Y_i}{X_{ij}}\left(\frac{Y_i}{B_2}\right)^{Y_i - X_{ij}} \\ &\leq \sum \left(\frac{s}{B_1}\right)^{sd_1 - \sum Y_i}\left(\frac{s}{B_2}\right)^{d_2 \sum Y_i - \sum X_{ij}} \prod_i \binom{s}{Y_i} \prod_{i,j}\binom{Y_i}{X_{ij}} \end{aligned}$$

where the summation is taken over all possible configurations of $\{X_i\}$ and $\{Y_i\}$ so that $s \geq Y_i \geq \max_j X_{ij}$ and $\sum X_{ij} \leq (1-\epsilon)sd_1 d_2$.

Invoking the combinatorial equality (3.4) and letting $X = \sum X_{ij}$ and $Y = \sum Y_i$, we obtain that

$$\begin{aligned} p_s &\leq \sum_{Y=d_1}^{sd_1}\binom{sd_1}{Y}\left(\frac{s}{B_1}\right)^{sd_1 - Y} \sum_{X=d_1 d_2}^{\min\{d_2 Y, (1-\epsilon)sd_1 d_2\}}\binom{d_2 Y}{X}\left(\frac{s}{B_2}\right)^{d_2 Y - X} \\ &= \sum_{Y=d_1}^{(1-\epsilon/2)sd_1}\binom{sd_1}{Y}\left(\frac{s}{B_1}\right)^{sd_1 - Y}\sum_{X=d_1 d_2}^{d_2 Y}\binom{d_2 Y}{X}\left(\frac{s}{B_2}\right)^{d_2 Y - X} \\ &\quad + \sum_{Y=(1-\epsilon/2)sd_1}^{sd_1}\binom{sd_1}{Y}\left(\frac{s}{B_1}\right)^{sd_1 - Y}\sum_{X=d_1 d_2}^{(1-\epsilon)sd_1 d_2}\binom{d_2 Y}{X}\left(\frac{s}{B_2}\right)^{d_2 Y - X} \\ &=: S_1 + S_2 \end{aligned} \tag{3.8}$$

We bound S_1 and S_2 separately. First,

$$S_1 \le \left(1 + \frac{s}{B_2}\right)^{sd_1 d_2} \sum_{Y=\epsilon sd_1}^{sd_1} \binom{sd_1}{Y} \left(\frac{s}{B_1}\right)^Y$$

It follows from Chernoff bound (3.5) that

$$\begin{aligned} S_1 &\le \left(1 + \frac{s}{B_2}\right)^{sd_1 d_2} \left(\frac{es}{\frac{\epsilon}{2}B_1}\right)^{\epsilon sd_1/2} \\ &\le \exp\left\{-\frac{1}{2}\epsilon sd_1 \left(\ln \frac{\epsilon B_1}{2es}\right) + sd_1 d_2 \ln\left(1 + \frac{s}{B_2}\right)\right\} \\ &\le \exp\left\{-\frac{1}{4}\epsilon sd_1 \ln \frac{B_1}{k} + c_2 \epsilon sd_1 d_2\right\} \quad (\text{since } B_1 \gtrsim k/\epsilon^2) \\ &\le \exp\left\{-c_3 s \ln \frac{N}{k}\right\} \end{aligned} \tag{3.9}$$

where the absolute constant $c_2 > 0$ can be made arbitrarily close to 0 and the absolute constant c_3 can be made arbitrarily large.

Now we bound S_2. When $Y \ge (1 - \epsilon/2)sd_1 d_2$ then

$$\frac{(1-\epsilon)sd_1 d_2}{d_2 Y} \le 1 - \frac{\epsilon}{2}.$$

Again invoking Chernoff bound,

$$\sum_{X=d_1 d_2}^{(1-\epsilon)sd_1 d_2} \binom{d_2 Y}{X} \left(\frac{s}{B_2}\right)^{d_2 Y - X} \le \left(\frac{es}{\frac{\epsilon}{2}B_2}\right)^{d_2 Y - (1-\epsilon)sd_1 d_2} \le \left(\frac{1}{C'}\right)^{d_2 Y - (1-\epsilon)sd_1 d_2}$$

where $C' > 0$ is an absolute constant which can be made arbitrarily large. So

$$\begin{aligned} S_2 &\le \sum_{Y=(1-\epsilon/2)sd_1}^{sd_1} \binom{sd_1}{Y} \left(\frac{s}{B_1}\right)^{sd_1 - Y} \left(\frac{1}{C'}\right)^{\epsilon sd_1 d_2/2} \\ &\le \sum_{Y=0}^{(\epsilon/2)sd_1} \binom{sd_1}{Y} \left(\frac{s}{B_1}\right)^{Y} \left(\frac{1}{C'}\right)^{\epsilon sd_1 d_2/2} \\ &\le 2\left(\frac{1}{C'}\right)^{\epsilon sd_1 d_2/2} \end{aligned}$$

It immediately follows, similarly to upper-bounding S_1, that

$$S_2 \leq \exp\left\{-c_4 s \ln \frac{N}{k}\right\}, \tag{3.10}$$

where $c_4 > 0$ can be made arbitrarily large. Plugging (3.9) and (3.10) into (3.8) we see that (3.7) holds. This completes the proof. □

3.11 Proof of Lemma 3.20

Proof. Fix a set S of size s. Let event $\mathcal{E}$ be that at least $(1-\epsilon/2)s$ elements in S are isolated in at least $(1-\zeta/2)d_1$ first-layer buckets. Similarly to Lemma 3.17 we know that

$$\Pr\{\mathcal{E}^c\} \leq \left(\frac{c's}{\zeta\epsilon B_1}\right)^{\zeta\epsilon s d_1} \leq e^{-cs\log\frac{N}{k}}$$

where c' is an absolute constant and $c > 0$ can be made arbitrarily large. In the above we used that fact that since $B_1 = \Omega(k/(\zeta^\alpha \epsilon^{2\alpha}))$ it holds that

$$\ln \frac{\zeta\epsilon^2 B_1}{c_1 k} \geq \left(1 - \frac{1}{\alpha}\right) \ln \frac{B_1}{k}.$$

Now let us be conditioned on the event $\mathcal{E}$. Among the $(1-\epsilon/2)s$ elements we shall show that at least $(1-\epsilon)$ of them are isolated in at least $(1-\zeta)d_1 d_2$ second-layer buckets. That means there are a total of at least $\frac{\epsilon}{2}\frac{\eta}{2} s d_1 d_2$ failed pairs of element-reptition. But now, the probability of each collision is always bounded by s/B_2 even conditioned on previous outcomes, and we can proceed as in Lemma 3.17 to conclude that there are at least $\theta\zeta\epsilon s d_1 d_2$ (for some absolute constant θ) with probability at most

$$\left(\frac{es}{\theta\zeta\epsilon B_2}\right)^{\theta\zeta\epsilon s d_1 d_2} \leq e^{-c'' s \log\frac{N}{k}},$$

as desired, where the constant $c'' > 0$ can be made arbitrarily large. □

CHAPTER 4

Off-grid Fourier Sampling[1]

4.1 Introduction

As discussed in Section 1.4, there has been a lot of work to recover the signal with sparse discrete Fourier representation, which has broad applications in communication problems. Unfortunately, these assumptions are too strong for many practical applications where the discrete Fourier transform coefficients are only approximation of an underlying continuous Fourier transform. For example, if we want to measure the approaching speed (the 'doppler') of an object via the Doppler effect, we transmit a sinusoid wave $\mathrm{e}^{i\omega_0 t}$ (where t is time in this example) and receive a sinusoid wave whose frequency offset from ω_0 depends on the unknown doppler, v. Since v can be essentially any continuous value, so can be the received frequency. If there are two or more speeding objects in view, the received signal is of the form $f(t) = a_1\mathrm{e}^{i\omega_1 t}+a_2\mathrm{e}^{i\omega_2 t}$, where ω_1/ω_2 is not necessarily a rational number, so that $f(t)$ is not periodic. This practical and common example does not directly fit the discrete Fourier transform setting of [KM93, GGI$^+$02, GMS05, Iwe09, HIKP12b, HIKP12a].

A natural discretization approach, which takes a large number of samples at equidistant time points and reduces the problem to this discrete signal of samples, is complicated because there are inconveniences such as interpreting the result of the discrete version in the continuous setting and locating a real-valued frequency from a cluster of components in the discretized signal. Therefore, instead of developing a discretization reduction in this paper, we take a more direct approach of extending the existing techniques for the discrete setting, such as isolation by hashing and estimation, to the continuous setting, despite the fact that it introduces an additional $\log k$ factor into sampling duration.

[1]A preliminary version of the result in this chapter appeared in [BCG$^+$12].

A concrete application of our approach is the bearing (or angular direction) estimation of sources transmitting at fixed frequencies, a canonical array signal processing problem with applications to radar, sonar, and remote sensing. Other applications also include the finite rate of innovation problems [VMB02].

The organization of this chapter is as follows. In Section 4.2, we define our model and problem formulation. In Section 4.3, we present our algorithm and its analysis. In Section 4.4, we discuss the simple discretization approach. In Section 4.5 we give an application of this result to bearing estimation problems.

4.2 Preliminaries

In this section, we define the problem of sublinear recovery of sparse off-grid frequencies, set the stage notationally, and then detail our results.

4.2.1 The problem

We define a spectrally sparse function f with *off-grid* frequencies as a function $f : \mathbb{R} \to \mathbb{C}$ with k frequencies $\omega_1, \ldots, \omega_k \in \mathbb{S}^1$ (hereinafter $\mathbb{S}^1$ is identified with $(-\pi, \pi]$ and the arithmetic is to be modular), and we allow for noise ν in the spectrum that is supported on a set $I_\nu \subset \mathbb{S}^1$. We fix a minimum frequency resolution $\eta > 0$ and assume that $\{[\omega_j - \eta/2, \omega_j + \eta/2)\}_{j=1}^k$ and $I_\nu + [-\eta/2, \eta/2)$ are all mutually disjoint on $\mathbb{S}^1$. That is, the frequencies are not on a fixed, discrete grid but they are separated from each other and from the noise by a minimum frequency resolution. In our analysis below, we assume that $|\omega_j| \geq \eta$ without loss of generality.

Specifically, let $(I_\nu, \mathcal{M}, \mu)$ be a σ-finite measure space, $(\mathbb{R}, \mathcal{L}, \lambda)$ denote the canonical Lebesgue measure space of $\mathbb{R}$ and ν be in $L^1(I_\nu, d\mu)$. Furthermore, suppose that $\nu(\omega)\mathrm{e}^{i\omega x}$ is a $(\mu \times \lambda)$-measurable function on $I_\nu \times \mathbb{R}$. (The product measure $\mu \times \lambda$ is with respect to $\mathcal{M} \times \mathcal{L}$. We do not take completion of the product measure.) Note that this assumption is automatically satisfied if μ is a Borel measure on I_ν, which could be an important and common case. Now, $v(\omega)\mathrm{e}^{i\omega t}$ is μ-measurable for all $t \in \mathbb{R}$ and, as a consequence, $\nu\phi$ is μ-measurable for any $\phi \in C(I_\nu)$ because any continuous function on a compact set can be uniformly approximated by trignometric polynomials. We assume f is of the form

$$f(t) = \sum_{j=1}^{k} a_j \mathrm{e}^{i\omega_j t} + \int_{I_\nu} \nu(\omega)\mathrm{e}^{i\omega t} d\mu, \quad t \in \mathbb{R},\, a_j \in \mathbb{C} \tag{4.1}$$

Without loss of generality, we can assume that $a_j \neq 0$ for all j. It is clear that f is locally integrable on $\mathbb{R}$, and can therefore be viewed as a tempered distribution. Its Fourier transform $\hat{f}$ is also a tempered distribution. Define the spectrum of f to be the support of $\hat{f}$, which is a closed set. It can be readily verified that the spectrum of f defined in this way consists of $\{\omega_j\}$ and a subset $I' \subseteq I_\nu$ such that $\nu = 0$ almost everywhere w.r.t. μ on $I_\nu \setminus I'$. This agrees with our intuition of 'spectrum' being $\{\omega_j\}$. Alternatively, one can define the spectrum of a bounded function in a more elementary way, see [Hel95] for example.

Fix $\epsilon_1, \epsilon_2 \in (0, 1]$. Our goal is to find all (a_j, ω_j) with

$$|a_j| \geq \frac{\epsilon_1}{k}\|a\|_1 + \frac{\epsilon_2}{k}\int_{I_\nu} |\nu(\omega)| d\omega \tag{4.2}$$

making as few samples on $\mathbb{Z}$ as possible (and with the smallest support) from f and for the shortest duration and to produce such a list in runtime comparable to the number of samples. The number of samples and the size of the support set of the samples should be proportional to a polynomial in k and $\log(1/\eta)$, the number of desired frequencies and precision. We call the frequencies ω_j whose associated amplitude a_j meet the threshold condition (4.2) *significant*.

We observe that if we dilate the frequency domain $\mathbb{S}^1$ by a factor $1/d \in \mathbb{R}$ (i.e., map ω to ω/d), we produce an equivalent sequence of samples $f(t)$, at regularly spaced real-valued points $t = nd$, $n \in \mathbb{Z}$. While the points are indexed by the integers, the values themselves $t = nd$ are in $\mathbb{R}$. The dilation factor d determines the "rate" at which we sample the underlying signal and the total number of samples times the sampling rate is the duration over which we sample. Both the rate and the total number of samples are resources for our algorithm.

4.2.2 Notation

Let Ω be a domain (which can be either continuous or discrete). Roughly speaking, we call a function $K : \Omega \to \mathbb{R}$ a *filter* if K is or approximates the characteristic function χ_E of some set $E \subset \Omega$, which will be called the pass region of K. The signal resulting from applying filter K to signal f (viewed as a function on Ω) is the pointwise product $K \cdot f$.

Let $K_{m,\epsilon,\alpha}$ (often abbreviated as $K_{m,\epsilon}$ or K_m when there is no ambiguity on the parameters) be a kernel defined on $\mathbb{S}^1$ that satisfies the following properties:

- it is continuous on $\mathbb{S}^1$,

- it approximates $\chi_{[-\frac{\pi}{m},\frac{\pi}{m}]}$ (so K_m is a filter):

  1. $|K_{m,\epsilon}(x)| \leq \epsilon$ for $|x| \geq \frac{\pi}{m}$;
  2. $|K_{m,\epsilon}(x) - 1| \leq \epsilon$ for $|x| \leq (1-\alpha)\frac{\pi}{m}$;
  3. $K_{m,\epsilon}(x) \in [-\epsilon, 1+\epsilon]$ elsewhere;

- its Fourier transform $\widehat{K_{m,\epsilon}} : \mathbb{Z} \to \mathbb{C}$ has finite support: $|\operatorname{supp} \widehat{K_{m,\epsilon}}| = O(\frac{m}{\alpha} \log \frac{1}{\epsilon})$.

A Dolph-Chebyshev filter convolved with the characteristic function of an interval meets these criteria. See Figure 4.1 for a plot of K_m. We call the region $[-(1-\alpha)\frac{\pi}{m}, (1-\alpha)\frac{\pi}{m}]$ the *plateau* of K_m. The pass region of K_m is $[-\frac{\pi}{m}, \frac{\pi}{m}]$ and we define the transition region to be the complement of plateau in the pass region. A similar kernel was used in [HIKP12b] and [HIKP12a] with the only difference that their kernel was constructed by a Gaussian kernel convolved with the characteristic function of an interval. It is in our favor to use a kernel with a finite Fourier expansion.

4.2.3 Sampling with respect to a kernel

In this paper, we shall repeatedly query for values of the form

$$h(x) = \sum_{j=1}^{k} a_j K(\omega_j x) + \int_{I_\nu} \nu(\omega) K(\omega x) d\mu$$

for some kernel function K. Below we shall describe how to obtain $h(x)$ from appropriate samples of $f(t)$. Assume that k has a finite Fourier expansion

$$K(x) = \sum_{n=-N}^{N} \kappa_n e^{inx}.$$

Then

$$\begin{aligned} h(x) &= \sum_{j=1}^{k} \sum_{n=-N}^{N} \kappa_n a_j e^{in\omega_j x} + \int_{I_\nu} \sum_{n=-N}^{N} \kappa_n \nu(\omega) e^{in\omega_j x} d\mu \\ &= \sum_{n=-N}^{N} \kappa_n \left(\sum_{j=1}^{k} a_j e^{in\omega_j x} + \int_{I_\nu} \nu(\omega) e^{in\omega_j x} d\mu \right) \\ &= \sum_{n=-N}^{N} \kappa_n f(nx), \end{aligned}$$

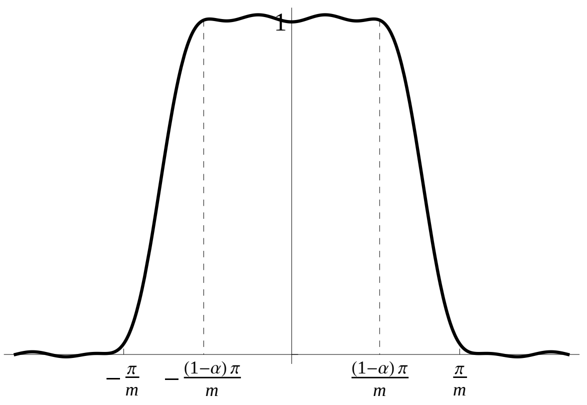

Figure 4.1: Plot of K_m with $\alpha = 0.476$

which means that $h(x)$ is a weighted sum of samples $\{f(nx)\}_{n=-N}^{N}$, where the weights are the Fourier coefficients of the kernel $K(x)$.

4.2.4 Main result

Theorem 4.1. *There exist a probability distribution $\mathcal{D}$ on a set of sampling points $t \in \mathbb{R}$ and an algorithm $\mathcal{A}$ such that for each perturbed exponential polynomial $f(t)$ as in* (4.1)*, with probability $\geq 1-\delta$, the algorithm returns a list $\Lambda = \{(a'_j, \omega'_j)\}_{j=1}^{k}$ of coefficients and frequencies such that*

1. *For each a_j that satisfies* (4.2) *there exists $\omega'_j \in \Lambda$ such that*

$$|\omega_j - \omega'_j| \leq \frac{\epsilon_2}{k}\eta.$$

2. *Let $\Lambda_0 = \left\{\omega'_j \in \Lambda : \exists \omega_{j_0} \text{ such that } \left|\omega_{j_0} - \omega'_j\right| \leq \frac{\epsilon_2 \eta}{k} \text{ and } |a_{j_0}| \text{ satisfies (4.2)}\right\}$. Then for each $\omega'_j \in \Lambda_0$ it holds that*

$$|a'_j - a_j| \leq \frac{\epsilon_1}{k}\|a\|_1 + \frac{\epsilon_2}{k}\|\nu\|_1.$$

3. *For each $\omega'_j \in \Lambda \setminus \Lambda_0$, it holds that*

$$|a'_j| \leq \frac{\epsilon_1}{k}\|a\|_1 + \frac{\epsilon_2}{k}\|\nu\|_1.$$

The algorithm takes

$$O\left(\frac{k}{\epsilon_2}\log\frac{k}{\delta}\log\frac{1}{\epsilon_2\eta}\log\frac{k}{\min\{\epsilon_1,\epsilon_2\}}\right)$$

samples and runs in time

$$O\left(\frac{k}{\epsilon_2}\log\frac{k}{\delta}\left(\log\frac{1}{\epsilon_2\eta}\log\frac{k}{\min\{\epsilon_1,\epsilon_2\}}+\log\frac{1}{\delta}\right)\right).$$

Furthermore, the size of the support of $\mathcal{D}$, i.e., the total duration of sampling, is

$$O\left(\frac{k}{\epsilon_2\eta}\log\frac{k}{\min\{\epsilon_1,\epsilon_2\}}\right).$$

This result gives ℓ_∞ bounds on both frequency estimates and coefficient estimates. In comparison with the discrete setting, frequency estimates are new and coefficient estimates are of the same kind as in [HIKP12b]. In the analogy of the discrete case, the minimum separation $\eta = 1/N$. For $\epsilon_1 = \epsilon_2 = \epsilon$ and constant δ, the number of samples and the runtime both become

$$O\left(\frac{k}{\epsilon}\log k\log\frac{N}{\epsilon}\log\frac{k}{\epsilon}\right)$$

and the sampling duration becomes

$$O\left(\frac{kN}{\epsilon}\log\frac{k}{\epsilon}\right).$$

4.3 Analysis

Almost all sublinear-time sparse recovery algorithms, including both the Fourier and the canonical basis cases, randomly hash coefficients into buckets. Since the representation is sparse, it is likely that each bucket contains exactly one coefficient and small noise so that its position can be found and then its value can be estimated. Our algorithm also follows this recipe. Later developments of these algorithms are iterative for the purpose of using a smaller number of samples or measurements or because of inability to obtain an accurate estimate. In contrast, our algorithm is not iterative. We hash the range of the frequencies into buckets and repeat sufficiently many times so that all frequencies are isolated, then we locate the frequency and estimate its amplitude. We do not need to iterate in the estimation procedure, because we use good kernel, as in [HIKP12b]. We shall briefly explain below why it is rather difficult to devise an iterative algorithm for the continuous case.

A main difference between the discrete and continuous case is that, in the continuous case, it is impossible to recover a frequency exactly (from finite samples) so that one can subtract off recovered signals at exact positions. Typically in the discrete setting, an iterative algorithm uses a loop invariant either as in [GLPS12, HIKP12a] or in [GMS05]. In the former case, the number of buckets decreases per round as the number of remaining heavy hitters decreases. In the continuous case, however, the accuracy of the frequency estimates produced by location procedure are dependent on the width the pass region of the filter: the wider the pass region, the more inaccurate the frequency estimate is. Unless the estimation procedure not only estimates the coefficient at given frequency but also improves the frequency estimate, we would have to increase the distance d between samples from $O(k/\eta)$ to $O(k^2/\eta)$ in order to the achieve the same accuracy for the final frequency estimate, (i.e., we must increase the duration over which samples are collected.)

In the latter case [GMS05], the number of buckets is kept the same at each round while the energy of the residual signal drops, and there are typically $\log \|a\|$ rounds. In hashing, we need to bound the inaccuracy $|K(h(\omega)) - K(h(\omega'))|$, where ω' is the recovered estimate of some real frequency ω, h the hash function and K the kernel. We can achieve this with a kernel that does not have a significant portion of its total energy outside of its pass region (i.e., a “non-leaking” kernel), but it is not obvious how to achieve such an accurate estimate using a Dirichlet or Fejér kernel which was used in [GMS05]. Besides, using a “non-leaking” kernel like the one used in [HIKP12b, HIKP12a] or the one used in this paper unfortunately introduces a factor $\log \|a\|$ into the number of samples in order to decrease the noise in a bucket.

4.3.1 Recovery algorithm

See Algorithm 4.1 for detailed pseudocode.

4.3.2 Analysis of algorithm

Here we provide a modular characterization of the algorithm.

Isolation.

This portion of the analysis is similar to that of [HIKP12a] but we emphasize the continuous frequency setting.

Let K_m be the kernel as described in Sec. 4.2 and set $D = 2\pi/\eta$. Define

$$\mathcal{H} = \{K_m(\omega d) = h_d(\omega) | d \in [D, 2D]\}$$

Algorithm 4.1 The overall recovery algorithm for the off-grid problem

```
function MAIN
    $y \leftarrow$ signal samples
    $L \leftarrow$ IDENTIFY($y$)
    $\Lambda \leftarrow$ ESTIMATE($L$)
    return $\sum_{\omega\in\Lambda} a_\omega e^{i\omega t}$
end function

function IDENTIFY($y$)
    $L \leftarrow \emptyset$
    for $t \leftarrow 1$ to $\Theta(\log\frac{k}{\delta})$ do
        Choose a random $d$ uniformly from $[D, 2D]$
        $b_i \leftarrow 0$ for all $i = 0, \ldots, m-1$
        for $r \leftarrow 1$ to $\lceil\log_2(1/\eta)\rceil$ do
            Collect samples and compute $\{u_\ell\}_{\ell=0}^{m-1}$ and $\{v_\ell\}_{\ell=0}^{m-1}$ according to
                Remark 4.8, where $u_\ell = \sum_j a_j K_m\left(\omega_j d - \frac{2\pi\ell}{m}\right) K_n\left(\frac{\omega_j d}{2^r} - \frac{2\pi}{2^r m}\ell - \frac{2b_\ell\pi}{2^r}\right)$
                and $v_\ell = \sum_j a_j K_m\left(\omega_j d - \frac{2\pi\ell}{m}\right) K_n\left(\frac{\omega_j d}{2^r} - \frac{2\pi}{2^r m}\ell - \frac{2b_\ell\pi}{2^r} - \pi\right)$
            for $\ell \leftarrow 0$ to $m-1$ do
                if $|v_\ell| > |u_\ell|$ then
                    $b_\ell \leftarrow b_\ell + 2^{r-1}$
        end for
        for $\ell \leftarrow 0$ to $m-1$ do
            $L \leftarrow L \cup \{\frac{2\pi\ell}{md} + \frac{2b_\ell\pi}{d}\}$
    end for
    return $L$
end function

function ESTIMATE($L$)
    Choose hash families $\mathcal{H}_1$ and $\mathcal{H}_2$ as described.
    for $r \leftarrow 1$ to $\Theta(\log\frac{m}{\delta})$ do
        for each $\omega \in L$ do
            $a_\omega^{(r)} \leftarrow$ measurement w.r.t. $\mathcal{H}_1$
            $b_\omega^{(r)} \leftarrow$ measurement w.r.t. $\mathcal{H}_2$
        end for
    for each $\omega \in L$ do
        $a_\omega \leftarrow \mathrm{median}_t\, a_\omega^{(r)}$
        $b_\omega \leftarrow \mathrm{median}_t\, b_\omega^{(r)}$
    end for
    $L' \leftarrow \{x \in L : |b_\omega| \geq |a_\omega|/2\}$.
    $\Lambda \leftarrow \{(\omega, a_\omega) : \omega \in L'\}$.
    Cluster $\Lambda = \{(\omega, a_\omega)\}$ by $\omega$ and retain only one element in the cluster.
    Retain top $k$ ones (w.r.t. $a_\omega$) in $\Lambda$
    return $\Lambda$
end function
```

to be a family of hash functions. We choose h_d randomly from $\mathcal{H}$ by drawing d from the interval $[D, 2D]$ uniformly at random. Observe that the map $\omega \mapsto \omega d$ is a random dilation of $\mathbb{S}^1$. Similar to [HIKP12a] and [GMS05], we shall consider m-translations of K_m, denoted by $\{K_m^{(j)}\}_{j=0}^{m-1}$, where $K_m^{(j)}(x) = K_m\left(x - \frac{2\pi j}{m}\right)$ $(x \in \mathbb{S}^1)$, so that their pass regions cover $\mathbb{S}^1$. The pass regions will be referred to as *buckets* and the pass region of $K_m^{(j)}$ as j-th bucket. For convenience we shall also call the plateau of $K_m^{(j)}$ the plateau of the j-th bucket. It is clear that each frequency ω, under the random dilation $\omega \mapsto \omega d$, will land in some bucket with index $b(\omega, d)$.

In the discrete setting, we hash N elements into m buckets so that each bucket contains N/m elements. Here we hash $(-\pi, \pi]$ into m buckets so that each bucket has measure $2\pi/m$. Similar to [HIKP12a] and [GMS05], the next lemmata show that this hashing is effective, immitating Claim 3.1, Claim 3.2 of [HIKP12a] and Lemma 3.1 of [GMS05].

The first lemma tells us that the probability of collision of two well-separated frequencies under a random hash function $h_d \in \mathcal{H}$ is small.

Lemma 4.2. *Suppose that $|\omega' - \omega| \geq \eta$. Then*

$$\Pr_d\{b(\omega, d) = b(\omega', d)\} \leq \frac{1}{m}\left(2 + \frac{1}{m}\right).$$

Proof. Without loss of generality, assume that $\Delta\omega = \omega' - \omega > 0$. Write $\omega d = x + \xi$ with $|\xi| < \pi/m$. Then the probability is equal to

$$\Pr\left\{(\Delta\omega)d \in \left[2s\pi - \xi - \frac{\pi}{m}, 2s\pi - \xi + \frac{\pi}{m}\right]\right\} \leq \frac{1}{D} \cdot \frac{2\pi}{m\Delta\omega} \cdot |I|,$$

where I is the set of possible s's,

$$I = \left\{\left\lfloor \frac{(\Delta\omega)D + \xi - \frac{\pi}{m}}{2\pi} \right\rfloor + 1, \ldots, \left\lfloor \frac{2(\Delta\omega)D + \xi + \frac{\pi}{m}}{2\pi} \right\rfloor\right\}.$$

The desired upper bound of the probability follows immediately from that

$$|I| \leq \frac{(\Delta\omega)D}{2\pi} + 1 + \frac{1}{m}. \qquad \square$$

While Lemma 4.2 guarantees that well-separated frequencies do not collide under our hash function, because we are in the continuous setting, there is some probability that a frequency is hashed into the transition region of the kernel K_m. The following lemma shows that with high probability, a frequency bounded away from zero is mapped to the region of the kernel that is very close to 1.

Lemma 4.3. *Assume that $\omega \geq \eta$ and let $0 < \alpha < 1/2$ be as given in the definition of the kernel K_m. Then*

$$\Pr_d \left\{ \omega d \in \left[\frac{2s\pi}{m} - (1-\alpha)\frac{\pi}{m}, \frac{2s\pi}{m} + (1-\alpha)\frac{\pi}{m} \right] \text{ for some } s \in \mathbb{Z} \right\} \geq (1-\alpha)\left(1 - \frac{1}{m}\right).$$

Proof. It is clear that the probability is at least

$$\frac{1}{D} \cdot (1-\alpha)\frac{2\pi}{\omega m} \cdot |I|,$$

where I is the set of possible s's, and that

$$I = \left\{ \left\lfloor \omega D \frac{m}{2\pi} - \frac{1-\alpha}{2} \right\rfloor + 1, \ldots, \left\lfloor 2\omega D \frac{m}{2\pi} + \frac{1-\alpha}{2} \right\rfloor \right\}.$$

Then

$$|I| \geq \frac{\omega D m}{2\pi} - 1.$$

and the result follows immediately. □

The next lemma will allow us to estimate the coefficient of an isolated frequency and to bound the inaccuracy of its estimate in terms of the noise $\|\nu\|_1$.

Lemma 4.4. *Suppose that ξ is a random variable such that $|\xi| \leq \pi/m$ and the parameter ϵ of $K_{m,\epsilon}$ satisfies $\epsilon \leq c/m$ for some constant $c > 0$. Let $\omega \geq \eta$. Then*

$$\mathbb{E}_d[|K_m(\omega d + \xi)|] \leq \frac{2(c+2)}{m}.$$

Proof. Define

$$\tilde{K}_m(x) = \sup_{|y-x| \leq \pi/m} |K_m(y)|,$$

it is not difficult to see that

$$\|\tilde{K}_m\|_1 \leq 2\pi\epsilon + \frac{4\pi}{m} \leq \frac{2\pi C}{m}$$

where $C = c + 2$. Let d be uniformly chosen from some interval I. Then

$$\mathbb{E}_d[|K_m(\omega d + \xi)|] \leq \mathbb{E}_d[|\tilde{K}_m(\omega d)|] = \frac{1}{|I|}\int_I |\tilde{K}_m(\omega t)| dt = \frac{1}{\omega |I|}\int_{\omega I} |\tilde{K}_m(x)| dx.$$

Then

$$\int_{\omega I} |\tilde{K}_m(x)|dx \leq \frac{2C\pi}{m}\left\lceil \frac{\omega|I|}{2\pi} \right\rceil,$$

and thus

$$\mathbb{E}_d[\tilde{K}_m(\omega d + \xi)] \leq \frac{C}{m} \cdot \frac{\lceil \frac{\omega|I|}{2\pi} \rceil}{\frac{\omega|I|}{2\pi}}.$$

For $I = [D, 2D]$,

$$\mathbb{E}_d[|\tilde{K}_m(\omega d)|] \leq \frac{C}{m}\left(1 + \frac{1}{[\frac{\omega D}{2\pi}]}\right).$$

Let $D = 2\pi/\eta$, since $\omega \geq \eta$, it follows that $[\omega D/(2\pi)] \geq 1$, and the desired result holds. $\square$

Now we are ready to show our algorithm isolates frequencies.

Fix j_0 and choose $m = \Omega(k)$. The hashing guarantees that ω_{j_0} is well-isolated with probability $\Omega(1)$ by taking a union bound. Also, it follows immediately from Lemma 4.4 that the expected contribution of ν to the bucket is at most $c\|\nu\|_1/m$. Therefore we conclude by Markov's inequality that

Lemma 4.5. *Conditioned on ω_{j_0} being well-isolated under $h_d \in \mathcal{H}$, with probability $\Omega(1)$,*

$$\left|\sum_{j\neq j_0} a_j h_d(\omega_j) + \int_{I_\nu} \nu(\omega)h_d(\omega)d\mu\right| \leq C_1\epsilon\|a\|_1 + \frac{C_2}{m}\|\nu\|_1$$

for some constants C_1, C_2 that depend on the failure probability.

Proof. Note that $h_d(\omega) = K_m((\omega - \omega_{j_0})d + \xi_d)$, where ξ_d is piecewise continuous on $[D, 2D]$, and thus $\mathrm{e}^{in\xi_d}$ is λ-measurable. Since K_m has a finite Fourier expansion, we can easily see that $h_d(\omega)$ is a finite sum of $(\mu \times \lambda)$-measurable functions and is thus $(\mu \times \lambda)$-measurable. Therefore we can apply Fubini's Theorem to an iterated integral of $|h_d(\omega)|$. By Lemma 4.4 and Fubini's Theorem,

$$\begin{aligned}\mathbb{E}\left[\left|\int_{I_\nu} \nu(\omega)h_d(\omega)d\mu\right|\right] &\leq \mathbb{E}\left[\int_{I_\nu} |\nu(\omega)|\,|h_d(\omega)|d\mu\right] \\ &= \int_{I_\nu} \nu(\omega)\mathbb{E}\,|h_d(\omega)|\;d\omega \\ &\leq \frac{c}{m}\int_{I_\nu} |\nu(\omega)|d\mu,\end{aligned}$$

Since ω_j $(j \neq j_0)$ lands in different bucket from ω_{j_0},

$$|h_d(\omega_j)| \leq \epsilon, \quad j \neq j_0$$

thus

$$\mathbb{E}\left[\left|\sum_{j\neq j_0} a_j h_d(\omega_j) + \int_{I_\nu} \nu(\omega) h_d(\omega) d\mu\right|\right] \leq \epsilon\|a\|_1 + \frac{c}{m}\|\nu\|_1$$

The result follows from Markov's inequality. □

Bit Testing.

The isolation precedure above reduces the problem to the following: The parameter d is known, and exactly one of $\{\omega_j d\}_{j=1}^k$, say $\omega_{j_0} d$, belongs to $\bigcup_{n=0}^{N-1}[2n\pi-\delta, 2n\pi+\delta]$ for some small $\delta = \pi/m$ and (large) N. Suppose that $\omega_{j_0} d \in [2s\pi - \delta, 2s\pi + \delta]$. We shall find s and thus recover ω_{j_0}. Assume that ω_{j_0} is significant; i.e., a_{j_0} satisfies (4.2).

We recover s from the least significant bit to the most significant bit, as in [GMS05]. Assume we have already recovered the lowest r bits of s, and by translation, the lowest r bits of s are 0s. We shall now find the $(r+1)$-st lowest bit.

Let K_n (n is a constant, possibly $n = 3$) be another kernel with parameter ϵ' (a small constant). The following lemma shows that Line 6–11 of Identify gives the correct s.

Lemma 4.6. *Suppose that the lowest r bits of s are 0, let*

$$G_1(x) = K_m(x) K_n\left(\frac{x}{2^r}\right), \quad G_2(x) = K_m(x) K_n\left(\frac{x}{2^r} - \pi\right)$$

and u be the sample taken using G_1 and v using G_2. Then $|u| > |v|$ if $s \equiv 0 \pmod{2^{r+1}}$ and $|u| < |v|$ if $s \equiv 2^r \pmod{2^{r+1}}$, provided that

$$m \geq \frac{C}{\epsilon_2}k \quad \text{and} \quad \epsilon \leq C \cdot \min\left\{\frac{\epsilon_1}{k}, \frac{\epsilon_2}{k}\right\}$$

for some $C > 0$.

Proof. It is straighforward from the isolation discussion. When $s \equiv 0 \pmod{2^r}$,

$$|u| \geq (1-\epsilon)(1-\epsilon')|a_{j_0}| - (1+\epsilon')\left(C_1\epsilon\|a\|_1 - \frac{C_2}{m}\|\nu\|_1\right). \tag{4.3}$$

and when $s \equiv 2^{r-1} \pmod{2^r}$,

$$|u| \leq (1+\epsilon)\epsilon'|a_{j_0}| + (1+\epsilon')\left(C_1\epsilon\|a\|_1 + \frac{C_2}{m}\|\nu\|_1\right). \tag{4.4}$$

Similar bounds hold for $|v|$. Thus it suffices to choose

$$m \geq \frac{2(1+\epsilon')C_2}{1-\epsilon-2\epsilon'} \cdot \frac{k}{\epsilon_2}. \qquad \square$$

Repeat this process until $r = \log_2(\pi D) = O(\log(\pi/\eta))$ to recover all bits of s. At each iteration step the number of samples needed is $O(|\operatorname{supp}\widehat{G_1}| + |\operatorname{supp}\widehat{G_2}|) = O(|\operatorname{supp}\widehat{K_m}| \cdot |\operatorname{supp}\widehat{K_n}|) = O(\frac{k}{\epsilon_2}\log\frac{1}{\epsilon})$, so the total number of samples used in a single execution of Line 7 of Identify is $O(\frac{k}{\epsilon_2}\log\frac{1}{\epsilon}\log\frac{1}{\eta})$.

The precision of $\omega_{j_0}d$ will be $\delta = \pi/m$, and thus the precision of ω_{j_0} will be $\pi/(md) \leq \pi/(mD) = \eta/m$.

In summary, the hashing process guarantees that

Lemma 4.7. *With probability at least $1 - O(\delta)$,* Identify *returns a list L such that for each ω_j with a_j satisfying (4.2), there exists $\omega' \in L$ such that $|\omega' - \omega_j| \leq \eta/m$.*

Remark 4.8. Notice that $\sigma(K_m) \subseteq [-M, M] \cap \mathbb{Z}$ for some integer $M > 0$. We shall show that, similar to [GMS05], despite Line 7 of Identify (for m translations altogether) requires mr numbers, each of which is a sum of $2M+1$ terms, this process can be done in $O((M + m\log m)r)$ time instead of $O(Mmr)$ time.

Suppose that at some step, for the j-th translation ($0 \leq j \leq m-1$), the translation that shifts the lowest bits of s_j to 0 is b_j. By Section 4.2.3, we shall take $\Theta(Mn)$ samples, corresponding to the the spectrum of K_mK_n, so we index the samples by (s,t) with $|s| \leq \Theta(n)$ and $|t| \leq M$. We need the numbers

$$\sum_{s=-\Theta(n)}^{\Theta(n)} e^{-2\pi i (b_j + \frac{j}{m})\frac{s}{2^r}} \sum_{t=-M}^{M} e^{-2\pi i \frac{jt}{m}} w_{st} z_{st}, \quad j = 0, \ldots, m-1,$$

where z_{st} is the sample with index (s,t) and w_{st}'s are the Fourier coefficients of K_mK_n. Notice that the inner sum can be rewritten as

$$\sum_{\ell=0}^{m} e^{-2\pi i \frac{j\ell}{m}} \sum_{t \in (m\mathbb{Z}+\ell)\cap[-M,M]} w_{st} z_{st},$$

which can be done in $O(M + m\log m)$ time using the FFT algorithm. The outer sum has only constantly many terms. Hence to compute m numbers, each is a double sum as above, takes only $O(M + m\log m)$ times. There are r steps, so the total time complexity is $O((M + m\log m)r)$.

4.3.2.1 Coefficient Estimation.

The isolation procedure generates a list L of candidate frequencies. Like [HIKP12a], we estimate the coefficient at each position in L by hasing it into buckets using the same kernel but with possibly different parameters. We shall show how to extract good estimates and eliminate unreliable estimates among $|L|$ estimates.

The following lemma states that if a frequency candidate is near a true frequency then they fall in the same bucket with a good probability and if a frequency candidate is adequately away from a true frequency then they fall in different buckets with a good probability.

Lemma 4.9. *Let $D = \Theta(1/\eta)$ and $\delta > 0$. Choose d uniformly at random from $[\theta_1 D, \theta_2 D]$.*

1. *if $|\omega - \omega'| \leq \beta_1\delta/D \leq \eta$ then*

$$\Pr\{b(\omega', d) = b(\omega, d)\} \geq 1 - \beta_1\theta_2.$$

2. *if $|\omega - \omega'| \geq \beta_2\delta/D$ then*

$$\Pr\{b(\omega', d) = b(\omega, d)\} \leq \frac{1}{\beta_2(\theta_2 - \theta_1)} + \frac{c\delta}{D}$$

for some universal constant $c > 0$.

Proof. Without loss of generality assume $\omega' > \omega$. Then the probability in case (1) can be rewritten as

$$\sum_{s\in\mathbb{Z}} \Pr\left\{\left\lceil\frac{\omega' d}{2\delta} + \frac{1}{2}\right\rceil = \left\lceil\frac{\omega d}{2\delta} + \frac{1}{2}\right\rceil = s\right\} = \sum_{\text{possible } s} \frac{1}{(\theta_2 - \theta_1)D} \cdot m\left(\left[\frac{2s\delta - \delta}{\omega}, \frac{2s\delta + \delta}{\omega'}\right]\right),$$

where $m(E)$ denotes the Lebesgue measure of set E. Note that

$$m\left(\left[\frac{2s\delta - \delta}{\omega}, \frac{2s\delta + \delta}{\omega'}\right]\right) = \delta\frac{\omega + \omega' - 2s(\omega' - \omega)}{\omega'\omega} \geq \delta\frac{\omega + \omega' - \omega\theta_2\beta_1 - \frac{\beta_1\delta}{D}}{\omega'\omega} \geq \delta\frac{1 - \theta_2\beta_1}{\omega'}$$

Thus the sum of probabilities can be bounded from below by

$$\frac{1}{(\theta_2 - \theta_1)D} \cdot \delta\frac{1 - \theta_2\beta_1}{\omega'}\left(\left\lceil\frac{\omega'\theta_2 D}{2\delta} + \frac{1}{2}\right\rceil - \left\lceil\frac{\omega'\theta_1 D}{2\delta} + \frac{1}{2}\right\rceil + 1\right) \geq 1 - \theta_2\beta_1.$$

The other case can be proved similarly as Lemma 4.2. □

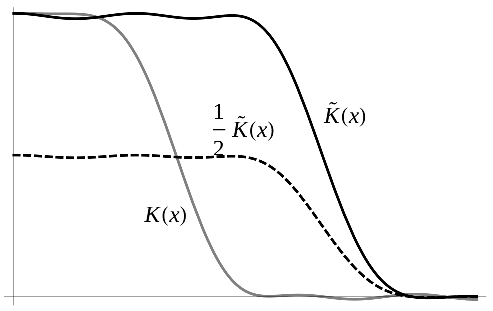

Figure 4.2: K and $\tilde{K}$

Now consider a fixed significant frequency, say, ω_1. Assume that ω_1 is isolated from other frequencies under hashing $\omega \mapsto \omega d$. If ω_1 lands in the plateau of the kernel K_m, that is, $K_m(\omega_1 d) \approx 1$, then the bucket value

$$\int K_m(xd)f(x)dx \approx a_1 K_m(\omega_1 d) \approx a_1,$$

is a good estimate to the coefficient. Similarly if it lands outside the pass region, the bucket value would be close 0. The only annoying situation is when it lands in the transition of K_m, in which case we may have a significant bucket value but much smaller than the desired a_1. Our plan is to detect the estimates from the transition region and remove them. To this end, consider two kernels K and $\tilde{K}$ such that the pass region of K falls within the plateau of $\tilde{K}$ (see Figure 4.2). Observe that the set $\{x : K(x) \geq \tilde{K}(x)/2\}$ consists of two parts: one part falls completely within the plateau of $\tilde{K}$, the other outside the pass region of K. With respect to the two kernels, we obtain two bucket values

$$a = \int K(xd)f(x)dx \approx a_1 K(\omega_1 d)$$

and

$$a' = \int \tilde{K}(xd)f(x)dx \approx a_1 \tilde{K}(\omega_1 d)$$

If $|a| > |a'|/2$, then we know that either $\omega_1 d$ falls in the plateau of $\tilde{K}$, which means that a' is a reliable estimate, or $\omega_1 d$ falls outside the pass region of K, which means that a is small (and thus is a'). Hence we can drop the estimates with $|a| < |a'|/2$ and retain reliable estimates (either in the plateau of $\tilde{K}$ or outside the pass region of K). We always take a' as our final estimate, which would be either significant or small. Furthermore, the plateau of K is contained in the set $\{x : K(x) \geq \tilde{K}(x)/2\}$, so we will always obtain a good estimate to a_1 if we can guarantee that $\omega_1 d$ will land in the plateau of K at least once.

This idea, together with the frequency estimate guarantee, is formalized in the next few lemmata.

Choose parameters $0 < \beta_1 < \beta_2$, $0 < \theta_1 < \theta_2$ such that $\beta_1\theta_2 + \alpha < 1/3$ and $1/(\beta_2(\theta_2 - \theta_1)) < 1/3$. Let $D = C\pi/\eta$. Define a hash family

$$\mathcal{H} = \{K_m(\omega d) = h_d(\omega) | d \in [\theta_1 D, \theta_2 D]\}.$$

It then holds that

Lemma 4.10. *Let $\omega' \geq \eta$ and $j_0 = \arg\min_j |\omega' - \omega_j|$. Obtain a measurement $a_{\omega'}$ w.r.t. $h_d \in \mathcal{H}$.*

1. *If $|\omega' - \omega_{j_0}| \leq \beta_1 C\eta/m$, with probability $\Omega(1)$, it holds that $|a_{\omega'} - a_{j_0}| \leq \epsilon\|a\|_1 + c'\|\nu\|_1/m$ for some $c' > 0$ dependent on the failure probability;*

2. *If $|\omega' - \omega_{j_0}| \geq \beta_2 C\eta/m$, with probability $\Omega(1)$, it holds that $|a_{\omega'}| \leq \epsilon\|a\|_1 + c'\|\nu\|_1/m$ for some $c' > 0$ dependent on the failure probability.*

Proof. As in hashing, since ω' is separated from all other ω_j ($j \neq j_0$), with probability $\Omega(1)$, ω_j does not land in the bucket for all $j \neq j_0$. It follows from Lemma 4.9 that

1. If $|\omega' - \omega_{j_0}| \leq \beta_1 C\eta/m$ then ω_{j_0} falls in the plateau of the same bucket as ω' except with probability $\beta_1\theta_2 + \alpha$.

2. If $|\omega' - \omega_{j_0}| \geq \beta_2 C\eta/m$ then ω_{j_0} falls in a different bucket from ω' except with probability $1/(\beta_2(\theta_2 - \theta_1)) + cC/m$.

Upon the success of either case, the noise in the bucket is at most $\epsilon\|a\|_1 + C\|\nu\|_1/m$ (by the argument in isolation the section) and the conclusion follows. □

Let $\Delta = \epsilon\|a\|_1 + c'\|\nu\|_1/m$, where c' is a constant dependent on the failure probability guaranteed in the preceding lemma and ϵ satisfies the condition in Lemma 4.6.

Take different $C_1 > C_2$ (and thus different D_1 and D_2) such that $\beta_1 C_2 \geq 1$ and that $C_2\beta_2 \leq C_1\beta_1$. Define hash families $\mathcal{H}_i$ ($i = 1, 2$) as

$$\mathcal{H}_i = \{K_m(\omega d) = h_d(\omega) | d \in [\theta_1 D_i, \theta_2 D_i]\}, \quad i = 1, 2.$$

It then follows that

Lemma 4.11. *Upon termination of execution of line 12 in* Estimate, *with probability* $\geq 1 - O(\delta)$, *for each* $\omega' \in L'$ *let* $j_0 = \arg\min_j |\omega' - \omega_j|$ *it holds that*

1. *If* $|\omega' - \omega_{j_0}| \leq \beta_1 C_1 \eta/m$, *then* $|a_{\omega'} - a_{j_0}| \leq \Delta$;
2. *If* $|\omega' - \omega_{j_0}| \geq \beta_2 C_1 \eta/m$, *then* $|a_{\omega'}| \leq \Delta$
3. *If* $\beta_1 C_1 \eta/m \leq |\omega' - \omega_{j_0}| \leq \beta_2 C_1 \eta/m$, *then* $|a_{\omega'}| \leq 2\Delta$.

Proof. Case (1) and (2) follow from the previous lemma. For ω' in case (3) it holds that $a_{\omega'} \leq 2b_{\omega'}$ and $b_{\omega'} \leq \Delta$ since $|\omega' - \omega_{j_0}| \geq \beta_2 C_2 \eta/m$. □

Loosely speaking, Lemma 4.11 guarantees a multiplicative gap between the coefficient estimates for the "good" estimates of significant frequencies and the coefficient estimates for all other frequency estimates. Next, we merge estimates of the same true source. In increasing order, for each $\omega' \in L'$ with coefficient estimate $a_{\omega'}$, find

$$I(\omega') = \left\{ \omega \in L' : \omega' \leq \omega \leq \omega' + \frac{C_1 \beta_1 \eta}{m} \text{ and } \frac{2}{\gamma - 1}|a_{\omega'}| < |a_\omega| < \frac{\gamma - 1}{2}|a_{\omega'}| \right\},$$

where $\gamma > 3$ is a constant to be determined later.

Choose any element from I as representative of all elements in I and add it to Λ. Continue this process from the next $\omega' \in L$ that is larger than I. Retain the top k items of Λ.

Lemma 4.12. *Suppose that* Estimate *is called with argument* L. *With probability* $\geq 1 - \delta$, *it produces a list* Λ *such that*

1. *For each* j *with* $|a_j| \geq \gamma\Delta$ *for some* $\gamma > 2 + \sqrt{5}$, *if there exists* $\omega' \in L$ *such that* $|\omega' - \omega_j| \leq \pi/m$, *then there exists* $(\omega'', a_{\omega''}) \in \Lambda$ *(we say that* $\omega'' \in \Lambda$ *is paired) such that*
$$|\omega'' - \omega_j| \leq \frac{C_1 \beta_1 \eta}{m}, \quad |a_{\omega''} - a_j| \leq \Delta.$$
2. *For each unpaired* $\omega \in \Lambda$ *it holds that*
$$|a_\omega| \leq 2\Delta.$$

Proof. In case (1), for all $\omega \in L'$ such that $|\omega - \omega_j| \leq C_1 \beta_1 \eta/m$ it holds that $|a_\omega| \geq (\gamma - 1)\Delta$ while for other ω it holds that $|a_\omega| \leq 2\Delta$. There is a multiplicative gap so the merging process does not mix frequencies that are close and far away from a true source. It is easy to verify that $\omega \in L'$ upon termination of Line 15 since $C_2 \beta_1 \geq 1$.

The rest is obvious. The conclusion of the theorem holds with probability at least $1 - O(\delta)$, which can be made $1 - \delta$ by rescaling δ by a constant factor. □

Now we are ready to prove our main result.

Proof of Theorem 4.1. It suffices to show that MAIN returns a desirable result with probability $\geq 1 - \delta$. Choose ϵ in the estimation procedure to be $\epsilon = \min\{\epsilon_1, \epsilon_2\}/(\gamma k)$ and $m \geq \gamma c' k$, where γ is as in Lemma 4.12, then $\Delta \leq \|\nu\|_1/(\gamma k)$. Thus whenever $|a_j|$ satisfies (4.2) it holds that $|a_j| \geq \gamma\Delta$. Combining Lemma 4.7 and Lemma 4.12 completes the proof. □

Number of Samples.

There are $O(\log \frac{k}{\delta})$ repetitions in isolation and each takes $O(\frac{k}{\epsilon_2} \log \frac{1}{\epsilon} \log \frac{1}{\eta})$ samples, hence the isolation procedure takes $O(\frac{k}{\epsilon_2} \log \frac{k}{\delta} \log \frac{1}{\epsilon} \log \frac{1}{\eta})$ samples in total.

The input of ESTIMATE is a list L of size $|L| = O(m \log \frac{k}{\delta})$. Use the same trick as in isolation, it takes $O(M) = O(\frac{k}{\epsilon_2} \log \frac{1}{\epsilon})$ samples for each of $O(\log \frac{|L|}{\delta}) = O(\log \frac{m}{\delta}) = O(\log \frac{k}{\delta\epsilon_2})$ repetitions. Hence the estimation takes $O(\frac{k}{\epsilon_2} \log \frac{k}{\delta\epsilon_2} \log \frac{1}{\epsilon})$ samples.

The total number of samples is therefore

$$O\left(\frac{k}{\epsilon_2} \log \frac{1}{\epsilon} \left(\log \frac{k}{\delta} \log \frac{1}{\eta} + \log \frac{k}{\delta\epsilon_2}\right)\right) = O\left(\frac{k}{\epsilon_2} \log \frac{k}{\delta} \log \frac{1}{\epsilon_2\eta} \log \frac{k}{\min\{\epsilon_1, \epsilon_2\}}\right).$$

Runtime.

It follows from Remark 4.8 that each isolation repetition takes $O((M + m \log m)r) = O(\frac{k}{\epsilon_2} \log \frac{k}{\epsilon\epsilon_2} \log \frac{1}{\eta}) = O(\frac{k}{\epsilon_2} \log \frac{1}{\epsilon} \log \frac{1}{\eta})$ time. There are $O(\log \frac{k}{\delta})$ repetitions so the total time for isolation is $O(\frac{k}{\epsilon_2} \log \frac{k}{\delta} \log \frac{1}{\epsilon} \log \frac{1}{\eta})$.

The input of ESTIMATE is a list L of size $|L| = O(m \log \frac{k}{\delta})$. Use the same trick in Remark 4.8, it stakes $O(M + m \log m + |L|)$ time to obtain values for all buckets and compute $a_\omega^{(s)}$ and $b_\omega^{(s)}$ for all $\omega \in L$ and each s. Hence line 3–6 of ESTIMATE takes time $O((M + m \log m + |L|) \log \frac{m}{\delta}) = O(\frac{k}{\epsilon_2} \log \frac{1}{\epsilon} \log \frac{k}{\delta\epsilon_2})$ time. Thus estimation takes time $O(\frac{k}{\epsilon_2} \log \frac{1}{\epsilon} \log \frac{k}{\delta\epsilon_2} + |L| \log \frac{m}{\delta} + |L| \log |L|) = O(\frac{k}{\epsilon_2} \log \frac{k}{\delta} (\log \frac{1}{\epsilon} \log \frac{1}{\epsilon_2} + \log \frac{1}{\delta}))$.

The total running time is therefore

$$\begin{aligned} O&\left(\frac{k}{\epsilon_2} \log \frac{k}{\delta} \left(\log \frac{1}{\epsilon} \log \frac{1}{\eta} + \log \frac{1}{\epsilon} \log \frac{1}{\epsilon_2} + \log \frac{1}{\delta}\right)\right) \\ &= O\left(\frac{k}{\epsilon_2} \log \frac{k}{\delta} \left(\log \frac{1}{\epsilon_2\eta} \log \frac{k}{\min\{\epsilon_1, \epsilon_2\}} + \log \frac{1}{\delta}\right)\right). \end{aligned}$$

Sample Duration.

It is clear that the sample duration is

$$O(Md) = O\left(\frac{M}{\eta}\right) = O\left(\frac{k}{\epsilon_2\eta}\log\frac{k}{\min\{\epsilon_1, \epsilon_2\}}\right).$$

Output Evaluation Metric.

Since we do not expect to recover the frequencies exactly, the typical approximation error of the form

$$\left\|\sum_j \left(a_j e^{i\omega_j x} - a'_j e^{i\omega'_j x}\right) + \nu(x)\right\|_p$$

contains both the coefficient approximation error $\|a - a'\|$ and a term of the form $\sum |a_j||\omega_j - \omega'_j|$, rather than the more usual bound in terms of the noise alone $\|\nu\|_p$ in the discrete case. Given bounds on both the coefficients $|a_j - a'_j|$ and the frequencies $|\omega_j - \omega'_j|$, it is possible to compute the two terms in the error. This is standard in the literature of polynomial-time algorithms to recover real frequencies (e.g., [PPT11], with which our result is comparable).

An alternative view is to treat the exponential sum $\sum a_i e^{i\omega x}$ as a distribution and consider

$$\sup_{g\in\mathcal{F}} |\langle f_1, g\rangle - \langle f_2, g\rangle|,$$

where f_1 and f_2 are two exponential sums and $\mathcal{F}$ some class of test functions. For instance, when f_1 and f_2 are probability measures, it gives total variation distance when $\mathcal{F}$ is the set of functions bounded by 1 and Wasserstein distance (or earth-mover's distance) when $\mathcal{F}$ consists of all 1-Lipschitz functions. But our algorithm has no guarantee that $\|a\| = \|a'\|$ so it is generally not a metric. Obviously this bound can be expressed in terms of $|a_i - a'_i|$ and $|\omega_i - \omega'_i|$ (for example, with $\mathcal{F}$ being the set of 1-Lipschitz function we obtain the L^1 norm) and can thus be bounded if the estimates of individual source are bounded. It remains unclear what class of test functions would give a less common but interesting bound for this problem.

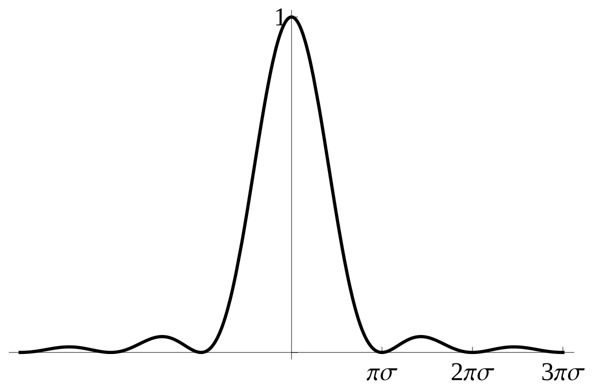

Figure 4.3: $\widehat{W}(\xi) = \sin^2(\xi/\sigma)/(\xi/\sigma)^2$

4.4 Discretization

As mentioned in the introduction, one may be tempted to reduce the continuous setting to the discrete setting, assuming

$$f(t) = \sum_{i=j}^{k} a_j e^{i\omega_j t}, \quad a_j \in \mathbb{C},$$

by simply taking samples of $f(t)w(t)$ for some window function $W(t)$ at N equidistant points $t = 0$, Δt, $2\Delta t$, $\dots$, $(N-1)\Delta t$. The discrete Fourier transform (DFT) of the samples are approximately

$$\frac{1}{\Delta t}\sum_{j=1}^{k} a_j \widehat{W}\left(\omega_j - \frac{\ell}{N\Delta t}\right), \quad \ell = 0, 1, \dots, N-1$$

by observing that

$$\sum_{k=0}^{N-1} e^{i\omega k\Delta t} W(k\Delta t) e^{-i\frac{k\ell}{N}} \approx \frac{1}{\Delta t}\int_0^T e^{i\omega x} W(x) e^{-i\ell\frac{x}{N\Delta t}} dt \approx \frac{1}{\Delta t}\widehat{W}\left(\omega - \frac{\ell}{N\Delta t}\right)$$

provided that $\Delta t \lesssim 1/\pi$ (so the Riemann sum is a good approximation to the integral) and $W(t)$ is supported on, or negligible outside, $[0, N\Delta t]$. A typical choice of $\widehat{W}$ is also a window function. Suppose that the pass region of $\widehat{W}$ has width $\sigma \lesssim \eta$ to avoid the interference of two different frequencies. The pass region of W is typically $1/\sigma \lesssim N\Delta t$, hence $1/\eta \lesssim 1/\sigma \lesssim N\Delta t$. Take $\widehat{W}(\xi) = \sin^2(\xi/\sigma)/(\xi/\sigma)^2$ (See Figure 4.3). Consider ω_1 and let ℓ be the nearest integer to $N\Delta t \cdot \omega_1$. Then the ℓ-th coefficient in the DFT is

$$\frac{1}{\Delta t}\left(a_1\widehat{W}\left(\omega_1-\frac{\ell}{N\Delta t}\right)+\sum_{j\neq 1}a_j\widehat{W}\left(\omega_j-\frac{\ell}{N\Delta t}\right)\right).$$

The first term in the bracket is close to a_1 because $|\omega_1-\ell/(N\Delta t)|\leq 1/(2N\Delta t)\lesssim\sigma$ and thus $\widehat{W}$ is close to 1. To bound the second term, notice that for $j\neq 1$,

$$\left|\omega_j-\frac{\ell}{N\Delta t}\right|\geq|\omega-\omega_1|-\frac{1}{2N\Delta t}\geq|\omega-\omega_1|-\frac{1}{C\eta}$$

for some absolute constant $C>0$. And thus the second term is bounded by (by rearranging the indices)

$$\frac{|a_2|+|a_3|}{\left(1-\frac{1}{C}\right)^2\left(\frac{\eta}{\sigma}\right)^2}+\frac{|a_4|+|a_5|}{\left(2-\frac{1}{C}\right)^2\left(\frac{\eta}{\sigma}\right)^2}+\frac{|a_6|+|a_7|}{\left(3-\frac{1}{C}\right)^2\left(\frac{\eta}{\sigma}\right)^2}+\cdots\lesssim\left(\frac{\sigma}{\eta}\right)^2\cdot\max_{i\geq 2}|a_i|$$

This means that the ℓ-th coefficient in the DFT is proportional to the cofficient associated with ω_1 corrupted by contributions from other cofficients associated with other frequencies. It is therefore conceivable that there exists a constant C (depending on W) such that the top Ck coefficients of the N DFT coefficients include constant approximations to those $\{a_i\}$'s which are at least a constant fraction of $\|a\|_1$ with the choice of $\sigma\lesssim\eta$ and thus the sample duration is $\Theta(1/\eta)$. Using existing sparse recovery results for discrete setting, it seems probable to recover the frequencies with coefficients at least a constant fraction of $\|a\|_1$ with sample duration of $N\Delta t\simeq 1/\eta$. Improving the guarantee to recovering frequencies with coefficients at least $1/k$ fraction of $\|a\|_1$ will increase the sample duration to $\Theta(k/\eta)$ with σ shrunk from $\Theta(\eta)$ to $\Theta(\eta/k)$.

This approach looks promising yet there are some inconveniences compared with the preceding direct approach. For instance, typical discrete case results give an ℓ_2/ℓ_2 error bound, that is, $\|\mathbf{x}-\mathbf{x}'\|_2\leq(1+\epsilon)\|\mathbf{x}-\mathbf{x}_{[k]}\|_2$, where $\mathbf{x}'$ is the approximation of $\mathbf{x}$ and $\mathbf{x}_{[k]}$ the best k-term approximation. It is not obvious how to interpret such result in the continuous setting. On the other hand, under the discretization scenario, a real-valued frequency spreads around so one may encounter a cluster of significant components in the discretized signal and thus an additional step of locating the frequency from a cluster of them is needed. This is not obvious either provided only the ℓ_2 error guarantee. An ℓ_∞ error guarantee is more desirable, however, it increases the complexity of the algorithms for the discrete case.

4.5 Application to Bearing Estimation

Humans use two ears to determine the location of the source of sound. This localization problem falls in the area of *array signal processing*, which considers the problem of locating signal sources (transmitters) using an array of receivers (sensors) that measure the ambient wavefield. This problem has a rich history of research and is found in a broad variety of fields such as radar, sonar, seismology and biomedicine. We refer the readers to the classical references [JD92, Tre92] for more details. In this section, we shall consider the problem of finding the sources under the assumption that the number of sources (or an upper bound thereof) is known *a priori*. Our formulation is as follows.

A source on the plane emits a sine wave at a single frequency ω and this wave travels at speed c isotropically in this medium. If we ignore the decay of the amplitude of the wave as it travels, the source is localized by a single bearing parameter $\theta \in \mathbb{S}^1$, if we were to express its position in polar coordinates. Formally, a source at angle θ produces a wave field

$$F_\theta(x,t) = a_\theta \exp\left(i\omega \left(t + \frac{\langle x, n_\theta \rangle}{c} \right)\right), \quad x \in \mathbb{R}^2, t \in \mathbb{R},$$

where $n_\theta = (\cos\theta, \sin\theta)$ is the unit vector in the direction θ. Restricting the wave field to $x \in \mathbb{R}$, the horizontal axis, we have

$$F_\theta(x,t) = a_\theta \exp\left(i\omega \left(t + \frac{x\cos\theta}{c} \right)\right), \quad x \in \mathbb{R}, t \in \mathbb{R},$$

or, writing $\omega_\theta = \omega\cos\theta$ and assuming without loss of generality $c = 1$,

$$F_\theta(x,t) = a_\theta \mathrm{e}^{i\omega t + i\omega_\theta x} = a_\theta \mathrm{e}^{i\omega t}\mathrm{e}^{i\omega_\theta x}.$$

On the horizontal axis, the wavefield oscillates in both time t and in position $x \in \mathbb{R}$, separately.

Suppose that there are k sources, each transmitting sine waves at the same frequency ω and at angles $\theta_1, \ldots, \theta_k \in \mathbb{S}^1$, and there is background noise at frequency ω supported on $I_\nu \subset \mathbb{S}^1$. We assume that $\{[\theta_j - \eta/2, \theta_j + \eta/2)\}_{j=1}^k$ and $[I_\nu - \eta/2, I_\nu + \eta/2)$ are all mutually disjoint. That is, the sources are not on a fixed, discrete grid but they are separated from each other and from the noise by a minimum resolution angle η.

To simplify notation, we denote ω_{θ_j} by ω_j and observe that $|\omega_j| \leq \omega$. A single

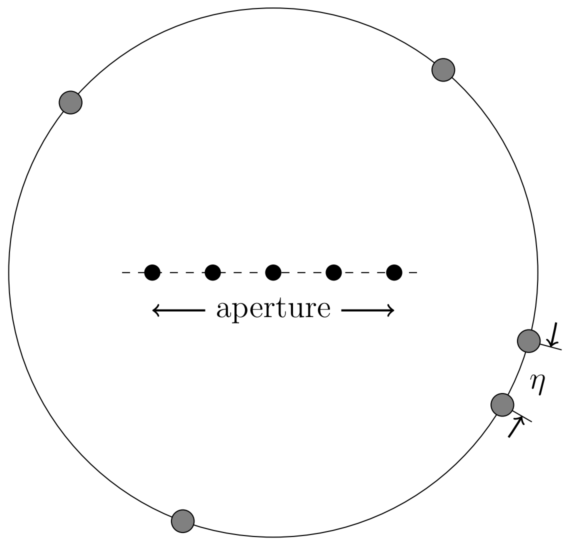

Figure 4.4: Receiver array and source configuration. The black nodes are receivers and the grey ones sources.

receiver at position x on the horizontal axis observes the wavefield as

$$f(x,t) = \sum_{j=1}^{k} a_j e^{i\omega t} e^{i\omega_j x} + e^{i\omega t} \left\{ \int_{I_\nu} a(\theta) e^{i\omega \cos\theta \cdot x} d\mu \right\}, \quad x, t \in \mathbb{R},$$

where μ is a measure on $\mathbb{S}^1$ that satisfies the assumptions prescribed in Section 4.2.1.

The goal of the bearing estimation problem is to construct a (distribution over) placements x_m of M receivers and, from observations $y(x_m, t_0)$ at a **fixed time** t_0, find the amplitudes $\{a_j\}$ and positions or angles $\{\omega_j\}$ of the sources. This problem has the form of an off-grid Fourier sampling problem where we seek the identity of the unknown k "frequencies" ω_j from M samples of a sparse signal plus background noise. Figure 4.4 shows the configuration of sources and receivers.

Choose $0 < \beta < \pi/2$ and consider $\theta \in J = [-(\pi - \beta), -\beta] \cup [\beta, \pi - \beta]$, whence $|\sin\theta| \geq \sin\beta$. Furthermore, we have

$$|\omega \cos\theta_1 - \omega \cos\theta_2| \geq \omega(\sin\beta)|\theta_1 - \theta_2|$$

for $\theta_1, \theta_2 \in J$. Thus, it follows for $\theta_j \in J$ that ω_j is separated from the other frequencies and the noise by at least $\omega\eta\sin\beta$. Because there is nothing special about placing the receivers on the horizontal axis, we can also consider receivers distributed on a line that is rotated a constant number of times with respect to the horizontal axes so that the translations of J cover $\mathbb{S}^1$ altogether, it suffices to find ω_j (and the associated a_j) with $\theta_j \in J$ for a fixed β. We also re-define the angular resolution η to be $\eta \sin\beta$. We hope that in this way we can reduce the problem to recovery of $\{\omega_j\}$, or, $\{\cos\theta_j\}$. However, ambiguity arises since $\cos\theta = \cos(-\theta)$, that is, sources symmetric around the horizontal line cannot be distinguished. Also a source that is close to the symmetric image of the other source may ruin the minimum separation in the reduced problem. Therefore we further make the following assumption: there exists an integer $q > 4$ such that for each pair of

$$E_p = \left\{\theta_j : \theta_j \in \left[\frac{\pi}{4} + (2p-1)\frac{\pi}{q} - 2\eta, \frac{\pi}{4} + (2p+1)\frac{\pi}{q} + 2\eta\right]\right\}$$

and

$$E'_p = \left\{\theta_j : \theta_j \in \left[-\frac{\pi}{4} + (2p-1)\frac{\pi}{q} - 2\eta, -\frac{\pi}{4} + (2p+1)\frac{\pi}{q} + 2\eta\right]\right\}, \quad p = 0, \ldots, q-1,$$

it holds that $d(-E_p + \{\frac{2p\pi}{q}\}, E'_p - \{\frac{2p\pi}{q}\}) \geq \eta$ and that $d(-E_p + \{\frac{2p\pi}{q}\}, I_\nu) \geq \eta$, where $d(\cdot,\cdot)$ is the metric on $\mathbb{S}^1$.

Consider a filter K on $\mathbb{S}^1$ with a finite Fourier transform $\hat{K}$, supported on $I \subset \mathbb{Z}$. Then, placing receivers down on a line at positions $\{nx\}_{n\in I}$ associated with weights $\{\hat{K}(n)\}_{n\in I}$, we find that the wavefield these receivers observe is

$$\begin{aligned}\sum_{n\in I} \hat{K}(n) y(nx) &= \sum_{j=1}^{k} a_j \sum_{n\in I} \hat{K}(n)\exp(i\omega_j nx) + \int_{I_\nu} a(\theta) \sum_{n\in I} \hat{K}(n)\exp\left(i\omega(\cos\theta)nx\right) d\mu \\ &= \sum_{j=1}^{k} a_j K(\omega_j x) + \int_{I_\nu} a(\theta) K(\omega(\cos\theta)x) d\mu.\end{aligned}$$

It is clear that any translation $K(u+\cdot)$ can be achieved by the same receiver array with associated weights $\{\hat{K}(n)\mathrm{e}^{iun}\}_{n\in I}\}$ and that scaling $K(\alpha x)$ can be achieved by scaling the receiver array by the same factor α. Thus, we can perform all of the required measurement techniques for sampling in a receiver array.

The following result is an immediate application of our main result.

Theorem 4.13. *There is a distribution $\mathcal{D}$ on uniform receiver arrays and an algorithm $\mathcal{A}$ such that for each wavefield emanating from k sources $f(s) = \sum_{j=1}^{k} a_j \mathrm{e}^{i\omega_j s} + \int_{I_\nu} a(\theta)\mathrm{e}^{i\omega s} d\mu$, with constant probability, given observations from the receiver arrays, the algorithm returns a list Λ of amplitudes and bearings $\Lambda = \{(a'_j, \omega'_j)\}_{j=1}^{k}$ such that*

1. *For each a_j that satisfies (4.2) there exists $\omega_j' \in \Lambda$ such that*

$$|\omega_j - \omega_j'| \leq \frac{\epsilon_2}{k}\eta.$$

2. *Let $\Lambda_0 = \left\{\omega_j' \in \Lambda : \exists \omega_{j_0} \text{ such that } \left|\omega_{j_0} - \omega_j'\right| \leq \frac{\epsilon_2 \eta}{k} \text{ and } |a_{j_0}| \text{ satisfies } (4.2)\right\}$, then for each $\omega_j' \in \Lambda_0$ it holds that*

$$|a_j' - a_j| \leq \frac{\epsilon_1}{k}\|a\|_1 + \frac{\epsilon_2}{k}\|\nu\|_1.$$

3. *For each $\omega_j' \in \Lambda \setminus \Lambda_0$, it holds that*

$$|a_j'| \leq \frac{\epsilon_1}{k}\|a\|_1 + \frac{\epsilon_2}{k}\|\nu\|_1.$$

The algorithm places

$$O\left(\frac{k}{\epsilon_2}\log k \log\frac{1}{\epsilon_2\eta}\log\frac{k}{\min\{\epsilon_1,\epsilon_2\}}\right)$$

receivers and runs in time proportional to number of receivers. Furthermore, the receiver aperture size is

$$O\left(\frac{k}{\epsilon_2\eta}\log\frac{k}{\min\{\epsilon_1,\epsilon_2\}}\right).$$

Proof. By our assumption on E_p and E_p', we can recover the sources in

$$\left[\frac{\pi}{4} + (2p-1)\frac{\pi}{q}, \frac{\pi}{4} + (2p+1)\frac{\pi}{q}\right] \cup \left[-\frac{\pi}{4} + (2p-1)\frac{\pi}{q}, -\frac{\pi}{4} + (2p+1)\frac{\pi}{q}\right]$$

with symmetry ambiguity. With rotations of the receiver arrays, the only ambiguity left will be to distinguish a source at θ from $\theta + \pi$. Suppose that $\theta \in E_{p_0}$. This ambiguity can be resolved by rotating the receiver array for E_{p_0} by η, the correctness of which is guaranteed by our assumption on E_p and E_p' again, noting the 2η brim on each side of each interval. $\square$

We remark that it is possible to handle more configurations of sources, such as sources at the vertices of a regular polygon, by starting with a random direction instead of x-axis.

4.6 Conclusion and Open Problems

In this chapter, we define a mathematically rigorous and practical signal model for sampling sparse Fourier signals with continuously placed frequencies and devise a sublinear time algorithm for recovering such signals. There are a number of technical difficulties in this model with directly applying the discrete sublinear Fourier sampling techniques, both algorithmic and mathematical. We leave the following problems open and make conjectures.

- Several direct techniques incur the penalty of extra measurements. We do not know if these additional measurements are necessary, if they are inherent in the model.

- Unlike the discrete case, the "duration" of the sampling or the extent of the samples is a resource for which we have no lower bounds. We think $\Theta(k/\eta)$ is the tight bound with $k \log^{O(1)}(1/\eta)$ samples while our algorithm takes $O((k/\eta)\log k)$ samples for $\epsilon_1 = \Theta(1/k)$ and $\epsilon_2 = \Theta(1)$. It remains future work to devise a sublinear time algorithm with smaller sample duration, for which a good reduction to discrete case looks a promising approach.

- If not reducing to discrete case, it would be good to design an iterative algorithm so the runtime can be lowered by an $O(\log k)$ factor. Notice that the number of samples is always bounded by the runtime from above.

BIBLIOGRAPHY

[Alo86] N. Alon. Eigenvalues and expanders. *Combinatorica*, 6(2):83–96, 1986.

[AMS99] N. Alon, Y. Matias, and M. Szegedy. The space complexity of approximating the frequency moments. *J. Comput. Syst. Sci.*, 58(1):137–147, 1999.

[BCG$^+$12] P. Boufounos, V. Cevher, A. C. Gilbert, Y. Li, and M. J. Strauss. What's the frequency, Kenneth?: Sublinear Fourier sampling off the grid. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, volume 7408 of *Lecture Notes in Computer Science*, pages 61–72. Springer, 2012.

[BCW10] R. Baraniuk, V. Cevher, and M. Wakin. Low-dimensional models for dimensionality reduction and signal recovery: A geometric perspective. *Proceedings of the IEEE*, 98(6):959–971, 2010.

[BDF$^+$11] J. Bourgain, S. Dilworth, K. Ford, S. Konyagin, and D. Kutzarova. Explicit constructions of rip matrices and related problems. *Duke Math. J.*, 159(1):145–185, 2011.

[BGI$^+$08] R. Berinde, A. C. Gilbert, P. Indyk, H. Karloff, and M. J. Strauss. Combining geometry and combinatorics: A unified approach to sparse signal recovery. In *Communication, Control, and Computing, 2008 46th Annual Allerton Conference on*, pages 798–805, 2008.

[BIPW10] K. D. Ba, P. Indyk, E. Price, and D. Woodruff. Lower bounds for sparse recovery. In *ACM SODA*, page to appear, 2010.

[CCFC02] M. Charikar, K. Chen, and M. Farach-Colton. Finding frequent items in data streams. In *Proceedings of the 29th International Colloquium on Automata, Languages and Programming (ICALP)*, pages 693–703, 2002.

[CDD09] A. Cohen, W. Dahmen, and R. Devore. Compressed sensing and best k-term approximation. *J. Amer. Math. Soc*, pages 211–231, 2009.

[CEPR09] R. Clifford, K. Efremenko, E. Porat, and A. Rothschild. From coding theory to efficient pattern matching. In *Proceedings of the twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '09, pages

778–784, Philadelphia, PA, USA, 2009. Society for Industrial and Applied Mathematics.

[CGV13] M. Cheraghchi, V. Guruswami, and A. Velingker. Restricted isometry of fourier matrices and list decodability of linear codes. In *SODA*, 2013.

[CM03] G. Cormode and S. Muthukrishnan. What's hot and what's not: Tracking most frequent items dynamically. In *Proc. ACM Principles of Database Systems*, pages 296–306, 2003.

[CM05] G. Cormode and S. Muthukrishnan. What's hot and what's not: tracking most frequent items dynamically. *ACM Trans. Database Syst.*, 30(1):249–278, 2005.

[CM06] G. Cormode and S. Muthukrishnan. Combinatorial algorithms for Compressed Sensing. In *Proc. 40th Ann. Conf. Information Sciences and Systems*, Princeton, Mar. 2006.

[CP07] R. Clifford and E. Porat. A filtering algorithm for k-mismatch with don't cares. In *String Processing and Information Retrieval*, volume 4726 of *Lecture Notes in Computer Science*, pages 130–136. Springer, 2007.

[CRT06] E. Candès, J. Romberg, and T. Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Inf. Theory*, 52(2):489–509, 2006.

[DDT$^+$08] M. F. Duarte, M. A. Davenport, D. Takhar, J. N. Laska, K. F. Kelly, and R. G. Baraniuk. Single-pixel imaging via compressive sampling. *IEEE Signal Processing Magazine*, 25(2):83–91, 2008.

[DM09] W. Dai and O. Milenkovic. Subspace pursuit for compressive sensing signal reconstruction. *IEEE Transactions on Information Theory*, 55(5):2230–2249, 2009.

[Don06] D. L. Donoho. Compressed Sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, Apr. 2006.

[DP96] D. P. Dubhashi and V. Priebe. Negative dependence through the FKG inequality. Technical Report MPI-I-96-1-020, Max-Planck Institut für Informatik, Saarbrücken, 1996.

[ECG$^+$09] Y. Erlich, K. Chang, A. Gordon, R. Ronen, O. Navon, M. Rooks, and G. J. Hannon. DNA sudoku—harnessing high-throughput sequencing for multiplexed specimen analysis. *Genome Research*, 19:1243—1253, 2009.

[FKS89] J. Friedman, J. Kahn, and E. Szemerédi. On the second eigenvalue of random regular graphs. In *Proceedings of the twenty-first annual ACM symposium on Theory of computing*, STOC '89, pages 587–598, 1989.

[GGI$^+$02] A. C. Gilbert, S. Guha, P. Indyk, S. Muthukrishnan, and M. Strauss. Near-optimal sparse fourier representations via sampling. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, STOC '02, pages 152–161, New York, NY, USA, 2002. ACM.

[GI10] A. C. Gilbert and P. Indyk. Sparse recovery using sparse matrices. *Proceedings of the IEEE*, 98(6):937–947, 2010.

[GLPS12] A. C. Gilbert, Y. Li, E. Porat, and M. J. Strauss. Approximate sparse recovery: Optimizing time and measurements. *SIAM J. Comput.*, 41(2):436–453, 2012.

[GMS05] A. C. Gilbert, S. Muthukrishnan, and M. J. Strauss. Improved time bounds for near-optimal sparse Fourier representation via sampling. In *Proc. SPIE Wavelets XI*, San Diego, 2005.

[GNP$^+$13] A. C. Gilbert, H. Q. Ngo, E. Porat, A. Rudra, and M. J. Strauss. ℓ_2/ℓ_2-foreach sparse recovery with low risk. In *Proceedings of ICALP*, 2013.

[GSTV06] A. C. Gilbert, M. J. Strauss, J. A. Tropp, and R. Vershynin. Algorithmic linear dimension reduction in the ℓ_1 norm for sparse vectors. In *Proceedings of 44th Annual Allerton Conference on Communication, Control, and Computing*, pages 1411–1418, 2006.

[GSTV07] A. C. Gilbert, M. J. Strauss, J. A. Tropp, and R. Vershynin. One sketch for all: fast algorithms for compressed sensing. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, STOC '07, pages 237–246, 2007.

[GUV09] V. Guruswami, C. Umans, and S. Vadhan. Unbalanced expanders and randomness extractors from Parvaresh-Vardy codes. *J. ACM*, 56(4):20:1–20:34, July 2009.

[Hel95] H. Helson. *Harmonic Analysis (Second Edition)*. Hindustan Book Agency, 1995.

[HIKP12a] H. Hassanieh, P. Indyk, D. Katabi, and E. Price. Nearly optimal sparse Fourier transform. In *Proceedings of the 44th symposium on Theory of Computing*, STOC '12, pages 563–578, New York, NY, USA, 2012. ACM.

[HIKP12b] H. Hassanieh, P. Indyk, D. Katabi, and E. Price. Simple and practical algorithm for sparse Fourier transform. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '12, pages 1183–1194. SIAM, 2012.

[INR10] P. Indyk, H. Q. Ngo, and A. Rudra. Efficiently decodable non-adaptive group testing. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '10, pages 1126–1142, 2010.

[IR08] P. Indyk and M. Ruzic. Near-optimal sparse recovery in the L_1 norm. *Foundations of Computer Science*, pages 199–207, 2008.

[Iwe09] M. Iwen. Combinatorial sublinear-time Fourier algorithms. *Foundations of Computational Mathematics*, 10(3):303–338, 2009.

[JD92] D. H. Johnson and D. E. Dudgeon. *Array Signal Processing: Concepts and Techniques*. Simon & Schuster, 1992.

[KM93] E. Kushilevitz and Y. Mansour. Learning decision trees using the fourier spectrum. *SIAM Journal on Computing*, 22(6):1331–1348, 1993.

[LDP07] M. Lustig, D. Donoho, and J. M. Pauly. Sparse MRI: The application of compressed sensing for rapid MR imaging. *Magnetic Resonance in Medicine*, 58(6):1182–1195, 2007.

[NNW12] J. Nelson, H. L. Nguyên, and D. P. Woodruff. On deterministic sketching and streaming for sparse recovery and norm estimation. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, volume 7408 of *Lecture Notes in Computer Science*, pages 627–638. Springer, 2012.

[NT09] D. Needell and J. A. Tropp. CoSaMP: Iterative signal recovery from incomplete and inaccurate samples. *Appl. Comp. Harmonic Anal.*, 26(3):301–321, 2009.

[PAW07] J. L. Paredes, G. R. Arce, and Z. Wang. Ultra-wideband compressed sensing: Channel estimation. *IEEE Journal of Selected Topics in Signal Processing*, 1(3):383–395, 2007.

[PPT11] T. Peter, D. Potts, and M. Tasche. Nonlinear approximation by sums of exponentials and translates. *SIAM J. Sci. Comput.*, 33(4):1920–1947, 2011.

[PS12] E. Porat and M. J. Strauss. Sublinear time, measurement-optimal, sparse recovery for all. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '12, pages 1215–1227, 2012.

[PV05] F. Parvaresh and A. Vardy. Correcting errors beyond the guruswami-sudan radius in polynomial time. In *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science*, FOCS '05, pages 285–294, 2005.

[PW11] E. Price and D. P. Woodruff. $(1+\epsilon)$-approximate sparse recovery. In *Proceedings of the 2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*, FOCS '11, pages 295–304, 2011.

[Ric] Rice DSP group. Compressive sensing resources. `http://dsp.rice.edu/cs`. Accessed: 3 April 2013.

[RV08] M. Rudelson and R. Veshynin. On sparse reconstruction from Fourier and Gaussian measurements. *Communications on Pure and Applied Mathematics*, 61:1025–1045, Mar. 2008.

[Tre92] H. L. V. Trees. *Detection, Estimation, and Modulation Theory: Radar-Sonar Signal Processing and Gaussian Signals in Noise.* Krieger Publishing Co., Inc., Melbourne, FL, USA, 1992.

[Upf92] E. Upfal. Tolerating linear number of faults in networks of bounded degree. In *Proceedings of the eleventh annual ACM symposium on Principles of distributed computing*, PODC '92, pages 83–89, 1992.

[VMB02] M. Vetterli, P. Marziliano, and T. Blu. Sampling signals with finite rate of innovation. *IEEE Transactions on Signal Processing*, 50(6):1417–1428, 2002.

[ZBSG05] Y. H. Zheng, D. J. Brady, M. E. Sullivan, and B. D. Guenther. Fiber-optic localization by geometric space coding with a two-dimensional gray code. *Applied Optics*, 44(20):4306–4314, 2005.

[ZPB06] Y. H. Zheng, N. P. Pitsianis, and D. J. Brady. Nonadaptive group testing based fiber sensor deployment for multiperson tracking. *IEEE Sensors Journal*, 6(2):490–494, 2006.