Nearly Optimal Sparse Fourier Transform

Haitham Hassanieh MIT
Piotr Indyk MIT
Dina Katabi MIT
Eric Price MIT

{haithamh,indyk,dk,ecprice}@mit.edu

Abstract

We consider the problem of computing the k-sparse approximation to the discrete Fourier transform of an n-dimensional signal. We show:

- An $O(k \log n)$-time randomized algorithm for the case where the input signal has at most k non-zero Fourier coefficients, and
- An $O(k \log n \log(n/k))$-time randomized algorithm for general input signals.

Both algorithms achieve $o(n \log n)$ time, and thus improve over the Fast Fourier Transform, for any $k = o(n)$. They are the first known algorithms that satisfy this property. Also, if one assumes that the Fast Fourier Transform is optimal, the algorithm for the exactly k-sparse case is optimal for any $k = n^{\Omega(1)}$.

We complement our algorithmic results by showing that any algorithm for computing the sparse Fourier transform of a general signal must use at least $\Omega(k \log(n/k)/\log \log n)$ signal samples, even if it is allowed to perform *adaptive* sampling.

1 Introduction

The discrete Fourier transform (DFT) is one of the most important and widely used computational tasks. Its applications are broad and include signal processing, communications, and audio/image/video compression. Hence, fast algorithms for DFT are highly valuable. Currently, the fastest such algorithm is the Fast Fourier Transform (FFT), which computes the DFT of an n-dimensional signal in $O(n \log n)$ time. The existence of DFT algorithms faster than FFT is one of the central questions in the theory of algorithms.

A general algorithm for computing the exact DFT must take time at least proportional to its output size, i.e., $\Omega(n)$. In many applications, however, most of the Fourier coefficients of a signal are small or equal to zero, i.e., the output of the DFT is (approximately) *sparse*. This is the case for video signals, where a typical 8x8 block in a video frame has on average 7 non-negligible frequency coefficients (i.e., 89% of the coefficients are negligible) [CGX96]. Images and audio data are equally sparse. This sparsity provides the rationale underlying compression schemes such as MPEG and JPEG. Other sparse signals appear in computational learning theory [KM91, LMN93], analysis of Boolean functions [KKL88, O'D08], compressed sensing [Don06, CRT06], multi-scale analysis [DRZ07], similarity search in databases [AFS93], spectrum sensing for wideband channels [LVS11], and datacenter monitoring [MNL10].

For sparse signals, the $\Omega(n)$ lower bound for the complexity of DFT no longer applies. If a signal has a small number k of non-zero Fourier coefficients – the *exactly k-sparse* case – the output of the Fourier transform can be represented succinctly using only k coefficients. Hence, for such signals, one may hope for a DFT algorithm whose runtime is sublinear in the signal size, n. Even for a general n-dimensional signal x – the *general case* – one can find an algorithm that computes the best *k-sparse approximation* of its Fourier transform, $\widehat{x}$, in sublinear time. The goal of such an algorithm is to compute an approximation vector $\widehat{x}'$ that satisfies the following ℓ_2/ℓ_2 *guarantee*:

$$\|\widehat{x} - \widehat{x}'\|_2 \leq C \min_{k\text{-sparse } y} \|\widehat{x} - y\|_2, \tag{1}$$

where C is some approximation factor and the minimization is over k-sparse signals. We allow the algorithm to be *randomized*, and only succeed with constant (say, 2/3) probability.

The past two decades have witnessed significant advances in sublinear sparse Fourier algorithms. The first such algorithm (for the Hadamard transform) appeared in [KM91] (building on [GL89]). Since then, several sublinear sparse Fourier algorithms for complex inputs have been discovered [Man92, GGI+02, AGS03, GMS05, Iwe10, Aka10, HIKP12b]. These algorithms provide[1] the guarantee in Equation (1).[2]

The main value of these algorithms is that they outperform FFT's runtime for sparse signals. For very sparse signals, the fastest algorithm is due to [GMS05] and has $O(k \log^c(n) \log(n/k))$ runtime, for some[3] $c > 2$. This algorithm outperforms FFT for any k smaller than $\Theta(n/\log^a n)$ for some $a > 1$. For less sparse signals, the fastest algorithm is due to [HIKP12b], and has $O(\sqrt{nk} \log^{3/2} n)$ runtime. This algorithm outperforms FFT for any k smaller than $\Theta(n/\log n)$.

Despite impressive progress on sparse DFT, the state of the art suffers from two main limitations:

1. None of the existing algorithms improves over FFT's runtime for the whole range of sparse signals, i.e., $k = o(n)$.
2. Most of the aforementioned algorithms are quite complex, and suffer from large "big-Oh" constants (the algorithm of [HIKP12b] is an exception, but has a running time that is polynomial in n).

Results. In this paper, we address these limitations by presenting two new algorithms for the sparse Fourier transform. We require that the length n of the input signal is a power of 2. We show:

- An $O(k \log n)$-time algorithm for the exactly k-sparse case, and
- An $O(k \log n \log(n/k))$-time algorithm for the general case.

The key property of both algorithms is their ability to achieve $o(n \log n)$ time, and thus improve over the FFT, for *any* $k = o(n)$. These algorithms are the first known algorithms that satisfy this property. Moreover, if one assume that FFT is optimal and hence the DFT cannot be computed in less than $O(n \log n)$ time, the algorithm for the exactly k-sparse case is *optimal*[4] as long as $k = n^{\Omega(1)}$. Under the same assumption, the result for the general case is at most one $\log \log n$ factor away from the optimal runtime for the case of "large" sparsity $k = n/\log^{O(1)} n$.

Furthermore, our algorithm for the exactly sparse case (depicted as Algorithm 3.1 on page 5) is quite simple and has low big-Oh constants. In particular, our preliminary implementation of a variant of this algorithm is faster than FFTW, a highly efficient implementation of the FFT, for $n = 2^{22}$ and $k \leq 2^{17}$ [HIKP12a]. In contrast, for the same signal size, prior algorithms were faster than FFTW only for $k \leq 2000$ [HIKP12b].[5]

We complement our algorithmic results by showing that any algorithm that works for the general case must use at least $\Omega(k \log(n/k)/\log \log n)$ samples from x. The lower bound uses techniques from [PW11], which shows a lower bound of $\Omega(k \log(n/k))$ for the number of *arbitrary* linear measurements needed to compute the k-sparse approximation of an n-dimensional vector $\widehat{x}$. In comparison to [PW11], our bound is slightly worse but it holds even for *adaptive* sampling, where the algorithm selects the samples based on the values of the previously sampled coordinates.[6] Note that our algorithms are *non-adaptive*, and thus limited by the more stringent lower bound of [PW11].

[1] The algorithm of [Man92], as stated in the paper, addresses only the exactly k-sparse case. However, it can be extended to the general case using relatively standard techniques.

[2] All of the above algorithms, as well as the algorithms in this paper, need to make some assumption about the precision of the input; otherwise, the right-hand-side of the expression in Equation (1) contains an additional additive term. See Preliminaries for more details.

[3] The paper does not estimate the exact value of c. We estimate that $c \approx 3$.

[4] One also needs to assume that k divides n. See Section 5 for more details.

[5] Note that both numbers ($k \leq 2^{17}$ and $k \leq 2000$) are for the exactly k-sparse case. The algorithm in [HIKP12b] can deal with the general case, but the empirical runtimes are higher.

[6] Note that if we allow *arbitrary* adaptive linear measurements of a vector $\widehat{x}$, then its k-sparse approximation can be computed using only $O(k \log \log(n/k))$ samples [IPW11]. Therefore, our lower bound holds only where the measurements, although adaptive, are limited to those induced by the Fourier matrix. This is the case when we want to compute a sparse approximation to $\widehat{x}$ from samples of x.

Techniques – overview. We start with an overview of the techniques used in prior works. At a high level, sparse Fourier algorithms work by binning the Fourier coefficients into a small number of bins. Since the signal is sparse in the frequency domain, each bin is likely[7] to have only one large coefficient, which can then be located (to find its position) and estimated (to find its value). The binning has to be done in sublinear time, and thus these algorithms bin the Fourier coefficients using an n-dimensional filter vector G that is concentrated both in time and frequency. That is, G is zero except at a small *number* of time coordinates, and its Fourier transform $\hat{G}$ is negligible except at a small *fraction* (about $1/k$) of the frequency coordinates, representing the filter's "pass" region. Each bin essentially receives only the frequencies in a narrow range corresponding to the pass region of the (shifted) filter $\hat{G}$, and the pass regions corresponding to different bins are disjoint. In this paper, we use filters introduced in [HIKP12b]. Those filters (defined in more detail in Preliminaries) have the property that the value of $\hat{G}$ is "large" over a constant fraction of the pass region, referred to as the "super-pass" region. We say that a coefficient is "isolated" if it falls into a filter's super-pass region and no other coefficient falls into filter's pass region. Since the super-pass region of our filters is a constant fraction of the pass region, the probability of isolating a coefficient is constant.

To achieve the stated running times, we need a fast method for locating and estimating isolated coefficients. Further, our algorithm is iterative, so we also need a fast method for updating the signal so that identified coefficients are not considered in future iterations. Below, we describe these methods in more detail.

New techniques – location and estimation. Our location and estimation methods depends on whether we handle the exactly sparse case or the general case. In the exactly sparse case, we show how to estimate the position of an isolated Fourier coefficient using only two samples of the filtered signal. Specifically, we show that the phase difference between the two samples is linear in the index of the coefficient, and hence we can recover the index by estimating the phases. This approach is inspired by the frequency offset estimation in orthogonal frequency division multiplexing (OFDM), which is the modulation method used in modern wireless technologies (see [HT01], Chapter 2).

In order to design an algorithm[8] for the general case, we employ a different approach. Specifically, we can use two samples to estimate (with constant probability) individual bits of the index of an isolated coefficient. Similar approaches have been employed in prior work. However, in those papers, the index was recovered bit by bit, and one needed $\Omega(\log \log n)$ samples per bit to recover *all* bits correctly with constant probability. In contrast, in this paper we recover the index one *block of bits* at a time, where each block consists of $O(\log \log n)$ bits. This approach is inspired by the fast sparse recovery algorithm of [GLPS10]. Applying this idea in our context, however, requires new techniques. The reason is that, unlike in [GLPS10], we do not have the freedom of using arbitrary "linear measurements" of the vector $\hat{x}$, and we can only use the measurements induced by the Fourier transform.[9] As a result, the extension from "bit recovery" to "block recovery" is the most technically involved part of the algorithm. Section 4.1 contains further intuition on this part.

New techniques – updating the signal. The aforementioned techniques recover the position and the value of any isolated coefficient. However, during each filtering step, each coefficient becomes isolated only with constant probability. Therefore, the filtering process needs to be repeated to ensure that each coefficient is correctly identified. In [HIKP12b], the algorithm simply performs the filtering $O(\log n)$ times and uses the median estimator to identify each coefficient with high probability. This, however, would lead to a running time of $O(k \log^2 n)$ in the k-sparse case, since each filtering step takes $k \log n$ time.

One could reduce the filtering time by subtracting the identified coefficients from the signal. In this way, the number of non-zero coefficients would be reduced by a constant factor after each iteration, so the

[7] One can randomize the positions of the frequencies by sampling the signal in time domain appropriately [GGI+02, GMS05]. See Preliminaries for the description.

[8] We note that although the two-sample approach employed in our algorithm works in theory only for the exactly k-sparse case, our preliminary experiments show that using a few more samples to estimate the phase works surprisingly well even for general signals.

[9] In particular, the method of [GLPS10] uses measurements corresponding to a random error correcting code.

cost of the first iteration would dominate the total running time. Unfortunately, subtracting the recovered coefficients from the signal is a computationally costly operation, corresponding to a so-called *non-uniform* DFT (see [GST08] for details). Its cost would override any potential savings.

In this paper, we introduce a different approach: instead of subtracting the identified coefficients from the *signal*, we subtract them directly from the *bins* obtained by filtering the signal. The latter operation can be done in time linear in the number of subtracted coefficients, since each of them "falls" into only one bin. Hence, the computational costs of each iteration can be decomposed into two terms, corresponding to filtering the original signal and subtracting the coefficients. For the exactly sparse case these terms are as follows:

- The cost of filtering the original signal is $O(B \log n)$, where B is the number of bins. B is set to $O(k')$, where k' is the the number of yet-unidentified coefficients. Thus, initially B is equal to $O(k)$, but its value decreases by a constant factor after each iteration.
- The cost of subtracting the identified coefficients from the bins is $O(k)$.

Since the number of iterations is $O(\log k)$, and the cost of filtering is dominated by the first iteration, the total running time is $O(k \log n)$ for the exactly sparse case.

For the general case, we need to set k' and B more carefully to obtain the desired running time. The cost of each iterative step is multiplied by the number of filtering steps needed to compute the location of the coefficients, which is $\Theta(\log(n/B))$. If we set $B = \Theta(k')$, this would be $\Theta(\log n)$ in most iterations, giving a $\Theta(k \log^2 n)$ running time. This is too slow when k is close to n. We avoid this by decreasing B more slowly and k' more quickly. In the r-th iteration, we set $B = k/\mathrm{poly}(r)$. This allows the total number of bins to remain $O(k)$ while keeping $\log(n/B)$ small—at most $O(\log \log k)$ more than $\log(n/k)$. Then, by having k' decrease according to $k' = k/r^{\Theta(r)}$ rather than $k/2^{\Theta(r)}$, we decrease the number of rounds to $O(\log k/ \log \log k)$. Some careful analysis shows that this counteracts the $\log \log k$ loss in the $\log(n/B)$ term, achieving the desired $O(k \log n \log(n/k))$ running time.

Organization of the paper. In Section 2, we give notation and definitions used throughout the paper. Sections 3 and 4 give our algorithm in the exactly k-sparse and the general case, respectively. Section 5 gives the reduction to the exactly k-sparse case from a k-dimensional DFT. Section 6 gives the sample complexity lower bound for the general case. Section 7 describes how to efficiently implement our filters. Finally, Section 8 discusses open problems arising from this work.

2 Preliminaries

This section introduces the notation, assumptions, and definitions used in the rest of this paper.

Notation. We use $[n]$ to denote the set $\{1, \ldots, n\}$, and define $\omega = e^{-2\pi \mathbf{i}/n}$ to be an nth root of unity. For any complex number a, we use $\phi(a) \in [0, 2\pi]$ to denote the *phase* of a. For a complex number a and a real positive number b, the expression $a \pm b$ denotes a complex number a' such that $|a - a'| \leq b$. For a vector $x \in \mathbb{C}^n$, its support is denoted by $\mathrm{supp}(x) \subset [n]$. We use $\|x\|_0$ to denote $|\mathrm{supp}(x)|$, the number of non-zero coordinates of x. Its Fourier spectrum is denoted by $\widehat{x}$, with

$$\widehat{x}_i = \frac{1}{\sqrt{n}} \sum_{j \in [n]} \omega^{ij} x_j.$$

For a vector of length n, indices should be interpreted modulo n, so $x_{-i} = x_{n-i}$. This allows us to define *convolution*

$$(x * y)_i = \sum_{j \in [n]} x_j y_{i-j}$$

and the *coordinate-wise product* $(x \cdot y)_i = x_i y_i$, so $\widehat{x \cdot y} = \widehat{x} * \widehat{y}$.

When $i \in \mathbb{Z}$ is an index into an n-dimensional vector, sometimes we use $|i|$ to denote $\min_{j \equiv i \pmod n} |j|$.

Definitions. The paper uses two tools introduced in previous papers: (pseudorandom) spectrum permutation [GGI+02, GMS05, GST08] and flat filtering windows [HIKP12b].

Definition 2.1. *Suppose σ^{-1} exists mod n. We define the* permutation $P_{\sigma,a,b}$ *by*

$$(P_{\sigma,a,b}x)_i = x_{\sigma(i-a)}\omega^{\sigma bi}.$$

We also define $\pi_{\sigma,b}(i) = \sigma(i-b) \bmod n$.

Claim 2.2. $\widehat{P_{\sigma,a,b}x}_{\pi_{\sigma,b}(i)} = \widehat{x}_i\omega^{a\sigma i}$.

Proof.

$$\begin{aligned}
\widehat{P_{\sigma,a,b}x}_{\sigma(i-b)} &= \frac{1}{\sqrt{n}}\sum_{j\in[n]}\omega^{\sigma(i-b)j}(P_{\sigma,a,b}x)_j \\
&= \frac{1}{\sqrt{n}}\sum_{j\in[n]}\omega^{\sigma(i-b)j}x_{\sigma(j-a)}\omega^{\sigma bj} \\
&= \omega^{a\sigma i}\frac{1}{\sqrt{n}}\sum_{j\in[n]}\omega^{i\sigma(j-a)}x_{\sigma(j-a)} \\
&= \widehat{x}_i\omega^{a\sigma i}.
\end{aligned}$$

□

Definition 2.3. *We say that $(G, \widehat{G'}) = (G_{B,\delta,\alpha}, \widehat{G'}_{B,\delta,\alpha}) \in \mathbb{R}^n \times \mathbb{R}^n$ is a* flat window function *with parameters $B \geq 1$, $\delta > 0$, and $\alpha > 0$ if $|\mathrm{supp}(G)| = O(\frac{B}{\alpha}\log(n/\delta))$ and $\widehat{G'}$ satisfies*

- $\widehat{G'}_i = 1$ *for* $|i| \leq (1-\alpha)n/(2B)$
- $\widehat{G'}_i = 0$ *for* $|i| \geq n/(2B)$
- $\widehat{G'}_i \in [0,1]$ *for all* i
- $\left\|\widehat{G'} - \widehat{G}\right\|_\infty < \delta$.

The above notion corresponds to the $(1/(2B), (1-\alpha)/(2B), \delta, O(B/\alpha\log(n/\delta))$-flat window function in [HIKP12b]. In Section 7 we give efficient constructions of such window functions, where G can be computed in $O(\frac{B}{\alpha}\log(n/\delta))$ time and for each i, $\widehat{G'}_i$ can be computed in $O(\log(n/\delta))$ time. Of course, for $i \notin [(1-\alpha)n/(2B), n/(2B)]$, $\widehat{G'}_i \in \{0,1\}$ can be computed in $O(1)$ time.

The fact that $\widehat{G'}_i$ takes $\omega(1)$ time to compute for $i \in [(1-\alpha)n/(2B), n/(2B)]$ will add some complexity to our algorithm and analysis. We will need to ensure that we rarely need to compute such values. A practical implementation might find it more convenient to precompute the window functions in a preprocessing stage, rather than compute them on the fly.

We use the following lemma from [HIKP12b]:

Lemma 2.4 (Lemma 3.6 of [HIKP12b])**.** *If $j \neq 0$, n is a power of two, and σ is a uniformly random odd number in $[n]$, then* $\Pr[\sigma j \in [-C, C] \pmod n] \leq 4C/n$.

Assumptions. Through the paper, we require that n, the dimension of all vectors, is an integer power of 2. We also make the following assumptions about the precision of the vectors $\widehat{x}$:

- For the exactly k-sparse case, we assume that $\widehat{x}_i \in \{-L, \ldots, L\}$ for some precision parameter L. To simplify the bounds, we assume that $L = n^{O(1)}$; otherwise the $\log n$ term in the running time bound is replaced by $\log L$.
- For the general case, we only achieve Equation (1) if $\|\widehat{x}\|_2 \leq n^{O(1)} \cdot \min_{k\text{-sparse } y} \|\widehat{x} - y\|_2$. In general, for any parameter $\delta > 0$ we can add $\delta\|\widehat{x}\|_2$ to the right hand side of Equation (1) and run in time $O(k\log(n/k)\log(n/\delta))$.

procedure HASHTOBINS(x, $\widehat{z}$, $P_{\sigma,a,b}$, B, δ, α)
  Compute $\widehat{y}_{jn/B}$ for $j \in [B]$, where $y = G_{B,\alpha,\delta} \cdot (P_{\sigma,a,b}x)$
  Compute $\widehat{y'}_{jn/B} = \widehat{y}_{jn/B} - (\widehat{G'_{B,\alpha,\delta}} * \widehat{P_{\sigma,a,b}z})_{jn/B}$ for $j \in [B]$
  return $\widehat{u}$ given by $\widehat{u}_j = \widehat{y'}_{jn/B}$.
end procedure
procedure NOISELESSSPARSEFFTINNER(x, k', $\widehat{z}$, α)
  Let $B = k'/\beta$, for sufficiently small constant β.
  Let $\delta = 1/(4n^2L)$.
  Choose σ uniformly at random from the set of odd numbers in $[n]$.
  Choose b uniformly at random from $[n]$.
  $\widehat{u} \leftarrow$ HASHTOBINS$(x, \widehat{z}, P_{\sigma,0,b}, B, \delta, \alpha)$.
  $\widehat{u}' \leftarrow$ HASHTOBINS$(x, \widehat{z}, P_{\sigma,1,b}, B, \delta, \alpha)$.
  $\widehat{w} \leftarrow 0$.
  Compute $J = \{j : |\widehat{u}_j| > 1/2\}$.
  for $j \in J$ **do**
    $a \leftarrow \widehat{u}_j/\widehat{u}'_j$.
    $i \leftarrow \sigma^{-1}(\text{round}(\phi(a)\frac{n}{2\pi})) \bmod n$. ▷ $\phi(a)$ denotes the phase of a.
    $v \leftarrow \text{round}(\widehat{u}_j)$.
    $\widehat{w}_i \leftarrow v$.
  end for
  return $\widehat{w}$
end procedure
procedure NOISELESSSPARSEFFT(x, k)
  $\widehat{z} \leftarrow 0$
  for $t \in 0, 1, \ldots, \log k$ **do**
    $k_t \leftarrow k/2^t$, $\alpha_t \leftarrow \Theta(2^{-t})$.
    $\widehat{z} \leftarrow \widehat{z} +$ NOISELESSSPARSEFFTINNER$(x, k_t, \widehat{z}, \alpha_t)$.
  end for
  return $\widehat{z}$
end procedure

Algorithm 3.1: Exact k-sparse recovery

3 Algorithm for the exactly sparse case

In this section we assume $\widehat{x}_i \in \{-L, \ldots, L\}$, where $L \leq n^c$ for some constant $c > 0$, and $\widehat{x}$ is k-sparse. We choose $\delta = 1/(4n^2L)$. The algorithm (NOISELESSSPARSEFFT) is described as Algorithm 3.1. The algorithm has three functions:

- HASHTOBINS. This permutes the spectrum of $\widehat{x - z}$ with $P_{\sigma,a,b}$, then "hashes" to B bins. The guarantee will be described in Lemma 3.3.
- NOISELESSSPARSEFFTINNER. Given time-domain access to x and a sparse vector $\widehat{z}$ such that $\widehat{x - z}$ is k'-sparse, this function finds "most" of $\widehat{x - z}$.
- NOISELESSSPARSEFFT. This iterates NOISELESSSPARSEFFTINNER until it finds $\widehat{x}$ exactly.

We analyze the algorithm "bottom-up", starting from the lower-level procedures.

Analysis of NOISELESSSPARSEFFTINNER and HASHTOBINS. For any execution of NOISELESSSPARSEFFTINNER, define the support $S = \text{supp}(\widehat{x} - \widehat{z})$. Recall that $\pi_{\sigma,b}(i) = \sigma(i - b) \bmod n$. Define $h_{\sigma,b}(i) = \text{round}(\pi_{\sigma,b}(i)B/n)$ and $o_{\sigma,b}(i) = \pi_{\sigma,b}(i) - h_{\sigma,b}(i)n/B$. Note that therefore $|o_{\sigma,b}(i)| \leq n/(2B)$.

We will refer to $h_{\sigma,b}(i)$ as the "bin" that the frequency i is mapped into, and $o_{\sigma,b}(i)$ as the "offset". For any $i \in S$ define two types of events associated with i and S and defined over the probability space induced by σ and b:

- "Collision" event $E_{coll}(i)$: holds iff $h_{\sigma,b}(i) \in h_{\sigma,b}(S \setminus \{i\})$, and
- "Large offset" event $E_{off}(i)$: holds iff $|o_{\sigma,b}(i)| \geq (1-\alpha)n/(2B)$.

Claim 3.1. *For any $i \in S$, the event $E_{coll}(i)$ holds with probability at most $4|S|/B$.*

Proof. Consider distinct $i, j \in S$. By Lemma 2.4,

$$\begin{aligned}\Pr[h_{\sigma,b}(i) = h_{\sigma,b}(j)] &\leq \Pr[\pi_{\sigma,b}(i) - \pi_{\sigma,b}(j) \bmod n \in [-n/B, n/B]] \\ &= \Pr[\sigma(i-j) \bmod n \in [-n/B, n/B]] \\ &\leq 4/B.\end{aligned}$$

By a union bound over $j \in S$, $\Pr[E_{coll}(i)] \leq 4\,|S|\,/B$. □

Claim 3.2. *For any $i \in S$, the event $E_{off}(i)$ holds with probability at most α.*

Proof. Note that $o_{\sigma,b}(i) \equiv \pi_{\sigma,b}(i) \equiv \sigma(i-b) \pmod{n/B}$. For any odd σ and any $l \in [n/B]$, we have that $\Pr_b[\sigma(i-b) \equiv l \pmod{n/B}] = B/n$. Since only $\alpha n/B$ offsets $o_{\sigma,b}(i)$ cause $E_{off}(i)$, the claim follows. □

Lemma 3.3. *Suppose B divides n. The output $\widehat{u}$ of* HASHTOBINS *satisfies*

$$\widehat{u}_j = \sum_{h_{\sigma,b}(i)=j} \widehat{(x-z)}_i (\widehat{G'_{B,\delta,\alpha}})_{-o_{\sigma,b}(i)} \omega^{a\sigma i} \pm \delta \left\|\widehat{x}\right\|_1.$$

Let $\zeta = |\{i \in \mathrm{supp}(\widehat{z}) \mid E_{off}(i)\}|$. The running time of HASHTOBINS *is $O(\frac{B}{\alpha}\log(n/\delta) + \|\widehat{z}\|_0 + \zeta\log(n/\delta))$.*

Proof. Define the flat window functions $G = G_{B,\delta,\alpha}$ and $\widehat{G'} = \widehat{G'}_{B,\delta,\alpha}$. We have

$$\begin{aligned}\widehat{y} &= \widehat{G \cdot P_{\sigma,a,b} x} = \widehat{G} * \widehat{P_{\sigma,a,b} x} \\ \widehat{y'} &= \widehat{G'} * \widehat{P_{\sigma,a,b}(x-z)} + (\widehat{G} - \widehat{G'}) * \widehat{P_{\sigma,a,b} x}\end{aligned}$$

By Claim 2.2, the coordinates of $\widehat{P_{\sigma,a,b}x}$ and $\widehat{x}$ have the same magnitudes, just different ordering and phase. Therefore

$$\left\|(\widehat{G} - \widehat{G'}) * \widehat{P_{\sigma,a,b}x}\right\|_\infty \leq \left\|\widehat{G} - \widehat{G'}\right\|_\infty \left\|\widehat{P_{\sigma,a,b}x}\right\|_1 \leq \delta \left\|\widehat{x}\right\|_1$$

and hence

$$\begin{aligned}\widehat{u}_j = \widehat{y'}_{jn/B} &= \sum_{|l|<n/(2B)} \widehat{G'}_{-l} (\widehat{P_{\sigma,a,b}(x-z)})_{jn/B+l} \pm \delta \left\|\widehat{x}\right\|_1 \\ &= \sum_{|\pi_{\sigma,b}(i) - jn/B| < n/(2B)} \widehat{G'}_{jn/B - \pi_{\sigma,b}(i)} (\widehat{P_{\sigma,a,b}(x-z)})_{\pi_{\sigma,b}(i)} \pm \delta \left\|\widehat{x}\right\|_1 \\ &= \sum_{h_{\sigma,b}(i)=j} \widehat{G'}_{-o_{\sigma,b}(i)} \widehat{(x-z)}_i \omega^{a\sigma i} \pm \delta \left\|\widehat{x}\right\|_1\end{aligned}$$

as desired.

We can compute HASHTOBINS via the following method:

1. Compute y with $\|y\|_0 = O(\frac{B}{\alpha}\log(n/\delta))$ in $O(\frac{B}{\alpha}\log(n/\delta))$ time.

2. Compute $v \in \mathbb{C}^B$ given by $v_i = \sum_j y_{i+jB}$.

3. Because B divides n, by the definition of the Fourier transform (see also Claim 3.7 of [HIKP12b]) we have $\widehat{y}_{jn/B} = \widehat{v}_j$ for all j. Hence we can compute it with a B-dimensional FFT in $O(B \log B)$ time.

4. For each coordinate $i \in \mathrm{supp}(\widehat{z})$, decrease $\widehat{y}_{\frac{n}{B} h_{\sigma,b}(i)}$ by $\widehat{G'}_{-o_{\sigma,b}(i)} \widehat{z}_i \omega^{a\sigma i}$. This takes $O(\|\widehat{z}\|_0 + \zeta \log(n/\delta))$ time, since computing $\widehat{G'}_{-o_{\sigma,b}(i)}$ takes $O(\log(n/\delta))$ time if $E_{off}(i)$ holds and $O(1)$ otherwise. □

Lemma 3.4. *Consider any $i \in S$ such that neither $E_{coll}(i)$ nor $E_{off}(i)$ holds. Let $j = h_{\sigma,b}(i)$. Then*

$$\textit{round}(\phi(\widehat{u}_j / \widehat{u}'_j)) \frac{n}{2\pi}) = \sigma i \pmod n,$$

$$\textit{round}(\widehat{u}_j) = \widehat{x}_i - \widehat{z}_i,$$

and $j \in J$.

Proof. We know that $\|\widehat{x}\|_1 \le k \|\widehat{x}\|_\infty \le kL < nL$. Then by Lemma 3.3 and $E_{coll}(i)$ not holding,

$$\widehat{u}_j = \widehat{(x-z)}_i \widehat{G'}_{-o_{\sigma,b}(i)} \pm \delta nL.$$

Because $E_{off}(i)$ does not hold, $\widehat{G'}_{-o_{\sigma,b}(i)} = 1$, so

$$\widehat{u}_j = \widehat{(x-z)}_i \pm \delta nL. \tag{2}$$

Similarly,

$$\widehat{u}'_j = \widehat{(x-z)}_i \omega^{\sigma i} \pm \delta nL$$

Then because $\delta nL < 1 \le \left|\widehat{(x-z)}_i\right|$, the phase is

$$\phi(\widehat{u}_j) = 0 \pm \sin^{-1}(\delta nL) = 0 \pm 2\delta nL$$

and $\phi(\widehat{u}'_j) = -\sigma i \frac{2\pi}{n} \pm 2\delta nL$. Thus $\phi(\widehat{u}_j / \widehat{u}'_j) = \sigma i \frac{2\pi}{n} \pm 4\delta nL = \sigma i \frac{2\pi}{n} \pm 1/n$ by the choice of δ. Therefore

$$\mathrm{round}(\phi(\widehat{u}_j / \widehat{u}'_j) \frac{n}{2\pi}) = \sigma i \pmod n.$$

Also, by Equation (2), $\mathrm{round}(\widehat{u}_j) = \widehat{x}_i - \widehat{z}_i$. Finally, $|\mathrm{round}(\widehat{u}_j)| = |\widehat{x}_i - \widehat{z}_i| \ge 1$, so $|\widehat{u}_j| \ge 1/2$. Thus $j \in J$. □

For each invocation of NOISELESSSPARSEFFTINNER, let P be the the set of all pairs (i, v) for which the command $\widehat{w}_i \leftarrow v$ was executed. Claims 3.1 and 3.2 and Lemma 3.4 together guarantee that for each $i \in S$ the probability that P does not contain the pair $(i, (\widehat{x} - \widehat{z})_i)$ is at most $4|S|/B + \alpha$. We complement this observation with the following claim.

Claim 3.5. *For any $j \in J$ we have $j \in h_{\sigma,b}(S)$. Therefore, $|J| = |P| \le |S|$.*

Proof. Consider any $j \notin h_{\sigma,b}(S)$. From Equation (2) in the proof of Lemma 3.4 it follows that $|\widehat{u}_j| \le \delta nL < 1/2$. □

Lemma 3.6. *Consider an execution of* NOISELESSSPARSEFFTINNER*, and let $S = \mathrm{supp}(\widehat{x} - \widehat{z})$. If $|S| \le k'$, then*

$$E[\|\widehat{x} - \widehat{z} - \widehat{w}\|_0] \le 8(\beta + \alpha)|S|.$$

Proof. Let e denote the number of coordinates $i \in S$ for which either $E_{coll}(i)$ or $E_{off}(i)$ holds. Each such coordinate might not appear in P with the correct value, leading to an incorrect value of $\widehat{w}_i$. In fact, it might result in an arbitrary pair (i', v') being added to P, which in turn could lead to an incorrect value of $\widehat{w}_{i'}$. By Claim 3.5 these are the only ways that $\widehat{w}$ can be assigned an incorrect value. Thus we have

$$\|\widehat{x} - \widehat{z} - \widehat{w}\|_0 \le 2e.$$

Since $E[e] \le (4|S|/B + \alpha)|S| \le (4\beta + \alpha)|S|$, the lemma follows. □

Analysis of NOISELESSSPARSEFFT. Consider the tth iteration of the procedure, and define $S_t = \mathrm{supp}(\widehat{x} - \widehat{z})$ where $\widehat{z}$ denotes the value of the variable at the beginning of loop. Note that $|S_0| = |\mathrm{supp}(\widehat{x})| \leq k$.

We also define an indicator variable I_t which is equal to 0 iff $|S_t|/|S_{t-1}| \leq 1/8$. If $I_t = 1$ we say the the tth iteration was not *successful*. Let $\gamma = 8 \cdot 8(\beta + \alpha)$. From Lemma 3.6 it follows that $\Pr[I_t = 1 \mid |S_{t-1}| \leq k/2^{t-1}] \leq \gamma$. From Claim 3.5 it follows that even if the tth iteration is not successful, then $|S_t|/|S_{t-1}| \leq 2$.

For any $t \geq 1$, define an event $E(t)$ that occurs iff $\sum_{i=1}^{t} I_i \geq t/2$. Observe that if none of the events $E(1) \ldots E(t)$ holds then $|S_t| \leq k/2^t$.

Lemma 3.7. *Let $E = E(1) \cup \ldots \cup E(\lambda)$ for $\lambda = 1 + \log k$. Assume that $(4\gamma)^{1/2} < 1/4$. Then $\Pr[E] \leq 1/3$.*

Proof. Let $t' = \lceil t/2 \rceil$. We have

$$\Pr[E(t)] \leq \binom{t}{t'} \gamma^{t'} \leq 2^t \gamma^{t'} \leq (4\gamma)^{t/2}$$

Therefore

$$\Pr[E] \leq \sum_t \Pr[E(t)] \leq \frac{(4\gamma)^{1/2}}{1 - (4\gamma)^{1/2}} \leq 1/4 \cdot 4/3 = 1/3.$$

□

Theorem 3.8. *Suppose $\widehat{x}$ is k-sparse with entries from $\{-L, \ldots, L\}$ for some known $L = n^{O(1)}$. Then the algorithm* NOISELESSSPARSEFFT *runs in expected $O(k \log n)$ time and returns the correct vector $\widehat{x}$ with probability at least $2/3$.*

Proof. The correctness follows from Lemma 3.7. The running time is dominated by $O(\log k)$ executions of HASHTOBINS.

Assuming a correct run, in every round t we have

$$\|\widehat{z}\|_0 \leq \|\widehat{x}\|_0 + |S_t| \leq k + k/2^t \leq 2k.$$

Therefore

$$\mathbb{E}[|\{i \in \mathrm{supp}(z) \mid E_{off}(i)\}|] \leq \alpha \|\widehat{z}\|_0 \leq 2\alpha k,$$

so the expected running time of each execution of HASHTOBINS is $O(\frac{B}{\alpha} \log(n/\delta) + k + \alpha k \log(n/\delta)) = O(\frac{B}{\alpha} \log n + k + \alpha k \log n)$. Setting $\alpha = \Theta(2^{-t/2})$ and $\beta = \Theta(1)$, the expected running time in round t is $O(2^{-t/2} k \log n + k + 2^{-t/2} k \log n)$. Therefore the total expected running time is $O(k \log n)$. □

4 Algorithm for the general case

This section shows how to achieve Equation (1) for $C = 1 + \epsilon$. Pseudocode is in Algorithm 4.1 and 4.2.

4.1 Intuition

Let S denote the "heavy" $O(k/\epsilon)$ coordinates of $\widehat{x}$. The overarching algorithm SPARSEFFT works by first "locating" a set L containing most of S, then "estimating" $\widehat{x}_L$ to get $\widehat{z}$. It then repeats on $\widehat{x - z}$. We will show that each heavy coordinate has a large constant probability of both being in L and being estimated well. As a result, $\widehat{x - z}$ is probably nearly $k/4$-sparse, so we can run the next iteration with $k \to k/4$. The later iterations then run faster and achieve a higher success probability, so the total running time is dominated by the time in the first iteration and the total error probability is bounded by a constant.

In the rest of this intuition, we will discuss the first iteration of SPARSEFFT with simplified constants. In this iteration, hashes are to $B = O(k/\epsilon)$ bins and, with $3/4$ probability, we get $\widehat{z}$ so $\widehat{x - z}$ is nearly $k/4$-sparse. The actual algorithm will involve a parameter α in each iteration, roughly guaranteeing that with $1 - \sqrt{\alpha}$ probability, we get $\widehat{z}$ so $\widehat{x - z}$ is nearly $\sqrt{\alpha}k$-sparse; the formal guarantee will be given by Lemma 4.8. For this intuition we only consider the first iteration where α is a constant.

Location. As in the noiseless case, to locate the "heavy" coordinates we consider the "bins" computed by HASHTOBINS with $P_{\sigma,a,b}$. This roughly corresponds to first permuting the coordinates according to the (almost) pairwise independent permutation $P_{\sigma,a,b}$, partitioning the coordinates into $B = O(k/\epsilon)$ "bins" of n/B consecutive indices, and observing the sum of values in each bin. We get that each heavy coordinate i has a large constant probability that the following two events occur: no other heavy coordinate lies in the same bin, and only a small (i.e., $O(\epsilon/k)$) fraction of the mass from non-heavy coordinates lies in the same bin. For such a "well-hashed" coordinate i, we would like to find its location $\tau = \pi_{\sigma,b}(i) = \sigma(i-b)$ among the $\epsilon n/k < n/k$ consecutive values that hash to the same bin. Let

$$\theta^*_j = \frac{2\pi}{n}(j + \sigma b) \pmod{2\pi}. \tag{3}$$

so $\theta^*_\tau = \frac{2\pi}{n}\sigma i$. In the noiseless case, we showed that the difference in phase in the bin using $P_{\sigma,0,b}$ and using $P_{\sigma,1,b}$ is θ^*_τ plus a negligible $O(\delta)$ term. With noise this may not be true; however, we can say for any $\beta \in [n]$ that the difference in phase between using $P_{\sigma,a,b}$ and $P_{\sigma,a+\beta,b}$, as a distribution over uniformly random $a \in [n]$, is $\beta\theta^*_\tau + \nu$ with (for example) $\mathbb{E}[\nu^2] = 1/100$ (all operations on phases modulo 2π). We can only hope to get a constant number of bits from such a "measurement". So our task is to find τ within a region Q of size n/k using $O(\log(n/k))$ "measurements" of this form.

One method for doing so would be to simply do measurements with random $\beta \in [n]$. Then each measurement lies within $\pi/4$ of $\beta\theta^*_\tau$ with at least $1 - \frac{\mathbb{E}[\nu^2]}{\pi^2/16} > 3/4$ probability. On the other hand, for $j \neq \tau$ and as a distribution over β, $\beta(\theta^*_\tau - \theta^*_j)$ is roughly uniformly distributed around the circle. As a result, each measurement is probably more than $\pi/4$ away from $\beta\theta^*_j$. Hence $O(\log(n/k))$ repetitions suffice to distinguish among the n/k possibilities for τ. However, while the number of measurements is small, it is not clear how to decode in polylog rather than $\Omega(n/k)$ time.

To solve this, we instead do a t-ary search on the location for $t = \Theta(\log n)$. At each of $O(\log_t(n/k))$ levels, we split our current candidate region Q into t consecutive subregions $Q_1, \ldots, Q_t$, each of size w. Now, rather than choosing $\beta \in [n]$, we choose $\beta \in [\frac{n}{16w}, \frac{n}{8w}]$. By the upper bound on β, for each $q \in [t]$ the values $\{\beta\theta^*_j \mid j \in Q_q\}$ all lie within $\beta w \frac{2\pi}{n} \leq \pi/4$ of each other on the circle. On the other hand, if $|j - \tau| > 16w$, then $\beta(\theta^*_\tau - \theta^*_j)$ will still be roughly uniformly distributed about the circle. As a result, we can check a single candidate element e_q from each subregion: if e_q is in the same subregion as τ, each measurement usually agrees in phase; but if e_q is more than 16 subregions away, each measurement usually disagrees in phase. Hence with $O(\log t)$ measurements, we can locate τ to within $O(1)$ consecutive subregions with failure probability $1/t^{\Theta(1)}$. The decoding time is $O(t \log t)$.

This primitive LOCATEINNER lets us narrow down the candidate region for τ to a subregion that is a $t' = \Omega(t)$ factor smaller. By repeating LOCATEINNER $\log_{t'}(n/k)$ times, LOCATESIGNAL can find τ precisely. The number of measurements is then $O(\log t \log_t(n/k)) = O(\log(n/k))$ and the decoding time is $O(t \log t \log_t(n/k)) = O(\log(n/k) \log n)$. Furthermore, the "measurements" (which are actually calls to HASHTOBINS) are non-adaptive, so we can perform them in parallel for all $O(k/\epsilon)$ bins, with $O(\log(n/\delta))$ average time per measurement. This gives $O(k \log(n/k) \log(n/\delta))$ total time for LOCATESIGNAL.

This lets us locate every heavy and "well-hashed" coordinate with $1/t^{\Theta(1)} = o(1)$ failure probability, so every heavy coordinate is located with arbitrarily high constant probability.

Estimation. By contrast, estimation is fairly simple. As in Algorithm 3.1, we can estimate $\widehat{(x-z)}_i$ as $\widehat{u}_{h_{\sigma,b}(i)}$, where $\widehat{u}$ is the output of HASHTOBINS. Unlike in Algorithm 3.1, we now have noise that may cause a single such estimate to be poor even if i is "well-hashed". However, we can show that for a random permutation $P_{\sigma,a,b}$ the estimate is "good" with constant probability. ESTIMATEVALUES takes the median of $R_{est} = O(\log \frac{1}{\epsilon})$ such samples, getting a good estimate with $1 - \epsilon/64$ probability. Given a candidate set L of size k/ϵ, with $7/8$ probability at most $k/8$ of the coordinates are badly estimated. On the other hand, with $7/8$ probability, at least $7k/8$ of the heavy coordinates are both located and well estimated. This suffices to show that, with $3/4$ probability, the largest k elements J of our estimate $\widehat{w}$ contains good estimates of $3k/4$ large coordinates, so $\widehat{x - z - w_J}$ is close to $k/4$-sparse.

procedure SPARSEFFT(x, k, ϵ, δ)
  $R \leftarrow O(\log k / \log\log k)$ as in Theorem 4.9.
  $\widehat{z}^{(1)} \leftarrow 0$
  for $r \in [R]$ **do**
    Choose B_r, k_r, α_r as in Theorem 4.9.
    $R_{est} \leftarrow O(\log(\frac{B_r}{\alpha_r k_r}))$ as in Lemma 4.8.
    $L_r \leftarrow$ LOCATESIGNAL$(x, \widehat{z}^{(r)}, B_r, \alpha_r, \delta)$
    $\widehat{z}^{(r+1)} \leftarrow \widehat{z}^{(r)} +$ ESTIMATEVALUES$(x, \widehat{z}^{(r)}, 3k_r, L_r, B_r, \delta, R_{est})$.
  end for
  return $\widehat{z}^{(R+1)}$
end procedure
procedure ESTIMATEVALUES(x, $\widehat{z}$, k', L, B, δ, R_{est})
  for $r \in [R_{est}]$ **do**
    Choose $a_r, b_r \in [n]$ uniformly at random.
    Choose σ_r uniformly at random from the set of odd numbers in $[n]$.
    $\widehat{u}^{(r)} \leftarrow$ HASHTOBINS$(x, \widehat{z}, P_{\sigma,a_r,b}, B, \delta)$.
  end for
  $\widehat{w} \leftarrow 0$
  for $i \in L$ **do**
    $\widehat{w}_i \leftarrow \mathrm{median}_r\, \widehat{u}^{(r)}_{h_{\sigma,b}(i)} \omega^{-a_r \sigma i}$. ▷ Separate median in real and imaginary axes.
  end for
  $J \leftarrow \arg\max_{|J|=k'} \|\widehat{w}_J\|_2$.
  return $\widehat{w}_J$
end procedure

Algorithm 4.1: k-sparse recovery for general signals, part 1/2.

procedure LOCATESIGNAL(x, $\widehat{z}$, B, α, δ)
  Choose uniformly at random $\sigma, b \in [n]$ with σ odd.
  Initialize $l_i^{(1)} = (i-1)n/B$ for $i \in [B]$.
  Let $w_0 = n/B, t = \log n, t' = t/4, D_{max} = \log_{t'}(w_0 + 1)$.
  Let $R_{loc} = \Theta(\log_{1/\alpha}(t/\alpha))$ per Lemma 4.5.
  for $D \in [D_{max}]$ **do**
    $l^{(D+1)} \leftarrow$ LOCATEINNER$(x, \widehat{z}, B, \delta, \alpha, \sigma, \beta, l^{(D)}, w_0/(t')^{D-1}, t, R_{loc})$
  end for
  $L \leftarrow \{\pi_{\sigma,b}^{-1}(l_j^{(D_{max}+1)}) \mid j \in [B]\}$
  return L
end procedure

▷ δ, α parameters for G, G'
▷ $(l_1, l_1 + w), \ldots, (l_B, l_B + w)$ the plausible regions.
▷ $B \approx k/\epsilon$ the number of bins
▷ $t \approx \log n$ the number of regions to split into.
▷ $R_{loc} \approx \log t = \log\log n$ the number of rounds to run

procedure LOCATEINNER(x, $\widehat{z}$, B, δ, α, σ, b, l, w, t, R_{loc})
  Let $s = \Theta(\alpha^{1/3})$.
  Let $v_{j,q} = 0$ for $(j, q) \in [B] \times [t]$.
  for $r \in [R_{loc}]$ **do**
    Choose $a \in [n]$ uniformly at random.
    Choose $\beta \in \{\frac{snt}{4w}, \ldots, \frac{snt}{2w}\}$ uniformly at random.
    $\widehat{u} \leftarrow$ HASHTOBINS$(x, \widehat{z}, P_{\sigma,a,b}, B, \delta, \alpha)$.
    $\widehat{u}' \leftarrow$ HASHTOBINS$(x, \widehat{z}, P_{\sigma,a+\beta,b}, B, \delta, \alpha)$.
    for $j \in [B]$ **do**
      $c_j \leftarrow \phi(\widehat{u}_j / \widehat{u}'_j)$
      for $q \in [t]$ **do**
        $m_{j,q} \leftarrow l_j + \frac{q-1/2}{t} w$
        $\theta_{j,q} \leftarrow \frac{2\pi(m_{j,q}+\sigma b)}{n} \bmod 2\pi$
        if $\min(|\beta\theta_{j,q} - c_j|, 2\pi - |\beta\theta_{j,q} - c_j|) < s\pi$ **then**
          $v_{j,q} \leftarrow v_{j,q} + 1$
        end if
      end for
    end for
  end for
  for $j \in [B]$ **do**
    $Q^* \leftarrow \{q \in [t] \mid v_{j,q} > R_{loc}/2\}$
    if $Q^* \neq \emptyset$ **then**
      $l'_j \leftarrow \min_{q \in Q^*} l_j + \frac{q-1}{t} w$
    else
      $l'_j \leftarrow \perp$
    end if
  end for
  return l'
end procedure

Algorithm 4.2: k-sparse recovery for general signals, part 2/2.

4.2 Formal definitions

As in the noiseless case, we define $\pi_{\sigma,b}(i) = \sigma(i-b) \bmod n$, $h_{\sigma,b}(i) = \text{round}(\pi_{\sigma,b}(i)B/n)$ and $o_{\sigma,b}(i) = \pi_{\sigma,b}(i) - h_{\sigma,b}(i)n/B$. We say $h_{\sigma,b}(i)$ is the "bin" that frequency i is mapped into, and $o_{\sigma,b}(i)$ is the "offset". We define $h^{-1}_{\sigma,b}(j) = \{i \in [n] \mid h_{\sigma,b}(i) = j\}$.

Define

$$\text{Err}(x,k) = \min_{k\text{-sparse } y} \|x-y\|_2 .$$

In each iteration of SPARSEFFT, define $\widehat{x}' = \widehat{x} - \widehat{z}$, and let

$$\begin{aligned}\rho^2 &= \text{Err}^2(\widehat{x}',k) + \delta^2 n \|\widehat{x}\|_1^2 \\ \mu^2 &= \epsilon\rho^2/k \\ S &= \{i \in [n] \mid |\widehat{x}'_i|^2 \geq \mu^2\}\end{aligned}$$

Then $|S| \leq (1+1/\epsilon)k = O(k/\epsilon)$ and $\left\|\widehat{x}' - \widehat{x}'_S\right\|_2^2 \leq (1+\epsilon)\rho^2$. We will show that each $i \in S$ is found by LOCATESIGNAL with probability $1 - O(\alpha)$, when $B = \Omega(\frac{k}{\alpha\epsilon})$.

For any $i \in S$ define three types of events associated with i and S and defined over the probability space induced by σ and b:

- "Collision" event $E_{coll}(i)$: holds iff $h_{\sigma,b}(i) \in h_{\sigma,b}(S \setminus \{i\})$;
- "Large offset" event $E_{off}(i)$: holds iff $|o_{\sigma,b}(i)| \geq (1-\alpha)n/(2B)$; and
- "Large noise" event $E_{noise}(i)$: holds iff $\left\|\widehat{x}'_{h^{-1}_{\sigma,b}(h_{\sigma,b}(i))\setminus S}\right\|_2^2 \geq \text{Err}^2(\widehat{x}',k)/(\alpha B)$.

By Claims 3.1 and 3.2, $\Pr[E_{coll}(i)] \leq 4|S|/B = O(\alpha)$ and $\Pr[E_{off}(i)] \leq 2\alpha$ for any $i \in S$.

Claim 4.1. *For any $i \in S$, $\Pr[E_{noise}(i)] \leq 4\alpha$.*

Proof. For each $j \neq i$, $\Pr[h_{\sigma,b}(j) = h_{\sigma,b}(i)] \leq \Pr[|\sigma j - \sigma i| < n/B] \leq 4/B$ by Lemma 2.4. Then

$$\mathbb{E}[\left\|\widehat{x}'_{h^{-1}_{\sigma,b}(h_{\sigma,b}(i))\setminus S}\right\|_2^2] \leq 4\left\|\widehat{x}'_{[n]\setminus S}\right\|_2^2/B$$

The result follows by Markov's inequality. □

We will show for $i \in S$ that if none of $E_{coll}(i), E_{off}(i)$, and $E_{noise}(i)$ hold then SPARSEFFTINNER recovers $\widehat{x}'_i$ with $1 - O(\alpha)$ probability.

Lemma 4.2. *Let $a \in [n]$ uniformly at random, B divide n, and the other parameters be arbitrary in*

$$\widehat{u} = \text{HASHTOBINS}(x, \widehat{z}, P_{\sigma,a,b}, B, \delta, \alpha).$$

Then for any $i \in [n]$ with $j = h_{\sigma,b}(i)$ and none of $E_{coll}(i)$, $E_{off}(i)$, or $E_{noise}(i)$ holding,

$$\mathbb{E}[\left|\widehat{u}_j - \widehat{x}'_i\omega^{a\sigma i}\right|^2] \leq 2\frac{\rho^2}{\alpha B}$$

Proof. Let $\widehat{G'} = \widehat{G'}_{B,\delta,\alpha}$. Let $T = h^{-1}_{\sigma,b}(j) \setminus \{i\}$. We have that $T \cap S = \{\}$ and $\widehat{G'}_{-o_{\sigma,b}(i)} = 1$. By Lemma 3.3,

$$\widehat{u}_j - \widehat{x}'_i\omega^{a\sigma i} = \sum_{i' \in T} \widehat{G'}_{-o_\sigma(i')}\widehat{x}'_{i'}\omega^{a\sigma i'} \pm \delta\|\widehat{x}\|_1 .$$

Because the $\sigma i'$ are distinct for $i' \in T$, we have by Parseval's theorem

$$\mathbb{E}_a \left| \sum_{i' \in T} \widehat{G'}_{-o_\sigma(i')} \widehat{x'}_{i'} \omega^{a\sigma i'} \right|^2 = \sum_{i' \in T} (\widehat{G'}_{-o_\sigma(i')} \widehat{x'}_{i'})^2 \leq \left\| \widehat{x'_T} \right\|_2^2$$

Since $|X+Y|^2 \leq 2|X|^2 + 2|Y|^2$ for any X, Y, we get

$$\begin{aligned} \mathbb{E}_a[\left| \widehat{u}_j - \widehat{x'}_i \omega^{a\sigma i} \right|^2] &\leq 2 \|x'_T\|_2^2 + 2\delta^2 \|\widehat{x}\|_1^2 \\ &\leq 2 \operatorname{Err}^2(\widehat{x'}, k)/(\alpha B) + 2\delta^2 \|\widehat{x}\|_1^2 \\ &\leq 2\rho^2/(\alpha B). \end{aligned}$$

□

4.3 Properties of LOCATESIGNAL

In our intuition, we made a claim that if $\beta \in [n/(16w), n/(8w)]$ uniformly at random, and $i > 16w$, then $\frac{2\pi}{n}\beta i$ is "roughly uniformly distributed about the circle" and hence not concentrated in any small region. This is clear if β is chosen as a random real number; it is less clear in our setting where β is a random integer in this range. We now prove a lemma that formalizes this claim.

Lemma 4.3. *Let $T \subset [m]$ consist of t consecutive integers, and suppose $\beta \in T$ uniformly at random. Then for any $i \in [n]$ and set $S \subset [n]$ of l consecutive integers,*

$$\Pr[\beta i \bmod n \in S] \leq \lceil im/n \rceil (1 + \lfloor l/i \rfloor)/t \leq \frac{1}{t} + \frac{im}{nt} + \frac{lm}{nt} + \frac{l}{it}$$

Proof. Note that any interval of length l can cover at most $1 + \lfloor l/i \rfloor$ elements of any arithmetic sequence of common difference i. Then $\{\beta i \mid \beta \in T\} \subset [im]$ is such a sequence, and there are at most $\lceil im/n \rceil$ intervals $an + S$ overlapping this sequence. Hence at most $\lceil im/n \rceil (1 + \lfloor l/i \rfloor)$ of the $\beta \in [m]$ have $\beta i \bmod n \in S$. Hence

$$\Pr[\beta i \bmod n \in S] \leq \lceil im/n \rceil (1 + \lfloor l/i \rfloor)/t.$$

□

Lemma 4.4. *Let $i \in S$. Suppose none of $E_{coll}(i), E_{off}(i)$, and $E_{noise}(i)$ hold, and let $j = h_{\sigma,b}(i)$. Consider any run of* LOCATEINNER *with $\pi_{\sigma,b}(i) \in [l_j, l_j + w]$. Let $f > 0$ be a parameter such that*

$$B = \frac{Ck}{\alpha f \epsilon}.$$

for C larger than some fixed constant. Then $\pi_{\sigma,b}(i) \in [l'_j, l'_j + 4w/t]$ with probability at least $1 - tf^{\Omega(R_{loc})}$,

Proof. Let $\tau = \pi_{\sigma,b}(i) \equiv \sigma(i-b) \pmod n$, and for any $j \in [n]$ define

$$\theta^*_j = \frac{2\pi}{n}(j + \sigma b) \pmod{2\pi}$$

so $\theta^*_\tau = \frac{2\pi}{n}\sigma i$. Let $g = \Theta(f^{1/3})$, and $C' = \frac{B\alpha\epsilon}{k} = \Theta(1/g^3)$.

To get the result, we divide $[l_j, l_j + w]$ into t "regions", $Q_q = [l_j + \frac{q-1}{t}w, l_j + \frac{q}{t}w]$ for $q \in [t]$. We will first show that in each round r, c_j is close to $\beta\theta^*_\tau$ with $1 - g$ probability. This will imply that Q_q gets a "vote," meaning $v_{j,q}$ increases, with $1 - g$ probability for the q' with $\tau \in Q_{q'}$. It will also imply that $v_{j,q}$ increases with only g probability when $|q - q'| > 3$. Then R_{loc} rounds will suffice to separate the two with $1 - f^{-\Omega(R_{loc})}$ probability. We get that with $1 - tf^{-\Omega(R_{loc})}$ probability, the recovered Q^* has $|q - q'| \leq 3$

for all $q \in Q^*$. If we take the minimum $q \in Q^*$ and the next three subregions, we find τ to within 4 regions, or $4w/t$ locations, as desired.

In any round r, define $\widehat{u} = \widehat{u}^{(r)}$ and $a = a_r$. We have by Lemma 4.2 and that $i \in S$ that

$$\begin{aligned}\mathbb{E}[\left|\widehat{u}_j - \omega^{a\sigma i}\widehat{x'}_i\right|^2] &\leq 2\frac{\rho^2}{\alpha B} = \frac{2k}{B\alpha\epsilon}\mu^2 \\ &= \frac{2}{C'}\mu^2 \leq \frac{2}{C'}|\widehat{x'}_i|^2.\end{aligned}$$

Note that $\phi(\omega^{a\sigma i}) = -a\theta^*_\tau$. Thus for any $p > 0$, with probability $1 - p$ we have

$$\left|\widehat{u}_j - \omega^{a\sigma i}\widehat{x'}_i\right| \leq \sqrt{\frac{2}{C'p}}\left|\widehat{x'}_i\right|$$
$$\left\|\phi(\widehat{u}_j) - (\phi(\widehat{x'}_i) - a\theta^*_\tau)\right\|_\bigcirc \leq \sin^{-1}(\sqrt{\frac{2}{C'p}})$$

where $\|x - y\|_\bigcirc = \min_{\gamma\in\mathbb{Z}}|x - y + 2\pi\gamma|$ denotes the "circular distance" between x and y. The analogous fact holds for $\phi(\widehat{u'}_j)$ relative to $\phi(\widehat{x'}_i) - (a + \beta)\theta^*_\tau$. Therefore with at least $1 - 2p$ probability,

$$\begin{aligned}\|c_j - \beta\theta^*_\tau\|_\bigcirc &= \left\|\phi(\widehat{u}_j) - \phi(\widehat{u'}_j) - \beta\theta^*_\tau\right\|_\bigcirc \\ &= \left\|\left(\phi(\widehat{u}_j) - (\phi(\widehat{x'}_i) - a\theta^*_\tau)\right) - \left(\phi(\widehat{u'}_j) - (\phi(\widehat{x'}_i) - (a+\beta)\theta^*_\tau)\right)\right\|_\bigcirc \\ &\leq \left\|\phi(\widehat{u}_j) - (\phi(\widehat{x'}_i) - a\theta^*_\tau)\right\|_\bigcirc + \left\|\phi(\widehat{u'}_j) - (\phi(\widehat{x'}_i) - (a+\beta)\theta^*_\tau)\right\|_\bigcirc \\ &\leq 2\sin^{-1}(\sqrt{\frac{2}{C'p}})\end{aligned}$$

by the triangle inequality. Thus for any $s = \Theta(g)$ and $p = \Theta(g)$, we can set $C' = \frac{2}{p\sin^2(s\pi/4)} = \Theta(1/g^3)$ so that

$$\|c_j - \beta\theta^*_\tau\|_\bigcirc < s\pi/2 \tag{4}$$

with probability at least $1 - 2p$.

Equation (4) shows that c_j is a good estimate for i with good probability. We will now show that this means the approprate "region" $Q_{q'}$ gets a "vote" with "large" probability.

For the q' with $\tau \in [l_j + \frac{q'-1}{t}w, l_j + \frac{q'}{t}w]$, we have that $m_{j,q'} = l_j + \frac{q'-1/2}{t}w$ satisfies

$$|\tau - m_{j,q'}| \leq \frac{w}{2t}$$

so

$$|\theta^*_\tau - \theta_{j,q'}| \leq \frac{2\pi}{n}\frac{w}{2t}.$$

Hence by Equation (4), the triangle inequality, and the choice of $B \leq \frac{snt}{2w}$,

$$\begin{aligned}\|c_j - \beta\theta_{j,q'}\|_\bigcirc &\leq \|c_j - \beta\theta^*_\tau\|_\bigcirc + \|\beta\theta^*_\tau - \beta\theta_{j,q'}\|_\bigcirc \\ &< \frac{s\pi}{2} + \frac{\beta\pi w}{nt} \\ &\leq \frac{s\pi}{2} + \frac{s\pi}{2} \\ &= s\pi.\end{aligned}$$

Thus, $v_{j,q'}$ will increase in each round with probability at least $1 - 2p$.

Now, consider q with $|q - q'| > 3$. Then $|\tau - m_{j,q}| \geq \frac{7w}{2t}$, and (from the definition of $\beta > \frac{snt}{4w}$) we have

$$\beta \left|\tau - m_{j,q}\right| \geq \frac{7sn}{8} > \frac{3sn}{4}. \tag{5}$$

We now consider two cases. First, suppose that $|\tau - m_{j,q}| \leq \frac{w}{st}$. In this case, from the definition of β it follows that

$$\beta \left|\tau - m_{j,q}\right| \leq n/2.$$

Together with Equation (5) this implies

$$\Pr[\beta(\tau - m_{j,q}) \bmod n \in [-3sn/4, 3sn/4]] = 0.$$

On the other hand, suppose that $|\tau - m_{j,q}| > \frac{w}{st}$. In this case, we use Lemma 4.3 with parameters $l = 3sn/2$, $m = \frac{snt}{2w}$, $t = \frac{snt}{4w}$, $i = (\tau - m_{j,q})$ and $n = n$, to conclude that

$$\begin{aligned}
\Pr[\beta(\tau - m_{j,q}) \bmod n \in [-3sn/4, 3sn/4]] &\leq \frac{4w}{snt} + 2\frac{|\tau - m_{j,q}|}{n} + 3s + \frac{3sn}{2}\frac{st}{w}\frac{4w}{snt} \\
&\leq \frac{4w}{snt} + \frac{2w}{n} + 9s \\
&< \frac{6}{sB} + 9s \\
&< 10s
\end{aligned}$$

where we used that $|i| \leq w \leq n/B$, the assumption $\frac{w}{st} < |i|$, $t \geq 1$, $s < 1$, and that $s^2 > 6/B$ (because $s = \Theta(g)$ and $B = \omega(1/g^3)$).

Thus in either case, with probability at least $1 - 10s$ we have

$$\left\|\beta\theta_{j,q} - \beta\theta^*_\tau\right\|_{\bigcirc} = \left\|\frac{2\pi\beta(m_{j,q} - \tau)}{n}\right\|_{\bigcirc} > \frac{2\pi}{n}\frac{3sn}{4} = \frac{3}{2}s\pi$$

for any q with $|q - q'| > 3$. Therefore we have

$$\left\|c_j - \beta\theta_{j,q}\right\|_{\bigcirc} \geq \left\|\beta\theta_{j,q} - \beta\theta^*_\tau\right\|_{\bigcirc} - \left\|c_j - \beta\theta^*_\tau\right\|_{\bigcirc} > s\pi$$

with probability at least $1 - 10s - 2p$, and $v_{j,q}$ is not incremented.

To summarize: in each round, $v_{j,q'}$ is incremented with probability at least $1 - 2p$ and $v_{j,q}$ is incremented with probability at most $10s + 2p$ for $|q - q'| > 3$. The probabilities corresponding to different rounds are independent.

Set $s = g/20$ and $p = g/4$. Then $v_{j,q'}$ is incremented with probability at least $1 - g$ and $v_{j,q}$ is incremented with probability less than g. Then after R_{loc} rounds, if $|q - q'| > 3$,

$$\Pr[v_{j,q} > R_{loc}/2] \leq \binom{R_{loc}}{R_{loc}/2} g^{R_{loc}/2} \leq (4g)^{R_{loc}/2} = f^{\Omega(R_{loc})}$$

for $g = f^{1/3}/4$. Similarly,

$$\Pr[v_{j,q'} < R_{loc}/2] \leq f^{\Omega(R_{loc})}.$$

Hence with probability at least $1 - tf^{\Omega(R_{loc})}$ we have $q' \in Q^*$ and $|q - q'| \leq 3$ for all $q \in Q^*$. But then $\tau - l'_j \in [0, 4w/t]$ as desired.

Because $\mathbb{E}[|\{i \in \operatorname{supp}(\widehat{z}) \mid E_{off}(i)\}|] = \alpha \left\|\widehat{z}\right\|_0$, the expected running time is $O(R_{loc}Bt + R_{loc}\frac{B}{\alpha}\log(n/\delta) + R_{loc}\left\|\widehat{z}\right\|_0(1 + \alpha\log(n/\delta)))$. □

Lemma 4.5. *Suppose $B = \frac{Ck}{\alpha^2\epsilon}$ for C larger than some fixed constant. The procedure* LOCATESIGNAL *returns a set L of size $|L| \leq B$ such that for any $i \in S$, $\Pr[i \in L] \geq 1 - O(\alpha)$. Moreover the procedure runs in expected time*

$$O((\frac{B}{\alpha}\log(n/\delta) + \|\widehat{z}\|_0 (1 + \alpha\log(n/\delta)))\log(n/B)).$$

Proof. Consider any $i \in S$ such that none of $E_{coll}(i), E_{off}(i)$, and $E_{noise}(i)$ hold, as happens with probability $1 - O(\alpha)$.

Set $t = \log n, t' = t/4$ and $R_{loc} = O(\log_{1/\alpha}(t/\alpha))$. Let $w_0 = n/B$ and $w_D = w_0/(t')^{D-1}$, so $w_{D_{max}+1} < 1$ for $D_{max} = \log_{t'}(w_0 + 1) < t$. In each round D, Lemma 4.4 implies that if $\pi_{\sigma,b}(i) \in [l_j^{(D)}, l_j^{(D)} + w_D]$ then $\pi_{\sigma,b}(i) \in [l_j^{(D+1)}, l_j^{(D+1)} + w_{D+1}]$ with probability at least $1 - \alpha^{\Omega(R_{loc})} = 1 - \alpha/t$. By a union bound, with probability at least $1 - \alpha$ we have $\pi_{\sigma,b}(i) \in [l_j^{(D_{max}+1)}, l_j^{(D_{max}+1)} + w_{D_{max}+1}] = \{l_j^{(D_{max}+1)}\}$. Thus $i = \pi_{\sigma,b}^{-1}(l_j^{(D_{max}+1)}) \in L$.

Since $R_{loc}D_{max} = O(\log_{1/\alpha}(t/\alpha)\log_t(n/B)) = O(\log(n/B))$, the running time is

$$\begin{aligned} &O(D_{max}(R_{loc}\frac{B}{\alpha}\log(n/\delta) + R_{loc}\|\widehat{z}\|_0(1 + \alpha\log(n/\delta)))) \\ &= O((\frac{B}{\alpha}\log(n/\delta) + \|\widehat{z}\|_0(1 + \alpha\log(n/\delta)))\log(n/B)). \end{aligned}$$

□

4.4 Properties of ESTIMATEVALUES

Lemma 4.6. *For any $i \in L$,*

$$\Pr[\left|\widehat{w}_i - \widehat{x'}_i\right|^2 > \mu^2] < e^{-\Omega(R_{est})}$$

if $B > \frac{Ck}{\alpha\epsilon}$ for some constant C.

Proof. Define $e_r = \widehat{u}_j^{(r)}\omega^{-a_r\sigma i} - \widehat{x'}_i$ in each round r. Suppose none of $E_{coll}^{(r)}(i), E_{off}^{(r)}(i)$, and $E_{noise}^{(r)}(i)$ hold, as happens with probability $1 - O(\alpha)$. Then by Lemma 4.2,

$$\mathbb{E}_{a_r}[|e_r|^2] \leq 2\frac{\rho^2}{\alpha B} = \frac{2k}{\alpha\epsilon B}\mu^2 < \frac{2}{C}\mu^2$$

Hence with $3/4 - O(\alpha) > 5/8$ probability in total,

$$|e_r|^2 < \frac{8}{C}\mu^2 < \mu^2/2$$

for sufficiently large C. Then with probability at least $1 - e^{-\Omega(R_{est})}$, both of the following occur:

$$\left|\operatorname*{median}_r \text{real}(e_r)\right|^2 < \mu^2/2$$
$$\left|\operatorname*{median}_r \text{imag}(e_r)\right|^2 < \mu^2/2.$$

If this is the case, then $|\text{median}_r\, e_r|^2 < \mu^2$. Since $\widehat{w}_i = \widehat{x'}_i + \text{median}\, e_r$, the result follows. □

Lemma 4.7. *Let $R_{est} \geq C\log\frac{B}{\gamma fk}$ for some constant C and parameters $\gamma, f > 0$. Then if* ESTIMATEVALUES *is run with input $k' = 3k$, it returns $\widehat{w_J}$ for $|J| = 3k$ satisfying*

$$\text{Err}^2(\widehat{x'_L} - \widehat{w_J}, fk) \leq \text{Err}^2(\widehat{x'_L}, k) + O(k\mu^2)$$

with probability at least $1 - \gamma$.

Proof. By Lemma 4.6, each index $i \in L$ has

$$\Pr[\left|\widehat{w}_i - \widehat{x'}_i\right|^2 > \mu^2] < \frac{\gamma f k}{B}.$$

Let $U = \{i \in L \mid \left|\widehat{w}_i - \widehat{x'}_i\right|^2 > \mu^2\}$. With probability $1-\gamma$, $|U| \le fk$; assume this happens. Then

$$\left\|(\widehat{x'} - \widehat{w})_{L\setminus U}\right\|_\infty^2 \le \mu^2. \tag{6}$$

Let T contain the top $2k$ coordinates of $\widehat{w}_{L\setminus U}$. By the analysis of Count-Sketch (most specifically, Theorem 3.1 of [PW11]), the ℓ_∞ guarantee (6) means that

$$\left\|\widehat{x'}_{L\setminus U} - \widehat{w}_T\right\|_2^2 \le \mathrm{Err}^2(\widehat{x'}_{L\setminus U}, k) + 3k\mu^2. \tag{7}$$

Because J is the top $3k > (2+f)k$ coordinates of $\widehat{w_L}$, $T \subset J$. Let $J' = J \setminus (T \cup U)$, so $|J'| \le k$. Then

$$\begin{aligned}
\mathrm{Err}^2(\widehat{x'_L} - \widehat{w_J}, fk) &\le \left\|\widehat{x'_{L\setminus U}} - \widehat{w_{J\setminus U}}\right\|_2^2 \\
&= \left\|\widehat{x'}_{L\setminus(U\cup J')} - \widehat{w_T}\right\|_2^2 + \left\|(\widehat{x'} - \widehat{w})_{J'}\right\|_2^2 \\
&\le \left\|\widehat{x'}_{L\setminus U} - \widehat{w_T}\right\|_2^2 + |J'|\left\|(\widehat{x'} - \widehat{w})_{J'}\right\|_\infty^2 \\
&\le \mathrm{Err}^2(\widehat{x'}_{L\setminus U}, k) + 3k\mu^2 + k\mu^2 \\
&= \mathrm{Err}^2(\widehat{x'}_{L\setminus U}, k) + O(k\mu^2)
\end{aligned}$$

where we used Equations (6) and (7). □

4.5 Properties of SPARSEFFT

We will show that $\widehat{x} - \widehat{z}^{(r)}$ gets sparser as r increases, with only a mild increase in the error.

Lemma 4.8. *Define $\widehat{x}^{(r)} = \widehat{x} - \widehat{z}^{(r)}$. Consider any one loop r of* SPARSEFFT*, running with parameters $(B, k, \alpha) = (B_r, k_r, \alpha_r)$ such that $B \ge \frac{Ck}{\alpha^2\epsilon}$ for some C larger than some fixed constant. Then for any $f > 0$,*

$$\mathrm{Err}^2(\widehat{x}^{(r+1)}, fk) \le (1+\epsilon)\,\mathrm{Err}^2(\widehat{x}^{(r)}, k) + O(\epsilon\delta^2 n\,\|\widehat{x}\|_1^2)$$

with probability $1 - O(\alpha/f)$, and the running time is

$$O((\|\widehat{z}^{(r)}\|_0(1 + \alpha\log(n/\delta)) + \frac{B}{\alpha}\log(n/\delta))(\log\frac{1}{\alpha\epsilon} + \log(n/B))).$$

Proof. We use $R_{est} = O(\log\frac{B}{\alpha k}) = O(\log\frac{1}{\alpha\epsilon})$ rounds inside ESTIMATEVALUES.

The running time for LOCATESIGNAL is

$$O((\frac{B}{\alpha}\log(n/\delta) + \|\widehat{z}^{(r)}\|_0(1 + \alpha\log(n/\delta)))\log(n/B)),$$

and for ESTIMATEVALUES is

$$O((\frac{B}{\alpha}\log(n/\delta) + \|\widehat{z}^{(r)}\|_0(1 + \alpha\log(n/\delta)))\log\frac{1}{\alpha\epsilon})$$

for a total running time as given.

Recall that in round r, $\mu^2 = \frac{\epsilon}{k}(\mathrm{Err}^2(\widehat{x}^{(r)}, k) + \delta^2 n \left\|\widehat{x}\right\|_1^2)$ and $S = \{i \in [n] \mid \left|\widehat{x}_i^{(r)}\right|^2 > \mu^2\}$. By Lemma 4.5, each $i \in S$ lies in L_r with probability at least $1 - O(\alpha)$. Let H contain the largest k entries of $\left|\widehat{x}_i^{(r)}\right|^2$. Hence $|(H \cap S) \setminus L| < fk$ with probability at least $1 - O(\alpha/f)$. If this happens, then

$$\begin{aligned}
\mathrm{Err}^2(\widehat{x}^{(r)}_{[n]\setminus L}, fk) &\le \mathrm{Err}^2(\widehat{x}^{(r)}_{H\setminus L}, fk) + \left\|\widehat{x}^{(r)}_{[n]\setminus(H\cup L)}\right\|_2^2 \\
&\le \left\|\widehat{x}^{(r)}_{H\setminus(S\cup L)}\right\|_2^2 + \mathrm{Err}^2(\widehat{x}^{(r)}, k) \\
&\le |H \setminus S| \cdot \left\|\widehat{x}^{(r)}_{H\setminus S}\right\|_\infty^2 + \mathrm{Err}^2(\widehat{x}^{(r)}, k) \\
&\le k\mu^2 + \mathrm{Err}^2(\widehat{x}^{(r)}, k) \qquad (8)
\end{aligned}$$

Let $\widehat{w} = \widehat{z}^{(r+1)} - \widehat{z}^{(r)} = \widehat{x}^{(r)} - \widehat{x}^{(r+1)}$ by the vector recovered by ESTIMATEVALUES. Then $\mathrm{supp}(\widehat{w}) \subset L$, so

$$\begin{aligned}
\mathrm{Err}^2(\widehat{x}^{(r+1)}, 2fk) &= \mathrm{Err}^2(\widehat{x}^{(r)} - \widehat{w}, 2fk) \\
&\le \mathrm{Err}^2(\widehat{x}^{(r)}_{[n]\setminus L}, fk) + \mathrm{Err}^2(\widehat{x}^{(r)}_L - \widehat{w}, fk) \\
&\le \mathrm{Err}^2(\widehat{x}^{(r)}_{[n]\setminus L}, fk) + \mathrm{Err}^2(\widehat{x}^{(r)}_L, k) + O(k\mu^2)
\end{aligned}$$

by Lemma 4.7. But by Equation (8), this gives

$$\begin{aligned}
\mathrm{Err}^2(\widehat{x}^{(r+1)}, 2fk) &\le \mathrm{Err}^2(\widehat{x}^{(r)}_{[n]\setminus L}, k) + \mathrm{Err}^2(\widehat{x}^{(r)}_L, k) + O(k\mu^2) \\
&\le \mathrm{Err}^2(\widehat{x}^{(r)}, k) + O(k\mu^2) \\
&= (1 + O(\epsilon))\,\mathrm{Err}^2(\widehat{x}^{(r)}, k) + O(\epsilon\delta^2 n \left\|\widehat{x}\right\|_1^2).
\end{aligned}$$

The result follows from rescaling f and ϵ by constant factors. □

Given the above, this next proof follows a similar argument to [IPW11], Theorem 3.7.

Theorem 4.9. *With $2/3$ probability,* SPARSEFFT *recovers $\widehat{z}^{(R+1)}$ such that*

$$\left\|\widehat{x} - \widehat{z}^{(R+1)}\right\|_2 \le (1+\epsilon)\,\mathrm{Err}(\widehat{x}, k) + \delta \left\|\widehat{x}\right\|_2$$

in $O(\frac{k}{\epsilon}\log(n/k)\log(n/\delta))$ time.

Proof. Define $f_r = O(1/r^2)$ so $\sum f_r < 1/4$. Choose R so $\prod_{r\le R} f_r < 1/k \le \prod_{r<R} f_r$. Then $R = O(\log k/\log\log k)$, since $\prod_{r\le R} f_r < (f_{R/2})^{R/2} = (2/R)^R$.

Set $\epsilon_r = f_r\epsilon$, $\alpha_r = \Theta(f_r^2)$, $k_r = k\prod_{i<r} f_i$, $B_r = O(\frac{k}{\epsilon}\alpha_r f_r)$. Then $B_r = \omega(\frac{k_r}{\alpha_r^2\epsilon_r})$, so for sufficiently large constant the constraint of Lemma 4.8 is satisfied. For appropriate constants, Lemma 4.8 says that in each round r,

$$\mathrm{Err}^2(\widehat{x}^{(r+1)}, k_{r+1}) = \mathrm{Err}^2(\widehat{x}^{(r+1)}, f_r k_r) \le (1 + f_r\epsilon)\,\mathrm{Err}^2(\widehat{x}^{(r)}, k_r) + O(f_r\epsilon\delta^2 n \left\|\widehat{x}\right\|_1^2) \qquad (9)$$

with probability at least $1 - f_r$. The error accumulates, so in round r we have

$$\mathrm{Err}^2(\widehat{x}^{(r)}, k_r) \le \mathrm{Err}^2(\widehat{x}, k)\prod_{i<r}(1+f_i\epsilon) + \sum_{i<r} O(f_r\epsilon\delta^2 n \left\|\widehat{x}\right\|_1^2)\prod_{i<j<r}(1+f_j\epsilon)$$

with probability at least $1 - \sum_{i<r} f_i > 3/4$. Hence in the end, since $k_{R+1} = k\prod_{i\le R} f_i < 1$,

$$\left\|\widehat{x}^{(R+1)}\right\|_2^2 = \mathrm{Err}^2(\widehat{x}^{(R+1)}, k_{R+1}) \le \mathrm{Err}^2(\widehat{x}, k)\prod_{i\le R}(1+f_i\epsilon) + O(R\epsilon\delta^2 n \left\|\widehat{x}\right\|_1^2)\prod_{i\le R}(1+f_i\epsilon)$$

with probability at least $3/4$. We also have

$$\prod_i (1 + f_i\epsilon) \le e^{\epsilon \sum_i f_i} \le e$$

making

$$\prod_i (1 + f_i\epsilon) \le 1 + e\sum_i f_i\epsilon < 1 + 2\epsilon.$$

Thus we get the approximation factor

$$\left\|\widehat{x} - \widehat{z}^{(R+1)}\right\|_2^2 \le (1+2\epsilon)\,\mathrm{Err}^2(\widehat{x}, k) + O((\log k)\epsilon\delta^2 n\,\|\widehat{x}\|_1^2)$$

with at least $3/4$ probability. Rescaling δ by poly(n), using $\|\widehat{x}\|_1^2 \le n\,\|\widehat{x}\|_2$, and taking the square root gives the desired

$$\left\|\widehat{x} - \widehat{z}^{(R+1)}\right\|_2 \le (1+\epsilon)\,\mathrm{Err}(\widehat{x}, k) + \delta\,\|\widehat{x}\|_2\,.$$

Now we analyze the running time. The update $\widehat{z}^{(r+1)} - \widehat{z}^{(r)}$ in round r has support size $3k_r$, so in round r

$$\|\widehat{z}^{(r)}\|_0 \le \sum_{i<r} 3k_r = O(k).$$

Thus the expected running time in round r is

$$\begin{aligned}
&O((k(1+\alpha_r\log(n/\delta)) + \frac{B_r}{\alpha_r}\log(n/\delta))(\log\frac{1}{\alpha_r\epsilon_r} + \log(n/B_r))) \\
&= O((k + \frac{k}{r^4}\log(n/\delta) + \frac{k}{\epsilon r^2}\log(n/\delta))(\log\frac{r^2}{\epsilon} + \log(n\epsilon/k) + \log r)) \\
&= O((k + \frac{k}{\epsilon r^2}\log(n/\delta))(\log r + \log(n/k)))
\end{aligned}$$

We split the terms multiplying k and $\frac{k}{\epsilon r^2}\log(n/\delta)$, and sum over r. First,

$$\begin{aligned}
\sum_{r=1}^{R}(\log r + \log(n/k)) &\le R\log R + R\log(n/k) \\
&\le O(\log k + \log k\log(n/k)) \\
&= O(\log k\log(n/k)).
\end{aligned}$$

Next,

$$\sum_{r=1}^{R}\frac{1}{r^2}(\log r + \log(n/k)) = O(\log(n/k))$$

Thus the total running time is

$$O(k\log k\log(n/k) + \frac{k}{\epsilon}\log(n/\delta)\log(n/k)) = O(\frac{k}{\epsilon}\log(n/\delta)\log(n/k)).$$

□

5 Reducing the full k-dimensional DFT to the exact k-sparse case in n dimensions

In this section we show the following lemma. Assume that k divides n.

Lemma 5.1. *Suppose that there is an algorithm A that, given an n-dimensional vector y such that $\hat{y}$ is k-sparse, computes $\hat{y}$ in time $T(k)$. Then there is an algorithm A' that given a k-dimensional vector x computes $\hat{x}$ in time $O(T(k))$.*

Proof. Given a k-dimensional vector x, we define $y_i = x_{i \bmod k}$, for $i = 0 \dots n-1$. Whenever A requests a sample y_i, we compute it from x in constant time. Moreover, we have that $\hat{y}_i = \hat{x}_{i/(n/k)}$ if i is a multiple of (n/k), and $\hat{y}_i = 0$ otherwise. Thus $\hat{y}$ is k-sparse. Since $\hat{x}$ can be immediately recovered from $\hat{y}$, the lemma follows. □

Corollary 5.2. *Assume that the n-dimensional DFT cannot be computed in $o(n \log n)$ time. Then any algorithm for the k-sparse DFT (for vectors of arbitrary dimension) must run in $\Omega(k \log k)$ time.*

6 Lower Bound

In this section, we show any algorithm satisfying Equation (1) must access $\Omega(k \log(n/k)/\log\log n)$ samples of x.

We translate this problem into the language of compressive sensing:

Theorem 6.1. *Let $F \in \mathbb{C}^{n \times n}$ be orthonormal and satisfy $|F_{i,j}| = 1/\sqrt{n}$ for all i, j. Suppose an algorithm takes m adaptive samples of Fx and computes x' with*

$$\|x - x'\|_2 \leq 2 \min_{k\text{-sparse } x^*} \|x - x^*\|_2$$

for any x, with probability at least $3/4$. Then it must have $m = \Omega(k \log(n/k)/\log\log n)$.

Corollary 6.2. *Any algorithm computing the approximate Fourier transform must access $\Omega(k \log(n/k)/\log\log n)$ samples from the time domain.*

If the samples were chosen non-adaptively, we would immediately have $m = \Omega(k \log(n/k))$ by [PW11]. However, an algorithm could choose samples based on the values of previous samples. In the sparse recovery framework allowing general linear measurements, this adaptivity can decrease the number of measurements to $O(k \log\log(n/k))$ [IPW11]; in this section, we show that adaptivity is much less effective in our setting where adaptivity only allows the choice of Fourier coefficients.

We follow the framework of Section 4 of [PW11]. In this section we use standard notation from information theory, including $I(x; y)$ for mutual information, $H(x)$ for discrete entropy, and $h(x)$ for continuous entropy. Consult a reference such as [CT91] for details.

Let $\mathcal{F} \subset \{S \subset [n] : |S| = k\}$ be a family of k-sparse supports such that:

- $|S \oplus S'| \geq k$ for $S \neq S' \in \mathcal{F}$, where $\oplus$ denotes the exclusive difference between two sets, and
- $\log |\mathcal{F}| = \Omega(k \log(n/k))$.

This is possible; for example, a random code on $[n/k]^k$ with relative distance $1/2$ has these properties.

For each $S \in \mathcal{F}$, let $X^S = \{x \in \{0, \pm 1\}^n \mid \operatorname{supp}(x^S) = S\}$. Let $x \in X^S$ uniformly at random. The variables x_i, $i \in S$, are i.i.d. subgaussian random variables with parameter $\sigma^2 = 1$, so for any row F_j of F, $F_j x$ is subgaussian with parameter $\sigma^2 = k/n$. Therefore

$$\Pr_{x \in X^S}[|F_j x| > t\sqrt{k/n}] < 2e^{-t^2/2}$$

hence for each S, we can choose an $x^S \in X^S$ with

$$\left\|Fx^S\right\|_\infty < O(\sqrt{\frac{k \log n}{n}}). \tag{10}$$

Let $X = \{x^S \mid S \in \mathcal{F}\}$ be the set of such x^S.

Let $w \sim N(0, \alpha\frac{k}{n} I_n)$ be i.i.d. normal with variance $\alpha k/n$ in each coordinate.

Consider the following process:

Procedure. First, Alice chooses $S \in \mathcal{F}$ uniformly at random, then selects the $x \in X$ with $\mathrm{supp}(x) = S$. Alice independently chooses $w \sim N(0, \alpha\frac{k}{n} I_n)$ for a parameter $\alpha = \Theta(1)$ sufficiently small. For $j \in [m]$, Bob chooses $i_j \in [n]$ and observes $y_j = F_{i_j}(x + w)$. He then computes the result $x' \approx x$ of sparse recovery, rounds to X by $\hat{x} = \arg\min_{x^* \in X} \|x^* - x'\|_2$, and sets $S' = \mathrm{supp}(\hat{x})$. This gives a Markov chain $S \to x \to y \to x' \to \hat{x} \to S'$.

We will show that deterministic sparse recovery algorithms require large m to succeed on this input distribution $x + w$ with $3/4$ probability. By Yao's minimax principle, this means randomized sparse recovery algorithms also require large m to succeed with $3/4$ probability.

Our strategy is to give upper and lower bounds on $I(S; S')$, the mutual information between S and S'.

Lemma 6.3 (Analog of Lemma 4.3 of [PW11] for $\epsilon = O(1)$)**.** *There exists a constant $\alpha' > 0$ such that if $\alpha < \alpha'$, then $I(S; S') = \Omega(k \log(n/k))$.*

Proof. Assuming the sparse recovery succeeds (as happens with 3/4 probability), we have $\|x' - (x + w)\|_2 \leq 2\|w\|_2$, which implies $\|x' - x\|_2 \leq 3\|w\|_2$. Therefore

$$\begin{aligned}\|\hat{x} - x\|_2 &\leq \|\hat{x} - x'\|_2 + \|x' - x\|_2 \\ &\leq 2\|x' - x\|_2 \\ &\leq 6\|w\|_2 .\end{aligned}$$

We also know $\|x' - x''\|_2 \geq \sqrt{k}$ for all distinct $x', x'' \in X$ by construction. Because $\mathbb{E}[\|w\|_2^2] = \alpha k$, with probability at least $3/4$ we have $\|w\|_2 \leq \sqrt{4\alpha k} < \sqrt{k}/6$ for sufficiently small α. But then $\|\hat{x} - x\|_2 < \sqrt{k}$, so $\hat{x} = x$ and $S = S'$. Thus $\Pr[S \neq S'] \leq 1/2$.

Fano's inequality states $H(S \mid S') \leq 1 + \Pr[S \neq S'] \log |\mathcal{F}|$. Thus

$$I(S; S') = H(S) - H(S \mid S') \geq -1 + \frac{1}{2}\log|\mathcal{F}| = \Omega(k \log(n/k))$$

as desired. □

We next show an analog of their upper bound (Lemma 4.1 of [PW11]) on $I(S; S')$ for adaptive measurements of bounded ℓ_∞ norm. The proof follows the lines of [PW11], but is more careful about dependencies and needs the ℓ_∞ bound on Fx.

Lemma 6.4.

$$I(S; S') \leq O(m \log(1 + \frac{1}{\alpha}\log n)).$$

Proof. Let $A_j = F_{i_j}$ for $j \in [m]$, and let $w'_j = A_j w$. The w'_j are independent normal variables with variance $\alpha\frac{k}{n}$. Because the A_j are orthonormal and w is drawn from a rotationally invariant distribution, the w' are also independent of x.

Let $y_j = A_j x + w'_j$. We know $I(S; S') \leq I(x; y)$ because $S \to x \to y \to S'$ is a Markov chain. Because the variables A_j are deterministic given $y_1, \ldots, y_{j-1}$,

$$\begin{aligned}I(x; y_j \mid y_1, \ldots, y_{j-1}) &= I(x; A_j x + w'_j \mid y_1, \ldots, y_{j-1}) \\ &= h(A_j x + w'_j \mid y_1, \ldots, y_{j-1}) - h(A_j x + w'_j \mid x, y_1, \ldots, y_{j-1}) \\ &= h(A_j x + w'_j \mid y_1, \ldots, y_{j-1}) - h(w'_j).\end{aligned}$$

By the chain rule for information,

$$\begin{aligned} I(S;S') &\le I(x;y) \\ &= \sum_{j=1}^{m} I(x; y_j \mid y_1, \ldots, y_{j-1}) \\ &= \sum_{j=1}^{m} h(A_j x + w'_j \mid y_1, \ldots, y_{j-1}) - h(w'_j) \\ &\le \sum_{j=1}^{m} h(A_j x + w'_j) - h(w'_j). \end{aligned}$$

Thus it suffices to show $h(A_j x + w'_j) - h(w'_j) = O(\log(1 + \frac{1}{\alpha}\log n))$ for all j.

Note that A_j depends only on $y_1, \ldots, y_{j-1}$, so it is independent of w'_j. Thus

$$\mathbb{E}[(A_j x + w'_j)^2] = \mathbb{E}[(A_j x)^2] + \mathbb{E}[(w'_j)^2] \le O(\frac{k \log n}{n}) + \alpha \frac{k}{n}$$

by Equation (10). Because the maximum entropy distribution under an ℓ_2 constraint is a Gaussian, we have

$$\begin{aligned} h(A_j x + w'_j) - h(w'_j) &\le h(N(0, O(\frac{k\log n}{n}) + \alpha\frac{k}{n})) - h(N(0, \alpha\frac{k}{n})) \\ &= \frac{1}{2}\log(1 + \frac{O(\log n)}{\alpha}) \\ &= O(\log(1 + \frac{1}{\alpha}\log n)). \end{aligned}$$

as desired. □

Theorem 6.1 follows from Lemma 6.3 and Lemma 6.4, with $\alpha = \Theta(1)$.

7 Efficient Constructions of Window Functions

Claim 7.1. *Let* cdf *denote the standard Gaussian cumulative distribution function. Then:*

1. $\mathrm{cdf}(t) = 1 - \mathrm{cdf}(-t)$.
2. $\mathrm{cdf}(t) \le e^{-t^2/2}$ *for* $t < 0$.
3. $\mathrm{cdf}(t) < \delta$ *for* $t < -\sqrt{2\log(1/\delta)}$.
4. $\int_{x=-\infty}^{t} \mathrm{cdf}(x)dx < \delta$ *for* $t < -\sqrt{2\log(3/\delta)}$.
5. *For any* δ, *there exists a function* $\widetilde{\mathrm{cdf}}_\delta(t)$ *computable in* $O(\log(1/\delta))$ *time such that* $\left\|\mathrm{cdf} - \widetilde{\mathrm{cdf}}_\delta\right\|_\infty < \delta$.

Proof.

1. Follows from the symmetry of Gaussian distribution.
2. Follows from a standard moment generating function bound on Gaussian random variables.
3. Follows from (2).
4. Property (2) implies that $\mathrm{cdf}(t)$ is at most $\sqrt{2\pi} < 3$ times larger than the Gaussian pdf. Then apply (3).

5. By (1) and (3), $\mathrm{cdf}(t)$ can be computed as $\pm\delta$ or $1 \pm \delta$ unless $|t| < \sqrt{2(\log(1/\delta))}$. But then an efficient expansion around 0 only requires $O(\log(1/\delta))$ terms to achieve precision $\pm\delta$.

   For example, we can truncate the representation [Mar04]

$$\mathrm{cdf}(t) = \frac{1}{2} + \frac{e^{-t^2/2}}{\sqrt{2\pi}}\left(t + \frac{t^3}{3} + \frac{t^5}{3\cdot 5} + \frac{t^7}{3\cdot 5\cdot 7} + \cdots\right)$$

   at $O(\log(1/\delta))$ terms.

□

Claim 7.2. *Define the continuous Fourier transform of $f(t)$ by*

$$\widehat{f}(s) = \int_{-\infty}^{\infty} e^{-2\pi \mathbf{i} s t} f(t) dt.$$

For $t \in [n]$, define

$$g_t = \sqrt{n} \sum_{j=-\infty}^{\infty} f(t + nj)$$

and

$$g'_t = \sum_{j=-\infty}^{\infty} \widehat{f}(t/n + j).$$

Then $\widehat{g} = g'$, where $\widehat{g}$ is the n-dimensional DFT of g.

Proof. Let $\Delta_1(t)$ denote the Dirac comb of period 1: $\Delta_1(t)$ is a Dirac delta function when t is an integer and zero elsewhere. Then $\widehat{\Delta}_1 = \Delta_1$. For any $t \in [n]$, we have

$$\begin{aligned}
\widehat{g}_t &= \sum_{s=1}^{n} \sum_{j=-\infty}^{\infty} f(s + nj) e^{-2\pi \mathbf{i} t s/n} \\
&= \sum_{s=1}^{n} \sum_{j=-\infty}^{\infty} f(s + nj) e^{-2\pi \mathbf{i} t (s+nj)/n} \\
&= \sum_{s=-\infty}^{\infty} f(s) e^{-2\pi \mathbf{i} t s/n} \\
&= \int_{-\infty}^{\infty} f(s) \Delta_1(s) e^{-2\pi \mathbf{i} t s/n} ds \\
&= \widehat{(f \cdot \Delta_1)}(t/n) \\
&= (\widehat{f} * \Delta_1)(t/n) \\
&= \sum_{j=-\infty}^{\infty} \widehat{f}(t/n + j) \\
&= g'_t.
\end{aligned}$$

□

Lemma 7.3. *For any parameters $B \geq 1, \delta > 0$, and $\alpha > 0$, there exist flat window functions G and $\widehat{G'}$ such that G can be computed in $O(\frac{B}{\alpha}\log(n/\delta))$ time, and for each i $\widehat{G'}_i$ can be evaluated in $O(\log(n/\delta))$ time.*

Proof. We will show this for a function $\widehat{G'}$ that is a Gaussian convolved with a rectangular filter. First we construct analogous window functions for the continuous Fourier transform. We then show that discretizing these functions gives the desired result.

For some parameters σ and 1 with $1 < \sigma \leq n$ and $C < 1$ to be determined later, define $\widehat{D}$ and $\widehat{F}$ to be Gaussian and rectangular filters, respectively, according to:

- $\widehat{D}(s) = \frac{\sigma}{\sqrt{2\pi}} e^{-\sigma^2 s^2/2}$ is a Gaussian pdf with standard deviation $1/\sigma$.
- $D(t) = e^{-2\pi^2 t^2/\sigma^2}$ is $\sigma/\sqrt{2\pi}$ times a Gaussian pdf with standard deviation $\sigma/2\pi$
- $\widehat{F}(s) = \text{rect}(s/(2C))$ is 1 if $|s| < C$ and 0 otherwise, is a rectangular filter of length $2C$.
- $F(t) = 2C\text{sinc}(2Ct) = \frac{\sin(2\pi Ct)}{\pi t}$.

Consider the filter

$$G^* = D \cdot F$$
$$\widehat{G^*} = \widehat{D} * \widehat{F}.$$

We have $|G^*(t)| \leq 2C\,|D(t)| < 2C\delta$ for $|t| > \frac{\sigma}{2\pi}\sqrt{2\log(1/\delta)}$. Furthermore, G^* is computable in $O(1)$ time.

Its Fourier transform is $\widehat{G^*}(s) = \text{cdf}(\sigma(s+C)) - \text{cdf}(\sigma(s-C))$. By Claim 7.1 we have for $|s| > C + \sqrt{2\log(1/\delta)}/\sigma$ that $\widehat{G^*}(s) = \pm\delta$. We also have, for $|s| < C - \sqrt{2\log(1/\delta)}/\sigma$, that $\widehat{G^*}(s) = 1 \pm 2\delta$.

This gives us efficient *continuous* flat window functions. To get *discrete* ones, for $i \in [n]$ let $H_i = \sqrt{n}\sum_{j=\infty}^{\infty} G^*(i+nj)$. By Claim 7.2 it has DFT $\widehat{H}_i = \sum_{j=\infty}^{\infty} \widehat{G^*}(i/n + j)$.

We show how to approximate H and $\widehat{H}$ efficiently. First, H:

$$\begin{aligned}
\sum_{|i|>1+\frac{\sigma}{2\pi}\sqrt{2\log(1/\delta)}} |G^*(i)| &\leq 4C \sum_{i<-1-\frac{\sigma}{2\pi}\sqrt{2\log(1/\delta)}} |D(i)| \\
&\leq 4C \int_{-\infty}^{-\frac{\sigma}{2\pi}\sqrt{2\log(1/\delta)}} |D(x)|\,dx \\
&\leq 4C \frac{\sigma}{\sqrt{2\pi}} \text{cdf}(-\sqrt{2\log(1/\delta)}) \\
&< 2C\sigma\delta \leq 2n\delta.
\end{aligned}$$

Thus if we let

$$G_i = \sqrt{n} \sum_{\substack{|j|<\frac{\sigma}{2\pi}\sqrt{2\log(1/\delta)} \\ j \equiv i \pmod n}} G^*(j)$$

for $|i| < \frac{\sigma}{2\pi}\sqrt{2\log(1/\delta)}$ and $G_i = 0$ otherwise, then $\|G - H\|_1 \leq 2\delta n^{3/2}$.

Now consider approximating $\widehat{H}$. Note that for integer i with $|i| \le n/2$,

$$\begin{aligned}
\widehat{H}_i - \widehat{G^*}(i/n) &= \sum_{\substack{j\in\mathbb{Z}\\ j\neq 0}} \widehat{G^*}(i/n+j) \\
\left|\widehat{H}_i - \widehat{G^*}(i/n)\right| &\le 2\sum_{j=0}^{\infty} \widehat{G^*}(-1/2-j) \\
&\le 2\sum_{j=0}^{\infty} \mathrm{cdf}(\sigma(-1/2-j+C)) \\
&\le 2\int_{-\infty}^{-1/2} \mathrm{cdf}(\sigma(x+C))dx + 2\,\mathrm{cdf}(\sigma(-1/2+C)) \\
&\le 2\delta/\sigma + 2\delta \le 4\delta
\end{aligned}$$

by Claim 7.1, as long as

$$\sigma(1/2 - C) > \sqrt{2\log(3/\delta)}. \tag{11}$$

Let

$$\widehat{G'}_i = \begin{cases} 1 & |i| \le n(C - \sqrt{2\log(1/\delta)}/\sigma) \\ 0 & |i| \ge n(C + \sqrt{2\log(1/\delta)}/\sigma) \\ \widetilde{\mathrm{cdf}}_\delta(\sigma(i/n + C)) - \widetilde{\mathrm{cdf}}_\delta(\sigma(i/n - C)) & \text{otherwise} \end{cases}$$

where $\widetilde{\mathrm{cdf}}_\delta(t)$ computes $\mathrm{cdf}(t)$ to precision $\pm\delta$ in $O(\log(1/\delta))$ time, as per Claim 7.1. Then $\widehat{G'}_i = \widehat{G^*}(i/n) \pm 2\delta = \widehat{H}_i \pm 6\delta$. Hence

$$\begin{aligned}
\left\|\widehat{G} - \widehat{G'}\right\|_\infty &\le \left\|\widehat{G'} - \widehat{H}\right\|_\infty + \left\|\widehat{G} - \widehat{H}\right\|_\infty \\
&\le \left\|\widehat{G'} - \widehat{H}\right\|_\infty + \left\|\widehat{G} - \widehat{H}\right\|_2 \\
&= \left\|\widehat{G'} - \widehat{H}\right\|_\infty + \|G - H\|_2 \\
&\le \left\|\widehat{G'} - \widehat{H}\right\|_\infty + \|G - H\|_1 \\
&\le (2n^{3/2} + 6)\delta.
\end{aligned}$$

Replacing δ by δ/n^2 and plugging in $\sigma = \frac{4B}{\alpha}\sqrt{2\log(n/\delta)} > 1$ and $C = (1-\alpha/2)/(2B) < 1$, we have the required properties of flat window functions:

- $|G_i| = 0$ for $|i| \ge \Omega(\frac{B}{\alpha}\log(n/\delta))$
- $\widehat{G'}_i = 1$ for $|i| \le (1-\alpha)n/(2B)$
- $\widehat{G'}_i = 0$ for $|i| \ge n/(2B)$
- $\widehat{G'}_i \in [0,1]$ for all i.
- $\left\|\widehat{G'} - \widehat{G}\right\|_\infty < \delta$.
- We can compute G over its entire support in $O(\frac{B}{\alpha}\log(n/\delta))$ total time.
- For any i, $\widehat{G'}_i$ can be computed in $O(\log(n/\delta))$ time for $|i| \in [(1-\alpha)n/(2B), n/(2B)]$ and $O(1)$ time otherwise.

The only requirement was Equation (11), which is that

$$\frac{4B}{\alpha}\sqrt{2\log(n/\delta)}(1/2 - \frac{1-\alpha/2}{2B}) > \sqrt{2\log(3n/\delta)}.$$

This holds if $B \geq 2$. The $B = 1$ case is trivial using the constant function $\widehat{G'}_i = 1$. □

8 Open questions

- Design an $O(k \log n)$-time algorithm for general signals. Alternatively, prove that no such algorithm exists, under "reasonable" assumptions.[10]
- Reduce the sample complexity of the algorithms. Currently, the number of samples used by each algorithm is only bounded by their running times.
- Extend the results to other (related) tasks, such as computing the sparse Walsh-Hadamard Transform.
- Extend the algorithm to the case when n is not a power of 2. Note that some of the earlier algorithms, e.g., [GMS05], work for any n.
- Improve the failure probability of the algorithms. Currently, the algorithms only succeed with constant probability. Straightforward amplification would take a $\log(1/p)$ factor slowdown to succeed with $1 - p$ probability. One would hope to avoid this slowdown.

Acknowledgements

The authors would like to thank Martin Strauss and Ludwig Schmidt for many helpful comments about the writing of the paper. This work is supported by the Space and Naval Warfare Systems Center Pacific under Contract No. N66001-11-C-4092, David and Lucille Packard Fellowship, and NSF grants CCF-1012042 and CNS-0831664. E. Price is supported in part by an NSF Graduate Research Fellowship.

References

[AFS93] R. Agrawal, C. Faloutsos, and A. Swami. Efficient similarity search in sequence databases. *Int. Conf. on Foundations of Data Organization and Algorithms*, pages 69–84, 1993.

[AGS03] A. Akavia, S. Goldwasser, and S. Safra. Proving hard-core predicates using list decoding. *Annual Symposium on Foundations of Computer Science*, 44:146–159, 2003.

[Aka10] A. Akavia. Deterministic sparse Fourier approximation via fooling arithmetic progressions. *COLT*, pages 381–393, 2010.

[CGX96] A. Chandrakasan, V. Gutnik, and T. Xanthopoulos. Data driven signal processing: An approach for energy efficient computing. *International Symposium on Low Power Electronics and Design*, 1996.

[CRT06] E. Candes, J. Romberg, and T. Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52:489–509, 2006.

[10] The $\Omega(k \log(n/k)/\log\log n)$ lower bound for the sample complexity shows that the running time of our algorithm, $O(k \log n \log(n/k))$, is equal to the sample complexity of the problem times (roughly) $\log n$. One could speculate that this logarithmic discrepancy is due to the need for using FFT to process the samples. Although we do not have any evidence for the optimality of our general algorithm, the "sample complexity times $\log n$" bound appears to be a natural barrier to further improvements.

[CT91] Thomas Cover and Joy Thomas. *Elements of Information Theory*. Wiley Interscience, 1991.

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

[DRZ07] I. Daubechies, O. Runborg, and J. Zou. A sparse spectral method for homogenization multiscale problems. *Multiscale Model. Sim.*, 6(3):711–740, 2007.

[GGI+02] A. Gilbert, S. Guha, P. Indyk, M. Muthukrishnan, and M. Strauss. Near-optimal sparse Fourier representations via sampling. *STOC*, 2002.

[GL89] O. Goldreich and L. Levin. A hard-corepredicate for allone-way functions. *STOC*, pages 25–32, 1989.

[GLPS10] Anna C. Gilbert, Yi Li, Ely Porat, and Martin J. Strauss. Approximate sparse recovery: optimizing time and measurements. In *STOC*, pages 475–484, 2010.

[GMS05] A. Gilbert, M. Muthukrishnan, and M. Strauss. Improved time bounds for near-optimal space Fourier representations. *SPIE Conference, Wavelets*, 2005.

[GST08] A.C. Gilbert, M.J. Strauss, and J. A. Tropp. A tutorial on fast Fourier sampling. *Signal Processing Magazine*, 2008.

[HIKP12a] H. Hassanieh, P. Indyk, D. Katabi, and E. Price. sFFT: Sparse Fast Fourier Transform. *http://groups.csail.mit.edu/netmit/sFFT/*, 2012.

[HIKP12b] H. Hassanieh, P. Indyk, D. Katabi, and E. Price. Simple and practical algorithm for sparse Fourier transform. *SODA*, 2012.

[HT01] Juha Heiskala and John Terry, Ph.D. *OFDM Wireless LANs: A Theoretical and Practical Guide*. Sams, Indianapolis, IN, USA, 2001.

[IPW11] P. Indyk, E. Price, and D. P. Woodruff. On the power of adaptivity in sparse recovery. *FOCS*, 2011.

[Iwe10] M. A. Iwen. Combinatorial sublinear-time Fourier algorithms. *Foundations of Computational Mathematics*, 10:303–338, 2010.

[KKL88] J. Kahn, G. Kalai, and N. Linial. The influence of variables on boolean functions. *FOCS*, 1988.

[KM91] E. Kushilevitz and Y. Mansour. Learning decision trees using the Fourier spectrum. *STOC*, 1991.

[LMN93] N. Linial, Y. Mansour, and N. Nisan. Constant depth circuits, Fourier transform, and learnability. *Journal of the ACM (JACM)*, 1993.

[LVS11] Mengda Lin, A. P. Vinod, and Chong Meng Samson See. A new flexible filter bank for low complexity spectrum sensing in cognitive radios. *Journal of Signal Processing Systems*, 62(2):205–215, 2011.

[Man92] Y. Mansour. Randomized interpolation and approximation of sparse polynomials. *ICALP*, 1992.

[Mar04] G. Marsaglia. Evaluating the normal distribution. *Journal of Statistical Software*, 11(4):1–7, 2004.

[MNL10] A. Mueen, S. Nath, and J. Liu. Fast approximate correlation for massive time-series data. In *Proceedings of the 2010 international conference on Management of data*, pages 171–182. ACM, 2010.

[O'D08] R. O'Donnell. Some topics in analysis of boolean functions (tutorial). *STOC*, 2008.

[PW11] E. Price and D. P. Woodruff. $(1+\epsilon)$-approximate sparse recovery. *FOCS*, 2011.