A Sublinear Algorithm of Sparse Fourier Transform for Nonequispaced Data*

Jing Zou †

12th August 2005

Abstract

We present a sublinear randomized algorithm to compute a sparse Fourier transform for nonequispaced data. More precisely, we address the situation where a signal S is known to consist of N equispaced samples, of which only $L < N$ are available. This includes the case of "equispaced data with gaps"; if the ratio $p = L/N$ is smaller than 1, the available data are typically non-equispaced samples, with little or no visible trace of the equispacing of the full set of N samples. We extend an approach for equispaced data that was presented in [21]; the extended algorithm reconstructs, from the incomplete data, a near-optimal B-term representation R with high probability $1-\delta$, in time and space $poly(B, \log(L), \log(1/(1-p)), \log(1/\delta), \epsilon^{-1})$, such that $\|S-R\|_2^2 \leq (1+\epsilon)\|S-R_{opt}^B\|_2^2$, where R_{opt}^B is the optimal B-term Fourier representation of signal S. The sublinear $poly(\log L)$ time is compared to the superlinear $O(L^{1+(d-1)/\beta}\log L)$ time requirement of the present best known Inverse Nonequispaced Fast Fourier Transform (INFFT) algorithms, in the sense of weighted norm with the number of dimensions d, smoothness parameter β. Numerical experiments support the advantage in speed of our algorithm over other methods for sparse signals: it already outperforms INFFT for large but realistic size N and works well even in the situation of a large percentage of missing data and in the presence of large noise.

1 Introduction

We consider the problem in which the recovery of a discrete time signal S of length N is sought when only L signal values are known. In general, this is of course an insoluble problem; we consider it here under the additional assumption that the signal has a sparse Fourier transform. Let us fix the notations: the signal is denoted by $S = (S(t))_{t=0,\ldots,N-1}$, but we have at our disposal only the $(S(i))_{i\in T}$, where the set T is a subset of $\{0,\ldots,N-1\}$ and $|T| = L$. The Fourier transform of signal S is $\hat{S} = (\hat{S}(0),\ldots,\hat{S}(N-1))$, defined by $\hat{S}(\omega) = \frac{1}{\sqrt{N}}\sum_{t=0}^{N-1} S(t)e^{-2\pi i\omega t/N}$. In terms

*This work was partially supported by NSF grant DMS-03168875 and AFOSR grant 109-6047.

†Program of Applied and Computational Mathematics, Princeton University, Fine Hall, Washington Road, Princeton, NJ 08544, (jzou@math.princeton.edu).

of the Fourier basis functions $\phi_\omega(t) = \frac{1}{\sqrt{N}} e^{2\pi i \omega t/N}$, S can be written as $S = \sum_{\omega=0}^{N-1} \hat{S}(\omega)\phi_\omega(t)$; this is the (discrete) Fourier representation of S. A signal S is said to have a B-sparse Fourier representation, if there exists a subset $\Omega \subset \{0, \ldots, N-1\}$ with $|\Omega| = B$, and values $c(\omega) \neq 0$ for $\omega \in \Omega$, such that $S(t) = \sum_{\omega\in\Omega} c(\omega)\phi_\omega$. For a signal that does not have a B-sparse Fourier representation, we denote by $R^B_{opt}(S)$ the optimal B-term Sparse Fourier representation of S.

This paper presents a sublinear algorithm to recover a B-sparse Fourier representation of a signal S from incomplete data. Our algorithm also extends to the case where the Fourier transform $\hat{S}$ is not B-sparse, where we aim to find a near-optimal B-term Fourier representation, i.e. $R = \sum_{\omega\in\Omega} c(\omega)\phi_\omega$, such that

$$\|S - R\|_2^2 \leq (1+\epsilon)\|S - R^B_{opt}(S)\|_2^2. \tag{1}$$

A typical situation where our study applies is the observation of non-equispaced data, where the samples are nevertheless all elements of $\tau\mathbb{Z}$ for some $\tau > 0$. For a signal with evenly spaced data, the famous Fast Fourier Transform (FFT) computes all the Fourier coefficients in time $O(N \log N)$. However, the requirement of equally distributed data by FFT raises challenges for many important applications. For instance, because of the occurrence of instrumental drop-outs, the data may be available only on a set of non-consecutive integers. Another example occurs in astronomy, where the observers cannot completely control the availability of observational data: a telescope can only see the universe on nights when skies are not cloudy. In fact, computing the Fourier representation from irregularly spaced data has wide applications [20] in processing astrophysical and seismic data, the spectral method on adaptive grids, the tracking of Lagrangian particles, and the implementation of semi-Lagrangian methods.

In many of these applications, a few large Fourier coefficients already capture the major time-invariant wave-like information of the signal, and we can thus ignore very small Fourier coefficients. To find a small set of the largest Fourier coefficients and hence a (near) optimal B-sparse Fourier representation of a signal that describes most of the signal characteristics is a fundamental task in applied Fourier Analysis.

An equivalent version of this problem is as follows: define the matrix $A := (e^{2\pi i k t_j})_{k=0,\ldots,N;\ j=0\ldots,L-1} \in \mathbb{C}^{L,N}$, where the t_j are the locations of the available samples. Given $S(t_j)$, we want to reconstruct the signal S, or equivalently, its Fourier coefficients $\hat{S}_k$, so that $A\hat{S} = S$. This linear system is under-determined. Several iterative algorithms [7][12][2][1][13] have provided efficient approaches to solve this problem. Among all INFFT algorithms, the iterative ACT (or equivalently CGNR) approach of [7] and CGNE algorithm with the fast Fourier Transforms at nonequsipaced nodes (NFFT) in [13] are among the fastest methods. The latter takes time $O(L^{1+(d-1)/\beta} \log L)$ to reconstruct the signal in the sense of weighted norm, where L is the number of available points, d is the number of dimensions, and $\beta > 1$ is the smoothness for the original signal. The super-linearity relationship between the running time and N (recall $L = pN$, where p is the percentage of available data) poses difficulties in processing large dimensional signals, which have nothing to do with the unequal spacing. It follows that identifying a sparse number of significant modes and amplitudes is expensive for even fairly modest N. Our goal in this paper is to discuss much faster (sublinear) algorithms that can identify the sparse representation or approximation with coefficients $a_1, \ldots, a_B$ and modes $\omega_1, \ldots, \omega_B$ for unevenly spaced data. These algorithms will not use all the samples $S(0), \ldots, S(N-1)$, but only a very sparse subset of them.

Our approach is based on the work presented in [8], that develops a theoretical algorithm how to construct the Fourier representation for a signal S with B-sparse Fourier representation in time and space $poly(B, \log N, 1/\epsilon, \log(1/\delta))$ on equal spacing data. The algorithm contains some random elements (which do not depend on the signal); their approach guarantees that the error of estimation is of order $\epsilon\|S\|_2^2$ with probability exceeding $1-\delta$. We have dubbed the whole family of algorithms RAℓSTA (for Randomized Algorithm for Sparse Transform Approximation); when dealing only with Fourier Transforms, as is the case here, we specialize it to RAℓSFA (F for Fourier). Zou, Gilbert, Strauss and Daubechies [21] presents a practical (and improved) implementation of the algorithm, showing that it is of interest, i.e. it outperforms FFT for reasonably large N. It convincingly beats FFT when the number of grid points N is reasonably large. The crossover point lies at $N \simeq 70000$ in one dimension, and at $N \simeq 900$ for data on a $N \times N$ grid in two dimensions for a eight-mode signal. When $B = 64$, RAℓSFA surpasses FFT at 3×10^7.

In this paper, we modify the RAℓSFA to solve the irregularly spaced data problem. The NERAℓSFA (Nonequispaced RAℓSFA) uses sublinear time and space $poly(B, \log L, \epsilon^{-1}, \log(1/\delta), \log(1/(1-p))$ to find a near-optimal B-term Fourier representation, such that $\|S - R\|_2^2 \leq (1+\epsilon)\|S - R_{opt}\|_2^2$ with high probability $1-\delta$. Similar to the RAℓSFA algorithm, it outperforms existing INFFT algorithms in processing sparse signals of large size.

Notation and Terminology Denote by χ_T a signal that equals 1 on a set T and zero elsewhere in the time domain. We say a signal S is q percent pure, if there exists a frequency ω and a signal ρ, such that $S = ae^{2\pi i\omega t/N} + \rho$, with $|a|^2 \geq (q\%)\|S\|_2^2$. If a signal is 90% pure, we call the frequency ω predominant. On the other hand, a frequency is significant, if $|\hat{S}(\omega)|^2 \geq \eta\|S\|_2^2$ for some constant $\eta \geq 0$. To quantify the unevenness of the data, we introduce a parameter $p = L/N$ to be the percentage of the available data over all the data, where L is the number of available data. Obviously a larger p corresponds to more information about the signal. We use ℓ^2-norm throughout the paper, which is denoted by $\|.\|_2$. The convolution $F * G$ is defined as $F * G(t) = \sum_s F(s)G(t-s)$. It follows that $\widehat{F * G}(\omega) = \sqrt{N}\hat{F}(\omega)\hat{G}(\omega)$.

A Box-car filter with width $2k+1$ is defined as follows:

$$\chi_k(t) \quad = \quad \begin{cases} \frac{\sqrt{N}}{2k+1} & \text{if } -k \leq t \leq k \\ 0 & \text{otherwise} \end{cases}$$

In the frequency domain, this filter is in the form of

$$\hat{\chi}_k(\omega) = \begin{cases} \frac{\sin((2k+1)\pi\omega/N)}{(2k+1)\sin(\pi\omega/N)} & \text{if } \omega \neq 0 \\ 1 & \text{otherwise} \end{cases} \tag{2}$$

A dilation operation on signal S with a dilation factor σ is defined as $S^{(\sigma)}(t) = S(\sigma t)$ for every point t.

Also, we define $S_1 = S\chi_T$, which is a new signal containing all the available information of S.

Organization The paper is organized as follows. In Section 2, we give the outline of the RAℓSFA algorithm. Section 3 presents the modification of RAℓSFA that deals with the unavailability of some samples by a greedy method. Finally, we compare numerical results with existing algorithms in Section 4.

2 Set-up of RAℓSFA

Given a signal S of length N, the optimal B-term Fourier representation $R^B_{opt}(S)$ uses only B frequencies; it is simply a truncated version of the Fourier representation of S, retaining only the B largest coefficients. The following theorem is the main result of [8].

Theorem 2.1. [8] *Let an accuracy factor ϵ, a failure probability δ, and a sparsity target $B \in \mathcal{O}(1), B \ll N$ be given. Then for an arbitrary signal S of length N, RAℓSFA will find a B-term approximation R to S, at a cost in time and space of order $poly(B, \log(N), 1/\epsilon, \log(1/\delta))$ and with probability exceeding $1-\delta$, so that $\|S-R\|_2^2 \leq (1+\epsilon)\|S - R^B_{opt}(S)\|_2^2$.*

The striking fact is that RAℓSFA can build a near-optimal representation R in sublinear time $poly(\log N)$ instead of the $O(N \log N)$ time requirement of other algorithms. Its speed surpasses FFT as long as the length of a signal is sufficiently large. If a signal is composed of only B modes, RAℓSFA constructs S without any error.

The main procedure is a Greedy Pursuit [21] with the following steps:

Algorithm 2.2. [8]TOTAL SCHEME
Input: signal S, the number of nonzero modes B or its upper bound, accuracy factor ϵ, success probability $1-\delta$, the standard deviation of the white Gaussian noise σ, a small number ξ for relative precision.

1. *Initialize the representation signal R to 0, set the maximum number of iterations $T = B\log(N)\log(1/\delta)/\epsilon^2$,*
2. *Test whether either $\|S-R\| \leq \xi\|R\|_2^2$. If yes, return the representation signal R and the whole algorithm ends; else go to step 3.*
3. *Locate Fourier Modes ω for the signal $S-R$ by the isolation and group test procedures below.*
4. *Estimate Fourier Coefficients at ω: $(\widehat{S-R})(\omega)$.*
5. *If the total number of iterations is less than T, go to 2; else return the representation R.*

The greedy pursuit procedure captures one significant frequency each time and then reduces the contribution of this frequency from the residual signal. In order to make this clear, we give an illustration about how to capture the B Fourier modes in successive sweeps. Note as a randomized algorithm, NERAℓFA has different performance in each run. Suppose the signal is $S = 100\phi_1 + 1.45\phi_2 + 1.4\phi_3 + \phi_4$, where $\phi_k = e^{2\pi ikt/N}$. We try to find a 2-term Fourier approximation. One possible run is as follows:

1. 1 Identify frequency $\omega = 1$; estimate $\hat{S}(1) = 102$, (not 100)
   Update: $R = 102\phi_1$, residual $S = original - R = -2\phi_1 + 1.45\phi_2 + 1.4\phi_3 + \phi_4$.
2. 2 Next, identify $\omega = 1$, estimate $\hat{S}(1) = -1.9$, $R = 100.1\phi_1$, and $S = -0.1\phi_1 + 1.45\phi_2 + 1.4\phi_3 + \phi_4$.

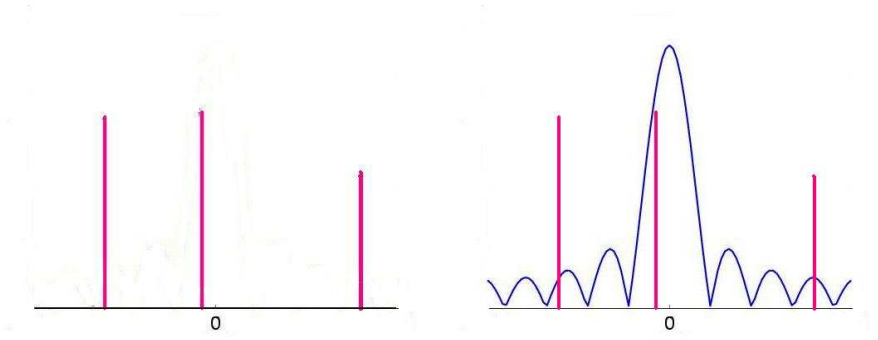

Figure 1: The graph illustration of the isolation procedure. Left: a signal with three significant modes. Right: Multiply the signal with a box-car filter in the frequency domain.

3. 3 Next, the frequency identification made some errors. Instead of the Fourier mode with the second largest coefficient, it identifies $\omega = 3$, estimate $\hat{S}(3) = 1.38$, $R = 100.1\phi_1 + 1.38\phi_3$, and $S = -0.1\phi_1 + 1.45\phi_2 + 0.02\phi_3 + \phi_4$.

From the above example, we notice there are errors in both frequency location and coefficient estimation. This is because we only use partial information about the signal. Fortunately, the greedy pursuit would help to correct the errors gradually. The final result is as follows.

1. Original signal $S = 100\phi_1 + 1.45\phi_2 + 1.4\phi_3 + \phi_4$.
2. Representation signal $R = 100.1\phi_1 + 1.38\phi_3$, error =$0.1^2 + 1.45^2 + 0.02^2 + 1^2 = 3.1129$.
3. Optimal representation signal $R_{opt} = 100\phi_1 + 1.45\phi_2$, optimal error $= \|S - R_{opt}\|_2^2 = 1.4^2 + 1^2 = 2.96$. Obviously, the error is within the $(1 + \epsilon)$ scope of the optimal error.

The most important part of the RAℓSFA [8, 21] is to identify significant frequencies and estimate their corresponding coefficients. In order to locate those nonzero frequencies, we first construct a new signal where a previous significant frequency becomes predominant, (see the first graph of Figure 1). This is implemented by convolving the original signal with a box-car filter, which is equivalent to the multiplication of both signals in the frequency domain, as shown the second graph of Figure 1. Therefore, we obtain a new signal with only one dominant frequency and other frequencies' amplitudes are very small.

Then a recursive approach called group test finds the exact label of this predominant mode, by splitting intervals, comparing energies, and keeping only intervals with large energies. After the frequency is located, the algorithm gives a good estimation of the coefficient, by taking means and medians of random samples.

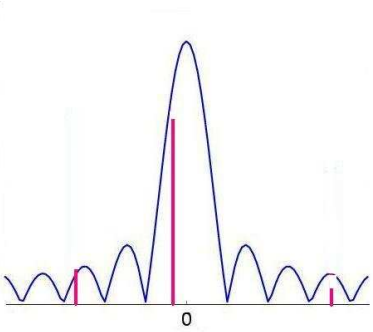

Figure 2: The graph illustration of the isolation procedure. After the multiplication, if the significant frequency falls in the pass region, most of its amplitudes is preserved. Otherwise, their amplitudes diminish. Hence, we obtain a new signal with only one predominant frequency.

3 NERAℓSFA with Greedy Technique

RAℓSFA samples from a signal with a fixed cost per sample, implicitly assuming that uniform and random sampling is possible. This raises challenges for processing unevenly spaced data. Specifically speaking, coefficients and norms can not be estimated properly. Thus one has to modify steps 3 and 4 of Algorithm 2.2 accordingly. In this section, NERAℓSFA, a modified version of RAℓSFA with greedy technique, is introduced to overcome these problems.

The basic new ideas, which distinguish NERAℓSFA from RAℓSFA, are a greedy pursuit from available data points or Lagrange interpolation to estimate the values of an unavailable data. We propose two different approaches here for processing an unavailable data. Both of them adapt greedy pursuit for an available data in estimating coefficients. They are distinct in norm estimation-the first method would search exhaustively for an available data point $S(t)$ as the substitute by generating random indices; in contrast, the second method takes Lagrange interpolation of its three nearest neighbors to estimate the value of the missing point.

A good data structure is important to save the running time cost. We denote the availability of a data point by a label, say +1 for available and 0 for unavailable. Hence, in order to check if its corresponding sample is valid, we only need to look at the value of the label. An alternative solution is to store all the sorted labels of available data in a long list. However, each search takes time $O(\log(N))$, which introduces a $O(\log N)^2$ factor into the whole computation. As the empirical results show, the running time of NERAℓSFA algorithm is linear to $\log N$. For this reason, we selected the first data structure.

We now give a more detailed discussion of the different procedures used in steps 3 and 4 of Algorithm 2.2; these correspond to the procedures detailed in [21]. For the sake of completeness, we give the full description of the adapted algorithm (rather than lost only the changes with respect to [21]) for the 1-dimensional case. It is an adaption of algorithms in [21], along the lines sketched above.

3.1 Estimating Fourier Coefficients

First, we give the procedure for estimating Fourier coefficients for unevenly spaced data as follows, which is very similar to Algorithm 3.3 in [21], except its greedy pursuit for an available data.

Algorithm 3.1. ESTIMATING INDIVIDUAL FOURIER COEFFICIENTS
Input: a signal S, a frequency ω, failure probability δ, accuracy factor ϵ.

Initialize: $n = \lfloor 2\log(1/\delta) \rfloor$, $m = \lfloor 8/\epsilon^2 \rfloor$.

1. *For* $i = 1, \ldots, n$
   For $j = 1, \ldots, m$
   Randomly generate the index t *until* $S(t)$ *is available.*
   Then let $t_{ij} = t$. *Evaluate* $k(t_{ij}) =< S(t_{ij}))\delta_{t_{ij}}, \phi_\omega(t_{ij}) >$.
2. *Take the means of* m *samples* $k(t_{ij})$, *i.e.* $p(i) = \sum_{j=1}^{m} k(t_{ij})$, *where* $i = 1, \ldots, n$.
3. *Take the median of* n *samples* $c = median_i(p(i))$, *where* $i = 1, \ldots, n$.
4. *Return* c *as the estimation of the Fourier coefficient* $\hat{S}(\omega)$.

Because of the high accuracy requirement of coefficient estimation, we typically only choose greedy pursuit for a satisfactory data. Similar to the observation in [21], fewer samples are already able to give an estimation with desirable accuracy and probability. Instead of the theoretical $16\epsilon^{-2}|\log(\delta)|$ samples requirement per coefficient, 150 samples per coefficient already obtain the relative accuracy $\epsilon = 10^{-4}$. Also note that the multiplication $k(t_{ij}) =< S(t_{ij})\delta_{t_{ij}}, \phi_\omega(t_{ij}) >= S(t_{ij})e^{2\pi\omega t_{ij}/N}$ can be easily computed, assume ω is already known.

Next, we show that using unevenly spaced data leads to a very good approximation to the true coefficient. We know that by repeating an experiment enough times, a small probability event will happen eventually. One easily checks that, in our case, only $p = L/N$ percentage of the data is available, so that $k > |\log\delta|/\log(1 - L/N)$ trials are needed to generate one available data point with success probability at least $1 - \delta$.

We shall also need the observation that most of the Fourier coefficients of a characteristic function on a typical set T are small, under some conditions. The following lemma makes this more explicit.

Lemma 3.2. *Suppose the components X_j of a discrete random variable $X = (X_j)_{j=0}^{N-1}$ are identically and independently distributed in $\{0, 1\}$, with $p = Prob(X_j = 1)$. Define the random set $T = \{j \in \{0, \ldots, N-1\} | X_j = 1\}$ to be the set of all available data; $\hat{\chi}_T(\omega) = \frac{1}{\sqrt{N}} \sum_{t\in T} e^{2\pi\omega t/N}$ is the discrete Fourier transform. If $\lambda < 2e\mu$, $p \geq 1 - \frac{\lambda^2(1-1/N)}{2\lambda+4|\log\tau|)}$, where $e = 2.71828$, then*

$$Prob(|\hat{\chi}_T(\omega)|^2 \geq \lambda) \leq \tau^2. \tag{3}$$

Proof. First, we claim that $E(|\hat{\chi}_T(\omega)|^2) = \frac{p(1-p)N}{(N-1)}$ for all $\omega \neq 0$. We have

$$|\hat{\chi}_T(\omega)|^2 = \frac{1}{N}\sum_{j,k\in T} e^{2\pi i\omega(j-k)/N} \tag{4}$$
$$= \frac{1}{N}\sum_{j\in T} 1 + \frac{1}{N}\sum_{j,k\in T, j\neq k} e^{2\pi i\omega(j-k)/N}.$$

It follows that

$$E(|\hat{\chi}_T(\omega)|^2) = p + \frac{1}{N}p\frac{pN-1}{N-1}\sum_{j,k=0,j\neq k}^{N-1} e^{2\pi i\omega(j-k)/N}.$$

Observe that $\sum_{j,k=0,j\neq k}^{N-1} e^{2\pi i\omega(j-k)/N} = |\sum_{j=0}^{N-1} e^{2\pi i\omega j/N}|^2 - \sum_{j=0}^{N-1} 1 = (N\delta_{\omega,0})^2 - N$, hence

$$E(|\hat{\chi}_T(\omega)|^2) = p + \frac{p}{N}\frac{pN-1}{N-1}(N^2\delta_{\omega,0} - N) = p\left\{1 + \frac{pN-1}{N-1}(N\delta_{\omega,0} - 1)\right\}$$
$$= \frac{p}{(N-1)}\left\{N - 1 + (pN-1)(N\delta_{\omega,0} - 1)\right\}.$$

By Chernoff's Inequality [5], since $\lambda < 2e\mu$,

$$Prob(|\hat{\chi}_T(\omega)|^2 \geq \lambda) \leq exp\left[-\frac{(\lambda/E(|\hat{\chi}_T(\omega)|^2) - 1)^2 E(|\hat{\chi}_T(\omega)|^2)}{4}\right] \tag{5}$$
$$= exp\left[-\frac{1}{4}\left(\lambda^2/E(|\hat{\chi}_T(\omega)|^2) - 2\lambda + E(|\hat{\chi}_T(\omega)|^2)\right)\right]$$

In order for $Prob(|\hat{\chi}_T(\omega)|^2 \geq \lambda) \leq \tau^2$, it suffices that

$$\lambda^2/E(|\hat{\chi}_T(\omega)|^2) - 2\lambda + E(|\hat{\chi}_T(\omega)|^2) \geq 4|\log\tau^2| \tag{6}$$

The above inequality holds if

$$\lambda^2/E(|\hat{\chi}_T(\omega)|^2) \geq \left(2\lambda + 4|\log\tau^2|\right) \tag{7}$$

Therefore,

$$E(|\hat{\chi}_T(\omega)|^2) \leq \frac{\lambda^2}{2\lambda + 4|\log\tau^2|} \tag{8}$$

Since $E(|\hat{\chi}_T(\omega)|^2) = \frac{p(1-p)N}{N-1}$, we only need $p(1-p) \leq \frac{\lambda^2(1-1/N)}{2\lambda+4|\log\tau^2|}$. Therefore, when $p \geq 1 - \frac{\lambda^2(1-1/N)}{2\lambda+4|\log\tau|)}$,

$$Prob(|\hat{\chi}_T(\omega)|^2 \geq \lambda) \leq \tau^2.$$

□

In particular, we want both λ and τ to be small, meaning that p cannot be too small itself.

Next, we consider the conditions for the two coefficients $\hat{S}(\omega)$ and $\hat{S}_1(\omega) = \widehat{S \cdot \chi_T}(\omega)$ to be close.

Lemma 3.3. *Suppose the parameters T, S, $\chi_T(t)$, λ, τ, p are as stated in Lemma 3.2, and define $S_1(t) = S(t)\chi_T(t)$. If $p \geq 1 - \frac{\lambda^2(1-1/N)}{2\lambda+4|\log\tau|)}$, and $\tau \leq \sqrt{1-(1-\delta)^{\frac{1}{B}}}$, $\sqrt{B\lambda} < 1$, then, for any ω,*

$$|\hat{S}(\omega) - \hat{S}_1(\omega)| \leq \sqrt{B\lambda}\|S\|_2. \tag{9}$$

with probability exceeding $(1-\tau^2)^B$.

Proof. Suppose the significant terms of signal S are ω_i, where $i = 1, \ldots, B$.
Since $S_1(t) = S(t)\chi_T(t)$ and thus $\hat{S}_1(\omega) = \hat{S}(\omega) * \hat{\chi}_T(\omega)$, then

$$\begin{aligned}\hat{S}_1(\omega) = \sum_{i=1}^{B} \hat{S}(\omega_i)\hat{\chi}_T(\omega-\omega_i) &= \hat{S}(\omega)\hat{\chi}_T(0) + \sum_{i=1,\omega\neq\omega_i}^{B} \hat{S}(\omega_i)\hat{\chi}_T(\omega-\omega_i) \\ &= \hat{S}(\omega) + \sum_{i=1,\omega\neq\omega_i}^{B} \hat{S}(\omega_i)\hat{\chi}_T(\omega-\omega_i).\end{aligned}$$

Therefore

$$|\hat{S}_1(\omega) - \hat{S}(\omega)| = |\sum_{i=1,\omega\neq\omega_i}^{B} \hat{S}(\omega_i)\hat{\chi}_T(\omega-\omega_i)| \tag{10}$$

$$\leq \sqrt{\sum_{i=1,\omega\neq\omega_i}^{B} |\hat{S}(\omega_i)|^2}\sqrt{\sum_{i=1,\omega\neq\omega_i}^{B} |\hat{\chi}_T(\omega-\omega_i)|^2} \leq \|S\|_2\sqrt{\sum_{i=1,\omega\neq\omega_i}^{B} |\hat{\chi}_T(\omega-\omega_i)|^2}.$$

Hence, we have $|\hat{\chi}_T(\omega)|^2 \leq \lambda$ with probability at least $1-\tau^2$ for any $\omega \neq 0$. This implies that $|\hat{S}_1(\omega) - \hat{S}(\omega)| \leq \sqrt{B\lambda}\|S\|_2$ with probability at least $(1-\tau^2)^B$.
For those $\omega \notin \{\omega_i, i = 1, \ldots, B\}$,

$$\hat{S}_1(\omega) = \sum_{i=1}^{B} \hat{S}(\omega)\hat{\chi}_T(\omega-\omega_i), \tag{11}$$

and we conclude similarly that $|\hat{S}_1(\omega) - \hat{S}(\omega)| \leq \sqrt{B\lambda}\|S\|_2.$, with probability at least $(1-\tau^2)^B$. □

We shall use Algorithm 3.1 to estimate $\hat{S}_1(\omega)$; we now look at how close the approximation A (i.e. the output of Algorithm 3.1) of $\hat{S}_1(\omega)$ is to the true coefficient $\hat{S}(\omega)$.

Theorem 3.4. *For a set of parameters T, S, $\chi_T(t)$, λ, τ, p as stated in Lemma 3.2, if $p \geq 1 - \frac{\lambda^2(1-1/N)}{2\lambda+4|\log\tau|)}$, then every application of Algorithm 3.1 produces, for each frequency ω and each signal S, and each $\lambda > 0$, with high probability exceeding $(1-\delta)(1-\tau^2)^B$, an output A (after inputting $(S, \omega, \epsilon, \delta)$), such that $|A - \hat{S}(\omega)|^2 \leq \lambda(1+\sqrt{B})^2\|S\|_2^2$.*

Proof. Lemma 4.2 in [21] says that the coefficient estimation algorithm returns A, such that

$$|A - \hat{S}_1(\omega)| \leq \sqrt{\lambda}\|S\|_2. \tag{12}$$

By Lemma 3.3

$$|\hat{S}_1(\omega) - \hat{S}(\omega)| \leq \sqrt{B\lambda}\|S\|_2. \tag{13}$$

Thus

$$|A - \hat{S}(\omega)| \leq |A - \hat{S}_1(\omega)| + |\hat{S}_1(\omega) - \hat{S}(\omega)| \leq (\sqrt{\lambda} + \sqrt{B\lambda})\|S\|_2. \tag{14}$$

From $(\sqrt{\lambda} + \sqrt{B\lambda})^2 \leq 2\lambda(B+1)$, it follows that

$$|A - \hat{S}(\omega)|^2 \leq \lambda(1 + \sqrt{B})^2\|S\|_2^2. \tag{15}$$

□

When we are able to get most of the data, the computational cost for estimating Fourier coefficients on unevenly spaced data is only slightly more than for the evenly spaced data case.

The time to compute the available signal value remains the same as for the evenly spaced data case. The extra time comes from visiting unavailable data.

Lemma 3.5. *Algorithm 3.1 takes time $c\frac{\log(1/\delta)\log\delta_1}{\epsilon^2\log(1-p)}$ in the worst case to pursue $c\frac{\log\delta}{\epsilon^2}$ available data with success probability greater than $(1-\delta_1)^{c\frac{\log\delta}{\epsilon^2}}$.*

Proof. We know that only $p = \frac{L}{N}$ data are available. In order to achieve an available data with probability $1 - \delta_1$ in k trials, we have

$$(1-p)^k \leq \delta_1$$

which implies $k \geq \frac{\log\delta_1}{\log(1-p)}$. That is, for obtaining one available data with probability $1-\delta_1$, we need at most $\frac{\log\delta_1}{\log(1-p)}$ trails. According to Lemma 3.4 in [21], we hope that $c\frac{\log(1/\delta)}{\epsilon^2}$ samples of $S(t)$ are able to produce a satisfactory estimation of $\hat{S}(\omega)$, where c is some constant. Hence, the total number of trials to obtain $c\frac{\log\delta}{\epsilon^2}$ available samples would be $c\frac{\log(1/\delta)\log\delta_1}{\epsilon^2\log(1-p)}$ in the worst case, with success probability exceeding $(1-\delta_1)^{c\frac{\log\delta}{\epsilon^2}}$. □

Fortunately, the operation of visiting samples is very fast and therefore contributes little to the total time, especially when most of the data are available.

The theoretical condition of Theorem 3.4 is very restricted. Fortunately, heuristically, this theorem holds for much broad situations.

Remark: as in [21], one can speed up the algorithm by using multi-step coarse-to-fine coefficient estimation procedures, which turns out to be more efficient than single-step accurate estimation; the proof is entirely analogous to Lemma 4.3 in [21].

3.2 Estimating Norms

The basic idea for locating the label of a significant frequency is to compare the energies (i.e. the L^2 norm) of signals restricted mostly in different frequency intervals. If the energy of some interval is relatively large, the significant mode would be in that region with higher probability. We construct the following new signals to focus on certain intervals

$$H_j(t) = \chi_1(t) e^{\frac{2\pi i j t}{4(2k+1)}} * \chi_{[-q_1,q_1]}(\sigma t) e^{\frac{2\pi i t \theta}{N}} * (S \cdot \chi_T) \tag{16}$$

where $2q_1 + 1$ is the filter width, $j = 0, \ldots, 8k + 4$, σ and θ are random dilation and modulation factors. (Please see [21] for an explanation of the role of σ and θ). For convenience, we denote $H_j(t)$ by $H(t)$.

We need to evaluate values of $H(t)$ for random indices $t \in \{0, \ldots, N-1\}$. Note that the signal H results from the convolutions of two finite bandwidth Box-car filters with the partial information of the original signal S. Therefore, any missing point needed by the two convolutions would lead to a failure of computing $H(t)$. The total number of signal points involved depends on the number of nonzero taps in these two filters. Moreover, random dilation and modulation factors of the second Box-car filter make computation more tricky. In this subsection, we propose two different approaches to estimate norms.

3.2.1 Greedy Pursuit for an Available Data Point

One naive way is to dive into the two convolutions and sample each related signal point. If it is not available, stop evaluating this $H(t)$ and start with a new index t. This definitely increases time cost by wasting abundant computation. For example, suppose five data are needed and only one of them is missing, then the algorithm may compute four data in vain in the worst case, where the missing data point is visited last in the sequence of 5.

To avoid the above situation, we first compute the locations of all the relevant points for the convolution; only if they are all available will we start the computation. The following lemma presents the rules of these location of points after random permutation.

Lemma 3.6. *Suppose we have a signal $H(t) = (\chi_1^{(\sigma_1)} * (\chi_{q_1}^{(\sigma_2)} * S)^{(\sigma_3)})^{(\sigma_4)})(t)$, where σ_1, σ_2, σ_3, and σ_4 are dilation factors. From the definition of Box car filter, the taps for χ_{q_1} lies in the interval $[-q_1, q_1]$, the taps for χ_2 in $[-q_2, q_2]$, then in order to evaluate $H(t)$, we need values of S with indices at $\sigma_3\sigma_4 t - \sigma_3\sigma_1 i - j\sigma_2$, where integers $i = q_1, \ldots, q_1$, $j = -q_2, \ldots, q_2$.*

Proof. To evaluate $H(t)$, first let signal $r = (\chi_{q_1}^{(\sigma_2)} * S)^{(\sigma_3)}$, then

$$H(t) = (\chi_{q_1}^{(\sigma_1)} * r)^{(\sigma_4)}(t) = \sum_{i=-q_1}^{q_1} \chi_{q_1}(\sigma_1 i) r(\sigma_4 t - \sigma_1 i) \tag{17}$$

where

$$r(\sigma_4 t - \sigma_1 i) = (\chi_{q_2}^{(\sigma_2)} * S)^{(\sigma_3)}(\sigma_4 t - \sigma_1 i) = (\chi_{q_2}^{(\sigma_2)} * S)(\sigma_3\sigma_4 t - \sigma_3\sigma_1 i)$$
$$= \sum_{j=-q_2}^{q_2} \chi_{q_2}(\sigma_2 j) S(\sigma_3\sigma_4 t - \sigma_3\sigma_1 i - \sigma_2 j). \quad (18)$$

Thus, in order to get the value of $H(t)$, we need values of all $S(t^{'})$, where $t^{'} = \sigma_3\sigma_4 t - \sigma_3\sigma_1 i - \sigma_2 j$, with $i = -q_1, \ldots, q_1$ and $j = -q_2, \ldots, q_2$. □

The scheme of the norm estimation algorithm is as follows. It is a variation form of Algorithm 3.6 in [21], with an added feature of exhaustive trials to obtain available data.

Algorithm 3.7. NORM ESTIMATION
Input: signal H, the failure probability δ
Initialize: the number of iterations $M = \lfloor 1.2 \ln(1/\delta) \rfloor$, $k = 0$.
While $k < M$:

1. *Randomly generate the index t_k.*
2. *Compute all indices needed by computing $H(t)$ in (16): $\Upsilon = \{t^{'} | t^{'} = \sigma_3\sigma_4 t - \sigma_3\sigma_1 i - \sigma_2 j, i = -1, 0, 1, j = -q_1, \ldots, q_1\}$.*
3. *If all the points $t^{'} \in \Upsilon$ are available, then compute $H(t_k)$ and $k = k + 1$; else go to step 1 and generate another index t_k.*
4. *estimate = 60-th percentile of the sequence $\{|H(t_k)|^2 N\}$, where $k = 0, \ldots, M - 1$.*

The following lemma investigates the number of repetitions to get a satisfactory data group for estimating norms.

Lemma 3.8. *Suppose χ_{q_1} and χ_{q_2} are two Box-car filters with numbers of taps $2q_1 + 1$ and $2q_2 + 1$ respectively. Define $D_{q_1,q_2} = \chi_{q_1} * \chi_{q_2}$. Then D_{q_1,q_2} has $2q_1 + 2q_2 + 1$ nonzero taps in the time domain.*

Lemma 3.9. *Randomly choose an index t for signal H, then after repetition for $k > \log \delta_2 / \log(1 - (1-p)^{2q_1+2q_2+1})$ times, we can at least find one computable $H(t)$ with all available points, with success probability $1 - \delta_2$.*

Proof. Randomly generate a sample point t. In order to compute $H(t)$, we need all the related $2q_1 + 2q_2 + 1$ points available. This would happen with probability $p^{2q_1+2q_2+1}$. Suppose the failure probability to find such an available data group is smaller than δ_2, and k is the number of trails for t, we have

$$(1 - p^{2q_1+2q_2+1})^k \leq \delta_2$$

which implies $k \geq \frac{\log \delta_2}{\log(1-p^{2q_1+2q_2+1})}$. □

If there exist satisfactory data groups, although maybe very few, the norm estimation will eventually find them. However, when most data are unavailable, the program may try repeatedly and take a huge amount of time. To avoid this situation, we introduce an upper bound on the number of the trials. When the bound is reached, just use the available sample points found so far to estimate the norms. This technique probably leads to a larger error, and thus hamper the success of correct frequency identification. However, by repeating the whole greedy pursuit process, the failure probability would be further reduced. Anyway we cannot hope to recover the signal, if there are very few available data.

The greedy algorithm described above is fast when p is sufficiently large (e.g. $p > 0.7$). For smaller p, the amount of time wasted to find available sample groups becomes unacceptably long. For example, when $B = 2$, $N = 100$, $p = 0.4$, the algorithm couldn't find the signal within 200 greedy pursuit iterations. This motivates us to use an interpolation technique for estimating the samples directly.

3.2.2 Lagrange Interpolation Techniques for Evaluating a Sample

Because the signals we seek to represent are supposed to have only very few modes and therefore very smooth, a Lagrange interpolation from 3 available neighbors sandwiching the unavailable point seemed an appropriate choice. We introduce the interpolation scheme only into estimating norms. It turns out that the resulting algorithm is more efficient and more successful in smaller p cases.

The idea is to estimate the value of a missing point by the Lagrange interpolation [18]. Suppose the three nearest neighbors of t are $(t_i(t), H(t_i))$, where $i = 1, 2, 3$. Then the Lagrange interpolation to approximate the value of $H(t)$ is

$$P(x) = \frac{(x - x_2)(x - x_3)}{(x_1 - x_2)(x_1 - x_3)} y_1 + \frac{(x - x_1)(x - x_3)}{(x_2 - x_1)(x_2 - x_3)} y_2 + \frac{(x - x_2)(x - x_1)}{(x_3 - x_2)(x_3 - x_1)} y_3 \qquad (19)$$

Since this sample interpolation procedure was already good enough to improve the practical performance of NERAℓSFA, we didn't try more fancier interpolation techniques. The detailed algorithm for estimating norms is as follows.

Algorithm 3.10. ESTIMATE NORM WITH INTERPOLATION TECHNIQUE
Input: signal H.
Initialize: $k = 0$, the maximum number of samples M.

1. *Randomly generate the index t_k, where $k = 0, \ldots, M - 1$.*
2. *For each k, if $H(t_k)$ is not available, estimate $H(t_k)$ by Lagrange interpolation; else compute $H(t_k)$ directly.*
3. *Estimation = 60-th percentile of the sequence $\{|H(t_k)|^2 N\}$, where $k = 0, \ldots, M - 1$.*

Note that we use interpolation *only* in norm estimation steps, where precision is less critical. With less precise norm estimation, the localization of important modes can still work well when

iterated. For coefficient estimation, which needs to be more precise, we always search for available samples.

Here is a new scheme for estimating norms, which uses much fewer samples than the original one and still achieves good estimation. In [21], we propose Lemma 3.7 that enabled us to achieve a good norm estimation by only a few samples. The following lemma is its adaption to the case of unevenly spaced data.

Lemma 3.11. *Suppose a signal S is 93% pure. Let three available nearest neighbors of an unavailable point t denoted by $t_i(t)$, where $i = 1, 2, 3$. For each t, there exists some number $M(t)$, such that $|f^{(3)}(t)| \leq M(t)$. Let $A = \frac{1}{6}M(t)(t - t_1(t))(t - t_2(t))(t - t_3(t))$. If $|A| \leq min(0.075\|S\|^2/\sqrt{N}, |S(t)|)$ for all t, the number of samples $r > [\ln(1.96C^{0.6}(1 - C)^{0.4})]^{-1} \ln(1/\delta)$, where $C = \frac{0.07}{(\sqrt{0.93}-\sqrt{0.3}-A\sqrt{N}/\|_2 S\|_2)^2}$, the output of Algorithm 3.10 gives an estimation of its energy which exceeds $0.3\|S\|_2^2$ with probability exceeding $1 - \delta$.*

Proof. Suppose the signal $S = a\phi_\omega + e$, where $|a|^2 > 0.93\|S\|_2^2$. We shall sample the signal S independently for r times. For those unavailable $S(t)$, we use Lagrange interpolation to substitute its value. Therefore, the sample at t is defined as follows.

For convenience, let t_1, t_2, and t_3 denote $t_1(t)$, $t_2(t)$, and $t_3(t)$.

$$P(t) = \begin{cases} S(t) & \text{if } S(t) \text{ is available,} \\ S(t) - \frac{f^{(3)}(\xi)}{6}(t - t_1)(t - t_2)(t - t_3) & \text{otherwise.} \end{cases}$$

as stated in Algorithm 3.10. For convenience, let $A = \frac{f^{(3)}(\xi)}{6}(t - t_1)(t - t_2)(t - t_3)$. Let

$$\begin{aligned} T &= \{t : N|P(t)|^2 < 0.3\|S\|_2^2\} \\ &= \{t : \sqrt{N}|P(t)| \leq \sqrt{0.3}\|S\|_2\} \\ &= \{t : \sqrt{N}|S(t) - A| \leq \sqrt{0.3}\|S\|_2\} \end{aligned}$$

Suppose $|S(t)| > |A|$, then

$$\begin{aligned} T \subset T_1 &= \{t : \sqrt{N}|S(t)| - |A| \leq \sqrt{0.3}\|S\|_2\} \\ &= \{t : \sqrt{N}|S(t)| \leq \sqrt{0.3}\|S\|_2 + A\sqrt{N}\} \\ &= \{t : \sqrt{N}|S(t)| \leq \left(\sqrt{0.3} + A\sqrt{N}/\|S\|_2\right)\|S\|_2\} \end{aligned}$$

Also by the purity of S, we have $\|e\|_2^2 \leq 0.07\|S\|_2^2$. Since $|S(t)| \geq |a\phi_\omega(t)| - |e(t)|$, we obtain

$$\sqrt{N}|e(t)| > a - \sqrt{N}|S(t)|. \tag{20}$$

then for any $t \in T$,

$$\sqrt{N}|e(t)| > \sqrt{0.93} - \sqrt{0.3} - A\sqrt{N}/\|S\|_2.$$

By the similar procedure in Lemma 3.7 in [21], it follows that

$$0.07N \geq N\|e\|_2^2 \geq N\sum_{t\in T}|e(t)|^2 \geq (\sqrt{0.93} - \sqrt{0.3} - A\sqrt{N}/\|S\|_2)^2|T|$$

Hence,

$$|T| \leq \frac{0.07}{(\sqrt{0.93} - \sqrt{0.3} - A\sqrt{N}/\|S\|_2)^2} N \tag{21}$$

Since $\alpha = \frac{|T|}{N}$, it implies that $\alpha < \frac{0.07}{(\sqrt{0.93} - \sqrt{0.3} - A\sqrt{N}/\|S\|_2)^2}$.

$$Prob\left(60\text{-}th\, percentile < 0.3\|S\|_2^2\right) \leq 1.96\alpha^{0.6}(1-\alpha)^{0.4} \tag{22}$$

The right hand side of (22) is increasing in α on the interval $[0, 0.6]$. Similar to Lemma 3.7 in [21], we want

$$\frac{0.07}{(\sqrt{0.93} - \sqrt{0.3} - A\sqrt{N}/\|S\|_2)^2} < 0.6. \tag{23}$$

Which can be satisfied by the condition $A < 0.075\|S\|_2/\sqrt{N}$. Also,

$$\left[(1-\alpha)e^{-0.6z} + \alpha e^{0.4z}\right]^r = \left[1.96\alpha^{0.6}(1-\alpha)^{0.4}\right]^r \leq e^{-Wr}$$

where $C = \frac{0.07}{(\sqrt{0.93} - \sqrt{0.3} - A\sqrt{N}/\|S\|_2)^2}$. So for $r \geq [\ln(1.96C^{0.6}(1-C)^{0.4})]^{-1} \ln(1/\delta)$, we have

$$\begin{aligned} Prob(Output\ of\ this\ algorithm \geq 0.3\|S\|_2^2) &= Prob(60\text{-}th\ percentile\ of\ N|S(t)|^2 \geq 0.3\|S\|_2^2) \\ &\geq 1-\delta. \end{aligned}$$

□

This norm estimation procedure will be used repeatedly in the group testing step below. It is true that the approximation effect of the interpolation remains open. Nevertheless, since only a rough estimation of the norms are desired for the Group test, the Lagrange interpolation could serve for this purpose in most cases. Also, we introduce certain tricks to avoid large errors. For example, if three neighboring points are too far away from the sample point, we will choose another $H(t)$.

3.3 Isolation

For a significant frequency in signal S, isolation aims to construct a series of new signals, such that this significant frequency becomes predominant in at least one of the new isolation signals. The following lemma is similar to Lemma 3.10 in [21].

Lemma 3.13. *Given signals S, S_1, and the parameters as stated in Lemma 3.2. Suppose $F_1^{(j)} = H_k * R_{\theta_j,\sigma_j}(S_1) = H_k * R_{\theta_j,\sigma_j}(S\chi_T)$, $F^{(j)} = H_k * R_{\theta_j,\sigma_j}(S)$, where $j = 0, \ldots, \log(1/\delta)$. If $p \geq 1 - \frac{\lambda^2(1-1/N)}{2\lambda + 4|\log\tau|)}$, $\eta \leq 1/B$ and $k \geq 12.25((B-1)\pi^2/\eta$, then for each ω with $|\hat{S}(\omega)|^2 > B\lambda\|S\|_2^2$, isolation algorithm can create a signal F_1^*, such that*

$$|\hat{F}_1^*(\omega)|^2 \geq 0.98\|F_1^*\|_2^2. \tag{24}$$

Proof. Since $|\hat{S}(\omega)|^2 > B\lambda\|S\|_2^2$, we have $|\hat{S}(\omega)| > \sqrt{B\lambda}\|S\|_2$. Then there exists some $\eta > 0$, such that $|\hat{S}(\omega)| \geq (\sqrt{\eta} + \sqrt{B\lambda})\|S\|_2$. Lemma 3.3 states that $|\hat{S}_1(\omega) - \hat{S}(\omega)| \leq \sqrt{B\lambda}\|S\|_2$. Therefore

$$|\hat{S}_1(\omega)| \geq \sqrt{\eta}\|S\|_2 \geq \sqrt{\eta}\|S_1\|_2. \tag{25}$$

Isolation algorithm returns $F_1^{(j)}$, as described in Lemma 3.9 in [21]. For any ω with $|\hat{S}_1(\omega)|^2 \geq \eta\|S_1\|_2^2$, there exists some j, such that

$$|\hat{F}_1^{(j)}(\omega)|^2 \geq 0.98\|F_1^{(j)}\|_2^2. \tag{26}$$

Let $F_1^* = F_1^{(j)}$, then

$$|\hat{F}_1^*(\omega)|^2 \geq 0.98\|F_1^*\|_2^2. \tag{27}$$

□

Not all of the F_is are 98% pure. Since it is difficult to discover its extent of purity, we shall apply the further steps of the algorithm to each of the F_i. A not-so-pure F_i may identify some insignificant mode $\tilde{\omega}$, which can be detected by the estimation of its coefficient. In the end, we only output those significant modes.

The isolation procedure construct a new signal, where a frequency ω with $|\hat{S}(\omega)|^2 \geq \|S\|^2/B$ would become dominant. This does not mean that NERAℓSFA can only find those large amplitude frequencies. Because the residual signal is updated by reducing the contribution of the relatively large frequencies, a previously small amplitude modes becomes more important gradually. The criteria of $|\hat{S}(\omega)|^2 \geq \|S\|^2/B$ would be eventually satisfied if the energy of the mode is well beyond that of the white noise.

3.4 Group Testing

The goal of group testing is thus to find the most significant mode of the signal F_1^* from isolation. It repeatedly zooms in the signal, and employ MSB (Most Significant Bit) of norm testing to select where to focus on.

The group test and most significant bit procedures for unevenly spaced data are the same as Algorithm 3.11 and 3.12 in [21]. For the sake of selfcontainedness, we list them here again. Note that MSB might find the wrong frequency sometimes. There are several possible reasons. For example, if the isolation procedure constructs a new signal, which does not contain one predominant frequency. Another reason is that the inaccurate information provided by the norm estimation leads to wrong judgment.

In each MSB step, we use a Box-car filter H_k to subdivide the whole region into $2k + 1$ subregions. By estimating the energies and comparing the estimates for all these new signals, we find the one with maximum energy, and we exclude those that have estimated energies much smaller than this maximum energy. We then repeat on the remaining region, a more precisely on the region obtained by removing the largest chain of excluded intervals; we dilate so that this new region fills the whole original interval, and split again. The successive outputs of the retained region gives an increasingly good approximation to the dominant frequencies. The following are the Group testing procedures:

Algorithm 3.12. *[21]* GROUP TESTING
Input: signal F, the length N of the signal F.
Initialize: set the signal F to F_0, iterative step $i = 0$, the length N of the signal, the accumulation factor $q = 1$.
In the ith iteration,

1. *If $q \geq N$, then return 0.*
2. *Find the most significant bit v and the number of significant intervals c by the procedure MSB.*
3. *Update $i = i+1$, modulate the signal F_i by $\frac{(v+0.5)N}{4(2k+1)}$ and dilate it by a factor of $4(2k+1)/c$. Store it in F_{i+1}.*
4. *Call the Group testing again with the new signal F_i, store its result in g.*
5. *Update the accumulation factor $q = q * 4(2k+1)/c$.*
6. *If $g > N/2$, then $g = g - N$.*
7. *return $mod(\lfloor \frac{cg}{4(2k+1)} + \frac{(v+1/2)N}{4(2k+1)} + 0.5 \rfloor, N)$;*

The MSB procedure is as follows.

Algorithm 3.13. *[21]*MSB (MOST SIGNIFICANT BIT)
Input: signal F with length N, a threshold $0 < \eta < 1$.

1. *Get a series of new signals $G_j(t) = F(t) \star (e^{2\pi ijt/4(2k+1)} H_k)$, $j = 0, \ldots, 8k+4$. That is, each signal G_j concentrates on the pass region $[\frac{(j-1/2)N}{4(2k+1)}, \frac{(j+1/2)N}{4(2k+1)}] := pass_j$.*
2. *Estimate the energies e_j of G_j, $j = 0, \ldots, 8k+4$.*
3. *Let l be the index for the signal with the maximum energy.*
4. *Compare the energies of all other signals with the lth signal. If $e_i < \eta e_l$, label it as an interval with small energy.*
5. *Take the center v_s of the longest chain of consecutive small energy intervals, suppose there are c_s intervals altogether in this chain.*
6. *The center of the large energy intervals is $v = 4(2k+1) - v_s$, the number of intervals with large energy is $c = 4(2k+1) - c_s$.*
7. *If $c > 4(2k+1)/2$, then do the original MSB* [8] *to get v and set $c = 2$, and $v = center\ of\ the interval\ with\ maximal\ energy$.*
8. *Output the dilation factor c and the most significant bit v.*

3.5 Adaptive Greedy Pursuit

In summary, given a signal S, for an accuracy ϵ and for B modes, we can find a very good approximation of the signal S by using Algorithm 2.2 in [21]. Since the Lagrange interpolation achieves better performance, we give the following theoretical result.

Theorem 3.14. *Given a signal S, an accuracy ϵ, success probability $1-\delta$. If Algorithm 2.2 can output a B-term representation R with sum-square-error $\|S-R\|_2^2 \leq (1+\epsilon)\|S-R_{opt}\|_2^2$, where R_{opt} is the B-term representation for S with the least sum-square-error, with time and space cost $poly(B, \log(N), \frac{1}{\epsilon}, \log(1/\delta))$ for computing and $\frac{poly(B,\log N,\log(1/\delta)}{\epsilon^2 \log(1-p)}$ for just visiting samples, where λ is defined in Lemma 3.2.*

Proof. We omit the proof since it is very similar to Theorem 9 in [8]. Now we need to show that the number of visiting operations are $O(\frac{B}{\epsilon \log(1-p)})$ in the worst case. The total number of greedy pursuit iterations is $O(\frac{B}{\epsilon})$. Hence, we need to estimate $O(\frac{B}{\epsilon})$ coefficients. For each coefficient, it visits $O(\frac{\log(1/\delta)}{\epsilon \log(1-p)}$ samples. Therefore, the theorem holds. □

A high-dimensional NERAℓSFA are very similar and trivial. With the same greedy pursuit for available points and interpolation techniques, it can derived from the high dimensional RAℓSFA.

Above theoretical analysis set very tight restriction on parameters. Heuristically, much looser parameter setting works fine, as the following numerical experiments support.

3.6 Extension to higher dimensions

The RAℓSFA in [21] proposed a higher dimensional RAℓSFA . With very similar techniques to two dimension RAℓSFA [8], this algorithm can be extended to process unevenly spaced data. As we already pointed out, the unevenly spaced data causes the introduction of special techniques in norm and coefficient estimation. For the coefficient estimation, we still pursuit an available data greedily. The norm estimation can either employ the greedy pursuit technique, or the high dimensional Lagrange interpolation.

First issue is to how to locate the four neighbors of a missing point and estimate the value. In one dimension, values of missing points can be interpolated by its few nearest left and right available neighbors. The idea can be extended to higher dimensional cases with more techniques.

For instance, in two dimensions, we first find four nearest available neighbors of a missing point in each quadrant. Suppose a missing point is (x, y), its four neighbors are (x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4). The weights of neighbors can be derived by solving the following linear system of equations.

$$\begin{pmatrix} x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \\ x_1y_1 & x_2y_2 & x_3y_3 & x_4y_4 \\ 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \end{pmatrix} = \begin{pmatrix} x \\ y \\ xy \\ 1 \end{pmatrix} \tag{28}$$

Since this content is standard, we omit the discussion of more complicated situations, for example the singular matrix in (28).

One easily sees that the system of equations (28) is translation invariant: the two linear system of equations

$$\begin{pmatrix} x_1+l & x_2+l & x_3+l & x_4+l \\ y_1+p & y_2+p & y_3+p & y_4+p \\ (x_1+l)(y_1+p) & (x_2+l)(y_2+p) & (x_3+l)(y_3+p) & (x_4+l)(y_4+p) \\ 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \end{pmatrix} = \begin{pmatrix} l \\ p \\ lp \\ 1 \end{pmatrix}$$

and

$$\begin{pmatrix} x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \\ x_1y_1 & x_2y_2 & x_3y_3 & x_4y_4 \\ 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

have the same solutions for any l and p. That means the location of the missing points does not influence the weights. Only the geometrical shape and relative distance of the available neighbors of a missing point matters.

Thus, we compute weights for the geometrical shapes of available neighboring points which occur most often. As we go through every missing point, we check if the shape of its neighboring available points matches those popular ones; if it does, we can directly get the weights without computation. This saves a huge amount of work, especially when p is large.

After computing all the weights ω_i, we estimate the value of the missing point by

$$S(x,y) = \sum_{i=1}^{4} \omega_i S(x_i, y_i). \tag{29}$$

4 Numerical Results

In this section, we present numerical results of NERAℓSFA and compared to the Inverse Non-equispaced Fast Fourier Transform (INFFT) algorithms. The popular software NFFT version 2.0 is used to give performance of INFFT, with default CGNE_R method and Dirichlet kernel. Its time cost excludes the precomputation of sample values, which takes $O(L)$. Numerical experiments show the advantage of our NERAℓSFA algorithm in processing large amount of data of sparse signals. We begin in Section 4.1 with comparing NERAℓSFA with INFFT for some one and two dimensional examples with different length. In Section 4.2, the performance for different number of modes is shown. Section 4.3 presents the result with different percentage of available data. We explore the robustness of the NERAℓSFA to noise in Section 4.4. Finally, we test the relationship between error and running time.

All the experiments were run on an AMD Athlon(TM) XP1900+ machine with Cache size 256KB, total memory 512 MB, Linux kernel version 2.4.20-20.9 and compiler gcc version 3.2.2. The numerical data is an average of 10 runs of the code; errors are given in the l^2 norm.

N	INFFT	NERAℓSFA (+sampling)	NERAℓSFA (w/o sampling)
2^9=512	0.01	0.63	0.31
2^{11}=2048	0.03	0.77	0.37
2^{13}=8192	0.17	0.90	0.46
2^{15}=32768	0.83	0.93	0.49
2^{17}=131072	4.30	1.03	0.51
2^{19}=524288	19.94	1.20	0.61

Table 3: Experiments with fixed $B = 8$, $p = 0.7$, $d = 1$ (one dimension), and varying length N of signals; an i.i.d. white noise is added with $\sigma = 0.5$, or $SNR < -12dB$ (see text). For each length of the signal, 10 different runs were carried out; the average result is shown. Two kinds of time costs for NERAℓSFA are provided. One is the total running time and another is the running time excluding the sampling time. The time of INFFT does not include the precomputation time for samples.

4.1 Experiments with Different Length of Signals

We ran the comparison for a 8-mode superposition signal $S(t) = \sum_{i=1}^{B} \phi_{\omega_i}$, plus white noise ν with the standard deviation $\sigma = 0.5$, damped by a factor of $1/\sqrt{N}$, (so that $\|\nu\|^2 = \sigma^2 = 0.25$; since $\|S\|^2 = 8$, this implies $SNR = 10 \log_{10} 32/N < -12dB$). Other parameters are $B = 8$, $\epsilon = 0.02$, $\delta = 0.01$, and $p = 70\%$. The missing data are randomly and uniformly distributed. NERAℓSFA outperforms INFFT in speed when N is large; see Figure 4 and Table 3. The corresponding crossover point is $N \geq 2^{15} = 32768$. For example, to process $2^{19} = 524,288$ data, more than nineteen minutes (estimated) are needed for INFFT versus approximately one second for NERAℓSFA. Experiments support the theoretical conclusion that NERAℓSFA would be faster than INFFT after some N for a sparse signal; whatever the sparsity, i.e. whatever the value of B, there always exists some crossover N.

In two dimensions, we test a noisy 6-mode superposition signal $S(t) = \sum_{i=1}^{B} \phi_{\omega_{xi}} \phi_{\omega_{yi}} + \nu$, where ν is a Gaussian white noise, $B = 6$, $\epsilon = 0.02$, $\delta = 0.01$, $p = 80\%$, and $\sigma = 0.1$. Missing data are randomly and uniformly distributed. As the number of grid points N in each dimension grows, two dimensional NERAℓSFA outperforms two dimensional INFFT at $N \geq 512$, as Figure 6 and Table 5 show. The crossover point becomes much smaller in high dimensions situation. It would not be surprising that for recovering a 6-mode three dimensional signal, NERAℓSFA surpasses INFFT at a hundred grid points in each dimension.

4.2 Experiments with Different Number of Modes

The number of modes has an important influence on the running time since the crossover point varies for signals with different B. To investigate this, we did the experiments with fixed $N = 2^{18} = 262,139$, $p = 0.7$ and varying B. We take $S(t) = 1/(a + \cos 2\pi t)$ with fast decaying Fourier coefficients and cut all the small coefficients with amplitudes smaller than 10^{-5}, where the

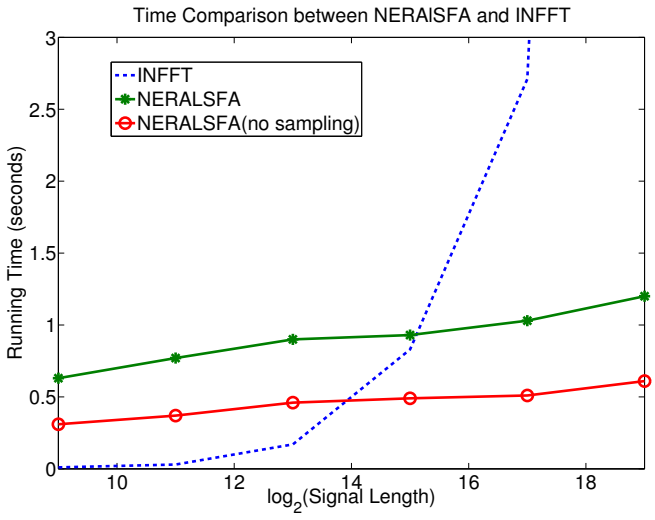

Figure 4: Time Comparison between INFFT and NERAℓSFA for different N with $B = 8, p = 0.7$, $d = 1$. The result in Table 3 is shown in the form of a graph here. The x coordinate is the $\log_2(N)$, the y coordinate presents the running time for each algorithm. NERAℓSFA without sampling surpasses INFFT at $N = 2^{14} = 16384$.

N	INFFT	NERAℓSFA (+sampling)	NERAℓSFA (w/o sampling)
128	0.13	2.86	1.57
256	0.73	2.60	1.46
512	3.00	3.70	2.13
1024	11.59	4.31	2.94
2048	54.94	6.56	4.90

Table 5: Experiments with fixed $B = 6$, $p = 0.8$, $d = 2$ (two dimensions), and varying length N of signals; an i.i.d white noise is added with $\sigma = 0.1$, or $SNR < -14dB$ (see text). For each length of the signal, 10 different runs were carried out; the average result is shown. Again, two kinds of time costs for NERAℓSFA, the one with and without sampling time is provided. The time of INFFT excludes the sampling time.

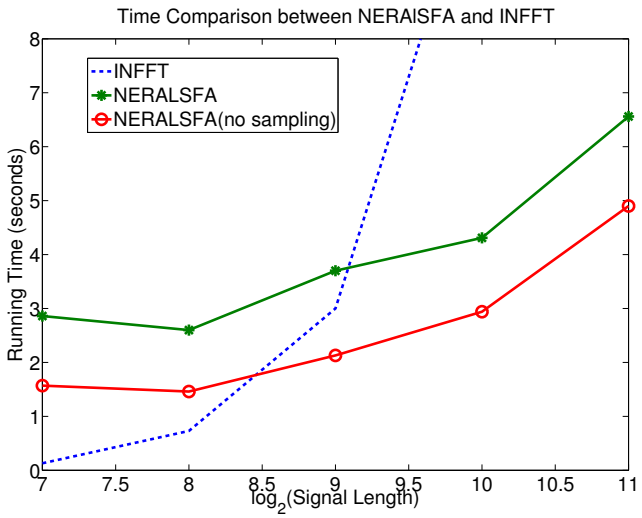

Figure 6: Time comparison between INFFT and NERAℓSFA for different N with fixed $B = 6$, $p = 0.8$, $d = 2$. The x coordinate is the logarithm of length N of signal in each dimension. INFFT is very fast when N is relatively small and slows down quickly as N increases. On the contrary, it takes NERAℓSFA similar time to process small and large N problem. NERAℓSFA without sampling outperforms INFFT at $N = 2^{8.5}$=362.

parameter a directly determines the number of Fourier modes with the absolute value greater than 10^{-5}. Available data are randomly distributed. Table 7 compares the running time for different B using INFFT and NERAℓSFA. At first, NERAℓSFA takes less time because N is so large. However, the execution time of INFFT keeps constant for different number of modes B, while that of modified RAℓSFA is polynomial of higher order. INFFT is faster than NERAℓSFA when $B \geq 18$. The regression techniques shows empirically that the order of B in NERAℓSFA is greater than quadratic. This is one of the characteristics of this version of the RAℓSFA algorithms and irrelevant to the nonequispaceness of the data. (A different version of RAℓSFA in [9] is linear in B, but maybe less easily used when not all equispaced data are available.)

4.3 Experiments for Different Percentage of Missing Data

The advantage of interpolation techniques is to recover a signal even when a large percentage of data is missing. Table 8 shows the recovery effect for a two-mode pure signal $c_1\phi_{\omega_1} + c_2\phi_{\omega_2}$, $N = 10^6$ with all the other parameters ϵ and δ the same as before. When the percentage of available data is large, both algorithms recover the signal well with similar running time.

We tried another example of signal when $N = 100$. NERAℓSFA without interpolation techniques fails to recover the signal with high probability if more than 45% data are unavailable. In contrast, with the help of interpolation technique, the NERAℓSFA can always recover the signal with only 25% available data.

number of modes B	value a	NERAℓSFA (+sampling)	INFFT
2	400.00	0.35	10.75
9	60.00	3.98	10.85
17	6.00	10.07	11.03
33	2.20	35.78	11.31
63	1.30	59.21	11.98

Table 7: Experiments with fixed $N = 262,139$, $p = 0.7$, $d = 1$ (one dimension), and varying number of modes B of signals. For each length of the signal, 10 different runs were carried out; the average result is shown. For this function, NERℓSFA only needs to sample $poly(\log N)$ function values, whereas INFFT computes all the data.

p	Time of NERAℓSFA (with interpolation)	success probability	Time of NERAℓSFA (w/o interpolation)	success probability
1	0.03	100 %	0.03	100 %
0.8	0.04	100 %	0.06	100 %
0.6	0.05	100 %	0.49	100 %
0.4	0.05	100 %	0.45	100 %
0.3	0.06	100 %	-	0 %
0.2	0.06	100 %	-	0 %
0.1	0.07	100 %	-	0 %
10^{-2}	0.11	100 %	-	0 %
10^{-3}	0.51	100 %	-	0 %
10^{-4}	4.58	100 %	-	0 %
0.00002	758.22	97 %	-	0 %

Table 8: Experiments with fixed $B = 2$, $N = 10^6$, no noise, and varying percentage of available data. Each entry is based on the average of 10 different runs. In each run, the number of iterations is limited to 200; (this also corresponds to a fixed limit to the number of samples taken.) the success probability indicates the number of runs in which all 6 modes were found. When only 30% of data is available, the NERAℓSFA without interpolation cannot find all two significant modes within 200 iterations.

σ	SNR (dB)	Time of NERAℓSFA (+sampling)	Time of NERAℓSFA (w/o sampling)	Relative Error (%)	Success probability
0	-	0.48	0.21	0.02	100%
0.5	-37.38	0.56	0.22	2.00	100%
1.0	-43.40	0.87	0.32	4.50	90%
1.5	-46.91	3.94	1.59	5.83	80%
2.0	-49.41	4.78	1.86	7.67	50%
2.5	-51.35	7.96	2.14	8.50	30%

Table 9: Experiments with fixed $B = 6$, $N = 2^{17}$, $p = 0.6$, and varying noise levels. For each noise level, 10 different runs were carried out; the average result is shown. In each run, the number of iterations is limited to 200; (this also corresponds to a fixed limit to the number of samples taken.) the success probability indicates the number of runs in which all 6 modes were found. The average relative error is the error of reconstructed signal with respect to the original signal.

Experiments also show that for NERAℓSFA with interpolation technique, the total number of available data, instead of the percentage of available data determines the success probability. On the contrary, The success of NERAℓSFA without interpolation is determined by the percentage.

These experiments also provide failure examples for the NERAℓSFA. The algorithm could fail when there are too few available data. Another failure example is as follows. Suppose the signal is $S = e^{ik\pi}$, where $k = 0, \ldots N-1$. We also assume the data is available only in the even grids. The Lagrangian interpolation would find the function values on odd grids are 1, instead of -1. Hence, the algorithm fails to find the correct significant mode and coefficient.

4.4 Experiments to Recover Noisy Signals

To recover a signal from very noisy data is a challenging problem. The following tests are done for $S(t) = \sum_{i=1}^{B} c_i \phi_{\omega_i} + \nu$, $B = 6$, $\epsilon = 0.02$, $N = 2^{17}$, $p = 0.6$, and different standard deviation σ for noise. As Table 9 shows, NERAℓSFA excels at extracting information from noisy data even in the case of small signal to noise ratio.

We also find this algorithm is able to recover the original signal from partial and noisy data. Suppose we only have contaminated and partial information about the signal. In some intervals, all the information are missing. Figure 10 shows the NERAℓSFA can recover the signal very well in the sense that it discovers most of the original pure signal information.

4.5 Experiments of the relationship between error and time

It is of particular interest to investigate how the running time increases for achieving different errors. We take a signal $S = \sum_{j=1}^{8} c_k \phi_{\omega_k}$, where c_k and ω_k are randomly taken from the interval $[1, 10]$ and all the integers in $\{0, \ldots, N-1\}$. As a randomized algorithm, RAℓSFA takes only a little bit more time to achieve higher accuracy, as shown in Figure 11. This is reasonable since the

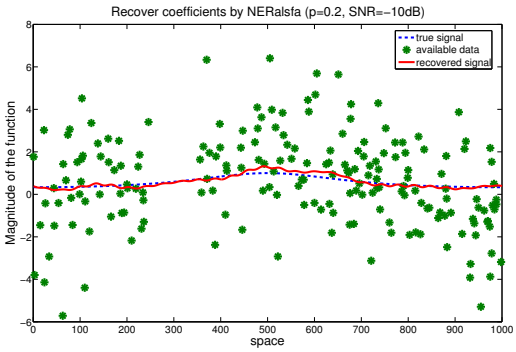

Figure 10: Experiments with $S(t) = 1.0./(2 + cos(2 * \pi * t)), p = 0.2, SNR = -10dB$. The true signal (denoted by the dotted line) is contaminated by very large noise and some of its information is missing. Hence, we only know the values of the dots. Then NERAℓ SFA recovers the signal (denoted by the solid line) from the dots.

total running time of RAℓSFA includes time for identifying significant modes and estimating their coefficients, where the second part only takes a small fraction of the time. Also, as a randomized algorithm, the performance of the algorithm varies in different runs. Hence, we notice that the error-time line is not strictly decreasing soemtimes.

5 Conclusion

We provide a sublinear sampling algorithm that recovers, with high probability, a B-term Fourier representation for an unevenly spaced signal. For those sparse signals of very large size, it is faster than any existed methods, for example, when $B = 8$, $N = 2^{15}$, and $p = 0.7$. Moreover, it recovers the signal in the situation of large percentage of missing data or large noise. The NERAℓSFA also denoises the signal much better successfully than INFFT.

6 Acknowledgments

For many helpful suggestions and discussions, I would thank my adviser Ingrid Daubechies. In addition, I thank Weinan E, Anna Gilbert, Martin Strauss for their suggestions.

References

[1] R. BASS AND K. GRÖCHENIG, *Random sampling of multivariate trigonometric polynomials*, SIAM J. Math. Anal., Vol. 36 (2004), pp. 773-795.

[2] A. BJÖRCK. *Numerical Methods for Least Squares Problems*. SIAM, Philadelphia, 1996.

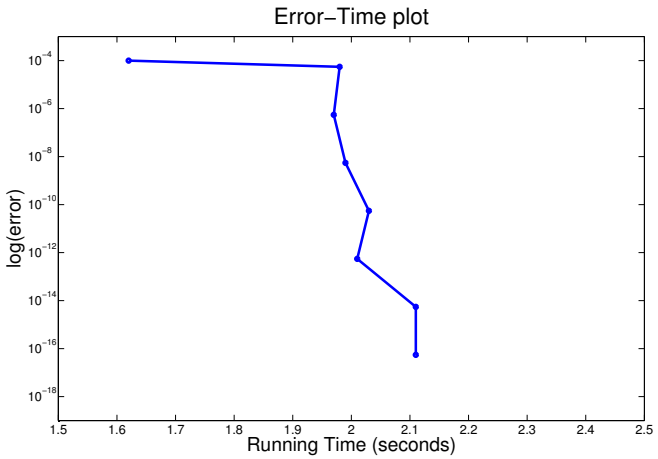

Figure 11: Experiments with fixed $B = 8$, $N = 10001$, $p = 0.7$. We did 500 experiments with different parameters to generate different errors. In each run, the number of iterations is limited to 100. All the data are categorized according to the errors. The x-label and y-label show the average running time and logarithmic of errors in different ranges.

[3] J.P. BOYD, *A fast algorithm for Chebyshev, Fourier and Sinc interpolation onto an irregular grid*, J. Comput. Phys., 103 (1992), pp. 243-257.

[4] E. CANDES, J. ROMBERG, AND T. TAO, Robust Uncertainty Principles: Exact Signal Reconstruction from Highly Incomplete Frequency Information, perprint, December 2004.

[5] JOHN CANNY, Lecture 10, Chernoff Bounds, http://www.cs.berkeley.edu/jfc/cs174/lecs.

[6] H. FASSBENDER, *On numerical methods for discrete least-squares approximation by trigonometric polynomials*, Math. Comput., 66(1997), pp719-741.

[7] H. FEICHTINGER, K. GRÖCHENIG, AND T. STROHMER, *Efficient numerical methods in non-uniform sampling theory*, Numer. Math., 69 (1995), pp423-440.

[8] A.C. GILBERT, S. GUHA, P. INDYK, S. MUTHUKRISHNAN AND M. STRAUSS, *Near-Optimal Sparse Fourier Representations via Sampling*, STOC, 2002

[9] A.C. GILBERT, S. MUTHUKRISHNAN AND M. STRAUSS, Improved Time Bounds for Near-Optimal Sparse Fourier Representation, to appear.

[10] L. GREENGARD AND J. LEE. Accelerating the Nonuniform Fast Fourier Transform, SIAM Review, 46 (2004), pp. 443-454.

[11] G. GRIMMETT AND D. STIRZAKER. *Probability and Random Processes*. Oxford University Press, 2001.

[12] M. HANKE. Conjugate gradient type method for ill-posed problems. Wiley, New York, 1995.

[13] S. KUNIS AND D. POTTS, *Stability results for scattered data interpolation by trigonometric polynomials*, preprint.

[14] S. KUNIS, D. POTTS, *NFFT, Software, C subroutine library,* http://www.math.uni-luebeck.de/potts/nfft, 2002-2004.

[15] S. KUNIS, D. POTTS, G. STEIDL, *Fast Fourier transform at nonequispaced knots: A user's guide to a C-library*, Manual of NFFT 2.0 software.

[16] Y. MANSOUR, *Randomized interpolation and approximation of sparse polynomials* , SIAM Journal on Computing 24:2 (1995).

[17] A. OPPENHEIM, A. WILLSKY WITH S. NOWAB. *Signals and Systems*. Prentice Hall, 1998.

[18] W. PRESS, S. TEUKOLSKY, W. VETTERLING AND B. FLANNERY. *Numerical Recipes in C: the art of scientific computing*. Cambridge University Press, 1992.

[19] L. REICHEL, G. S. AMMAR, AND W. B. GRAGG. *Discrete least squares approximation by trigonometric polynomials.* Math. Comput., 57(1991), pp. 273-289.

[20] A. F. WARE, *Fast Approximate Fourier Transforms for Irregularly Spaced Data*, SIAM Rev., 40 (1998), pp. 838–856.

[21] J. ZOU, A.C. GILBERT, M. STRAUSS AND I. DAUBECHIES, *Theoretical and Experimental Analysis of a Randomized Algorithm for Sparse Fourier Transform Analysis*, accepted by Journal of Computational Physics.