Appl. Comput. Harmon. Anal. 40 (2016) 553–574

Contents lists available at ScienceDirect

Applied and Computational Harmonic Analysis

www.elsevier.com/locate/acha

A multiscale sub-linear time Fourier algorithm for noisy data

Andrew Christlieb [a,1], David Lawlor [b,c,*], Yang Wang [a,2]

[a] Department of Mathematics, Michigan State University, East Lansing, MI 48824, United States
[b] Statistical and Applied Mathematical Sciences Institute, 19 T. W. Alexander Drive, P.O. Box 14006, Research Triangle Park, NC 27709-4006, United States
[c] Department of Mathematics, Duke University, Box 90320, Durham, NC 27708-0320, United States

ARTICLE INFO

Article history:
Received 25 March 2014
Received in revised form 10 October 2014
Accepted 12 April 2015
Available online 15 April 2015
Communicated by Gregory Beylkin

Keywords:
Fast Fourier algorithms
Multiscale algorithms
Fourier analysis

ABSTRACT

We extend the recent sparse Fourier transform algorithm of [1] to the noisy setting, in which a signal of bandwidth N is given as a superposition of $k \ll N$ frequencies and additive random noise. We present two such extensions, the second of which exhibits a form of error-correction in its frequency estimation not unlike that of the β-encoders in analog-to-digital conversion [2]. On k-sparse signals corrupted with additive complex Gaussian noise, the algorithm runs in time $O(k \log(k) \log(N/k))$ on average, provided the noise is not overwhelming. The error-correction property allows the algorithm to outperform FFTW [3], a highly optimized software package for computing the full discrete Fourier transform, over a wide range of sparsity and noise values.

1. Introduction

The Fast Fourier Transform (FFT) [4] is a fundamental numerical algorithm whose importance in a wide variety of applications cannot be overstated. The FFT reduces the runtime complexity of calculating the discrete Fourier transform (DFT) of a length N array from the naive $O(N^2)$ to $O(N \log(N))$. At the time of its introduction in the mid-1960s, it dramatically increased the size of problems that a typical computer could handle. Over the past fifty years the typical size of data sets has grown by orders of magnitude, and in certain application areas (e.g. cognitive radio and ultra-wideband radar [5,6]) the computation of the full FFT is no longer tractable on commodity hardware. In this and other instances, however, it is known a priori that the signals of interest have small frequency support; that is, their Fourier transforms are *sparse*. This problem has received attention from a number of research communities over the past decade, who have

* Corresponding author at: Department of Mathematics, Duke University, Box 90320, Durham, NC 27708-0320, United States. Fax: +919 685 9360.

E-mail addresses: christlieb@math.msu.edu (A. Christlieb), djl@math.duke.edu (D. Lawlor), ywang@math.msu.edu (Y. Wang).

[1] Fax: +517 432 1562.
[2] Fax: +517 355 5279.

http://dx.doi.org/10.1016/j.acha.2015.04.002
1063-5203/

shown that it is possible to significantly outperform the FFT in both runtime and sampling requirements when the number of significant Fourier modes k is much less than the nominal bandwidth N. Early works addressing this topic from the perspective of learning boolean functions include [7,8].

The sparse Fourier transform problem was first studied explicitly in [9,10], the latter of which gave a randomized algorithm with runtime and sampling complexity $\mathrm{O}(k^2 \operatorname{polylog}(N))$.[3] This was later improved to $\mathrm{O}(k \operatorname{polylog}(N))$ [11] through the use of unequally-spaced FFTs [12]. For a given failure probability δ and accuracy parameter ε, the algorithm returns a k-term approximation $\hat{y}$ to the DFT of the input $\hat{x}$ such that with probability $1-\delta$ it holds that

$$\|\hat{x}-\hat{y}\|_2^2 \leq (1+\varepsilon)\|\hat{x}-\hat{x}_k\|_2^2. \tag{1}$$

Here $\hat{x}_k$ is the best k-term approximation to $\hat{x}$ and $\|\cdot\|_2$ is the discrete ℓ_2 norm. In [13], a randomized $\mathrm{O}(k^2\operatorname{polylog}(N))$ algorithm for the sparse Fourier transform problem was given in the context of list decoding.[4] A separate group of authors [14] has developed a modified version of the algorithm of [11] with runtime $\mathrm{O}(\log(N)\sqrt{Nk\log(N)})$. While the dependence on N is sub-optimal asymptotically, in practice this algorithm is significantly faster than either [10] or [11]. The same authors presented an improved algorithm with runtime $\mathrm{O}(k\log(N)\log(N/k))$ in [15] whose frequency identification procedure is very similar to [1], upon which the present work is based. However, the performance of [15] in the presence of noise has yet to be evaluated empirically.

The algorithms described in the previous paragraph are all randomized, and so will fail on each signal with positive probability. Recognizing this as a potential detriment in failure-intolerant applications, two authors have independently given deterministic algorithms for the sparse Fourier transform problem. In [16,17] an algorithm with $\operatorname{poly}(k, \log(N))$ runtime was given where the exponent on k is at least six.[5] This high dependence on k renders the algorithm infeasible in practice, and it has not been implemented. However, we note that algorithms of [18,16,17] address a strictly wider class of signals than those with k-sparse Fourier spectrum, specifically those satisfying $\|\widehat{S}\|_1/\|\widehat{S}\|_2 \leq \operatorname{polylog}(N)$. In [19], the combinatorial properties of aliasing among frequencies were exploited to give an algorithm with runtime and sampling complexity $\mathrm{O}(k^2 \operatorname{polylog}(N))$. While this represented a major improvement over the theoretical runtime complexity of [16], in practice it only outperformed the FFT for relatively modest values of the sparsity k.

Most recently the authors of [1] gave a deterministic algorithm whose sampling and runtime complexity are $\mathrm{O}(k\log(k))$ in the average case and $\mathrm{O}(k^2\log(k))$ in the worst case. The worst-case bounds are asymptotically of the same order in k (up to log factors) as [19], but over a representative class of random signals it was shown to significantly outperform its deterministic and randomized competitors. This was achieved by sampling the input at two sets of equispaced points slightly offset in time. This time shift appears in the Fourier domain as a frequency modulation, which allows the authors to both detect when aliasing has occurred and, for frequencies that are isolated (i.e. not aliased), to calculate the frequency value directly. While [19] also uses properties of aliasing to reconstruct frequency values, it is not able to distinguish between aliased and non-aliased terms until sufficiently many DFTs of coprime lengths have been computed, and so is unable to perform any better in the average case than in the worst case. In the empirical evaluation of [1] an improvement of over two orders of magnitude was observed over [11] and [19].

In this paper we extend the algorithm of [1] to noisy environments in two distinct ways. The first of these, which is a minor modification of the noiseless algorithm, is based on a certain rounding of the frequency estimates and was previously reported in [1]. In this work we provide an improved algorithm and more detailed analysis of that earlier work. The second extension is the main result of this paper

[3] We write $f = \operatorname{polylog}(g)$ to indicate that $f = \mathrm{O}(\log^c(g))$ for some unspecified constant c.

[4] The runtime of this algorithm was incorrectly stated as $\mathrm{O}(k^{11/2}\operatorname{polylog}(N))$ in [1].

[5] This algorithm is a de-randomization of the randomized algorithm presented in [18].