Continuous Fast Fourier Sampling

Praveen K. Yenduri, Anna C. Gilbert

Continuous Fast Fourier Sampling

Praveen K. Yenduri[(1)] and Anna C. Gilbert[(2)]

(1) University of Michigan, 4438 EECS building, Ann Arbor, MI 48109, USA.
(2) University of Michigan, 2074 East Hall, Ann Arbor, MI 48109, USA.
ypkumar@umich.edu, annacg@umich.edu

Abstract:

Fourier sampling algorithms exploit the spectral sparsity of a signal to reconstruct it quickly from a small number of samples. In these algorithms, the sampling rate is sub-Nyquist and the time to reconstruct the dominate frequencies depends on the type of algorithm—some scale with the number of tones found and others with the length of the signal. The Ann Arbor Fast Fourier Transform (AAFFT) scales with the number of desired tones. It approximates the DFT of a spectrally sparse digital signal on a fixed block by taking a small number of structured random samples. Unfortunately, to acquire spectral information on a particular block of interest, the samples acquired must be appropriately correlated for that block. In other words, the sampling pattern, though random, depends on the block of interest. When blocks of interest overlap significantly, the union of the sampling patterns may not be an optimal one (it might not be sub-Nyquist anymore). Unlike the much slower algorithms, the sampling pattern does not accommodate an arbitrary block position. We propose a new sampling procedure called Continuous Fast Fourier Sampling which allows us to continuously sample the signal at a sub-Nyquist rate and then apply AAFFT on any arbitrary block. Thus, we have a highly resource-efficient continuous Fourier sampling algorithm.

1. Introduction

Let x be a discrete time signal of length n which is sparse or compressible in the frequency domain but the exact frequency content depends on time. We consider the problem of computing the frequency content present in different blocks of the signal in a resource efficient manner. This problem arises in many applications such as *cognitive radio* [2] where a wireless node alters its transmission or reception parameters based on active monitoring of radio frequency spectrum at various times. Another application is *incoherent demodulation* of communication signals [3] such as FSK, MSK, OOK, etc., where the computed frequency spectrum at different times represents the message being transmitted itself.

There are several Fourier sampling algorithms [1, 8, 9] with low sampling costs that reconstruct the entire spectrum of a sampled signal. These algorithms make use of a uniformly random (not structured) sample set for computations thus allowing us to compute frequencies in any arbitrary block of interest from the signal. However, the time to reconstruct the spectrum is superlinear in signal's size and hence are slow and inappropriate for the applications involving large signal sizes or bandwidths where just a few frequencies are of interest. Instead, we consider a sub-linear time computational method called the AAFFT (Ann Arbor Fast Fourier Transform) described in [4].
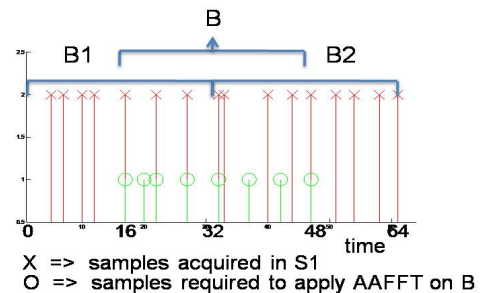

Figure 1: Figure showing the samples acquired in $S1$ and the samples required to apply AAFFT on $B = [16, 47]$.

Let y be a fixed block of interest of length N in the discrete time signal x. Since x is sparse in frequency domain, it can be assumed that y has only m dominant digital frequencies, where $m \ll N$. The AAFFT algorithm takes a small number of (correlated) random samples from the block of interest and produces an approximation of its DFT (identifies dominant tones), using time and storage $m\,\mathrm{poly}(\log(N))$. If we are interested in a windowed Fourier analysis of x over windows of length N, a straightforward approach towards solving our problem using AAFFT is to divide the signal x into consecutive non-overlapping blocks of length N, generate appropriately correlated sampling patterns for each block, acquire the samples and then apply AAFFT on each block. Let us call this sample set $S1$. Unfortunately, $S1$ does not accommodate arbitrary block positions. For example, consider samples acquired in $S1$ from two consecutive blocks $B1$ and $B2$. Lets say we are now interested in block B which consists of second half of $B1$ and first half of $B2$ (see Figure (1)). However AAFFT cannot be applied on B since the samples acquired from B will not be appropriately structured for its application. This is illustrated in Figure (1) for a simple case of $N = 32$ with a dummy y-axis and a few samples plotted for clarity.

We propose a new sampling procedure called the Continuous Fast Fourier Sampling that allows us to continu-

ously sample the signal (as opposed to division into discrete blocks) at a sub-nyquist rate and then apply AAFFT on any arbitrary block of interest. The article describes the algorithm in detail in Section (3.2), proves its correctness in Section (3.3), followed by a few numerical experiments and results in Section (3).

2. The Fourier Sampling Algorithm (AAFFT)

The Fourier Sampling algorithm is predicated upon non-evenly spaced samples unlike many traditional spectral estimation techniques [6, 7] and uses a highly nonlinear reconstruction method that is divided into two stages, *frequency identification* and *coefficient estimation*, each of which includes multiple repetitions of basic subroutines. A detailed description of the implementation of AAFFT is available in [5].

Frequency Identification consists of two steps, dominant frequency isolation and identification. Isolation is carried out by a two-stage process: (i) pseudo random permutations of the spectrum, followed by (ii) the application of a filter bank with $K = O(m)$ bands, where m = number of tones (dominant spikes) in the signal. With high probability, a significant fraction of the dominant tones fall into individual bands, isolating each tone from the others and this probability can be increased with additional repetitions. Note that all the above is carried out *conceptually* in the frequency domain but instantiated in the time domain. That is, we sample the permuted and filtered signal in the time domain. To carry out the computations, the algorithm uses signal samples at time points indexed by $P(t, \sigma) = \{(t + q\sigma) \mod N, q = 0, 1, .., K - 1\}$, where (t, σ) is randomly chosen for each repetition. The identification stage performs group testing to determine the dominant frequency value in each of the K outputs of the filterbank. This stage uses the samples indexed at arithmetic progressions $P(t^b, \sigma)$ formed from each element of the geometric progression $t^b = t + \frac{N}{2^{b+1}}$, $b = 0, 1, .., \log_2(N/2)$. The *estimation* stage uses the random sampling similar to the isolation stage for coefficient estimation of each of the dominant frequencies identified.

Note that although the (t, σ) pair is chosen randomly in each repetition, the samples that result from each pair are highly structured. Let $A_1 = \{(t, \sigma)\}$ used in the *frequency identification* stage and similarly let A_2 be defined for the *estimation* stage. These two sets define a sampling pattern.

3. Continuous Fast Fourier Sampling

3.1 Sample set construction

Let n be the length of signal x which has m dominant tones that vary over time. Let the block length be N. Let $K = O(m)$ and $\alpha = \log_2(N)$. Let (t, σ) be a fixed pair in A_1 or A_2. Define a sequence of time points $t(0) = t$, $t(j) = (t(j - 1) + Q(j - 1)\sigma) \mod N$ for $j = 1, .., J$, where $Q(j - 1) = $ smallest integer such that $t(j-1)+Q(j-1)\sigma \geq N$ and $J = \lceil \frac{K\sigma}{N} \rceil$. We call $t(j)$ the "N-wraparound" of $t(j-1)$. Figure (2) illustrates the calculation of a N-wraparound. The choice of J is such that

the theorems in Section (3.3) hold. For $j = 1, .., J$, denote by I_j the arithmetic progression formed by $(t(j), \sigma)$,

$$I_j = \{t(j) + q\sigma, \forall q \geq 0 : t(j) + q\sigma \leq n\} \quad (1)$$

Now, consider the geometric progression $t^b = t + \frac{N}{2^{b+1}}$ for all $b = 0, 1, .., \alpha - 1$. For each b, $\left(t + \frac{N}{2^{b+1}}, \sigma\right)$ is treated as another (t, σ) pair and the sequence $t^b(j)$ and the corresponding progressions I_j^b can be defined.

Do all the above, for each pair (t_ℓ, σ_ℓ) in A_1 and A_2 and denote the arithmetic progressions produced, by $I_{\ell,j}$, for $j = 1, .., J_\ell$. Define the union of all such arithmetic progressions as $I_\ell = \bigcup_{j=0}^{J_\ell} I_{\ell,j}$. Similarly define $I_\ell^b = \bigcup_{j=0}^{J_\ell} I_{\ell,j}^b$ for $b = 0, .., \alpha - 1$. Now define $I_\ell^B = \bigcup_{b=0}^{\alpha-1} I_\ell^b$. Finally define

$$I(A_1, A_2) = \left(\bigcup_{A_1}(I_\ell \cup I_\ell^B)\right) \cup \left(\bigcup_{A_2} I_\ell\right) \quad (2)$$

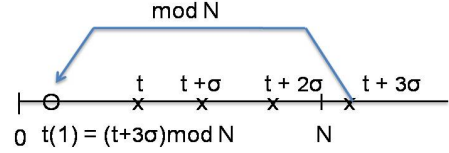Given a set of indices I, we denote by $S^x(I)$ the set of samples from signal x indexed by I.

Figure 2: Calculation of N-Wraparound $t(1)$ from t.

3.2 The CFFS Algorithm

Preprocessing:
INPUT: N // Block length (1) Sample-set generation : Choose A_1 and A_2 as defined and compute $I(A_1, A_2)$ (as in Equation (2)). **OUTPUT:** $I(A_1, A_2)$ // Index set
Sample Acquisition
INPUT: $I(A_1, A_2)$, x (2) sample signal x at I and obtain samples $S^x(I)$. **OUTPUT:** $S^x(I)$
Reconstruction
INPUT: $S^x(I)$, (n_1, n_2) // boundary indices of an arbitrary block y of length N from signal x (3) calculate A_1', A_2' (depend on (n_1, n_2), defined in Section (3.3)) and extract $S^y(I(A_1', A_2')) \subset S^x(I)$. (4) apply AAFFT on the sample-set $S^y(I(A_1', A_2'))$ **OUTPUT:** top m frequencies of x in block $y = x[n_1, n_2]$

3.3 Proof of Correctness of CFFS

The arbitrary block y has boundaries (n_1, n_2). To generate samples from this block, we define new sets A_1' and A_2' as follows. For every (t, σ) in A_1 and A_2, let

i be the smallest integer such that $t + i\sigma > n_1$. Define $t' = (t + i\sigma) \bmod n_1$. Note that t' is simply the n_1-wraparound of t. Put $A_1' = \{(t', \sigma) : (t, \sigma) \in A_1\}$ and similarly A_2'. Note that A_1' and A_2' are still random since A_1 and A_2 were chosen randomly. To apply AAFFT on block y we can now use samples of y indexed by the sampling pattern defined (as in Section (2.)) from A_1' and A_2'. The following theorems together show that the required samples of y are available in $S^x(I(A_1, A_2))$.

Theorem 1 *For sets A_1' and A_2' as defined above, $S^y(I(A_1', A_2')) \subset S^x(I(A_1, A_2))$.*

Theorem 2 *AAFFT can be applied on the sample-set $S^y(I(A_1', A_2'))$, i.e. the index set $I(A_1', A_2')$ has the required structure explained in Section (2.).*

Rather than giving detailed proofs, we prove a proposition that lies at the heart of the two theorems.

Proposition 3 *For every (t', σ) in A_1' or A_2', $S^y(P(t', \sigma)) \subset S^x(I(A_1, A_2))$.*

Proof: Let (t, σ) be the pair in A_1 or A_2 from which (t', σ) was obtained. We will prove that the arithmetic progressions I_j formed by the sequence of wraparounds $t(j), j = 1, .., J$ as defined in Section (3.1), induce mod-N arithmetic in the progression $P(t', \sigma)$ (P as defined in Section (2.)). Consider the first few terms in $P(t', \sigma)$, till $(t' + (q_0 - 1)\sigma) \bmod N$ where q_0 is the smallest integer such that $(t' + q_0\sigma) \geq N$. From definition of t' observe that $t' = (t + i\sigma - n_1)$. so $y(t') = x(n_1 + t') = x(t + \sigma) \in S^x(I_0)$, where I_0 is defined in Equation (1). Similarly it is easy to see that the first q_0 terms in $S^y(P(t', \sigma))$ are contained in $S^x(I_0)$. Now call the next term $(t' + q_0\sigma) \bmod N = t'(1)$. Observe that $t'(1) = t' + \sigma \left\lceil \frac{N - t'}{\sigma} \right\rceil - N$. Similarly observe that $t(1) = t + \sigma \left\lceil \frac{N - t}{\sigma} \right\rceil - N$. Now, Substituting $t' = (t + i\sigma - n_1)$ in the expression for $t'(1)$ we get, $t'(1) = t + i\sigma - n_1 + \sigma \left\lceil \frac{N - t + n_1 - i\sigma}{\sigma} \right\rceil - N = t + i\sigma - n_1 + \sigma \left\lceil \frac{N - t}{\sigma} \right\rceil + d\sigma - N = t(1) + (i + d)\sigma - n_1$, for an appropriately defined d, which can be shown to be positive. So $y((t' + q_0\sigma) \bmod N) = y(t'(1)) = x(t(1) + (i + d)\sigma) \in S^x(I_1)$, where again I_1 is defined in Equation (1). Let q_1 be the smallest integer such that $(t'(1) + q_1\sigma) \geq N$. Now it is easy to see that the next q_1 terms in $S^y(P(t', \sigma))$ are contained in $S^x(I_1)$. Repeat this until all the terms in $P(t', \sigma)$ are covered. ∎

Proposition 4 *On average, the storage requirement of CFFS algorithm is $O(\frac{n}{N} m \log^{O(1)} N)$, which is of the same order as a straightforward, fixed boundary sample set for AAFFT.*

4. Results and Discussion

The Continuous Fast Fourier Sampling algorithm has been implemented and tested in various settings. In particular, we performed following three experiments.

First, we consider a model problem for communication devices which use frequency-hopping modulation schemes. The signal we want to reconstruct has two tones that
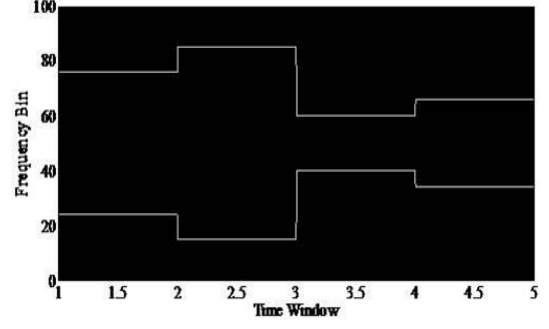

Figure 3: The Sparsogram for a synthetic frequency-hopping signal consisting of two tones, as computed by AAFFT ($S1$) and by CFFS.

change at regular intervals. We apply both the straightforward AAFFT on $S1$ and CFFS to identify the location of the tones. Figure (3) shows the obtained *sparsogram* which is a time-frequency plot that displays only the dominant frequencies in the signal. We get the same sparsogram in both cases, as expected. For $N = 2^{20}$, $S1$ samples about 0.94% of the signal whereas CFFS samples about 1.06% of the signal, which is only very slightly larger than $S1$. This experiment demonstrates the efficiency and similarity of the two methods and supports the proposition made in Section (3.3).
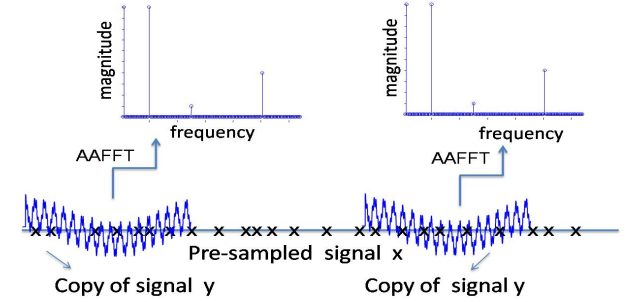

Figure 4: Applying CFFS to different blocks of signal x.

While $S1$-AAFFT cannot be applied to compute the dominant tones in any arbitrary block, the CFFS has no such limitation. This is demonstrated in the next experiment as follows. Let y be a signal of length $N = 2^{20}$, with $m = 4$ known dominant frequencies. Let x be an arbitrary signal of length n with $N \ll n$. Now let $x[n_1, n_2]$ be an arbitrary block of interest of length N. Set $x(n_1 + q) = y(q)$, for $q = 0, 1, \ldots, N - 1$. Thus we have placed a copy of the known signal y in the block of interest. The CFFS was then applied and the four dominant frequencies in the block of interest were computed. The obtained values for frequencies and their coefficients match closely with those of the signal y and satisfy the error guarantees of AAFFT. The whole experiment was repeated with different values for n_1 (and corresponding $n_2 = n_1 + N - 1$) and the same results were obtained. Figure (4) shows the sketch of a signal x, pre-sampled in a predetermined manner (according to CFFS), with copies of y placed at arbitrary positions. Application of AAFFT to any block with copy of y gives the same results thus demonstrating the correctness of CFFS.

In the final experiment, we consider the frequency hopping signal from the first experiment. Let the block size be $N = 2^{17}$ with unknown block boundaries. Let f_1 and f_2 be the respective frequencies in two adjacent blocks ($f1$ in the left block). We consider the problem of finding the block boundary using CFFS with an analysis window of size N. The center of the window can be varied and a binary search can be performed for the block boundary in the following manner. If the center is to the left of the actual boundary, then the coefficient of $f1$ produced by AAFFT will be higher than that of the $f2$. This indicates that the center has to be moved to the right from its current position. Also the search is not strictly binary since the amount by which $f1$ coefficient is higher than $f2$ can be used to shift the center of the window to the right by an equivalent amount. This step can be iterated a few times to make the center converge to the actual block boundary. We express the error as the distance to the true boundary and determine what percentage of the block this distance is. Table (1) displays the error and how the error increases with decreasing SNR. Note that even in the case

SNR(dB)	%Error	SNR(dB)	%Error
no noise	0.39	6	0.78
10	0.58	4	0.79
8	0.70	2	1.56

Table 1: Percentage error in boundary identification.

of no noise there is some inherent ambiguity in the identification of block boundary. This uncertainty is caused by two factors. First, when the analysis window has portions of both the $f1$-block and $f2$-block, the net signal is no longer sparse due to a sudden change in frequency and has a slowly decaying spectrum. With $m = 2$ the AFFT guarantees that the error made in signal approximation is about as much as the error in optimal 2-term approximation [5]. Hence a slowly decaying spectrum implies more error in the approximation. A second and more important factor is the number of samples actually acquired from the region of uncertainty around the block boundary. From the entire block, CFFS acquires about 8% samples from the $N = 2^{17}$ present. Assuming these samples are uniformly distributed (which is not true for CFFS), the number of samples present in the region of uncertainty (0.4%) is about 40. In practice, CFFS contains even fewer samples in the uncertainty region (about 30 on average). In terms of samples actually acquired in CFFS, the boundary estimation is off by only a few samples and hence is negligible, as it does not affect the computations. This will be true for any sparse sampling method like CFFS. Furthermore, if the uncertainty were to be reduced to 0.3% say, the boundary identification would improve by only about 6 samples on average, which again is negligible. Hence the boundary identification through the above method is accurate enough for all practical purposes.

5.   Conclusions and Future Work

We described and proved a sub-linear time sparse Fourier sampling algorithm called the CFFS which along with AAFFT can be applied to compute the frequency content of sparse digital signals at any point of time. Once the block length N is selected, a sub-nyquist sampling pattern can be pre-determined and the samples can be acquired from the signal (during the runtime if required). The AAFFT can be applied to the samples corresponding to any block of length N of the signal and the dominant frequencies in that block and their coefficients can be computed in sublinear time. The algorithm requires the block length N to be fixed beforehand. Designing or extending the algorithm to work for different values of N can be considered. Adapting the algorithm to further reduce the computational complexity by using known side information about the signal can also be considered. The algorithm is also highly parallelizable and can be adapted for hardware applications. Also, we may be able to extend this sample set generation to the deterministic sampling algorithm described in [10].

Acknowledgements

References:

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

[2] I.F.Akyildiz, W.Y.Lee, M.C.Vuran, and S.Mohanty, Next generation dynamic spectrum access cognitive radio wireless networks: A survey, *Computer Networks Journal (Elsevier)*,50: 2127–2159, Sep 2006.

[3] Simon Haykin, *Communication systems*. Fourth Edition, John Wiley and Sons, 2005.

[4] A.C.Gilbert, S.Muthukrishnan, and M.J.Strauss, Improved time bounds for near-optimal sparse Fourier representations. *Proc. SPIE Wavelets XI*, 59141(A):1–15, 2005.

[5] A.C.Gilbert, M.J.Strauss, and J.A.Tropp. A Tutorial on Fast Fourier Sampling *IEEE Signal Processing Magazine*, 25(2):57–66, March 2008.

[6] G.K. Smith and D.M. Hawkins, Robust frequency estimation using elemental sets. *J.Comput.Graph.Stat*, 9(1):196–214, 2000.

[7] G. Harikumar and Y. Bresler, FIR perfect signal reconstruction from multiple convolutions: minimum deconvolver orders. *IEEE Trans.Signal Processing*, 46(1):215–218, 1998.

[8] G. Cormode and S. Muthukrishnan. Combinatorial algorithms for compressed sensing. *Proc.2006 IEEE Int.Conf.Information Sciences Systems*, 230–294, April 2006.

[9] A.C.Gilbert,   M.J.Strauss,   J.A.Tropp,   and R.Vershynin, One sketch for all: Fast algorithms for Compressed Sensing. *Proc. 39th ACM Symposium on Theory of Computing*, 237–246, June 2007.

[10] M.A.Iwen,   A deterministic sub-linear time sparse Fourier algorithm via non-adaptive compressed sensing methods. *SODA '08*, 20–29, Jan 2008.