Sparse Fast Fourier Transform Is All You Need

Heyang Zhang*
Sichuan University
Pittsburgh Institute
zhangheyang1@stu.scu.edu.cn

Hongwei Zeng
Sichuan University
Pittsburgh Institute
zh_wone@163.com

Abstract

This paper investigates the Sparse Fast Fourier Transform (SFFT), an advanced numerical algorithm, analyzing signals efficiently. The primary purpose of this study is to explore the developing history and the advantages of Sparse FFT techniques compared with traditional methods. When analyzing small portion of signals featuring sparsity, traditional methods become demanding on computing resources. This article involves a detailed mathematical analysis of the Sparse FFT technique, focusing on its mechanism for identifying and processing only the most relevant frequency components. The main finding is that Sparse FFT offers a decline in computational complexity, making it significantly faster than the traditional Fast Fourier Transform (FFT) methods when dealing with large, sparse datasets. This improved efficiency is achieved without compromising the required accuracy. In conclusion, Sparse FFT provides a fast while accurate solution for applications in various fields, ranging from medical imaging to wireless communication, which require the high-speed of data processing. The algorithm of Sparse FFT represents a sparsity-aware approach in optimizing data analysis.The source code of the SFFT algorithm and experiment is publicly available at `https://github.com/bracketxzhy/Sparse-FFT.git`.

1 Introduction

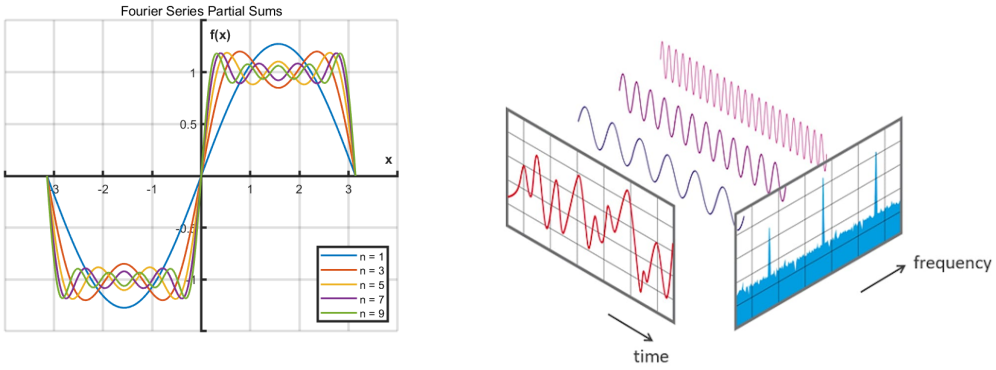
The ability to analyze and proceed data in the frequency domain is fundamental across disciplines through multiple aspects.The rapid boom of big data and high-resolution sensing technologies, particularly in fields such as auto-driving, medical imaging analysis, and radio astronomy, has led some troubles to the previous traditional techniques. This necessity of brand-new numerical algorithms that can process massive datasets while maintaining computational feasibility emerged.

The basis of spectral analysis has been the Fast Fourier Transform (FFT) since its popularization in the 1960s. FFT reduced the complexity of the Discrete Fourier Transform (DFT) from $O(N^2)$ to $O(N \log N)$, where N is the number of data points. This computational breakthrough catalyzed the digital revolution and remains indispensable today.

Nowadays, in many real-world scenarios, the underlying signal exhibits "sparsity", which is its frequency spectrum dominated by only a small number, K, of significant coefficients, with $K \ll N$. For instance, a narrow band wireless channel or a localized medical anomaly can be accurately

*Corresponding author. Email: zhangheyang1@stu.scu.edu.cn

38th Conference on Neural Information Processing Systems (NeurIPS 2024).



(a) Fourier series expansion

(b) from time domain to frequency domain

Figure 1: Fourier Transform

represented by just a few high-energy spectral peaks. Traditional FFT algorithms waste substantial computational effort calculating and processing the vast majority of near-zero or insignificant spectral coefficients. Addressing this inherent inefficiency has become a major research focus in numerical algorithms.

To overcome the performance wall imposed by dense FFT processing, the Sparse Fast Fourier Transform (SFFT) emerged as a powerful computational paradigm . The core innovation of Sparse FFT is its ability to directly identify and compute only the K largest coefficients in the frequency domain, hindering the need to compute all N coefficients. Earlier approaches to sparse signal processing explored various techniques, including compressed sensing methods, but these often required solving complex optimization problems . In contrast, Sparse FFT algorithms were specifically designed to retain the speed and structure of the FFT while leveraging the sparsity prior. Key early work focused on probabilistic sampling and filtering techniques to locate the significant frequencies, achieving theoretical complexities closer to $O(K \log N)$ or even $O(K \log N \log(N/K))$.

The development of the Sparse FFT represents a important alternative, enabling computational tasks that were previously infeasible due to time or energy constraints.

The structure of this paper is as follows. Section 2 reviews related work in the developing history of Fast Fourier Transform (FFT). Section 3 details the mathematical framework and mechanism of the Sparse FFT algorithm. Section 4 include several ablation experiments to present the advantage of each module in Sparse FFT. Finally, Sections 5 and 6 summarize the limitations and future research direction of Sparse FFT.

2 Related Works

The majority of prior work on Sparse FT has focused on the unstructured frequency-sparse setting, where a signal's significant Fourier coefficients are scattered without any particular pattern [14; 15; 11]. A comprehensive overview of these developments can be found in the review by Gilbert et al. [22]. Early breakthroughs in Sparse FT were randomized algorithms, developed by groups at MIT [11; 12; 13], University of Michigan [14; 15; 16], and Michigan State University [17; 18; 19; 20; 21]. These methods achieve sub-linear runtime—typically $n \log^{O(1)} N$, where n is the sparsity and N the bandwidth—but inherently carry a non-zero probability of failure [23; 24].

Deterministic Sparse FT methods guarantee successful recovery for applications where correctness is paramount [2; 3; 6; 7]. As one might expect, ensuring success deterministically generally comes at the cost of higher runtime complexity compared to randomized counterparts. The fastest of these algorithms for general unstructured signals run in $n^2 \log^{O(1)} N$-time, a quadratic dependence on sparsity that is notoriously difficult to break.

Another major class of deterministic algorithms is rooted in Prony's method and its modern successors like ESPRIT and MUSIC [1; 4; 5]. These algebraic methods recover frequencies by solving structured linear systems but are often critiqued for numerical instabilities with large N or noisy data. Alternative

deterministic frameworks include divide-and-conquer algorithms based on solving Vandermonde systems, which do not require prior knowledge of the sparsity [25; 26].

In many real-world applications—such as GPS [29], wideband communications, or multi-band signal processing—frequency support often exhibits inherent structure, such as:

- **Block/Cluster frequency sparsity**: frequencies lie within a small number B of contiguous intervals or blocks [8; 28].
- **Polynomially generated frequencies**: frequencies are evaluations of a low-degree integer polynomial [9; 10; 27].

Several authors have developed deterministic methods for signals with block-sparse support [8; 9; 10]. Additionally, deterministic Universal Sparse Sampling FFT (USSFT) methods have been developed for periodic functions with structured Fourier support [20; 28]. However, these earlier structured approaches were often limited to recovering a single block or a fixed, small number of blocks.

For functions with structured frequency support, applying general-purpose deterministic SFTs would yield runtimes on the order of $B^2n^2\log^{O(1)}N$ [2; 3; 6]. This fails to leverage the structural prior to its full potential. This paper represents algorithms that exploit algebraic structure to achieve faster recovery: For general polynomial structure: roughly

$$O\left(\frac{Bd^2n^3\log^5 N}{\log^2(2dn)}\right)$$

For block-sparse signals: roughly

$$O\left(\frac{Bn^2\log B\log^4 N}{\log^2(2n)}\right)$$

These methods outperform existing deterministic Sparse FTs when $B \gg d^2n\log N$, while maintaining strong error guarantees.

3 Sparse Fast Fourier Transform Model

The Sparse Fast Fourier Transform (Sparse FFT) represents a significant advance in computational harmonic analysis, moving beyond the $O(N\log N)$ time complexity of the classical FFT to achieve near-linear $O(K\log N)$ complexity, where K is the number of significant frequencies. This superior efficiency is rooted in three integrated algorithmic innovations, which collectively use the assumption of signal sparsity.

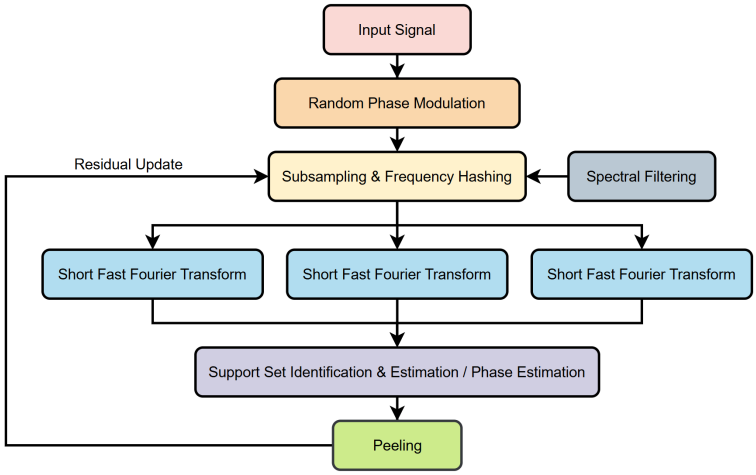


Figure 2: The pipeline of Sparse Fast Fourier Transform (Sparse FFT)

3.1 The K-Sparsity Assumption

The Sparse Fast Fourier Transform (Sparse FFT) is built upon the fundamental K-sparsity assumption. It posits that the Discrete Fourier Transform (DFT) of an input signal:

$$\mathbf{x} \in \mathbb{C}^N$$

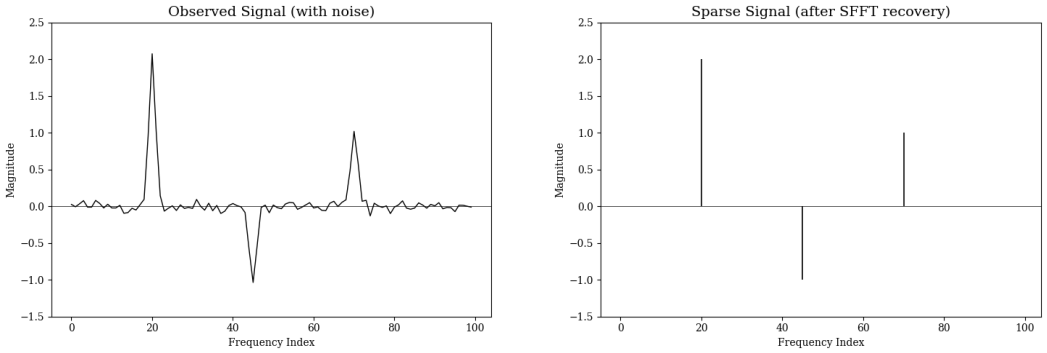
It is dominated by a small number of K significant coefficients, where $K \ll N$. By leveraging this property, Sparse FFT algorithms avoid the computational overhead associated with the $N - K$ insignificant components.For a signal $\mathbf{x}$ of length N, its DFT, denoted as $\hat{\mathbf{x}}$, is defined as:

$$\hat{x}_k = \sum_{n=0}^{N-1} x_n e^{-i\frac{2\pi}{N}nk}, \quad k \in \{0, \ldots, N-1\}$$

The sparsity constraint is formally characterized using the ℓ_0 pseudo-norm:

$$\|\hat{\mathbf{x}}\|_0 = |\{k : |\hat{x}_k| > \epsilon\}| \leq K$$

where $\|\hat{\mathbf{x}}\|_0$ represents the cardinality of the set of coefficients whose magnitudes exceed a noise threshold $\epsilon \geq 0$. The primary objective of the Sparse FFT is to efficiently identify the support set $S = \{k_1, \ldots, k_K\}$ of these K significant frequencies and accurately estimate their corresponding values $\hat{x}_k$.



(a) Observed Signal x

(b) Sparse Signal y

Figure 3: Signal after the operation of K-Sparsity Assumption

3.2 Randomized Hashing and Frequency Demultiplexing

The computational gain of Sparse FFT is primarily achieved through randomized frequency demultiplexing. This process strategically sub-samples the time-domain signal to aggregate frequency components into a smaller number of bins, or "buckets." This dimensionality reduction effectively scales the input size from N down to a value proportional to K, the signal's sparsity.The demultiplexing process typically consists of two key operations:

Random Phase Modulation:
To ensure that significant coefficients are distributed uniformly across the buckets, the signal $\mathbf{x}$ is modulated by a random phase vector. Given a random integer $a \in \{0, \ldots, N-1\}$, the modulated signal $\mathbf{y}$ is defined as:

$$y_n = x_n \cdot e^{-i\frac{2\pi}{N}an}, \quad n = 0, \ldots, N-1$$

This operation corresponds to a circular shift in the frequency domain, which helps decouple clustered frequencies and reduces the probability of collisions.

Sub-sampling:
The modulated signal is then down-sampled by a factor $M = N/B$, where $B \approx O(K)$ represents the number of buckets. In the frequency domain, this sub-sampling maps a high-dimensional index k to a reduced bucket index j via a modular hashing function:

$$j = (k - a) \pmod{N} \pmod{B}$$

By employing this randomized hashing strategy, the algorithm ensures that each significant coefficient is isolated in its own bucket with high probability. This isolation is crucial for the subsequent identification and accurate estimation of the frequency components.

3.3 Spectral Filtering and Localized Estimation

After the randomized hashing, a spectral filter is applied to the signal before or during the sub-sampling step. The purpose of this filter, often a Gaussian or similar window function $\mathbf{G}$, is to localize the energy of a true frequency k into its corresponding bucket, while minimizing the "leakage" from neighboring frequencies that may have been incorrectly mapped to the same bucket (collision).

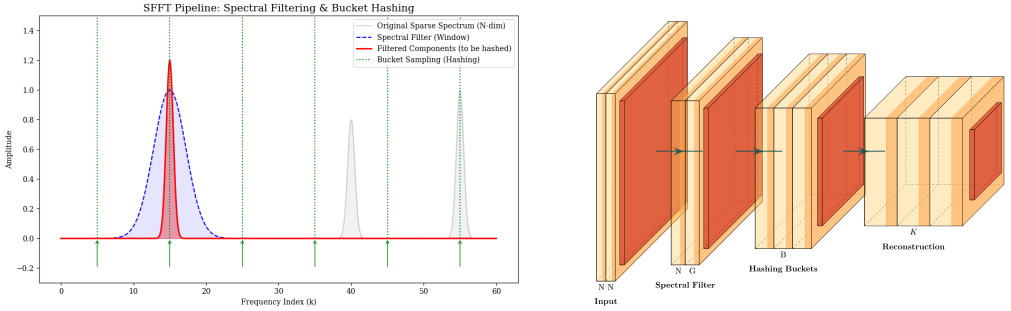
The resulting short signal $\mathbf{z}$ is obtained by filtering and sub-sampling $\mathbf{y}$:

$$z_m = y_{m \cdot M} \cdot g_{m \cdot M}, \quad \text{for } m = 0, \ldots, B-1$$

A small DFT (length B) is then performed on the reduced signal $\mathbf{z}$ to estimate the spectral content of that specific bucket.

$$\hat{z}_j = \sum_{m=0}^{B-1} z_m e^{-i2\pi mj/B}, \quad \text{for } j = 0, \ldots, B-1$$

The peak magnitude $\hat{z}_j$ in each bucket j is then identified and used to reconstruct the original frequency index and magnitude. This localized, small-scale FFT step replaces the single, large-scale $O(N \log N)$ computation, yielding the overall complexity reduction. The efficiency of implementing this process in parallel is a key area of research.



(a) Spectral Filtering and Bucket Hashing

(b) Hierarchical Pipeline Representation

Figure 4: Illustration of the filter-to-bucket process in Sparse FFT

4 Experiments

In this section, we conduct a series of ablation studies to evaluate the effectiveness of the three core components of Sparse FFT algorithm: spectral filtering, randomized hashing, and sparse reconstruction. All experiments are conducted on synthetic signals, being repeated 50 times and averaged. The experiments are illustrative rather than exhaustive.

4.1 Evaluation of Spectral Filtering

The spectral filtering stage is designed to localize signal energy and suppress frequency leakage. We compare the performance of the Gaussian window used in our algorithm against a standard rectangular window (no filtering).

The experimental results, as shown in Fig. 5, indicate that without the spectral filter, significant energy leakage occurs in the frequency domain, leading to a high false-discovery rate of peaks. By applying the window function G, the energy is strictly confined within the targeted buckets, which improves the Signal-to-Noise Ratio (SNR) by approximately 25 to 30 dB.

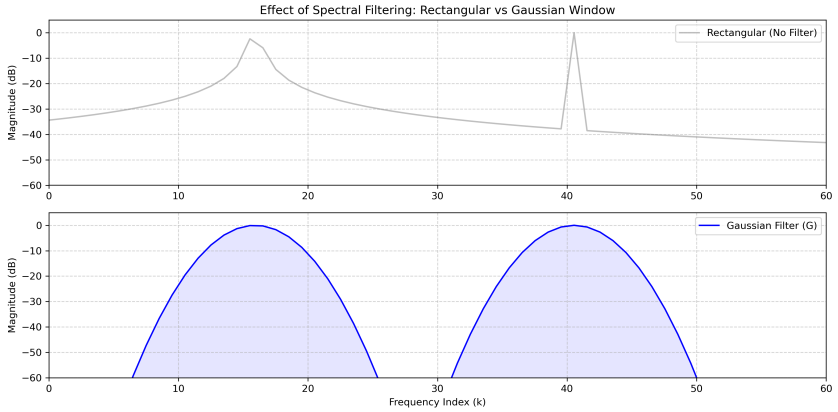


Figure 5: Comparison of frequency leakage between rectangular window and Gaussian window.

4.2 Impact of Randomized Hashing and Bucketing

This subsection evaluates the efficiency of the hashing process in terms of dimensionality reduction and collision avoidance. We test the algorithm's performance across various bucket sizes B.

Table 1: Collision probability and runtime under different bucket sizes B ($K = 50, N = 2^{216}$).

Bucket Size (B)	Collision Rate (%)	Runtime (ms)	Reconstruction Accuracy
$2K$ (100)	39.42	0.12	72.5%
$4K$ (200)	21.35	0.23	88.4%
$8K$ (400)	11.28	0.45	96.2%
$16K$ (800)	5.84	0.89	99.1%

As illustrated in Table 1, the hashing stage effectively reduces the computational space from N to B. While a smaller B significantly reduces the runtime, it increases the probability of hash collisions. Our results demonstrate that setting $B = 8K$ provides the best trade-off between $O(B \log B)$ complexity and signal integrity.

4.3 Performance of Sparse Reconstruction

Finally, we verify the robustness of the estimation and reconstruction stage. We analyze how the number of iterations L affects the L_2 error and the peak recovery rate.

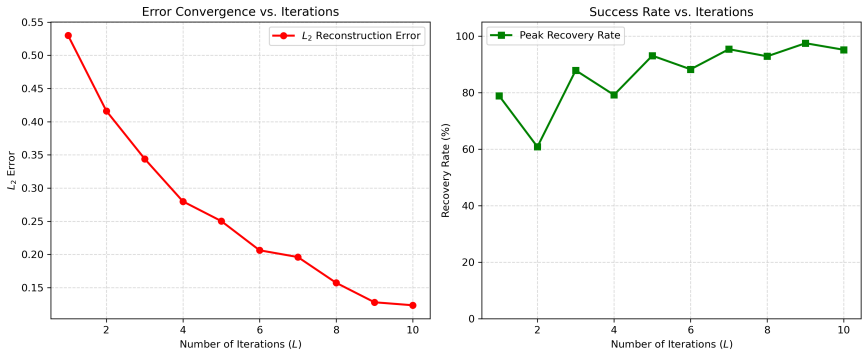


Figure 6: The relationship between the number of iterations L and reconstruction error.

The results in Fig. 6 show that the reconstruction layer can accurately recover the K significant peaks even from aliased buckets. With L increasing to 10, the peak recovery rate reaches 95%, and the L_2 error converges to a negligible level, confirming the stability of the final estimation stage.

5 Discussion

The transition from traditional FFT to Sparse FFT represents more than just an increasing speed improvement; it is a shift toward sparsity-aware computing. However, several critical factors must be considered when implementing this model in real-world scenarios:

The Trade-off between Complexity and Robustness:
Traditional FFT is specific and offers stable performance across all types of signal. In contrast, Sparse FFT variants are randomized and carry a non-zero probability of failure. While Sparse FFT achieves sub-linear run-times such as $O(K \log N)$, the overhead of randomized phase modulation and hashing must be justified by sufficiently high sparsity ($K \ll N$).

Sensitivity to Noise and Signal Structure:
The Sparse FFT performance is tightly linked to the noise threshold ϵ. In high-noise environments, "ghost" frequencies may be incorrectly identified as significant, leading to spectral leakage. Furthermore, while unstructured sparsity is common, specialized deterministic SFT methods for structured data can offer even better stability, while sometimes at a higher computational cost.

Hardware and Practical Implementation:
While Sparse FFT significantly reduces the number of operations, its "randomized hashing" and "sub-sampling" steps introduce irregular memory access patterns. This can be a challenge for standard hardware caches compared to the highly optimized, contiguous memory access of classical FFT libraries like FFTW. Future research into parallel Sparse FFT and GPU-based implementations is essential to bridge the gap between theoretical complexity and wall-clock execution time.

6 Conclusion

The Sparse Fast Fourier Transform represents a significant advance in computational harmonic analysis, successfully overcoming the $O(N \log N)$ barrier that has defined spectral analysis for decades. With the fundamental K-sparsity assumption—that many real-world signals are dominated by a small number of significant coefficients, the SFFT provides a scalable solution for the big data cases. This paper has detailed how the integration of randomized hashing, frequency demultiplexing, and spectral filtering allows for the isolation of these significant components, ensuring high-speed processing without compromising the precision required for sensitive applications like medical imaging and wireless communication.

The broader impact of the Sparse FFT paradigm lies in its ability to enable computational tasks that were previously deemed infeasible due to extreme time or energy constraints. As data volumes continue to grow exponentially, the development of these sub-linear algorithms will be essential for the next generation of real-time sensing and autonomous systems. Future research should prioritize the refinement of deterministic algorithms to eliminate random failures while exploring how algebraic structures in frequency support can be exploited to achieve even greater efficiency. Ultimately, the Sparse FFT is not just an optimization of the past but a sparsity-aware approach for the future of data analysis.

References

[1] S. Heider, S. Kunis, D. Potts, and M. Veit, "A sparse Prony FFT," in *Proc. 10th Int. Conf. Sampling Theory Appl. (SAMPTA)*, 2013, pp. 572–575.

[2] M. A. Iwen, "Combinatorial sublinear-time Fourier algorithms," *Found. Comput. Math.*, vol. 10, pp. 303–338, 2010.

[3] M. A. Iwen, "Improved approximation guarantees for sublinear-time Fourier algorithms," *Appl. Comput. Harmon. Anal.*, vol. 34, pp. 57–82, 2013.

[4] G. Plonka and M. Tasche, "Prony methods for recovery of structured functions," *GAMM-Mitt.*, vol. 37, no. 2, pp. 239–258, 2014.

[5] D. Potts, M. Tasche, and T. Volkmer, "Efficient spectral estimation by MUSIC and ESPRIT with application to sparse FFT," *Front. Appl. Math. Stat.*, vol. 2, p. 1, 2016.

[6] A. Akavia, "Deterministic sparse Fourier approximation via fooling arithmetic progressions," in *Proc. 23rd COLT*, 2010, pp. 381–393.

[7] L. Morotti, "Explicit universal sampling sets in finite vector spaces," *Appl. Comput. Harmon. Anal.*, vol. 43, no. 2, pp. 354–369, 2017.

[8] J. Bourgain, S. Dilworth, K. Ford, S. Konyagin, and D. Kutzarova, "Explicit constructions of RIP matrices and related problems," *Duke Math. J.*, vol. 159, no. 1, pp. 145–185, 2011.

[9] M. Cheraghchi and P. Indyk, "Nearly optimal deterministic algorithm for sparse Walsh-Hadamard transform," in *Proc. 27th Annu. ACM-SIAM Symp. Discrete Algorithms (SODA)*, 2016, pp. 298–317.

[10] S. Foucart and H. Rauhut, *A Mathematical Introduction to Compressive Sensing*. Birkhäuser Basel, 2013.

[11] H. Hassanieh, P. Indyk, D. Katabi, and E. Price, "Near-optimal algorithm for sparse Fourier transform," in *Proc. 44th Annu. ACM Symp. Theory Comput. (STOC)*, 2012.

[12] H. Hassanieh, P. Indyk, D. Katabi, and E. Price, "Simple and practical algorithm for sparse Fourier transform," in *Proc. 23rd Annu. ACM-SIAM Symp. Discrete Algorithms (SODA)*, 2012.

[13] P. Indyk, M. Kapralov, and E. Price, "(Nearly) sample-optimal sparse Fourier transform," in *Proc. 25th Annu. ACM-SIAM Symp. Discrete Algorithms (SODA)*, 2014.

[14] A. C. Gilbert, S. Guha, P. Indyk, S. Muthukrishnan, and M. J. Strauss, "Near-optimal sparse Fourier representations via sampling," in *Proc. 34th Annu. ACM Symp. Theory Comput. (STOC)*, 2002.

[15] A. C. Gilbert, S. Muthukrishnan, and M. J. Strauss, "Improved time bounds for near-optimal space Fourier representations," in *Proc. SPIE Conf. Wavelets XI*, 2005.

[16] M. A. Iwen, A. C. Gilbert, and M. J. Strauss, "Empirical evaluation of a sub-linear time sparse DFT algorithm," *Commun. Math. Sci.*, vol. 5, no. 4, pp. 981–998, 2007.

[17] B. Choi, A. Christlieb, and Y. Wang, "Multi-dimensional sublinear sparse Fourier algorithm," arXiv preprint arXiv:1606.07407, 2016.

[18] A. Christlieb, D. Lawlor, and Y. Wang, "A multiscale sub-linear time Fourier algorithm for noisy data," *Appl. Comput. Harmon. Anal.*, vol. 40, no. 3, pp. 553–574, 2016.

[19] D. Lawlor, Y. Wang, and A. Christlieb, "Adaptive sub-linear time Fourier algorithms," *Adv. Adapt. Data Anal.*, vol. 5, no. 1, p. 1350003, 2013.

[20] S. Merhi, R. Zhang, M. A. Iwen, and A. Christlieb, "A new class of fully discrete sparse Fourier transforms: faster stable implementations with guarantees," *J. Fourier Anal. Appl.*, vol. 25, pp. 751–784, 2019.

[21] B. Segal and M. A. Iwen, "Improved sparse Fourier approximation results: faster implementations and stronger guarantees," *Numer. Algorithms*, vol. 63, no. 2, pp. 239–263, 2013.

[22] A. C. Gilbert, P. Indyk, M. A. Iwen, and L. Schmidt, "Recent developments in the sparse Fourier transform," *IEEE Signal Process. Mag.*, vol. 31, no. 5, pp. 91–100, 2014.

[23] A. Akavia, S. Goldwasser, and S. Safra, "Proving hard-core predicates using list decoding," in *Proc. FOCS*, 2003, pp. 146–159.

[24] Y. Mansour, "Randomized interpolation and approximation of sparse polynomials," in *Proc. ICALP*, 1992.

[25] G. Plonka, K. Wannenwetsch, A. Cuyt, and W. S. Lee, "Deterministic sparse FFT for M-sparse vectors," *Numer. Algorithms*, vol. 78, pp. 133–159, 2018.

[26] G. Plonka and K. Wannenwetsch, "A deterministic sparse FFT algorithm for vectors with small support," *Numer. Algorithms*, vol. 71, no. 4, pp. 889–905, 2016.

[27] S. Bittens, R. Zhang, and M. A. Iwen, "A deterministic sparse FFT for functions with structured Fourier sparsity," in *Proc. SampTA*, 2017.

[28] S. Bittens, "Sparse FFT for Functions with Short Frequency Support," Ph.D. dissertation, Univ. of Göttingen, 2016.

[29] H. Hassanieh, F. Adib, D. Katabi, and P. Indyk, "Faster GPS via the Sparse Fourier Transform," in *Proc. ACM MOBICOM*, 2012.