Open access • Proceedings Article • DOI:10.1109/ICASSP.2013.6638743

Sparse Fast Fourier Transform by downsampling — Source link

Sung-Hsien Hsieh, Chun-Shien Lu, Soo-Chang Pei

Institutions: National Taiwan University

Published on: 26 May 2013 - International Conference on Acoustics, Speech, and Signal Processing

Topics: Split-radix FFT algorithm, Prime-factor FFT algorithm, Fast Fourier transform, Discrete Fourier transform and Non-uniform discrete Fourier transform

SPARSE FAST FOURIER TRANSFORM BY DOWNSAMPLING

Sung-Hsien Hsieh[*,**]*, Chun-Shien Lu*[*]*, and Soo-Chang Pei*[**]

[*]Institute of Information Science, Academia Sinica, Taipei, Taiwan
[**]Graduate Inst. Comm. Eng., National Taiwan University, Taipei, Taiwan
email: {parvaty316, lcs}@iis.sinica.edu.tw, pei@cc.ee.ntu.edu.tw

ABSTRACT

Sparse Fast Fourier Transform (sFFT) [1][2], has been recently proposed to outperform FFT in reducing computational complexity. Assume that an input signal of length N in the frequency domain is K-sparse, where $K \leq N$. sFFT costs $O(K \log N)$ instead of $O(N \log N)$ in FFT.

In this paper, a new fast sFFT algorithm is proposed and costs $O(K \log K)$ averagely without any operations being related to N. The idea is to downsample the original input signal at the beginning. Subsequent processing operates under downsampled signals, which length is proportional to $O(K)$. However, downsampling possibly leads to "aliasing." By shift theorem of DFT, the aliasing problem can be formulated as the "Moment-preserving problem." In addition, a top-down iterative strategy combined with different downsampling factors further saves computational costs. Complexity analysis and experimental results show that our method outperforms FFT and sFFT.

Index Terms— Sparsity, FFT, Sparse FFT, Downsampling

1. INTRODUCTION

Fast Fourier transform (FFT) is a well-known approach for computing DFT with $O(N \log N)$, where N is the length of a signal. How to outperform FFT is a significant challenge in the signal processing community.

Recently, the researchers in MIT propose, as a breakthrough, a new technique, called Sparse Fast Fourier Transform (sFFT) [1][2], that is proved to outperform FFT. Let $\boldsymbol{x} \in C^N$ be the input signal in the time domain and let $\boldsymbol{X} \in C^N$ be the Fourier transform of $\boldsymbol{x}$. Assume that x is K-sparse in that there are K non-zero entries in $\boldsymbol{X}$, *i.e.*, $supp(\boldsymbol{X}) = K$ and $K \leq N$. sFFT costs $O(K \log N)$.

The idea behind sFFT is to sample fewer (proportional to K) instead of keeping all frequency grids since most frequency grids are zero and do not need to be calculated. FFT based on such subsampling strategy will only cost $O(K \log K)$ calculations. However, because the locations and values of the K non-zero entries are unknown, subsampled frequency grids often lead to data loss and cannot achieve perfect reconstruction.

In order to cope with this difficulty, sFFT is proposed to include the strategies of filtering and permutation that can increase the probability of capturing useful information from subsampled frequency grids. These operations cost $O(K \log N)$. According to [1][2], sFFT is faster than FFTW [3] (a very fast C subroutine library for computing FFT) when $\boldsymbol{X}$ is an exact K-sparse signal with $K \leq \frac{N}{2^6}$. sFFT also outperforms previous works such as [4][5].

Even though sFFT is outstanding, there are some limitations summarized as follows. 1) Filtering and permutation are operated on $\boldsymbol{x}$. Since $\boldsymbol{x} \in C^N$, these operations are related to N. Thus, sFFT is still influenced by N and cannot achieve the most ideal complexity $O(K \log K)$. 2) sFFT only succeeds with a constant probability; *i.e.,* it possibly fails.

In this paper, a new fast sFFT algorithm with complexity of $O(K \log K)$ by downsampling in the time domain is proposed and dubbed as sFFT-DT. The idea behind sFFT-DT is to downsample the original input signal first and then all subsequent operations are conducted on the downsampled signals. When the length of a downsampled signal is $O(K)$, no operations related to N are required in our method. However, downsampling possibly leads to "aliasing," where different signals become indistinguishable in terms of their locations and values. To overcome this problem, we consider the locations and values of K non-zero entries as variables and the "aliasing problem" is found to be equivalent to "Moment-preserving problem," which can be solved via orthogonal polynomials [6]. Moreover, our method, conducted in a manner of top-down iterative strategy under different downsampling factors, can further reduce computational complexity. Our method sFFT-DT is analytically and experimentally verified to outperform FFT and sFFT.

2. PROPOSED METHOD: SFFT-DT

We describe the proposed method and analyze its computational complexity. The proposed method contains three steps: 1) Downsample the original signal. 2) Calculate Fourier transform of the downsampled signal by FFT. 3) The Fourier transform of the downsampled signal is used to locate and estimate K non-zero entries of $\boldsymbol{X}$. Steps 1 and 2 are simple and straightforward. Thus, we focus on Step 3 here.

2.1. Problem Formulation

Let $\boldsymbol{x_d}$ be the downsampled signal, where $x_d[k] = x[dk]$, $k \in [0, \frac{N}{d} - 1]$, and d is the downsampling factor. Let $\boldsymbol{X_d}$ be discrete Fourier transform (DFT) of $\boldsymbol{x_d}$, where

$$\begin{aligned} X_d[k] =& (X[k] + X[k + \frac{N}{d}] + X[k + 2\frac{N}{d}] \\ & ... + X[k + (d-1)\frac{N}{d}])/d. \end{aligned} \tag{1}$$

Note that each frequency grid of $\boldsymbol{X_d}$ is a summation of d terms of $\boldsymbol{X}$. When more than two terms of $\boldsymbol{X}$ are non-zero, "aliasing" occurs, as illustrated in Fig. 1. Fig. 1(a) shows an original signal in the frequency domain, where only 3 frequency grids are non-zero. Fig. 1(b) shows the downsampled signal in the frequency domain when $d = 2$. Aliasing appears at 0.8π because two terms are summed together. Fig. 1 will be further explained in detail in Sec. 2.3.

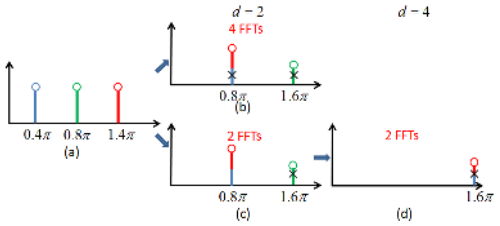


Fig. 1. Aliasing and its iterative solver. (a) Original signal in frequency domain. (b) Downsampled signal in frequency domain with $d = 2$. If we want to solve all frequency grids once, it requires 4 FFTs. (c) Similar to (b), however, frequency grids at $d = 2$ are solved first and require 2 FFTs. (d) Remaining frequency grids require 2 extra FFTs at $d = 4$.

Another useful property of DFT is the shift theorem. Let $x_{d,l}[k] = x[dk + l]$, where l denotes the shift factor. Each element of $\boldsymbol{X_{d,l}}$ is denoted as:

$$\begin{aligned} X_{d,l}[k] = (X[k]e^{i2\pi kl/N} + X[k + \frac{N}{d}]e^{i2\pi(k+\frac{N}{d})l/N} + \\ ... + X[k + (d-1)\frac{N}{d}]e^{i2\pi(k+(d-1)\frac{N}{d})l/N})/d. \end{aligned} \tag{2}$$

Thus, Eq. (2) degenerates to Eq. (1) when $l = 0$.

In practice, all we can obtain are $X_{d,l}[k]$'s for different l's. For each downsampling factor d, there will be no more than d terms on the right side of Eq. (2), where each term contains two unknown variables. For example, $X[k]e^{i2\pi kl/N}$ is composed of two variables, $X[k]$ and $e^{i2\pi kl/N}$. Let a, $1 \leq a \leq d$, denote the number of terms on the right side of Eq. (2). Therefore, we need $2a$ equations to solve these $2a$ variables, and l is within the range of $[0, 2a - 1]$. By taking the above into consideration, the problem can be formulated as:

$$\begin{aligned} p_0 z_0^0 + p_1 z_1^0 + ... + p_{a-1} z_{a-1}^0 &= m_0, \\ p_0 z_0^1 + p_1 z_1^1 + ... + p_{a-1} z_{a-1}^1 &= m_1, \\ &\vdots \\ p_0 z_0^{2a-1} + p_1 z_1^{2a-1} + ... + p_{a-1} z_{a-1}^{2a-1} &= m_{2a-1}, \end{aligned} \tag{3}$$

where $X_{d,l}[k]$ is known and is denoted as m_l while p_j and z_j^l, respectively, represent unknown $X[s_j]$ and $e^{i2\pi s_j l/N}$ for $s_j \in \{k,\ k + \frac{N}{d}\ , ... , k + (d-1)\frac{N}{d}\}$ and $j \in [0, a-1]$.

It is trivial that no aliasing occurs if $a = 1$, irrespective of whatever the downsampling factor is. Under this circumstance, we have $m_0 = X_{d,0}[k]$, $m_1 = X_{d,1}[k]$, $m_0 = p_0 z_0^0 = X[s_0]/d$, and $m_1 = p_0 z_0^1 = X[s_0]e^{i2\pi s_0/N}/d$, according to Eq. (3). It is easy to obtain that $|m_0| = |X[s_0]|/d = |m_1|$ and $m_1/m_0 = e^{i2\pi s_0/N}$. After some derivations, we can solve s_0 and obtain $X[s_0] = dX_{d,0}[k]$ at the position s_0. The above solution is based on the shift theorem of DFT and can only work under a non-aliasing environment. However, when aliasing appears (*i.e.*, $a > 1$), Eq. (3) is unsolvable because $\frac{m_1}{m_0} = \frac{p_0 z_0^1 + p_1 z_1^1}{p_0 z_0^0 + p_1 z_1^0}$ for $a = 2$.

To cope with the aliasing problem, we consider Eq. (3) from another point of view. It is observed that the i'th row in Eq. (3) is the i'th moment with $m_i = \sum_{j=0}^{a-1} p_j z_j^i$. The issue of solving p_j's and z_j's given different moments (m_i's) is the "Moment-preserving problem (MPP)." We find that the solution to MPP [6][7] based on orthogonal polynomials is useful and will be discussed in the next subsection.

2.2. The Solution to Moment-Preserving Problem

Note that the moment-preserving problem (Eq. (3)) is nonlinear and cannot be solved by simple matrix operations. On the contrary, we have to solve z_j's first such that Eq. (3) becomes linear. Then, p_i's can be solved by matrix inversion. Thus, the main difficulty is how to solve z_j's given known moments. According to [6], given the unique moments with m_0, m_1, ..., m_{2a-1}, there must exist the corresponding orthogonal polynomial equation, $P(z)$, with roots z_j's for $0 \leq j \leq a-1$. Then, z_j's can be obtained as the roots of $P(z)$. The steps of solving MPP are as follows.
(i) Let the orthogonal polynomial equation $P(z)$ be:

$$P(z) = z^a + c_{a-1} z^{a-1} + ... + c_1 z + c_0. \tag{4}$$

The relationship between $P(z)$ and the moments is:

$$\begin{aligned} c_0 m_0 + c_1 m_1 + ... + c_{a-1} m_{a-1} &= -m_a, \\ c_0 m_1 + c_1 m_2 + ... + c_{a-1} m_a &= -m_{a+1}, \\ &\vdots \\ c_0 m_{a-1} + c_1 m_a + ... + c_{a-1} m_{2a-2} &= -m_{2a-1}. \end{aligned} \tag{5}$$

Eq. (5) is solved by matrix inversion to obtain c_j's.
(ii). Find the roots of $P(z)$ in Eq. (4). These roots are the solutions of z_0, z_1,...z_{a-1}, respectively.

(iii). Substitute all z_j's into Eq. (3) and solve the resulting equations to obtain p_j's.

Tsai [7] proposed a complete analytic solution composed of the above three steps for $a \leq 4$ based on the constraint that $p_0+p_1+...+p_{a-1} = 1$. Nevertheless, for the aliasing problem considered here, the constraint is $p_0 + p_1 + ... + p_{a-1} = X_{d,0}[k]$, as indicated in Eq. (2). Thus, the complete analytic solution is derived for $a = 2$ as:

$$
\begin{aligned}
&c_d = \begin{vmatrix} m_0 & m_1 \\ m_1 & m_2 \end{vmatrix}, \\
&c_0 = (\frac{1}{c_d}) \begin{vmatrix} -m_2 & m_1 \\ -m_3 & m_2 \end{vmatrix}, \; c_1 = (\frac{1}{c_d}) \begin{vmatrix} m_0 & -m_2 \\ m_1 & m_3 \end{vmatrix}, \\
&z_0 = \frac{1}{2}[-c_1 - (c_1^2 - 4c_0)^{\frac{1}{2}}], \; z_1 = \frac{1}{2}[-c_1 + (c_1^2 - 4c_0)^{\frac{1}{2}}], \\
&p_d = z_1 - z_0, \\
&p_0 = (\frac{1}{p_d}) \begin{vmatrix} m_0 & 1 \\ m_1 & z_1 \end{vmatrix}, \; p_2 = m_0 - p_0.
\end{aligned}
\tag{6}
$$

Eq. (6) costs $O(a^3)$ operations. In other words, even though aliasing occurs for all K non-zero entries of $\boldsymbol{X}$, $O(Ka^3)$ is required under the worst case, which is unrelated to N. However, there is no close-form solution to step (ii) for $a > 4$. Under the situation, (ii) can be solved by numerical analysis like Newton's method.

In general, aliasing seldom occurs if the locations of non-zero entries of $\boldsymbol{X}$ are random. Let $N^+ = \frac{N}{dK}$ be the ratio of the length ($\frac{N}{d}$) of a downsampled signal to K. Fig. 2 shows the probability of aliasing ($a \geq 2$) at different N^+'s. For $a > 4$, the probability is very low. In other words, $6 \sim 8$ FFTs in downsampled signals are enough to recover most frequency grids of $\boldsymbol{X}$. However, when the signal is not so sparse with K approaching N (*e.g.*, $K = \frac{N}{8}$ and $N^+ = 2^0$), the cost of 8 FFTs in downsampled signals is almost equivalent to that of one FFT in the original signal. To further reduce the cost, a top-down iterative strategy is proposed in the next subsection.

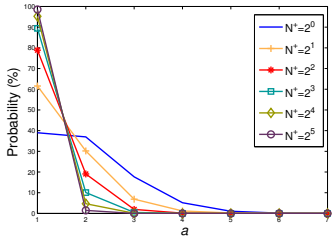


Fig. 2. Probability of aliasing at different N^+'s. a denotes the number of terms on the right side of Eq. (2). The results show that aliasing ($a \geq 2$), in fact, seldom occurs.

2.3. Top-Down Iterative Strategy

An iterative strategy is proposed to solve the aliasing problem with iterative increase of the downsampling factor d. Fig 1 illustrates an example. In Fig. 1(b), if we try to solve all aliasing problems at one iteration, 4 FFTs are required since the maximum value of a is 2. If we first solve the problem with $a = 1$, it costs 2 FFTs, as shown in Fig. 1(c). Since 2 FFTs are not enough for solving the aliasing problem under $a = 2$, 2 extra FFTs are required.

The key is how to calculate 2 extra FFTs with less cost in the above example. The idea motivated by sFFT is to discard the solved frequency grids in the frequency domain. Thus, if K' frequency grids are subtracted from the original X, the sparsity of the remaining signal is $K - K'$. Since a more sparse signal is generated in an iterative manner, d can be set to be larger under fixed N^+. As shown in Fig. 1(d), 2 extra required FFTs can be fast done with a larger d (=4). From Fig. 2, the probability at $a = i$ is at least 2 times larger than that at $a = i + 1$ for $N^+ = 2^1$ and $i \geq 1$. Consequently, d is doubled iteratively and the cost of total FFTs is bound by that required at the first iteration.

2.4. Algorithm

Algorithm 1 depicts the proposed algorithm, sFFT-DT, which is composed of three functions, **main**, **SubFreq** and **MPP**. At the initialization stage, the sets S and T record the positions of solved and unsolved frequency grids, respectively.

The function **main** is executed in a top-down manner by doubling the downsampling factor iteratively. It should be noted that in the function **main**, $X_{d,j}[k] = 0$, initially defined in Eq. (2), may imply: 1) $X[k + j\frac{N}{d}]$'s for all $j \in [0, d-1]$ are zero and 2) $X[k + j\frac{N}{d}]$'s are non-zero but their sum is zero. To distinguish both, $|X_{d,j}[k]| > 0$, $j \in [0, 2l + 1]$, is a sufficient condition. If the number of aliasing frequency grids (unknowns) is less than or equal to $2l + 2$, it is enough to distinguish both by checking whether any one of the $2l + 2$ equations is not equal to 0. If yes, it implies that at least a frequency gird is non-zero; otherwise, all $X[k + j\frac{N}{d}]$'s are definitely zero. More specifically, the condition (Line 9) is equivalent to checking $2l + 2$ equations at l'th iteration. At $l = 0$, two equations ($X_{d,0}[k]$ and $X_{d,1}[k]$) are checked to ensure that all frequency grids with $a \leq 2$ are distinguished. At $l = 1$, if $k \in T$, it is confirmed that $X[k + j\frac{N}{d}]$'s are non-zero at the previous iteration. On the contrary, if $k \notin T$, extra 2 equations ($X_{d,2}[k]$ and $X_{d,3}[k]$) are added to ensure that all frequency grids with $a \leq 4$ are distinguished. Thus, at l'th iteration, total $2l + 2$ equations are checked.

In addition, due to the iterative framework, we solve the aliasing problem via the function **MPP** by assuming the number of unknown $\boldsymbol{s}$ (locations) and $\boldsymbol{p}$ (estimations) in advance. If the assumption is true, the condition, $s_j \; mod \; d = k$ for all $j \in [0, l]$ (Line 30), must be true. For each iteration, the solved frequency grids are subtracted from the original frequency grids via the function **SubFreq** such that the resultant signal is more sparser and can be recovered using larger downsampling factors.

Algorithm 1 The Proposed Algorithm: sFFT-DT

Input: $\boldsymbol{x}, t, K$; **Output:** $\boldsymbol{X}$;
Initialization: $\boldsymbol{X} = \mathbf{0}, d = O(\frac{N}{K}), S = \{\}, T = \{\}$;

01. **function main**()
02. **for** $l = 0$ to $t-1$
03. $x_{d,2l}[k] = x[dk + 2l]$ for $k \in [0, \frac{N}{d} - 1]$;
04. $x_{d,2l+1}[k] = x[dk + 2l + 1]$ for $k \in [0, \frac{N}{d} - 1]$;
05. $\boldsymbol{X_{d,2l}} = \mathbf{FFT}(\boldsymbol{x_{d,2l}}) \times d$;
06. $\boldsymbol{X_{d,2l+1}} = \mathbf{FFT}(\boldsymbol{x_{d,2l+1}}) \times d$;
07. $\mathbf{SubFreq}(\boldsymbol{X_{d,2l}}, \boldsymbol{X_{d,2l+1}}, \boldsymbol{X}, d, l, S)$;
08. **for** $k = 0$ to $\frac{N}{d} - 1$
09. **if** $(k \in T$ or $|X_{d,2l}[k]| > 0$ or $|X_{d,2l+1}[k]| > 0)$
10. $m_j = X_{d,j}[k]$ for $j \in [0, 2l + 1]$;
11. $\mathbf{MPP}(\boldsymbol{m}, l, d, k, \boldsymbol{X}, S, T)$;
12. **end if**
13. **end for**
14. $d = 2d$;
15. All elements in T modulo $\frac{N}{d}$.
16. **end for**
17. **function SubFreq** $(\boldsymbol{X_{d,2l}}, \boldsymbol{X_{d,2l+1}}, \boldsymbol{X}, d, l, S)$
18. **for** $k \in S$
19. $k_d = k$ mod $\frac{N}{d}$;
20. $X_{d,2l}[k_d] = X_{d,2l}[k_d] - X[k]e^{\frac{i2\pi k(2l)}{N}}$;
21. $X_{d,2l+1}[k_d] = X_{d,2l+1}[k_d] - X[k]e^{\frac{i2\pi k(2l+1)}{N}}$;
22. **end for**
23. **function MPP** $(\boldsymbol{m}, l, d, k, X, S, T)$
24. **if** $l = 0$
25. $z_0 = (\frac{m_{l+1}}{m_l})$; $p_0 = m_0$;
26. **else**
27. Solve the aliasing problem with $a = l + 1$ by the solution described in Sec. 2.2.
28. **end if**
29. $s_j = (\ln z_j)N/i2\pi$ for all $j \in [0, l]$;
30. **if** $(s_j \ mod \ d) = k$ for all $j \in [0, l]$
31. $S = S \cup \boldsymbol{s}$;
32. $X[s_j] = p_j$ for all $j \in [0, l]$;
33. **else**
34. $T = T \cup \boldsymbol{s}$;
35. **end if**

2.5. Computational Complexity of sFFT-DT

The outer loop of **main** runs t times. If non-zero frequency grids are distributed uniformly, given $d = O(\frac{N}{K})$ and $N^+ \in [2^1\ 2^2]$, t is set to be 4 because most frequency grids are non-aliasing for $a = 1$ or suffer aliasing for $2 \leq a \leq 4$ according to Fig. 2. The cost of outer loop is bounded by two FFTs. Since $d = O(\frac{N}{K})$ is set, the dimension of $x_{d,2l}$ and $x_{d,2l+1}$ is $O(K)$ and FFT costs $O(K \log K)$ at the first iteration. Since d is doubled iteratively, the total cost of t iterations is still bounded by $O(K \log K)$. In addition, **SubFreq** costs $O(K)$ operations due to $|S| \leq K$.

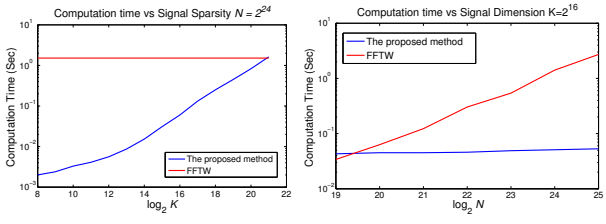
The inner loop of **main** totally runs $O(K)$ times, which is not related to the outer loop, since at most K frequency grids are necessary to be solved. The cost at each iteration is bounded by **MPP**. For $0 \leq l \leq 3$, the analytic solution, described in Sec. 2.2, costs $O(a^3)$ (without optimization). For $l > 3$, steps (i) and (iii) in Sec. 2.2 cost $O(a^3)$. Though step (ii) has no close-form solution, it can be solved in $O(da)$ because we know the roots must belong to the set, $\{e^{i2\pi(k+\frac{N}{K}l)/N} \mid l \in [0, d-1]\}$. Since a is a constant, MPP costs $O(d)$. Thus, the inner loop costs $O(dK = N)$, given $d = O(\frac{N}{K})$. In sum, the proposed algorithm, sFFT-DT, is dominated by FFT and costs $O(K \log K)$ operations.

3. EXPERIMENTAL RESULTS

Our method, sFFT-DT, was numerically verified and compared with FFTW (http://www.fftw.org/). The simulations were conducted under Visual Studio 2008 with an Intel CPU Q6600 and 2.99 GB RAM under Win 7. The signal $\boldsymbol{x}$ in time domain is produced as follows: 1) Generate a K-sparse signal $\boldsymbol{X_{ori}}$ and 2) Produce $\boldsymbol{x}$ as inverse FFT of $\boldsymbol{X_{ori}}$. The approximation error is defined as $\frac{\|\boldsymbol{X}-\boldsymbol{X_{ori}}\|_1}{\|\boldsymbol{X_{ori}}\|_1}$.

Fig. 3(a) shows the computational time versus sparsity, where $N = 2^{24}$ and $t = 4$. The initial d is set according to $N^+ = 2^2$. For $K \leq \frac{N}{2^4}$, our algorithm outperforms FFTW. The approximation error is less than 0.07%. Moreover, sFFT [1] is only faster than FFTW when $K \leq \frac{N}{2^6}$ (results of sFFT can be found in http://groups.csail.mit.edu/netmit/sFFT/results.html). Compared to sFFT, our method, sFFT-DT, is able to deal with non-sparser signals (with large K). Fair and direct comparisons with sFFT [1][2] will be conducted in the future.

Fig. 3(b) shows the computational time versus signal dimension under fixed K. d is initially set based on $N^+ = 2^2$. The computational time of our method is invariant to N.



(a) (b)

Fig. 3. (a) Computational time vs. Sparsity under $N = 2^{24}$ and $a = 4$. (b) Computational time vs. Signal dimension under $K = 2^{16}$ and $a = 4$.

4. ACKNOWLEDGMENT

This work was supported by National Science Council, Taiwan (ROC), under grant NSC 100-2628-E-001-005-MY2.

5. REFERENCES

[1] H. Hassanieh, P. Indyk, D Katabi, and Eric Price, "Nearly optimal sparse fourier transform," *STOC*, 2012.

[2] H. Hassanieh, P. Indyk, D Katabi, and Eric Price, "Simple and practical algorithm for sparse fourier transform," *SODA*, 2012.

[3] M. Frigo and S. G. Johnson, "The design and implementation of fftw3," in *Proceedings of the IEEE*, 2005, pp. 216–231.

[4] A. Gilbert, M. Muthukrishnan, and M. Straussn, "Improved time bounds for near-optimal space fourier representations," in *SPIE Conference, Wavelets*, 2005.

[5] M. A. Iwen, A. Gilbert, and M. Straussn, "Empirical evaluation of a sub-linear time sparse dft algorithm," in *Communications in Mathematical Sciences*, 2007.

[6] G. Szego, *Orthogonal Polynomials*, Amer. Math. Sot., 1975.

[7] W. H. Tsai, "Moment-preserving thresholding," *Comput. Vision, Graphics, Image Processing*, vol. 29, pp. 377–393, 1985.