EE451: SRE Report
Fourier Transform of PsuedoRandom Sequences

Badal Varshney
19D070015
Under the guidance of
Prof. Virendra R. Sule

Department of Electrical Engineering
IIT Bombay

Autumn 2022-2023

Contents

1 Abstract

Discrete Fourier Transform (DFT) finds application in various engineering domains such as communication systems, image and speech processing, biomedical signal and seismic signal processing etc. A computationally efficient algorithm for computing DFT was given by Cooley-Tukey, thereby reducing the time complexity of DFT calculation from O(N^2) to O(NlogN). In this report, we present an algorithm to compute DFT of large sequences. Here, we demonstrate that for these large sequences, the complexity of DFT computation is reduced to O(Nlog(logN)). The idea of finding the DFT of the whole sequence by first calculating the DFT of subsequences and then, combining the DFT of these sub-sequences to get the DFT of whole Sequences.

2 Introduction

Many digital signals expressed as a function of time, are further represented as their frequency components because frequency domain analysis of signals and systems is much easier than the time domain counterpart. Digital signal processing makes use of Fourier Transform tools to map a signal from the time domain to the frequency domain. Discrete Fourier Transform translates the finite length time-domain discrete sequence into its frequency domain counterpart having the same length (or greater) so that no spectral information is lost and perfect reconstruction is possible. Suppose x(n) is a sequence of finite length 'N' then its DFT is

$$y(k) \;=\; \sum_{n=0}^{N-1} x(n).e^{-i2nk\pi/N} \quad k \,\epsilon\, \{0, 1, 2, ..., N-1\} \tag{1}$$

Using this definition directly, the time complexity of calculating DFT results to O(N^2). With the introduction of a fast Fourier transform by Cooley-Tukey, Radix-2 algorithm was developed with time complexity O(NlogN). The main aim of this report is to further reduce the time complexity for large sequences.

3 Methodology

A sequence S = $(s_0, s_1, s_2, ..., s_m, ...s_{M-1})$ with terms in a finite field GF(q) with q elements is called a linear recurring sequence over GF(q) with minimal polynomial

$$X^m = a_0 + a_1X + ... + a_{m-1}X^{m-1} \quad \epsilon \;\; GF(q)[x] \tag{2}$$

If,

$$a_0s_i + a_1s_{i+1} + ... + a_ms_{i+m} = 0 \quad for \;\; any \quad i \geq 1. \tag{3}$$

Now, we make the Henkel Matrix of sequence $(s_0, s_1, s_2, ...s_{M-1})$ as defined as-

$$H(m) = \begin{pmatrix} s_0 & s_1 & \cdots & s_{m-1} \\ s_1 & s_2 & \cdots & s_m \\ \vdots & \vdots & \ddots & \vdots \\ s_{m-1} & s_{m-2} & \cdots & s_{2m-1} \end{pmatrix} \tag{4}$$

And take the coefficient of minimal as column vector 'α'-

$$\alpha = \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_{m-1} \end{pmatrix} \tag{5}$$

Now, we want the shifted sequence vector of our original sequence. For that, we defined the column vector h(m+1) as

$$h(m+1) = H(m) * \alpha \tag{6}$$

Which comes out to be like-

$$h(m+1) = \begin{pmatrix} s_m \\ s_{m+1} \\ \vdots \\ s_{2m-1} \end{pmatrix} \tag{7}$$

So that 2m-1 = M-1, and hence we get the m $=\frac{M}{2}$, which means we have to take half of the original sequence as a subsequence. But, It is true for full rank Matrix H(m). If it is not so then, reduce the row-column to make the matrix H(m) full rank matrix so that it satisfy this relation-

$$RankH(m) = RankH(m+j), \quad j = \{1, 2, ...\} \tag{8}$$

and 'm' is the degree of maximum rank minimum polynomial.
Now, Fourier Transform matrix (T) of DFT of length m is defined as

$$T = \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega & \omega^2 & \cdots & \omega^{n-1} \\ 1 & \omega^2 & \omega^3 & \cdots & \omega^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{n-1} & \omega^{2(n-1)} & \cdots & \omega^{(n-1)^2} \end{pmatrix} \tag{9}$$

where w is primitive Nth root of unity Primitive nth root of unity.

Definition of Primitive Nth root of unity-
Let n be a positive integer. A primitive n^{th} root of unity is an n^{th} root of unity that is not a k^{th} root of unity for any positive k n. That is, ζ is a primitive n^{th} root of unity if and only if

$$\zeta^n = 1, \text{ and } \zeta^k \neq 1 \text{ for any positive integer } k < n. \tag{10}$$

So, we take the primitive n^{th} root of unity as follows-

$$\zeta^m = 1 = e^{j2\pi} = cos(2\pi) + jsin(2\pi)$$

$$\implies \zeta = e^{\frac{j2\pi}{m}} = cos(\frac{2\pi}{m}) + jsin(\frac{2\pi}{m}) \tag{11}$$

Convert the binary sequence to real sequence of period n

$$c = (-1)^a, \text{where a is sequence element} \tag{12}$$

DFT of sequences(F)

$$F = T * c \tag{13}$$

This is simply the DFT formula used here. We use this to make our approach to find the DFT of subsequences to reduce complexity of DFT.
From here, we make the X(f) matrix which is defined as

$$X(f) = T * x(0) \tag{14}$$

Making of Ω Matrix-
It is defined as identity matrix with first column embedded with zero vector and last row is embedded with minimal polynomial coefficient which is from 5

$$\Omega = \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ a_0 & a_1 & a_2 & \cdots & a_{m-1} \end{pmatrix} \tag{15}$$

where a_0, a_1, a_2, ..., a_{m-1} are the coefficients of minimal polynomial that we obtained in 5
Now, We have Shifted version of subsequences that we have formulate

$$x(0) = [s_0 s_1 .... s_{m-1}]$$
$$x(1) = [s_1 s_2 .... s_m]$$
$$x(m) = [s_m s_{m+1} .... s_{2m-1}]$$

We can get these shifted version of original sequence of length m by

$$x(k+1) = \Omega * x(k), \text{where k} = \{0,1,2,....\} \tag{16}$$

And we are also come up with condition which is depend on k and x(0)

$$x(k) = \Omega^k * x(0) \tag{17}$$

From here, we can easily find the any shifted subsequences of length m by the value of k and first subsequences x(0) which is (s_0 s_1 .... s_{m-1}).
Here, from the 14, we get this relation

$$x(0) = T^{-1} * x(f) \tag{18}$$

We can also write 17 as

$$x(k) = \Omega^k * T^{-1} * x(f) \tag{19}$$

Find the DFT of subsequences
The Multification of sequences x(k) from 19 with T matix from 9

$$\text{DFT of x(k)} = T * x(k)$$
$$\implies T * x(k) = T * (\Omega^k * T^{-1} * x(f))$$
$$\implies T * x(k) = (T * \Omega^k * T^{-1}) * x(f) \tag{20}$$

4 Future Work

Find the way to merge all the DFT of subsequences to determine the DFT of large sequences to reduce the complexity of DFT and Verify the property of this DFT techniques like linearity, time-shifting, frequency-shifting, convolution etc.

5 References

References

[1] Prof. R. Fateman, "The (finite field) Fast Fourier Transform", https://people.eecs.berkeley.edu/ fateman/282/readings/fftnotes.pdf

[2] Zhi-Han Gaoa, Fang-Wei Fu, "The minimal polynomial of a sequence obtained from the componentwise linear transformation of a linear recurring sequence", Theor Comput Sci, 2010, vol 411: 3883–3893

[3] James L Massey and Shirlei Scrconek, "Linear Complexity of Periodic Sequences: A General Theory", N. Koblitz (Ed.): Advances in Cryptology - CRYPTO '96, LNCS 1109, pp. 358-371, 1996

[4] Graham H. Norton, "Minimal Polynomial Algorithms for Finite Sequences", arXiv:0911.0130v3 [cs.IT]

[5] Graham H. Norton. "The Berlekamp-Massey Algorithm via Minimal Polynomials", http://arXiv.org, 1001.1597.

[6] Howard W. Levinson and Vadim A. Markel, "Binary Discrete Fourier Transform and its Inversion", arXiv:2011.10130v2 [math.NA]

[7] ianqin Zhou, "A fast algorithm for determining the linear complexity of periodic sequences", arXiv:cs/0512040v1 [cs.CR]

[8] Blackburn,S.R., A generalization of the discrete Fourier transform: determining the minimal polynomial of a periodic sequence[J]. IEEE Trans on Information Theory, 1994, 40(5): 1702-1704.

[9] R. Jha, R. Prasad, R. Khemka and A. Mandpura, "A Faster DFT Algorithm for Specific Binary Pulse Sequences," 2020 International Conference on Communication and Signal Processing (ICCSP), 2020, pp. 1459-1461, doi: 10.1109/ICCSP48568.2020.9182054.

[10] S. V. Fedorenko, "A Method for Computation of the Discrete Fourier Transform over a Finite Field", Probl Inf Transm 42, 139–151 (2006). https://doi.org/10.1134/S0032946006020074

[11] J.W. Cooley and J. Tukey, "An Algorithm for the Machine Calculation of Complex Fourier Series" Math Comp. 19 29–301 (1965)