AN ELEMENTARY INTRODUCTION TO FAST FOURIER TRANSFORM ALGORITHMS

ANNA BOUNCHALEUN

Abstract. This paper provides a brief overview of a family of algorithms known as the fast Fourier transforms (FFT), focusing primarily on two common methods. Before considering its mathematical components, we begin with a history of how the algorithm emerged in its various forms. We then introduce the motives and central ideas leading to the theory of Fourier analysis, one such motive being the study of partial differential equations. The Fourier transforms used in this analysis seek to represent a sufficiently "nice" function as a linear combination of exponential components of different frequencies. In the discrete setting, we can efficiently compute the coefficients of the Fourier transform of a function defined on finite cyclic groups using FFT algorithms. In particular, these algorithms are especially helpful in reducing the number of steps needed for processing large amounts of data. From this, we see some applications to problems in signal processing.

Contents

1. Introduction

The famous divide-and-conquer algorithm, considered by C. Gauss in the early 1800s, decomposes a problem into smaller subproblems that are similar in some sense and can be solved recursively. This approach can be interpreted to be the foundation for the modern fast Fourier transform, but FFT methods did not become well-known or well-established until J. Cooley and J. Tukey developed their algorithm in response to a problem in digital signal processing [3]. Particularly, the task involved sampling and analyzing seismological data to test for nuclear activity. Cooley and Tukey's work was a critical breakthrough in improving the efficiency

and operation count of machine computations for data analysis. Their 1965 paper *An algorithm for the machine calculation of complex Fourier series* acknowledged the work of I. Good, whose earlier prime-factor algorithm was fairly unknown at the time. While Cooley and Tukey's algorithm addresses problems of composite size, the Good-Thomas algorithm involves sizes with coprime factors [7]. Through the computational ability of computers, FFT algorithms such as these find applications in converting signals of several forms, including visual images, sound waves, and electrical signals. As such, the usage of the fast Fourier transform cannot be overstated, and the surge in interest in FFT methods as well as its clever operational optimization are both fascinating preludes to the mathematical algorithm itself.

2. Preliminaries

2.1. **The group $\mathbb{Z}_N$.** We will consider periodic information about a discrete set of N points, which can be formalized using the notions of modular arithmetic and finite cyclic groups. The simplest example of such a group of order N will be the focus of this section.

Definition 2.1. Let m be an integer and N a positive integer. The **residue class** of m, modulo N, is the set

$$[m] \coloneqq \{m + kN : k \in \mathbb{Z}\}.$$

In other words, $[m]$ is the set of integers congruent to m modulo N.

Definition 2.2. Let $\mathbb{Z}_N$, or $\mathbb{Z}/N\mathbb{Z}$, denote the set $\{[0], [1], \ldots, [N-1]\}$ of residue classes modulo N. We define the addition and multiplication of these classes by $[m] + [n] \coloneqq [m+n]$ and $[m][n] \coloneqq [mn]$; it is possible to show that these are well-defined. We call $(\mathbb{Z}_N, +)$ and $(\mathbb{Z}_N, +, \cdot)$ the **group** and **ring of integers modulo** N.

Under addition, we note that $(\mathbb{Z}_N, +)$ is a finite abelian group, meaning that the elements of the group commute. Furthermore, the product is commutative, so the ring is also commutative.

Definition 2.3. We say $[m]$ is a **unit** in the ring $\mathbb{Z}_N$ if there exists $[n]$ in $\mathbb{Z}_N$ such that

$$[m] \cdot [n] = [1].$$

We call n the **multiplicative inverse** of m modulo N and denote it m^{-1}.

For an integer m, its residue class $[m]$ is a unit if and only if m is coprime to N. Without loss of clarity, we will not distinguish between residue classes and their integer representatives. The set of units in the ring of integers modulo N forms an abelian group under multiplication, which is denoted by $(\mathbb{Z}_N^*, \cdot)$.

Definition 2.4. Let z be a complex number. We say that z is an **N-th root of unity** if $z^N = 1$.

Letting m range from 0 to $N-1$, the N values of $e^{2\pi i m/N}$ exhaust the set of N-th roots of unity. Moreover, there is a natural isomorphism between the additive group of integers modulo N and the multiplicative group of N-th roots of unity [9]. Thus, we will adopt the notation $\mathbb{Z}_N$ throughout this paper to refer to either group.

2.2. **Fourier series.** To preface the idea of the fast Fourier transform, we begin with a brief introduction to Fourier analysis to better understand its motive, purpose, and development. In 1807, J. Fourier introduced what is now known as the Fourier series to express the solution to the heat equation on a metal plate, by representing an integrable function on the circle as an infinite sum of sines and cosines. In general, the Fourier series is frequently used in solving linear, constant-coefficient PDEs; other common examples of these include the wave equation and Laplace equation on a rectangle. Interestingly enough, Fourier's paper, which can be understood as the origin of Fourier analysis, was after Gauss's work in the divide-and-conquer approach, an earlier incarnation of the fast Fourier transform.

We now define the Fourier series and consider questions of convergence. Throughout this paper, we will define the one-dimensional torus $\mathbb{T}^1 \coloneqq \mathbb{R}/\mathbb{Z}$ and identify it with the unit circle. In addition, $\mathbb{T}^1$ is a group under addition, a fact we will use later when introducing the discrete Fourier transform. At the moment, let us consider the L^2-space of complex-valued, square-integrable functions on the unit circle with corresponding inner product

$$\langle f, g\rangle_{L^2} \coloneqq \int_{\mathbb{T}^1} f(x)\overline{g(x)}dx \tag{2.5}$$

for $f, g \in L^2(\mathbb{T}^1)$. If we define $e_n(x) \coloneqq e^{2\pi inx}$ where $x \in \mathbb{T}^1$, then $\{e_n\}_{n\in\mathbb{Z}}$ is an orthonormal family of functions.

Definition 2.6. The **n-th Fourier coefficient** of $f \in L^2(\mathbb{T}^1)$ is defined by

$$\hat{f}(n) \coloneqq \langle f, e_n\rangle_{L^2} = \int_{\mathbb{T}^1} f(x)e^{-2\pi inx}dx.$$

Definition 2.7. The **N-th partial sum** of the Fourier series of $f \in L^2(\mathbb{T}^1)$ is defined by

$$S_N f \coloneqq \sum_{|n|\leq N} \hat{f}(n)e_n.$$

Without proof, we recall the convergence of the **Fourier series**—the trigonometric series with the Fourier coefficients as "weights" and exponentials defined by e_n. Thus, consideration of the behavior of $S_N f$ as $N \to \infty$ yields the following theorem.

Theorem 2.8. *If $f \in L^2(\mathbb{T}^1)$, then*

$$\lim_{N\to\infty} \int_{\mathbb{T}^1} |f(x) - S_N f(x)|^2 dx = 0.$$

As the proof is well-written in mathematical literature and can be found in [9] and [4], we will omit it. The result of Theorem 2.8 is what is known as the **mean-square convergence** of the Fourier series. We note that the question of pointwise convergence is far more complicated and has been considered since Fourier's time. Ultimately, in 1966, L. Carleson proved that if $f \in L^2$, then the Fourier series converges almost everywhere [9]. Based on the concepts behind the proof of mean-square convergence, the following theorem also holds.

Theorem 2.9. *If $f \in L^2(\mathbb{T}^1)$, then*

$$\|f\|^2_{L^2(\mathbb{T}^1)} \coloneqq \int_{\mathbb{T}^1} |f(x)|^2 dx = \sum_{n\in\mathbb{Z}} |\hat{f}(n)|^2.$$

Proof. From Theorem 2.8 and the continuity of the norm, we have that

$$\|f\|_{L^2(\mathbb{T}^1)} = \lim_{N\to\infty} \|S_N f\|_{L^2(\mathbb{T}^1)}.$$

The orthogonality of $\{e_n\}_{n\in\mathbb{Z}}$ gives that

$$\|f\|^2_{L^2(\mathbb{T}^1)} = \lim_{N\to\infty} \|S_N f\|^2_{L^2(\mathbb{T}^1)} = \lim_{N\to\infty} \sum_{|n|\leq N} |\hat{f}(n)|^2 = \sum_{n\in\mathbb{Z}} |\hat{f}(n)|^2.$$

□

Consequently, there is an isomorphism between $L^2(\mathbb{T}^1)$ and $\ell^2(\mathbb{Z})$ where f is a square-integrable function on the circle and $\{\hat{f}(n)\}_{n\in\mathbb{Z}}$ is the square-summable sequence of corresponding Fourier coefficients [4].

2.3. **Continuous Fourier transform.** Before introducing the discrete Fourier transform, we will outline the defining properties of the (continuous) Fourier transform, as the notions of the Fourier series and transform can be translated into the discrete setting. The Fourier transform can be thought of as the analogue of the Fourier series for non-periodic functions on $\mathbb{R}$. This method implements continuous objects in place of the discrete components in the periodic case [9]. From this understanding, applications of the Fourier transform can be found in the heat equation on an infinite rod and the Laplace equation on the upper half-plane. Although the discrete Fourier transform is more similar to the Fourier series in terms of periodicity, we will only briefly show how these formulas manifest in the non-periodic case, without discussing the explicit derivations. Assuming the appropriate conditions are satisfied, we can define the Fourier transform and the subsequent inversion formula.

Definition 2.10. The **Fourier transform** of a function $f \in L^1(\mathbb{R})$ is defined by

$$\hat{f}(\xi) = \int_{\mathbb{R}} f(x) e^{-2\pi i x\xi} dx.$$

Given a smooth, compactly supported function, the following formulas hold.

Theorem 2.11. *If $f \in C_0^\infty(\mathbb{R})$, then*

$$f(x) = \int_{\mathbb{R}} \hat{f}(\xi) e^{2\pi i x\xi} d\xi$$

and

$$\|f\|^2_{L^2(\mathbb{R})} \coloneqq \int_{\mathbb{R}} |f(x)|^2 dx = \int_{\mathbb{R}} |\hat{f}(\xi)|^2 d\xi =: \|\hat{f}\|^2_{L^2(\mathbb{R})}.$$

This proof is also well-written and will be excluded; further discussion of the continuous Fourier transform, including this proof, can be found in [9] and [10].

2.4. **Discrete Fourier transform.** Following our introduction to finite cyclic groups and Fourier transforms on $\mathbb{T}^1$ and $\mathbb{R}$, we naturally consider how to define the Fourier transform on $\mathbb{Z}_N$. This setting of finite Fourier analysis will serve as the foundation for our exploration of FFT algorithms.

Now, let us define V to be the vector space of all functions $f : \mathbb{Z}_N \to \mathbb{C}$. We can endow this space with the Hermitian inner product

$$\langle f, g \rangle \coloneqq \sum_{k=0}^{N-1} f(k)\overline{g(k)}.$$

We want to obtain a discrete analogue for the function $e_n(x) = e^{2\pi inx}$ defined on the unit circle $\mathbb{T}^1$. In particular, we recall that the family of exponentials $\{e_n\}_{n\in\mathbb{Z}}$ is orthonormal with respect to the inner product from (2.5). Furthermore, these functions satisfy the property $e_n(x + y) = e_n(x)e_n(y)$, implying these are homomorphisms from $\mathbb{T}^1$ to $\mathbb{C}^* \coloneqq \mathbb{C} \setminus \{0\}$.

With this motivation and the fact that $e^{2\pi i} = 1$, let us define $\tilde{e}_n(k) \coloneqq e^{2\pi ink/N}$ where $k, n = 0, \ldots, N-1$. We note that $\tilde{e}_n$ is also a well-defined function in $\mathbb{Z}_N$ since two representatives in the same residue class differ by a multiple of N. Moreover, $\{\tilde{e}_0, \ldots, \tilde{e}_{N-1}\}$ forms an orthogonal set, and the function $\tilde{e}_n$ is a homomorphism from $\mathbb{Z}_N$ to $\mathbb{C}^*$, as desired. Now, we can build an orthonormal set $\{e_0, \ldots, e_{N-1}\}$ by defining $e_n \coloneqq \frac{1}{\sqrt{N}}\tilde{e}_n$. Due to their orthonormality, these N functions are linearly independent. Thus, we have found a basis for the N-dimensional vector space V [9].

Definition 2.12. The **n-th Fourier coefficient** of $f \in V$ is defined by

$$\hat{f}(n) \coloneqq \langle f, e_n \rangle = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} f(k)e^{-2\pi ikn/N}. \tag{2.13}$$

Using the properties of the set $\{e_0, \ldots, e_{N-1}\}$, we are able to derive the following results.

Theorem 2.14. *Let f be a function on $\mathbb{Z}_N$. Then*

$$f = \sum_{n=0}^{N-1} \hat{f}(n)e_n$$

and

$$\|f\|^2 \coloneqq \langle f, f \rangle = \sum_{n=0}^{N-1} |\hat{f}(n)|^2.$$

Proof. This follows directly from the orthonormality of the basis for V. □

These two relations are known as the **Fourier inversion** and **Parseval formula** for the discrete Fourier transform. In addition, Theorem 2.14 is also the analogue of Theorems 2.9 and 2.11 in the preceding sections. Similar to the Fourier series and Fourier transform, there are variations of these formulas depending on how the Fourier coefficient is defined. For example, by scaling $\hat{f}(n)$, we can shift the constant $N^{-1/2}$ from the DFT into the inverse DFT, which would then introduce a constant N^{-1} in the Plancherel formula. For the purposes of this paper, we will continue with the representations defined previously.

3. The Cooley-Tukey FFT algorithm

We will consider the most common form of the Cooley-Tukey FFT algorithm, which assumes that N is a power of 2. By making use of repeated computations in the Fourier coefficients of a function f on $\mathbb{Z}_N$, we can significantly reduce the

number of operations. In fact, if we count complex addition and multiplication each as one operation, calculating the coefficients of the discrete Fourier transform requires $O(n^2)$ operations. In comparison, the Cooley-Tukey algorithm gives the following result.

Theorem 3.1. *Let $N = 2^n$. Then the Fourier coefficients of a function on $\mathbb{Z}_N$ can be computed with $O(N \log_2 N)$ operations.*

Proof. Let a_n be the least amount of steps required to compute the N-point DFT. We will split our initial DFT of size N into two smaller DFTs of size N_1 and N_2 where $N = N_1 N_2$. In the case of length $N = 2^n$, a common and natural division is $N_1 = 2$ and $N_2 = N/2 = 2^{n-1}$. Let us define $\omega_N \coloneqq e^{-2\pi i/N}$. This allows us to re-express the DFT by summing over the even and odd numbered inputs, and find

$$\begin{aligned}\hat{f}(n) &= \frac{1}{\sqrt{N}}\left[\sum_{m=0}^{N/2-1} f(2m)\omega_N^{n(2m)} + \sum_{m=0}^{N/2-1} f(2m+1)\omega_N^{n(2m+1)}\right] \\ &= \frac{1}{\sqrt{N}}\left[\sum_{m=0}^{N/2-1} f(2m)\omega_{N/2}^{n\cdot m} + \omega_N^n \sum_{m=0}^{N/2-1} f(2m+1)\omega_{N/2}^{n\cdot m}\right].\end{aligned}$$

Thus, we have reduced the number of operations to compute each $\hat{f}(n)$ to only two multiplications and one addition, a total of three steps. By taking into account two $N/2$-point DFTs and 2^n computations of $\{\omega_N^0, \ldots, \omega_N^{N-1}\}$, we obtain the following recursive relation on a_n:

$$a_n \leq 2 \cdot a_{n-1} + 2^n + 3 \cdot 2^n = 2a_{n-1} + 2^{n+2}. \tag{3.2}$$

Let us compute the base case a_1. For a function f defined on $\mathbb{Z}_2$, we have two coefficients:

$$\begin{aligned}\hat{f}(0) &= \frac{1}{\sqrt{2}}[f(0) + f(1)] \\ \hat{f}(1) &= \frac{1}{\sqrt{2}}[f(0) + (-1)f(1)],\end{aligned}$$

which require a total of five operations.

Using our result from (3.2), we obtain the following inequalities:

$$\begin{aligned}a_n &\leq 2a_{n-1} + 2^{n+2} \\ 2a_{n-1} &\leq 4a_{n-2} + 2 \cdot 2^{n+1} \\ &\vdots \\ 2^{n-2}a_2 &\leq 2^{n-1}a_1 + 2^{n-2} \cdot 2^4.\end{aligned}$$

By adding these inequalities, we form a telescoping sum and find that

$$a_n \leq 5 \cdot 2^{n-1} + 2^{n+2}(n-1) \leq 4 \cdot 2^n n = O(N \log_2 N),$$

as desired. □

Critical to this proof, we note that the roots of unity repeat in these smaller DFTs, as $\omega_{N/2} = e^{-4\pi i/N} = \omega_N^2$. In the literature of fast Fourier transforms, these roots are known as twiddle factors. While FFT methods reduce the number of steps required by breaking down the DFT, the operation count of the divide-and-conquer

approach is the combined counts of the mapping and subproblems, which we want to make smaller than that of the initial problem itself. For this case of the Cooley-Tukey algorithm, complex multiplications by the twiddle factors largely contribute to operations needed for the mapping [3]. Nonetheless, by applying the process of halving the DFTs recursively, it is in fact possible to decrease the runtime to $O(N \log_2 N)$, a significant improvement in the case of large N.

Due to the construction of this specific algorithm for splitting the DFT into $N/2$ pairs of inputs sampled from the time domain, this method is known as the radix-2 decimation-in-time FFT [3]. However, the Cooley-Tukey FFT algorithm is not just limited to sequences of length $N = 2^n$, although it was originally believed to be. If we have an N-point DFT where N is a power of m, we can implement what is known as a radix-m algorithm. Expectedly, this allows us to compute the Fourier coefficients in $O(N \log_m N)$ operations, using the convenient properties of the twiddle factors [1].

In the 1980s, with the rise of interest in fast transforms, another method was developed by multiple mathematicians. The split-radix algorithm built on the structure of the Cooley-Tukey algorithm and was able to further minimize the operation count for DFTs of length $N = 2^n$ by re-expressing the N-point DFT as one $N/2$-point DFT (on the even components) and two $N/4$-point DFTs (on the odd components) [3]. A significant improvement was made in 2007, when S. Johnson and M. Frigo were able to adjust the split-radix DFT to require even fewer steps. Finally, for $N = m_1^{n_1} m_2^{n_2}$, we can implement radix-m_1 and radix-m_2 fast Fourier transforms, and this procedure is appropriately named the mixed-radix algorithm.

4. The Good-Thomas FFT algorithm

The Good-Thomas prime factor algorithm (PFA) reduces the runtime for sequences of length $N = N_1 N_2$ where N_1 and N_2 are coprime. This method involves the re-indexing of the inputs k and outputs n through orderings known as Good's mapping and the Chinese Remainder Theorem mapping, respectively [3]. We begin by establishing this important theorem in number theory, which is oftentimes implemented in calculations involving large integers.

Theorem 4.1. *Let $n_1, \ldots, n_k$ be pairwise coprime positive integers and $a_1, \ldots, a_k$ be any integers. Then the system of congruences*

$$\begin{aligned} x &\equiv a_1 \mod n_1 \\ &\vdots \\ x &\equiv a_k \mod n_k \end{aligned}$$

has a solution x uniquely determined modulo $N = n_1 n_2 \cdots n_k$.

Proof. For each $i = 1, \ldots, k$, define $m_i = N/n_i$. Now let $y_i \equiv m_i^{-1} \mod n_i$ where m_i^{-1} is the multiplicative inverse of m_i. Since $n_1, \ldots, n_k$ are pairwise coprime, each y_i must exist. If we define

$$x = \sum_{i=1}^{k} a_i m_i y_i,$$

then x is a solution to the system of congruences. For each i, reducing x modulo n_i yields $x \equiv a_i m_i y_i \mod n_i \equiv a_i \mod n_i$ since $m_i y_i \equiv 1 \mod n_i$.

To prove uniqueness modulo N, let x_1 and x_2 be solutions to our system. Thus, $x_1 \equiv x_2 \mod n_i$, implying that n_i divides $x_1 - x_2$ for each i. Now, since $\gcd(n_i, n_j) = 1$ whenever $i \neq j$, we have that $N = n_1 n_2 \cdots n_k$ also divides $x_1 - x_2$. As we have shown that $x_1 \equiv x_2 \mod N$, the solution is unique modulo N. □

Through the Chinese Remainder Theorem, we can construct an isomorphism between $\mathbb{Z}_{mn}$ and $\mathbb{Z}_m \times \mathbb{Z}_n$. To show the reverse direction, we take an element of $\mathbb{Z}_{mn}$ modulo m and modulo n to find the system of congruences. We can further apply this relation to prime powers for $\mathbb{Z}_N$ where $N = p_1^{n_1} p_2^{n_2} \cdots p_k^{n_k}$ [6].

To initialize the prime factor algorithm, we re-index the input sequence of size $N = N_1 N_2$ in the following way:

$$k = k_1 N_2 + k_2 N_1 \mod N, \quad k_1 = 0, \ldots, N_1 - 1, k_2 = 0, \ldots, N_2 - 1. \tag{4.2}$$

This correspondence is appropriately named Good's mapping, acknowledging Good's work on the PFA in his 1958 paper *The interaction algorithm and practical Fourier analysis* [2]. By using the Chinese Remainder theorem, the output sequence is also re-indexed:

$$n = n_1 N_2^{-1} N_2 + n_2 N_1^{-1} N_1 \mod N, \quad n_1 = 0, \ldots, N_1 - 1, n_2 = 0, \ldots, N_2 - 1 \tag{4.3}$$

where $N_1 N_1^{-1} \equiv 1 \mod N_2$ and $N_2 N_2^{-1} \equiv 1 \mod N_1$. Recall that N_1^{-1} and N_2^{-1} exist because N_1 and N_2 are relatively prime. Table 1 provides the results of Good's mapping and the CRT mapping on a DFT of size $N = 21$, with $N_1 = 3$ and $N_2 = 7$.

Good's mapping

		k_2						
		0	1	2	3	4	5	6
	0	0	3	6	9	12	15	18
k_1	1	7	10	13	16	19	1	4
	2	14	17	20	2	5	8	11

CRT mapping

		n_2						
		0	1	2	3	4	5	6
	0	0	15	9	3	18	12	6
n_1	1	7	1	16	10	4	19	13
	2	14	8	2	17	11	5	20

TABLE 1. Re-indexing of the DFT using prime factor mappings for $N = 21$.

Now, substituting (4.2) and (4.3) into (2.13), we find

$$\begin{aligned}
&\hat{f}(n_1N_2^{-1}N_2 + n_2N_1^{-1}N_1) \\
&\qquad = \frac{1}{\sqrt{N}} \sum_{k_1N_2+k_2N_1}^{N-1} f(k_1N_2 + k_2N_1)\omega_N^{(k_1N_2+k_2N_1)(n_1N_2^{-1}N_2+n_2N_1^{-1}N_1)} \\
&\qquad = \frac{1}{\sqrt{N}} \sum_{k_1N_2+k_2N_1}^{N-1} f(k_1N_2 + k_2N_1)\omega_N^{n_1k_1N_2+n_2k_2N_1} \\
&\qquad = \frac{1}{\sqrt{N}} \sum_{n_1=0}^{N_1-1} \left(\sum_{n_2=0}^{N_2-1} f(k_1N_2 + k_2N_1)\omega_{N_2}^{n_2k_2} \right) \omega_{N_1}^{n_1k_1}.
\end{aligned}$$

Due to the correspondence between discrete samples and periodic transforms, we have that $\{\hat{f}(n)\}$ and $\{f(k)\}$ are N-periodic sequences, so we can compute the indices modulo N [3]. Furthermore, since ω_N is an N-th root of unity, the exponentials raised to a power including the term N_1N_2 simplify to 1 [2]. By re-expressing $k_1N_2 + k_2N_1$ as the pair k_1, k_2 and $n_1N_2^{-1}N_2 + n_2N_1^{-1}N_1$ as n_1, n_2, we can write the result of the PFA algorithm more elegantly:

$$\hat{f}(n_1, n_2) = \frac{1}{\sqrt{N}} \sum_{n_1=0}^{N_1-1} \sum_{n_2=0}^{N_2-1} f(k_1, k_2)\omega_{N_1}^{n_1k_1}\omega_{N_2}^{n_2k_2}.$$

Thus, we have traded our one-dimensional DFT of length $N = N_1N_2$ for a two-dimensional DFT with lengths N_1 and N_2. This is about as efficient as the Cooley-Tukey radix-2 algorithm, reducing the operation count to $O(N \log_2 N)$ steps [7]. However, in comparison to the Cooley-Tukey algorithm, the mappings only require permutations, but the N_1-point and N_2-point DFTs admit more computations due to N_1 and N_2 being relatively prime [3]. The prime factor algorithm can also be used in tandem with the Cooley-Tukey algorithm, an example of which is described in C. Temperton's 1992 paper *A generalized prime factor FFT algorithm for any* $N = 2^p3^q5^r$.

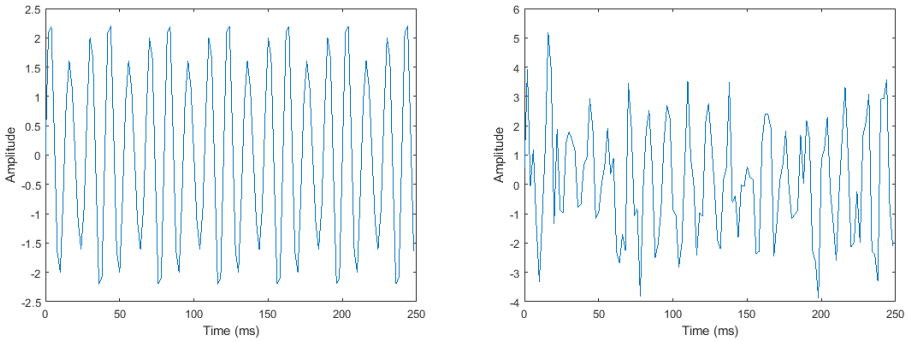
5. Visualization and further applications

In this section, we will establish some of the applications of the discrete Fourier transform and the FFT. We define a *signal* to be a function describing a physical quantity that changes in time or space. Common examples of signals include electrical voltage, sound waves, and color signals in an image. Signals can either be continuous (analog signal) or discrete in time (digital signal), and the discrete Fourier transform is a way to convert a signal sampled over a discrete set of points in the time domain to the frequency domain. The time domain indicates how the amplitude of the signal varies over time and the frequency domain shows which frequency components make up the signal. We refer to the difference between the highest and lowest frequencies of a signal as its bandwidth [5].

Given a continuous-time signal of finite bandwidth, we can recreate the original signal accurately from a discrete sequence of sampled points if the sampling rate is at least twice as fast as the highest frequency of the signal. This fundamental result in the intersection of mathematics and digital signal processing is known as the Nyquist-Shannon Sampling Theorem, and its applications in digitizing analog signals to improve transmission rates are substantial [5]. The result of this theorem

also allows for less data to be stored to preserve the most important frequency components and consequently, reduces the conversion time between analog and digital signals.

Before we begin to visualize the result of the DFT using MATLAB, let us first address how we can describe a signal. The *sampling rate* tells us how many samples are recorded from the original analog signal in each second. Furthermore, we will sample the signal at regular and equal intervals; this *sampling interval* is given by dividing one second by the sampling rate. Now, we can construct an analog signal $s(t) = 2\sin(2\pi \cdot 75t) + 0.5\sin(2\pi \cdot 100t)$, which is displayed in Figure 1 (A).



(A) (B)

FIGURE 1. Comparison of signal before (A) and after (B) corruption by random noise.

After adding random noise (following the standard normal distribution) to this signal, we can implement the FFT function to compute the coefficients of the DFT, for which the signal is sampled 500 times per second. This allows us to extract the approximate original frequency components from the noise-corrupted signal. The visualization of the signal analysis is shown in Figures 1 and 2. Considering the peaks of the amplitude spectrum, we see that the FFT function slightly underestimates the amplitude of 2 at a frequency of 75 hertz and 0.5 at 100 hertz, due to the noise applied.

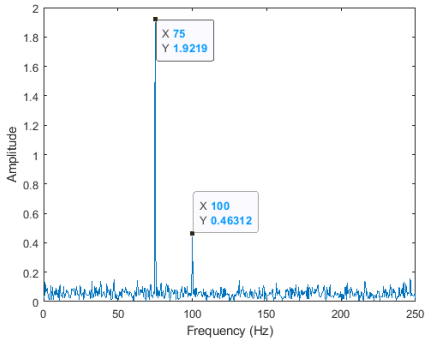


FIGURE 2. FFT-derived amplitude spectrum of noisy signal.

More information on the usage of the FFT algorithm in various programming languages can be found through FFT documentation on MathWorks and the FFTW (The Fastest Fourier Transform in the West) library for C. Interestingly, the FFTW was developed in the 1990s by Johnson and Frigo, the mathematicians mentioned previously who reduced the split-radix FFT to even fewer computations. Moreover, the FFT function in MATLAB is also influenced by the FFTW, which significantly optimizes the runtime by decomposing the transform through the prime factors and utilizing different FFT algorithm variants on these smaller transforms [8].

We will briefly leave a word on the widespread applications of frequency analysis, the discrete Fourier transform, and fast algorithms to conclude our study of the FFT. A common yet no less significant implementation of the FFT in modern technology is through image and audio recognition software, including mobile applications designed to quickly identify music, speech-to-text translators, and facial detection systems for added security to sensitive data. In these processes, the FFT is used in order to quickly analyze and compare frequencies to achieve these results. Additionally, after a signal is converted into the frequency domain, it can be manipulated more easily before applying the inverse DFT to return to the time domain. This is the case for audio filtering, which compresses data by filtering out the less important frequency components [8].

In the digital signal processing world, the ability to accurately represent and rapidly communicate data is critical. For an optimally-sized transform, the fast Fourier transform allows for a near-linear rate of conversion, implying data can be processed almost in real-time. Ultimately, in being able to identify the frequencies that compose some signal—whether it be the seismic waves of Cooley's initial problem or the signals that make up the images in this paper, we are closer to being able to amplify, filter, or modify the frequencies of our interest.

6. Acknowledgments

With great appreciation, I would like to thank my mentor, Daniel Campos, for his excellent guidance during my research study and for helping me strengthen my understanding of this subject. I would also like to thank my professor, Dr. Andrew Shulman, for his overwhelming support and encouragement throughout this project. Finally, I express my sincere gratitude to Dr. Peter May for directing this program and creating this memorable experience to grow as a student of mathematics.

7. Bibliography

References

[1] Marcel D. van de Burgwal, Pascal T. Wolkotte, and Gerard J. M. Smit. Non-Power-of-Two FFTs: Exploring the Flexibility of the Montium TP. International Journal of Reconfigurable Computing. 2009. https://www.hindawi.com/journals/ijrc/2009/678045/

[2] James W. Cooley, Peter A. W. Lewis, and Peter D. Welch. Historical Notes on the Fast Fourier Transform. IEEE Transactions on Audio and Electroacoustics, **15**, 76-79. 1967.

[3] P. Duhamel and M. Vetterli. Fast Fourier Transforms: A Tutorial Review and A State of the Art. Signal Processing, **19**, 259-299. 1990.

[4] Gerald B. Folland. Fourier Analysis and its Applications. Wadsworth. 1992.

[5] R.S. Kaler, M. Kulkarni, and Umesh Gupta. A Textbook of Digital Signal Processing. K International Publishing House. 2009.

[6] Ben Lynn. The Chinese Remainder Theorem. https://crypto.stanford.edu/pbc/notes/numbertheory/crt.html

[7] K. Marcolini. Variations and Applications of the Fast Fourier Transform Algorithms. University of Miami.
[8] Timothy Sauer. Numerical Analysis. Second Edition. Pearson Education. 2012.
[9] Elias M. Stein and Rami Shakarchi. Fourier Analysis: An Introduction. Princeton University Press. 2003.
[10] Michael E. Taylor. Fourier Analysis and the FFT. http://mtaylor.web.unc.edu/files/2018/04/fft.pdf