SJÄLVSTÄNDIGA ARBETEN I MATEMATIK

MATEMATISKA INSTITUTIONEN, STOCKHOLMS UNIVERSITET

# The Discrete Fourier Transform: Some Properties and Applications

av

**Luciano Egusquiza Castillo**

2024 - No K22

# The Discrete Fourier Transform: Some Properties and Applications

Luciano Egusquiza Castillo

Självständigt arbete i matematik 15 högskolepoäng, grundnivå

Handledare: Salvador Rodriguez Lopez

2024

# Abstract

The discrete Fourier transformation (DFT) is a map that takes a finite sequence from one domain into another. Typically, from time to frequency, generating the amplitude and phase of the constituent complex-sinusoidal components, which are its basis. The aim of this thesis is to gain an understanding of how the discrete Fourier transformation (DFT) is constructed in terms of its mathematical foundation. Some of its properties are delved into, amongst a few are linearity, one-to-oneness, periodicity, and its property of simplifying convolution operations.

Once a mathematical foundation and some properties have been explored, the fast Fourier transformation (FFT) is introduced. The FFT is an algorithm used in practice in order to reduce the number of complex multiplications needed to perform the DFT. This is done by exploiting some nice properties of the DFT matrix. Such as its symmetries, the fact that it is unitary, and that its inverse has similar elements.

Finally, applicative examples are demonstrated. A case of denoising a signal by applying the FFT is shown. Following the denoising example is an example of differentiation, where the error is compared with the Euler forward method. Lastly, an example of integration using the FFT is presented.

# Sammanfattning

Den diskreta Fouriertransformationen (DFT) är en avbildning som tar en ändlig sekvens från en domän till en annan. Typiskt sätt från tid till frekens, vilket genererar amplitud och fas av de komplexa sinusodiala komponenterna som är dess bas. Syftet med denna uppsats är att få förståelse kring hur DFT:n är uppbygd utifrån en matematisk grund. Några av dess egenskaper undersöks, varav ett urval är linjäritet, injektivitet, periodisitet samt dess egenskap att simplifiera konvolutionsoperationer.

Efter att en matematisk grund och egenskaper berörts introduceras den snabba Fouriertransformationen (FFT). FFT:n är en algoritm som används i praktiken för att förenkla beräkningen av DFT:n genom att reducera antalet komplexvärda multiplikationer. Detta uppnås genom att nyttja egenskaper hos DFT:n. Såsom dess symmetri, unitäritet samt att dess invers har snarlika element.

Avslutningsvis demonsteras några utav dess aplikationer. Ett fall av att avbrusa en signal genom tillämning av FFT demostreras. Följt av detta är ett exempel av differentiering där beräkningsfelet jämförs med Eulers framåtmetod. Slutligen tas ett exempel av integration upp.

# Acknowledgments

Firstly, I would like to thank my supervisor and mentor, Dr Salvador Rodriguez Lopez for all of his support, guidance, and teaching on my journey of writing this. I would also like to thank my dear friends: Amanuensis Leo G. Levenius, for his insightfulness and keen eye for details. Amanuensis Jack Zhan, for his amazing mathematical point of view and conceptualizing. Amanuensis Lars Lidvall, for his impeccable mathematical knowledge. Thank you, my dear friends, for your support and valid input. Last, but certainly not least, my father José Egusquiza Castillo. *Agradecekuyki tukuy imamanta papay, mana qanwanqa manan imapas kaymanchu. Munakuyki.*

# Contents

# Introduction

The *discrete Fourier transformation*, commonly denoted as DFT, is closely related to Fourier series. But instead of representing functions on an interval of real numbers, the DFT is used on functions on $\mathbb{Z}_N$. This means that the DFT takes as input elements from $\mathbb{Z}$ modulo $N$. One of many fields of applications is within signal processing, where the input is a discrete-time signal of finite length. The output of the DFT is then the amplitudes of the frequencies of the signal. In other words, when using the DFT on a signal, it can be used to gain knowledge about and alter the amplitude of different frequencies. We delve into this towards the end of the thesis. We take a linear algebraic approach to the DFT, representing our input functions as sequences, or rather vectors.

# Jean Baptise Joseph Fourier

Jean Baptiste Joseph Fourier was born at Auxerre in the province of Yonne, France on 21 March 1768. The name "Joseph" was that of his father, and the name "Fourier" is a variant of the word *fourrier* which means in a military sense a quartermaster [6, p. 1]. Fourier was a mathematician, famous for his work within mathematics and physics. His work on heat flow is arguably among the more well-known and in 1822 Fourier published *Théorie analytique de la chaleur*, in which he claimed any function of a variable could be expressed as a series of sines and cosines. This was the beginning of what today is known as the *Fourier transformation* [2, p. 47].

Figure 1: Portrait by Jules Boilly.

# Chapter 1

# Functions on $\mathbb{Z}_N$

Before we begin looking at the DFT, we need to go through some useful definitions and theorems. We begin by refreshing our memory of some linear algebra, followed by defining a set of vectors, $\{E_0, \ldots, E_{N-1}\} \subset \ell^2(\mathbb{Z}_N)$. It turns out these vectors will play a key role in the construction of the so-called *Fourier basis* [4, p. 107]. We also need to know what a *geometric sum* [13, p. 833] is, which is why it is defined in this section.

## 1.1 Some Useful Linear Algebra

**Definition 1.** We define a vector of complex numbers as $z$ where $z = (z(0), z(1), \ldots, z(N-1))$, and each entry or component of the vector is complex. That is, $z(i) \in \mathbb{C}$, for $i = 0, 1, \ldots, N-1$ and because $z$ is $N$-periodic, we have

$$z(m) = z(N+m) = z(2N+m) = \cdots = z(nN+m);$$

for all $m, n \in \mathbb{Z}$. This means $z$ regarded as a function is defined on all of $\mathbb{Z}$, not only $\mathbb{Z}_N$.

**Example 1.** For $m = 1, 2$ we have

$$\begin{aligned} z(0) &= z(N) = z(2N) = \cdots = z(nN), \\ z(1) &= z(N+1) = z(2N+1) = \cdots = z(nN+1). \end{aligned}$$

Since $z$ is defined on all of $\mathbb{Z}$, we can regard it as the function

$$\begin{aligned} z : \mathbb{Z} &\to \mathbb{C}, \\ k &\mapsto z(k). \end{aligned}$$

Alternatively, $z$ can be viewed as a sequence

$$z = (z(0), z(1), \dots, z(N-1)) \in \mathbb{C}^N. \tag{1.1}$$

The point of bringing up a few different ways of interpreting $z$ is simply because depending on how one approaches the study of Fourier analysis, different interpretations may be made. As previously mentioned, we take a linear algebraic approach to the study and will refer to $z$ as a vector. The vector space $\mathbb{C}^N$ is endowed with an inner product.

**Definition 2.** A basis for a subspace $V \in \mathbb{C}^N$ can be characterized by the following three equivalent definitions:

- It is a maximal set of linearly independent vectors in $V$; adding another vector to the set makes it no longer linearly independent.
- It is a minimal spanning set of vectors in $V$; removing a vector from the set makes it no longer span $V$.
- It is a set of linearly independent vectors that span $V$.

**Definition 3.** For any two vectors $z, w \in \mathbb{C}^N$ the inner product is defined as

$$\langle z, w \rangle = \sum_{k=0}^{N-1} z(k)\overline{w(k)}, \tag{1.2}$$

where $\overline{w(k)}$ is the complex conjugate of the $k$th component of the vector $w$.

This means $\mathbb{C}^N$ is a *complex inner product space* [5, p. 332]. We clarify this with the following definition.

**Definition 4.** Let $V$ be a vector space over $\mathbb{C}$. A *complex inner product* is a map $\langle \cdot, \cdot \rangle : V \times V \to \mathbb{C}$ with the following properties:

- *Additivity* $\langle u + v, w \rangle = \langle u, w \rangle + \langle v, w \rangle$ for all $u, v, w \in V$.

- *Scalar homogeneity* $\langle \alpha u, v\rangle = \alpha\langle u, v\rangle$ for all scalars $\alpha \in \mathbb{C}$ and all $u, v \in V$.
- *Positive definiteness* $\langle u, u\rangle \geq 0$ for all $u \in V$ and $\langle u, u\rangle = 0$ if and only if $u = \mathbf{0}$.
- *Conjugate symmetry* $\langle u, v\rangle = \overline{\langle v, u\rangle}$.

Furthermore, the norm associated to the inner product is

$$||z||_2 \coloneqq \sqrt{\langle z, z\rangle} = \left(\sum_{k=0}^{N-1} |z(k)|^2\right)^{\frac{1}{2}}. \tag{1.3}$$

The set of elements of $\mathbb{C}^N$, associated with the norm which square sum is finite can be defined as the so-called *l-2 space*, or *Lebesgue space* [3, p. 6], and is commonly denoted as $\ell^2(\mathbb{Z}_N)$. Another way of expressing $\ell^2(\mathbb{Z}_N)$ would be as the set of all $N$-sized square-summable complex-valued sequences

$$\ell^2(\mathbb{Z}_N) = \left\{ \{z(n)\}_{n=0}^{N-1} : z(n) \in \mathbb{C},\ \forall\, n\ \in\ \{0, 1, \ldots, N-1\}, \text{and} \sum_{n=0}^{N-1} |z(n)|^2 < \infty \right\}. \tag{1.4}$$

**Definition 5.** A *geometric sum* can be interpreted as a sum of numbers where each term is obtained by multiplying the previous term with some fixed constant $a \neq 1$, that is

$$S_{N-1} = \sum_{n=0}^{N-1} a^n = \frac{1 - a^N}{1 - a},$$

which converges as $N \to \infty$ if and only if $|a| < 1$.

**Definition 6.** Suppose $V$ is a complex inner product space. Let $B$ be a collection of vectors in $V$. $B$ is an *orthogonal set* if any two different elements of $B$ are orthogonal. $B$ is an *orthonormal set* if $B$ is an orthogonal set and $||v|| = 1$ for all $v \in B$. Two vectors $v, u \in B$, are orthogonal, denoted $u \perp v$, if their inner product is equal to zero. That is, if $\langle u, v\rangle = 0$.

**Definition 7.** The dimension of a vector space is equal to the number of vectors in its basis.

**Lemma 1.** *Suppose $V$ is a complex inner product space. Suppose $B$ is an orthogonal set of vectors in $V$ and $\mathbf{0} \notin B$. Then $B$ is a linearly independent set, i.e., for $B = \{b_0, b_1, \ldots, b_{N-1}\}$ and some scalars $\alpha_0, \alpha_1, \ldots, \alpha_{N-1}$, the only solution to*

$$\alpha_0 b_0 + \alpha_1 b_1 + \cdots + \alpha_{N-1} b_{N-1} = 0, \tag{1.5}$$

*is $\alpha_0 = \alpha_1 = \cdots = \alpha_{N-1} = 0$.*

*Proof.* If we take the inner product of both sides of Equation (1.5) with $b_n$, where $n$ is chosen arbitrarily and $n \in \{0, 1, 2, \ldots, N-1\}$ we will, by the properties in Definition 4, get

$$\begin{aligned} &\langle \alpha_0 b_0 + \alpha_1 b_1 + \cdots + \alpha_{N-1} b_{N-1}, b_n \rangle = \langle 0, b_n \rangle \\ \Leftrightarrow\ \ &\alpha_0 \langle b_0, b_n \rangle + \alpha_1 \langle b_1, b_n \rangle + \cdots + \alpha_{N-1} \langle b_{N-1}, b_n \rangle = 0. \end{aligned}$$

Now, by the assumption of orthogonality, note that $\langle b_k, b_n \rangle = 0$ if $k \neq n$. Thus all but one term above is zero and what remains is

$$\alpha_n \langle b_n, b_n \rangle = 0.$$

Since $b_n \neq \mathbf{0}$ and by assumption $\langle b_n, b_n \rangle \neq 0$, this implies $\alpha_n = 0$. Because $n$ was chosen arbitrarily, this can be duplicated for any $n \in \{0, 1, 2, \ldots, N-1\}$. Thus, proving $B$ is linearly independent. In particular, if $N = \dim V$, $B$ is a basis of $V$. □

**Theorem 2.** Let $V$ be a vector space over a field $\mathbb{F}$. Suppose that $V$ has a basis consisting of $N$ vectors. Then any other basis of $V$ also has $N$ vectors.

*Proof.* This can be proven by contradiction. Let $B = \{b_0, \ldots, b_{N-1}\}$ denote a basis of $V$ with $N$ vectors. Assume that there is another set of vectors, $A_0$ consisting of $K$ vectors, $K < N$ such that $A_0 = \{a_0, \ldots, a_{K-1}\}$ spans $V$. We add $b_0$ to $A_0$, denoting this set of vectors as

$$A_1' = \{\underbrace{b_0}_{\in V}, \underbrace{a_0, \ldots, a_{K-1}}_{\textit{spans } V}\}.$$

This means $b_0$ can be written as a linear combination of the rest of the vectors of $A_1'$, since they span $V$ and $B \subset V$, i.e.

$$b_0 = \alpha_0 a_0 + \alpha_1 a_1 + \cdots + \alpha_r a_r + \cdots + \alpha_{K-1} a_{K-1},$$

where at least one of the scalars, $\alpha_r \neq 0$. This in return means

$$a_r = -\frac{1}{\alpha_r}(-b_0 + \alpha_0 a_0 + \cdots + \alpha_{r-1}a_{r-1} + \alpha_{r+1}a_{r+1} + \cdots + \alpha_{K-1}a_{K-1}).$$

Hence $a_r$ can be removed from $A_1^{'}$, forming a set we call $A_1$. This process is repeated, adding $b_1$ to $A_1$ yields the set $A_2^{'}$ which will also contain some vector, $a_i$, which can be expressed by a linear combination of the rest of the vectors by the same analogy as in the case of $A_1^{'}$. Repeating this process $K$ times, where for each iterative set there is at least one element $\alpha_i \neq 0$, we end up with the set $A_K = \{b_0, \ldots, b_K\}$ which spans $V$. This, however, is impossible, since $A_K \subset B$ and $B$ is a basis of $V$, i.e. $B$ consists of $N$ linearly independent vectors. Hence, we reach a contradiction. So $K \nless N$. If $K > N$, $A_0$ is linearly dependent, since if it was linearly independent $B$ could not span it which is impossible since $B$ is a basis of $V$. Thus, if $A_0$ is a basis, $K = N$. □

**Theorem 3.** Suppose $V$ is an $N$-dimensional vector space and $v_0, \ldots, v_{N-1}$ are vectors in $V$, then these vectors are linearly independent if and only if

$$\operatorname{span}\{v_0, \ldots, v_{N-1}\} = V.$$

*Proof.* Since $V$ is $N$-dimensional, $V$ has a basis consisting of $N$ vectors, $W = \{W_0, \ldots, W_{N-1}\}$ by Definition 2. Suppose $v = \{v_0, \ldots, v_{N-1}\}$ is a set of $N$ linearly independent vectors. Let $u \in V$ be chosen arbitrarily but with the restriction $u \notin v$ and $u \neq \mathbf{0}$. Then $\{u, v_0, \ldots, v_{N-1}\}$ are linearly dependent by Theorem 2. This means $\{u, v_0, \ldots, v_{N-1}\} \in \operatorname{span} W$ and $u \in \operatorname{span}\{v_0, \ldots, v_{N-1}\}$. But since $u$ was chosen arbitrarily with the restrictions, this can be repeated for any $u \in V$, still fulfilling the restrictions. Hence $\operatorname{span} v = V$.

Now, instead assume $v$ is a set of $N$ vectors, and $\operatorname{span} v = V$. Since $V$ is $N$-dimensional, and $v$ consists of $N$ vectors, $v$ must then indeed be linearly independent. Since, if it was linearly dependent, we could by Theorem 2 find a subset of, at most $N - 1$ vectors that still spans $V$. This would, however, be impossible, since $W$ is linearly independent and consists of $N$ vectors. □

## 1.2 The Normalized Exponentials

**Definition 8.** We define the vectors $E_m \in \ell^2(\mathbb{Z}_N)$, $m = 0, 1, \dots, N-1$ by

$$E_m(n) := \frac{1}{\sqrt{N}} e^{\frac{2\pi imn}{N}}, \quad 0 \le m, n \le N-1. \tag{1.6}$$

What we mean by this is that the vector $E_m$ consists of the entries $\frac{1}{\sqrt{N}} e^{\frac{2\pi imn}{N}}$, having $n$ range from 0 to $N-1$. We further clarify with some simple examples.

**Example 2.** For $m = 0$, the vector $E_0$ is

$$E_0(n) = \left\{ \frac{1}{\sqrt{N}}, \frac{1}{\sqrt{N}}, \dots, \frac{1}{\sqrt{N}} \right\}.$$

**Example 3.** For $m = 1$, the vector $E_1$ is

$$E_1 = \left\{ \frac{1}{\sqrt{N}} e^{\frac{2\pi i}{N}}, \frac{1}{\sqrt{N}} e^{\frac{4\pi i}{N}}, \dots, \frac{1}{\sqrt{N}} e^{\frac{2\pi i(N-1)}{N}} \right\}.$$

**Lemma 4.** *The set of vectors $\{E_0, E_1, \dots, E_{N-1}\}$ is an orthonormal basis for $\ell^2(\mathbb{Z}_N)$.*

*Proof.* By Definition 6, an orthonormal set of vectors is a set of vectors in which all of the vectors are of length one and where the inner product between two distinct vectors $z, w$ is equal to 0, $z \neq w$. In other words, all vectors are unit vectors and orthogonal to one another. By Lemma 1, an orthonormal set of vectors is linearly independent, and by Theorem 3 a set of $N$ elements, or vectors which are all linearly independent and elements of some complex inner product space will span that inner product space, and be a basis for that space. This means if we can prove the inner product between any two distinct vectors $E_j, E_k$ are equal to 0, and that the norm of any vector $E_j$ is equal to one, we are done since the set consists of $N$ elements, all of which are elements of $\ell^2(\mathbb{Z}_N)$. For $0 \le j, k \le N-1$ the inner product between $E_j$ and $E_k$ as previously defined in Equation (1.2) is

$$\begin{aligned}
\langle E_j, E_k \rangle = \sum_{n=0}^{N-1} E_j(n)\overline{E_k(n)} &= \sum_{n=0}^{N-1} \frac{1}{\sqrt{N}} e^{\frac{2\pi ijn}{N}} \cdot \overline{\frac{1}{\sqrt{N}} e^{\frac{2\pi ikn}{N}}} \\
&= \frac{1}{N} \sum_{n=0}^{N-1} e^{\frac{2\pi ijn}{N}} \cdot e^{-\frac{2\pi ikn}{N}} = \frac{1}{N} \sum_{n=0}^{N-1} e^{\frac{2\pi in(j-k)}{N}}.
\end{aligned}$$

We divide this into two cases, either $j = k$ or $j \neq k$. The first case is somewhat trivial, since if $j = k$ we simply end up with

$$\frac{1}{N}\sum_{n=0}^{N-1} e^0 = \frac{1}{N}\cdot N = 1.$$

Hence, according to Equation (1.3) all vectors $E_j$, $0 \leq j \leq N-1$, are unit vectors since taking the square root of 1 is just 1.

In the case where $j \neq k$ we use Definition 5. Continuing with our proof, note that since $j \neq k$ we have $e^{\frac{2\pi in(j-k)}{N}} \neq 1$. Also note that, for the same reason $-N < j-k < N$, and lastly that $e^{\frac{2\pi in(j-k)}{N}}$ can be expressed as $\left(e^{\frac{2\pi i(j-k)}{N}}\right)^n$. Using the definition of a geometric sum, we get

$$\frac{1}{N}\sum_{n=0}^{N-1}\left(e^{\frac{2\pi i(j-k)}{N}}\right)^n = \frac{1-\left(e^{\frac{2\pi i(j-k)}{N}}\right)^N}{1-e^{\frac{2\pi i(j-k)}{N}}} = \frac{1-e^{2\pi i(j-k)}}{1-e^{\frac{2\pi i(j-k)}{N}}}.$$

Since $j$ and $k$ are integers, $j-k$ will also be an integer, this together with Euler's identity [12, p. 28] gives us $e^{2\pi i(j-k)} = 1$, which means for $j \neq k$

$$\langle E_j, E_k\rangle = \frac{1-e^{2\pi i(j-k)}}{1-e^{\frac{2\pi i(j-k)}{N}}} = \frac{1-1}{1-e^{\frac{2\pi i(j-k)}{N}}} = 0. \tag{1.7}$$

Thus, the set $\{E_0, E_1, \ldots, E_{N-1}\}$ is an orthonormal basis for $\ell^2(\mathbb{Z}_N)$. $\square$

The vectors forming this basis are called the *normalized complex exponentials*, or the *pure tones* of order $N$ [11, p. 51]. A pure tone is a sinusoidal wave at a single specific frequency, intuitively what this means is that the set consists of $N$ sinusoidal waves, where, for each value of $m$ a certain specific frequency is expressed. We will momentarily see that this basis bears close resemblance to the so-called Fourier basis $F$ which we will be looking at a little bit later.

Within most applicative uses of the DFT, the main focus is to change coordinates from the standard basis to the Fourier basis, in order to, with greater simplicity and efficiency perform some sort of operation on our given vector. Depending on the field of application, this vector can be anything from functions to signals. In the case of signals, the input domain is in time, and when we perform the DFT we map to the frequency domain. Once a signal is represented in the Fourier basis, that is, once we are representing our signal in the frequency domain, we want to perform some operations on the signal before changing coordinates back to the standard basis. This is the key essence of the DFT and has many different applications. One of them is noise filtering. Once we have frequencies expressed in the Fourier basis, we can remove certain frequencies with low overall magnitude in the signal-input, or frequencies that are very high and worsen the signal quality. Often this can be done at the cost of minor if not unnoticeable changes in the actual signal. But by removing unnecessary frequencies the signal input can be decompressed and will take up less memory while stored.

## 1.3 Defining the DFT

From [2, p. 105], since the complex exponentials are a basis for $\ell^2(\mathbb{Z}_N)$, for any $z, w \in \ell^2(\mathbb{Z}_N)$,

$$z = \sum_{m=0}^{N-1} \langle z, E_m \rangle E_m, \tag{1.8}$$

$$\langle z, w \rangle = \sum_{m=0}^{N-1} \langle z, E_m \rangle \overline{\langle w, E_m \rangle}, \tag{1.9}$$

$$||z||^2 = \sum_{m=0}^{N-1} |\langle z, E_m \rangle|^2. \tag{1.10}$$

Recall, by Definition 3 of inner product between two complex-valued vectors

$$\langle z, E_m \rangle = \sum_{n=0}^{N-1} z(n)\overline{E_m(n)} = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} z(n) e^{-\frac{2\pi i m n}{N}}. \tag{1.11}$$

**Definition 9.** Suppose $z \in \ell^2(\mathbb{Z}_N)$, for $0 \leq m, n \leq N-1$ we define

$$\widehat{z}(m) := \sqrt{N} \langle z, E_m \rangle = \sum_{n=0}^{N-1} z(n) e^{-\frac{2\pi i m n}{N}}. \tag{1.12}$$

We call the map that takes $z$ to the vector $\widehat{z} = (\widehat{z}(0), \widehat{z}(1), \ldots, \widehat{z}(N-1))$ the *discrete Fourier transformation* [4, p. 105], or DFT. That is, for two vectors $z, \widehat{z}$, the DFT is the map

$$\begin{aligned} \widehat{\ } : \ell^2(\mathbb{Z}_N) &\to \ell^2(\mathbb{Z}_N), \\ z &\mapsto \widehat{z}. \end{aligned}$$

In fact, the DFT is a linear transformation. Recall a linear transformation is a map which preserves vector addition and scalar multiplication. This will be useful later on when we define the DFT as a matrix.

**Definition 10.** A *linear transformation* [5, p. 65], $T$ is a map between two vector spaces that preserves additivity and scalar multiplication, for two vectors $u, v$, and some scalar $\alpha$, that is:

$$\begin{aligned} T(u+v) &= T(u) + T(v), \\ T(\alpha v) &= \alpha T(v). \end{aligned}$$

**Theorem 5.** The DFT is a linear transformation.

*Proof.* For $z, w \in \ell^2(\mathbb{Z}_N)$, and some scalar $\alpha \in \mathbb{C}$,

$$\begin{aligned} (\widehat{z+w})(m) &= \sum_{n=0}^{N-1} (z(n) + w(n)) e^{-\frac{2\pi i m n}{N}} \\ &= \sum_{n=0}^{N-1} z(n) e^{-\frac{2\pi i m n}{N}} + \sum_{n=0}^{N-1} w(n) e^{-\frac{2\pi i m n}{N}} = \widehat{z}(m) + \widehat{w}(m), \\ \widehat{\alpha z}(m) &= \alpha \sum_{n=0}^{N-1} z(n) e^{-\frac{2\pi i m n}{N}} = \alpha \widehat{z}(m). \end{aligned}$$

Thus, the DFT is a linear transformation. □

There is also a formula for inverting the vector $\widehat{z}$ back to $z$, this formula is, perhaps not too surprisingly called *the Fourier inversion formula* [4, p. 109].

**Theorem 6.** Letting $z \in \ell^2(\mathbb{Z}_N)$ the Fourier inversion formula is

$$z(n) = \frac{1}{N} \sum_{m=0}^{N-1} \widehat{z}(m) e^{\frac{2\pi i m n}{N}}, \ 0 \leq n \leq N-1. \tag{1.13}$$

*Proof.* Using Equation (1.8),

$$z(n) = \sum_{m=0}^{N-1} \langle z, E_m \rangle E_m(n) = \sum_{m=0}^{N-1} \frac{1}{\sqrt{N}} \widehat{z}(m) \frac{1}{\sqrt{N}} e^{\frac{2\pi i m n}{N}} = \frac{1}{N} \sum_{m=0}^{N-1} \widehat{z}(m) e^{\frac{2\pi i m n}{N}}.$$

□

Using the Fourier inversion formula, we can define the so-called *Fourier basis.*

**Definition 11.** We define the vectors $F_m \in \ell^2(\mathbb{Z}_N)$ by

$$F_m(n) := \frac{1}{N} e^{\frac{2\pi i m n}{N}}, \quad 0 \leq m, n \leq N-1.$$

Much like what we did in Definition 8, we can define the set of vectors $F = \{F_0, F_1, \ldots, F_{N-1}\}$ by fixing a value of $m$ for each vector and letting $n$ assume each integer value from 0 to $N-1$, for each vector. We may also, similarly to when we looked at the normalized exponential basis have a look at an example.

**Example 4.** Let $m = 1$ and $0 \leq n \leq N-1$, we then get

$$F_1 = \left\{ \frac{1}{N} e^{\frac{2\pi i n}{N}} \right\}_{n=0}^{N-1} = \left\{ \frac{1}{N}, \frac{1}{N} e^{\frac{2\pi i}{N}}, \frac{1}{N} e^{\frac{4\pi i}{N}}, \ldots, \frac{1}{N} e^{\frac{2\pi i (N-1)}{N}} \right\}.$$

The set of vectors $\{F_0, F_1, \ldots, F_{N-1}\}$ is called the Fourier basis for $\ell^2(\mathbb{Z}_N)$. Note that this is simply the normalised complex exponentials basis multiplied with $\frac{1}{\sqrt{N}}$, $F_m = \frac{1}{\sqrt{N}} E_m$. Using the Fourier basis, we can express our vector $z$ as

$$z = \sum_{m=0}^{N-1} \widehat{z}(m) F_m. \tag{1.14}$$

When we express $z$ in terms of the Fourier basis the coefficients used are $\widehat{z}(m)$. A common way of denoting this is

$$\widehat{z} = [z]_F, \tag{1.15}$$

which can be interpreted as *expressing the vector $z$ in the Fourier basis by using the components of the $\widehat{z}$-vector* [4, p. 36].

## 1.4 The Inverse DFT

**Definition 12.** Given a vector $z \in \ell^2(\mathbb{Z}_N)$, we define

$$\check{z}(n) := \frac{1}{N}\sum_{m=0}^{N-1} z(m)e^{\frac{2\pi i mn}{N}}, \quad 0 \leq n \leq N-1. \tag{1.16}$$

And much like when we defined the DFT, we define the vector $\check{z} = (\check{z}(0), \check{z}(1), \ldots, \check{z}(N-1))$. Note that $\check{}$ is inverting the $\hat{}$ mapping, and it is this map that is commonly known as the inverse DFT, or simply IDFT. That is, given two vectors $\hat{z}, z \in \ell^2(\mathbb{Z}_N)$

$$\begin{aligned} \check{} : \ell^2(\mathbb{Z}_N) &\to \ell^2(\mathbb{Z}_N), \\ \hat{z} &\mapsto z. \end{aligned}$$

## 1.5 The DFT as a Matrix

Depending on what literature one chooses to study, the notation used for the discrete Fourier matrix varies. We will be using the notation $W_N$ for the Fourier matrix which, to be clear is the change-of-base matrix *from* the standard basis *to* the Fourier basis. Before we get to the matrix, we do some quick simplifications to our notations, we define

$$\omega_N := e^{-\frac{2\pi i}{N}},$$

such that, for $m, n \in \{0, 1, \ldots, N-1\}$

$$\begin{aligned} e^{\frac{2\pi i mn}{N}} &= \omega_N^{-mn}, \\ e^{-\frac{2\pi i mn}{N}} &= \omega_N^{mn}. \end{aligned}$$

Using this notation, we can express the $\hat{z}$-components as

$$\hat{z}(m) = \sum_{n=0}^{N-1} z(n)\omega_N^{mn}. \tag{1.17}$$

Since we are working with indexation ranging from 0, rather than 1 up till now, we continue this habit for our matrices. So columns and rows are indexed from 0 to $N-1$.

**Definition 13.** The DFT matrix, here denoted as $W_N$ is defined as $[\omega_{mn}]_{0\le m,n\le N-1}$ writing this out we get

$$W_N := \frac{1}{\sqrt{N}}\begin{bmatrix} \omega_N^{00} & \omega_N^{01} & \omega_N^{02} & \cdots & \omega_N^{0(N-1)} \\ \omega_N^{10} & \omega_N^{11} & \omega_N^{12} & \cdots & \omega_N^{1(N-1)} \\ \omega_N^{20} & \omega_N^{21} & \omega_N^{22} & \cdots & \omega_N^{2(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \omega_N^{(N-1)0} & \omega_N^{(N-1)1} & \omega_N^{(N-1)2} & \cdots & \omega_N^{(N-1)^2} \end{bmatrix}$$
$$= \frac{1}{\sqrt{N}}\begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega_N & \omega_N^2 & \cdots & \omega_N^{N-1} \\ 1 & \omega_N^2 & \omega_N^4 & \cdots & \omega_N^{2(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega_N^{N-1} & \omega_N^{2(N-1)} & \cdots & \omega_N^{(N-1)^2} \end{bmatrix}.$$

Note that using this matrix, we obtain $\widehat{z}$ by

$$\widehat{z} = W_N z. \tag{1.18}$$

**Definition 14.** A Matrix $A$ is said to be *invertible* [5, p. 100] if there exists a matrix $B$ such that

$$AB = I, \tag{1.19}$$

where $I$ is the identity matrix.

A very useful property that the DFT matrix possesses is that it is *unitary.* We can use this to show it is invertible is the definition on a *unitary matrix.*

**Definition 15.** We say a matrix $U$ is *unitary* [8, p. 84] if

$$U^*U = UU^* = I,$$

where $U^*$ is the so called *conjugate transpose*, or *Hermitian transpose* [8, p. 6]. The conjugate transpose is obtained, perhaps not too surprisingly by transposing $U$ and taking the complex conjugates of each of its entries. This means, for each entry $U_{ij}$, we get

$$(U^*)_{ij} = \overline{U}_{ji}.$$

**Theorem 7.** The DFT matrix is unitary.

*Proof.* We prove this by simply performing the matrix multiplications $WW^*$, as well as $W^*W$. If both of these turn out to be equal to the identity matrix $I$, we are done.

We let $W_j$ denote the vector consisting of the entries on row $j$ of the matrix $W$, and $W_j(n)$ denote the $n$th entry of this vector. Similarly, we let $W_k$ denote the vector consisting of elements on column $k$ of matrix $W^*$ and $W_k(n)$ denote its $n$th entry. Using this notation, for $0 \leq j, k \leq N-1$ the entries of $WW^*$ are

$$(WW^*)_{jk} = \sum_{n=0}^{N-1} W_j(n)\overline{W_k(n)} = \sum_{n=0}^{N-1} e^{-\frac{2\pi ijn}{N}} \cdot e^{\frac{2\pi ikn}{N}} = \sum_{n=0}^{N-1} e^{\frac{2\pi i(k-j)n}{N}}.$$

We end up with the two cases $j = k$ and $j \neq k$. The first one means that all of the $N$ terms in the sums are just 1, meaning when added up we get $N$, multiplied with the constant $(\frac{1}{\sqrt{N}})^2$ we end up with all 1:s. Of course, the case where $j = k$ only occurs on the diagonal, so all the diagonal entries are 1. In the other case, where $j \neq k$

$$(WW^*)_{jk} = \sum_{n=0}^{N-1} e^{\frac{2\pi i(k-j)n}{N}} = \sum_{n=0}^{N-1} \left(e^{\frac{2\pi i(k-j)}{N}}\right)^n.$$

We remind ourselves of Definition 5 of a geometric sum, and the proof of Lemma 4. Using this we get

$$(WW^*)_{jk} = \frac{1 - \left(e^{\frac{2\pi i(j-k)}{N}}\right)^N}{1 - e^{\frac{2\pi i(j-k)}{N}}} = \frac{1 - e^{2\pi i(j-k)}}{1 - e^{\frac{2\pi i(j-k)}{N}}} = \frac{1-1}{1 - e^{\frac{2\pi i(j-k)}{N}}} = 0. \quad (1.20)$$

The proof for $W^*W$ is analogous, we would end up with the same situation, due to the symmetry of the matrix we would just have the conjugate as the first terms, but the result would be the same. This completes the proof, the DFT matrix is unitary. □

## 1.6 The Inverse DFT Matrix

**Proposition 8.** *A unitary matrix is invertible.*

*Proof.* Since a matrix $U$ is unitary if $UU^* = U^*U = I$, the inverse of $U$ has to exist per definition and is in fact $U^*$. □

**Proposition 9.** *The DFT matrix is invertible and its inverse,* $W_N^{-1}$ *is* $\overline{W}_N$.

*Proof.* If we transpose the matrix $W_N$, then in place of entry $\omega_N^{mn}$, the element $\omega_N^{nm}$ will take place. But recall we defined $\omega_N^{mn}$ as $e^{-\frac{2\pi imn}{N}}$, thus $\omega_N^{nm} = e^{-\frac{2\pi imn}{N}}$, so $\omega_N^{mn} = \omega_N^{nm}$. This means the transpose of $W_N$ is equal to $W_N$ itself, $W_N^T = W_N$. A matrix which is equal to its transpose is called a *symmetric* matrix [8, p. 7]. Thus, when computing the hermitian transpose, the conjugation of the matrix is the only operation which changes the matrix which means $W_N^* = \overline{W}_N$, and therefore

$$W_N^{-1} = \overline{W}_N. \tag{1.21}$$

□

This is a proof that is of great importance to us. Remember, we started in Lemma 4 by showing that the complex exponentials were a basis for $\ell^2(\mathbb{Z}_N)$. We then saw that the Fourier basis was strikingly similar to this basis. In fact, it is the complex exponential basis multiplied by the scalar $\frac{1}{\sqrt{N}}$. Since the complex exponentials were an orthonormal basis, the Fourier basis is orthogonal. We then proceeded to express the DFT in matrix form, and now we have shown that the DFT matrix is unitary and that it is invertible. This also tells us that the DFT is one-to-one or injective.

## 1.7 Properties of the DFT

**Definition 16.** We define $R_k z$ as the translation of $z$ by $k$ as

$$(R_k z)(n) = z(n-k),$$

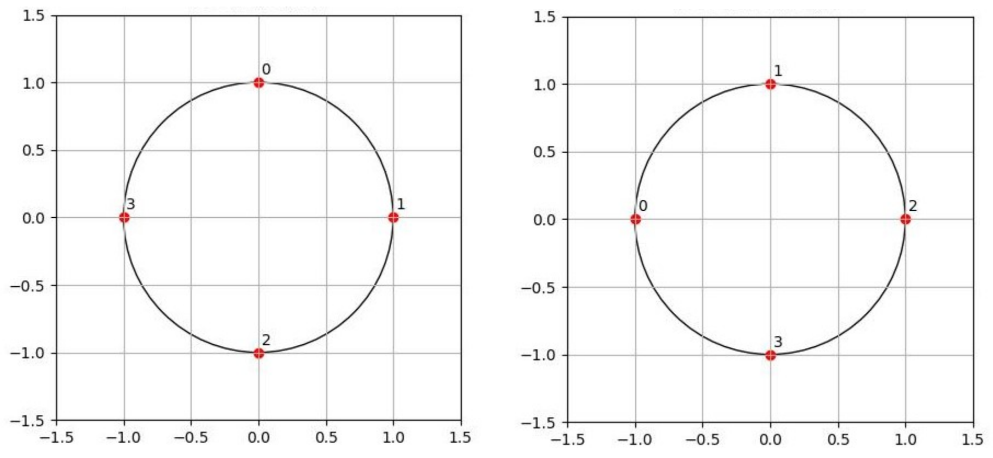
where $z \in \ell^2(\mathbb{Z}_N)$ and $n, k \in \mathbb{Z}$.

The operation known as *translation*, or sometimes *rotation* or *circular translation* [4, p. 132] could be described as a shift of each of the components $z(n)$ of $z$ by $k$ steps to the right. The best way of explaining this is by an illustrative example.

Let $z = (0, 1, 2, 3)$, then $N = 4$. We let $k = 3$, performing translation on $z$ gives us

$$\begin{aligned}(R_3z)(0) &= z(0-3) = z(-3) = z(1) = 1,\\ (R_3z)(1) &= z(1-3) = z(-2) = z(2) = 2,\\ (R_3z)(2) &= z(2-3) = z(-1) = z(3) = 3,\\ (R_3z)(3) &= z(3-3) = z(0) = 0.\end{aligned}$$

We end up with the shifted vector $R_3(z) = (1, 2, 3, 0)$, where each of the components has been shifted 3 steps to the right. This operation is illustrated in the two graphs above. It is not too hard to see where the names rotation or circular transformation come from. As depicted in Figure 2, when presented in a circular fashion, the translation rotates the elements 3 steps to the right.



(a) Pre-translation. (b) Post-translation.

Figure 2: Demonstration of translation, with $k = 3$.

The DFT's component magnitude, $|\widehat{z}(m)|$, is unchanged by translation. However, the phase might change.

**Lemma 10.** *Suppose $z \in \ell^2(\mathbb{Z}_N)$ and $k \in \mathbb{Z}$. Then for any $m \in \mathbb{Z}$,*

$$\widehat{(R_kz)}(m) = e^{-\frac{2\pi imk}{N}}\widehat{z}(m).$$

What Lemma 10 states is essentially what we claimed in words just above. That taking the DFT of a translation of $z$ by $k$ units will at most affect $z$ by an angular (phase) factor of $e^{-\frac{2\pi imk}{N}}$, sometimes referred to as *modulation*.

*Proof.* Recalling Definition 9 of the DFT, and Definition 16 of translation,

$$\widehat{(R_k z)}(m) = \sum_{n=0}^{N-1} (R_k z)(n) e^{\frac{-2\pi i m n}{N}} = \sum_{n=0}^{N-1} z(n-k) e^{\frac{-2\pi i m n}{N}}.$$

Since $k$ is fixed, it will of course have a constant value when we sum over $n$. We can change variables from $n$ to $p$, where we let $p = n - k$, note that when defining $p$ in such a way $n = p + k$.

Furthermore, if $n = 0$ then $p = -k$ and if $n = N - 1$ then $p = N - k - 1$, hence if we take the sum over $p$ instead of $n$ we get

$$\sum_{p=-k}^{N-k-1} z(p) e^{\frac{-2\pi i m (p+k)}{N}} = \sum_{p=-k}^{N-k-1} z(p) e^{\frac{-2\pi i m p}{N}} \cdot e^{\frac{-2\pi i m k}{N}},$$

but again, since $k$ is fixed we can move around the term $e^{\frac{-2\pi i m k}{N}}$ and end up with

$$e^{\frac{-2\pi i m k}{N}} \sum_{p=-k}^{N-k-1} z(p) e^{\frac{-2\pi i m p}{N}}.$$

This looks very similar to what we claimed would be the outcome of taking the DFT of the translation, what remains to show is that this sum equals $\widehat{z}(m)$. That is, what remains to show is that

$$\sum_{p=-k}^{N-k-1} z(p) e^{\frac{-2\pi i m p}{N}} = \sum_{n=0}^{N-1} z(n) e^{\frac{-2\pi i m n}{N}}. \tag{1.22}$$

Observe that in Equation (1.22) both $z(p)$ and $e^{\frac{2\pi i m p}{N}}$ are periodic functions of $p$ with periodicity $N$. We can split up Equation (1.22) into two cases. Either $k = 0$, or $1 \leq k \leq N - 1$. In the first case, we do not really have anything to show since we end up with exactly $\widehat{z}(m)$, this perhaps is not too surprising since if $k = 0$ there is no translation being performed on $z$ to begin with. The relevant case to look at and prove is when $1 \leq k \leq N - 1$.

$$\sum_{p=-k}^{N-k-1} z(p) e^{\frac{-2\pi i m p}{N}} = \sum_{p=-k}^{-1} z(p+N) e^{\frac{-2\pi i m (p+N)}{N}} + \sum_{p=0}^{N-k-1} z(p) e^{\frac{-2\pi i m p}{N}}.$$

We let $n = p + N$ in the first sum and in the second sum we simply let $n = p$, giving us

$$\sum_{n=N-k}^{N-1} z(n)e^{\frac{-2\pi imn}{N}} + \sum_{n=0}^{N-k-1} z(n)e^{\frac{-2\pi imn}{N}} = \sum_{n=0}^{N-1} z(n)e^{\frac{-2\pi imn}{N}}.$$

We now need to show this equality will hold for any arbitrary $k$. Therefore, we choose $k \in \mathbb{Z}$ arbitrarily and use some clever tricks. We note that there will always be some integer $r$ such that $(k + rN) \in \{0, 1, \dots, N-1\}$, this is not too hard to see. If $k \in \{0, 1, \dots, N-1\}$, we just let $r = 0$, if $k \in \alpha \cdot \{0, 1, \dots, N-1\}$, where $\alpha \in \mathbb{Z}$ and $N \leq |\alpha|$, we can "adjust" for this by letting $r$ be the integer-factor "correcting" $k$, placing $(k + rN)$ into some value in $\{0, 1, \dots, N-1\}$. For simplicity, we let $k' = (k + rN)$. We now do the same for $p$, we let $p' = (p - rN)$ Finally, we change from the sum over the variable $p$ to variable $p'$. Since $p = n - k$, and $p' = p + rN$, our new limits of summation thus become $-k - rN$ to $N - k - rN - 1$. But since $k' = k - rN$, these limits of summation can be expressed as $-k'$ to $N - k' - 1$. Thus we get

$$\sum_{p=-k}^{N-k-1} z(p)e^{\frac{-2\pi imp}{N}} = \sum_{p'=-k'}^{N-k'-1} z(p')e^{\frac{-2\pi imp'}{N}},$$

since, once again both $z$ and the exponential have periodicity $N$ and $k' \in \{0, 1, \dots, N-1\}$ we sum over the entirety of $\{0, 1, \dots, N-1\}$, and this adds up to $\widehat{z}(m)$. $\square$

This is an important property, so it will not hurt to clarify exactly why this means the magnitude remains unchanged. Recall from properties of Euler's identity, $\left|e^{i\theta}\right| = 1$, for any $\theta \in \mathbb{R}$. Also from basic analysis, we know for any two numbers $a, b \in \mathbb{C}$, $|a \cdot b| = |a| \cdot |b|$. This means, looking at the magnitude, we get

$$|\widehat{(R_k z)}(m)| = \left|e^{\frac{-2\pi imk}{N}} \widehat{z}(m)\right| = \left|e^{\frac{-2\pi imk}{N}}\right| \cdot |\widehat{z}(m)| = |\widehat{z}(m)|.$$

## 1.8 Translation Invariance

Recall in Definition 10, we said a linear transformation is a function from one vector space to another, preserving addition and scalar multiplication.

In our case, we are looking at functions from $\ell^2(\mathbb{Z}_N)$ to $\ell^2(\mathbb{Z}_N)$. We will momentarily see that any function which is both a linear transformationation and translation invariant has a very interesting synergy with the Fourier basis. Arguably one of the most fascinating properties of the DFT is that its basis, namely the Fourier basis $F$, is that *any* translation-invariant linear transformation from $\ell^2(\mathbb{Z}_N)$ to $\ell^2(\mathbb{Z}_N)$ is diagonalized by $F$. This of course needs to be proven. We start by reminding ourselves of an important property of linear transformations and vector spaces as well as some definitions from linear algebra.

**Definition 17.** Given a linear transformationation $T : V \to V$, a non-zero vector $F$ and a scalar $a \in \mathbb{C}$ if

$$T(F) = aF, \tag{1.23}$$

we say $F$ is an *eigenvector* of $T$ and $a$ is an *eigenvalue* [5, p. 246] of $T$.

**Definition 18.** Let V be a finite-dimensional vector space and $T : V \to V$ a linear transformation. If $V$ has a basis consisting of eigenvectors of $T$, then $T$ is said to be *diagonalizable* [4, p. 109].

In our case, the basis in question is of course the Fourier basis, $F$. This means each element of $F$ will be an eigenvector of any translation-invariant linear transformationation $T$.

**Theorem 11.** Let $T : \ell^2(\mathbb{Z}_N) \to \ell^2(\mathbb{Z}_N)$ be a translation-invariant linear transformation. Then each element of the Fourier basis $F$ is an eigenvector of $T$, and $T$ is in particular diagonalizable.

*Proof.* We start by fixing a value for $m$, where $m \in \{0, 1, \dots, N-1\}$. Recall that $F_m$ is the $m$th vector of the basis $F$, and that we denote the $n$th entry of the $m$th vector as $F_m(n)$. Since $T$ is a linear transformationation, there exist scalars $\{a_0, a_1, \dots, a_{N-1}\} \in \mathbb{C}$ such that

$$T(F_m)(n) = \sum_{k=0}^{N-1} a_k F_k(n) = \frac{1}{N} \sum_{k=0}^{N-1} a_k e^{\frac{2\pi i k n}{N}}, \tag{1.24}$$

for all $n \in \mathbb{Z}$. In the last step in the Equation (1.24), we have simply used the definition of the Fourier basis from Definition 11. Note that this is nothing other than the change of basis formula, where $F_m(n)$ represents an entry of the current basis, and $T(F_m)(n)$ represents our entries expressed in the new basis. Also note that if we perform translation on $F_m(n)$ by shifting one unit, that is, by letting $k = 1$ we get

$$(R_1F_m)(n) = F_m(n-1) = a_k e^{\frac{2\pi im(n-1)}{N}} = \frac{1}{N}e^{\frac{2\pi imn}{N}} \cdot e^{-\frac{2\pi im}{N}} = e^{-\frac{2\pi im}{N}} F_m(n).$$

Since $m$ is fixed and $F_m(n)$ only depends on $n$, we can treat $e^{-\frac{2\pi im}{N}}$ as a constant. This in combination with the property of scalar multiplication of a linear transformationation, applying the linear transformationation $T$ to our translation gives us

$$T(R_1F_m)(n) = e^{-\frac{2\pi im}{N}} T(F_m)(n) = e^{-\frac{2\pi im}{N}} \sum_{k=0}^{N-1} a_k F_k(n) = \sum_{k=0}^{N-1} a_k e^{-\frac{2\pi im}{N}} F_k(n).$$

This corresponds to the middle part of Equation (1.24) but where our scalar is $a_k e^{-\frac{2\pi im}{N}}$. However, looking at the right part of Equation (1.24) we also see that

$$\begin{aligned}(R_1T(F_m))(n) = (T(F_m))(n-1) &= \frac{1}{N}\sum_{k=0}^{N-1} a_k e^{\frac{2\pi ik(n-1)}{N}} \\ &= \sum_{k=0}^{N-1} a_k \frac{1}{N} e^{-\frac{2\pi ik}{N}} \cdot e^{\frac{2\pi in}{N}} = \sum_{k=0}^{N-1} a_k \frac{1}{N} e^{-\frac{2\pi ik}{N}} \cdot F_m(n).\end{aligned}$$

But because of the assumption that $T$ is translation invariant, we must have that $T(R_1F_M(n)) = R_1(T(F_m))(n)$ for all $n$. If we look at our two results above, both implied by Equation (1.24) we see that for every fixed value for $k \in \{0, 1, \ldots, N-1\}$ we get

$$a_k e^{-\frac{-2\pi m}{N}} = a_k e^{-\frac{-2\pi ik}{N}}. \tag{1.25}$$

If $k \neq m$ we will have $e^{-\frac{-2\pi m}{N}} \neq e^{-\frac{-2\pi ik}{N}}$, but for Equation (1.25) to hold, this must imply that $a_k = 0$. This means, for $k \neq m$ we have proven that $a_k = 0$. Thus, returning to Equation (1.25), all terms are equal to 0 except for when $k = m$, in which case

$$T(F_m)(n) = a_m F_m(n),$$

which exactly corresponds to what we previously defined would mean $a_m$ are eigenvalues of $T$, and $F_m$ are eigenvectors of $T$. Since we chose $m$ arbitrarily, this can be repeated for any $m$ defined as above. Hence, every vector of the Fourier basis $F$, $F_m$ is an eigenvector of $T$ with eigenvalues $a_m$. Furthermore, since all terms but where $k = m$ are equal to 0, $T$ is diagonalizable. □

Before looking at other interesting properties of the DFT, we go through some useful definitions. First off, we define what it means for a matrix $A$ to have periodicity with period $N$.

**Definition 19.** We say $A$ with entries $[a_{m,n}]$ is *periodic* [4, p. 132] with period $N$ if

$$a_{m+N,n} = a_{m,n} \quad \text{and} \quad a_{m,n+N} = a_{m,n},$$

where $m, n, N \in \mathbb{Z}$.

Recall from Definition 1 that this is pretty much exactly the way we defined functions on $\mathbb{Z}_N$, which we denoted as $z$, except now we are doing it in relation to a matrix. We explain this further with an example.

**Example 5.** Let $N = 5$, then

$$a_{-3,-2} = a_{-3+5,-2+5} = a_{2,3}.$$

**Definition 20.** A matrix $A$ with entries $[a_{m,n}]$ and period $N$ is called *circulant* [4, p. 132] if

$$a_{m+k,n+k} = a_{m,n}, \tag{1.26}$$

for all $m, n, k \in \mathbb{Z}$.

What we are saying is if we shift the indices for the row and column positions by the same integer $k$, then the entry we will end up at will be identical to the entry $a_{m,n}$, we could also interpret it as $A$ having identical elements along any diagonal. Furthermore, note that we defined this for all $m, n, k \in \mathbb{Z}$. This is fine because of the periodicity with period $N$.

We illustrate with an example.

**Example 6.**

$$A = \begin{bmatrix} 1 & 1+i & 2 \\ 2 & 1 & 1+i \\ 1+i & 2 & 1 \end{bmatrix},$$

the above matrix $A$ is circulant.

**Definition 21.** Let $z, w \in \ell^2(\mathbb{Z}_N)$, we denote the *convolution* [4, p. 134] of $z$ and $w$ as $z * w$, where $z * w \in \ell^2(\mathbb{Z}_N)$ with components

$$z * w(m) = \sum_{n=0}^{N-1} z(m-n)w(n),$$

for all $m \in \mathbb{Z}$.

The usefulness of the convolution operation will become apparent momentarily. As we will see, applying the DFT to convolution will greatly simplify the operation computationally.

**Example 7.** Let $z = (1, i, 2)$, $w = (0, 2, 1) \in \ell^2(\mathbb{Z}_N)$, then

$$\begin{aligned} z * w(0) &= \sum_{n=0}^{2} z(0-n)w(n) = z(0)w(0) + z(-1)w(1) + z(-2)w(2) \\ &= z(0)w(0) + z(2)w(1) + z(1)w(2) = 0 + 4 + i = 4 + i, \end{aligned}$$

analogously

$$\begin{aligned} z * w(1) &= \sum_{n=0}^{2} z(1-n)w(n) = 4, \\ z * w(2) &= \sum_{n=0}^{2} z(2-n)w(n) = 3 + 2i. \end{aligned}$$

So $z * w = (4 + i, 4, 3 + 2i)$.

**Definition 22.** Let $b \in \ell^2(\mathbb{Z}_N)$, we define

$$T_b : \ell^2(\mathbb{Z}_N) \to \ell^2(\mathbb{Z}_N) \tag{1.27}$$

by

$$T_b(z) = b * z,$$

for all $z \in \ell^2(\mathbb{Z}_N)$.

Thus, $T_b$ performs convolution on any function $z$ with some fixed function $b$. Furthermore, $T_b$ is sometimes called a *convolution operator* [4, p. 135].

**Lemma 12.** *A convolution operator $T_b$ for some $b \in \ell^2(\mathbb{Z}_N)$ is a linear transformation.*

*Proof.* This is not too hard to prove. We recall a function is a linear transformation if scalar multiplication and addition are preserved. That is if

$$T_b(\alpha z) = \alpha T_b(z) = \alpha(b * z),$$

for some $\alpha \in \mathbb{C}$ and

$$T_b(z + w) = T_b(z) + T_b(w) = b * z + b * w.$$

We begin by performing multiplication with a scalar, $\alpha$

$$T_b(\alpha z)(m) = \sum_{n=0}^{N-1} b(m-n) \cdot \alpha z(n) = \alpha \sum_{n=0}^{N-1} b(m-n)z(n) = \alpha(b * z)(m).$$

Now looking at addition for any $z, w \in \ell^2(\mathbb{Z}_N)$

$$\begin{aligned}T_b(z + w)(m) &= \sum_{n=0}^{N-1} b(m-n)(z+w)(n) = \sum_{n=0}^{N-1} b(m-n)(z(n) + w(n)) \\ &= \sum_{n=0}^{N-1} b(m-n)z(n) + \sum_{n=0}^{N-1} b(m-n)w(n) = b * z(m) + b * w(m).\end{aligned}$$

So $T_b$ is a linear transformation. □

We now have a look at how the DFT interacts with convolution. We will see that the DFT can greatly simplify the convolution operation.

**Lemma 13.** *Suppose $z, w \in \ell^2(\mathbb{Z}_N)$, then*

$$\widehat{(z * w)}(m) = \hat{z}(m)\hat{w}(m).$$

*Proof.* Recall we defined the DFT as

$$\hat{z}(m) = \sum_{n=0}^{N-1} z(n) e^{\frac{-2\pi i m n}{N}},$$

using this on convolution we get

$$\widehat{(z*w)}(m) = \sum_{n=0}^{N-1} (z*w)(n)e^{\frac{-2\pi imn}{N}} = \sum_{n=0}^{N-1} \left( \sum_{k=0}^{N-1} z(n-k)w(k)e^{\frac{-2\pi imn}{N}} \right).$$

We can now use a clever trick, note that

$$e^{\frac{-2\pi imn}{N}} = e^{\frac{-2\pi im(n-k)}{N}} \cdot e^{\frac{-2\pi imk}{N}},$$

using this, we can express the sums as

$$\sum_{n=0}^{N-1} \left( \sum_{k=0}^{N-1} z(n-k)w(k)e^{\frac{-2\pi imn}{N}} \right) = \sum_{k=0}^{N-1} w(k)e^{\frac{-2\pi imk}{N}} \sum_{n=0}^{N-1} z(n-k)e^{\frac{-2\pi im(n-k)}{N}}.$$

The left sum is already on the desired form and is equal to $\widehat{w}(m)$. For the right sum we change variables and let $p = n - k$, thus adjusting the summation to sum from $p = -k$ to $p = N - k - 1$. We get

$$\sum_{p=-k}^{N-k-1} z(p)e^{\frac{-2\pi imp}{N}} = \sum_{p=0}^{N-1} z(p)e^{\frac{-2\pi imp}{N}} = \widehat{z}(m).$$

The second step above holds since both $z$ and the exponential are periodic with period $N$. Thus, we have shown the DFT simplifies convolution to multiplication. $\square$

To finish off this chapter, we have a look at how one can go about adjusting the magnitude of specific components of the Fourier basis in ones DFT. Recall we defined the inverse DFT, that is the IDFT as

$$z = \sum_{k=0}^{N-1} \widehat{z}(k)F_k.$$

We previously expressed the IDFT using the variable $m$. We now make a small adjustment, replacing the notation $m$ with $k$, to be consistent with the literature. But we are otherwise defining it in the same way. We remind ourselves that $F_k$ is, as we recall, the $k$th component of the Fourier basis $F$ and $\widehat{z}(k)$ is the $k$th DFT coefficient or Fourier coefficient. The magnitude of these coefficients determines the strength, or as it sometimes is called, the weight of $F_k$. If we could adjust these coefficients $\widehat{z}(k)$, we could adjust the strength of specific components of the basis $F$ in our function $z$.

**Definition 23.** Let $m \in \ell^2(\mathbb{Z}_N)$. We define $T_m : \ell^2(\mathbb{Z}_N) \to \ell^2(\mathbb{Z}_N)$ by

$$T_m(z) = \widetilde{(m\widehat{z})}, \tag{1.28}$$

where $(m\widehat{z})$ is obtained by multiplying $m$ and $\widehat{z}$ componentwise, so $(m\widehat{z})(n) = m(n)\widehat{z}(n)$, for each $n$. Such a transformationation is called a *Fourier multiplier operator.*

Note that if we were to look at Equation (1.28) term-wise, then for each term $k$ we have

$$T_m(z)(k) = \widetilde{(m(k)\widehat{z}(m))}. \tag{1.29}$$

Thus, applying the DFT to the above equation yields

$$(\widehat{T_m(z)})(k) = m(k)\widehat{z}(m).$$

If we have a look at the components, $\widehat{z}(k)$ in the IDFT, we see that this is rather similar, but with a strength, or magnitude component $m(k)\widehat{z}(k)$, rather then just the $\widehat{z}(m)$. Thus, per the definition of the IDFT, multiplying this with the Fourier basis component $F_k$ yields the IDFT of $m(k)\widehat{z}(k)$, rather than just $\widehat{z}(k)$.

What this means is, if we look at $T_m(z)$ term-wise, where each term is $(\widehat{T_m(z)})(k)$, rather than the $\widehat{z}(k)$ in the definition of the IDFT, we can express it as

$$T_m(z) = \sum_{k=0}^{N-1} (\widehat{T_m(z)})(k)F_k = \sum_{k=0}^{N-1} m(k)\widehat{z}(k)F_k.$$

This clarifies and shows the usefulness of the Fourier multiplier operator. This is widely used within fields such as signal processing, where, using a Fourier multiplier operator, one can remove unwanted components of a signal. Such a method is within the field of signal processing called *filtering* [4, p. 139].

Furthermore, we note the similarities between the Fourier multiplier operator and convolution. Recall we proved in Lemma 13 that applying the DFT to convolution resulted in the multiplication of two functions $z, w$ componentwise. This is more or less exactly what we have performed in the Fourier multiplier operator, except we now perform multiplication with the Fourier basis thus performing IDFT. This means convolution is in fact, in terms of signal processing, also filtering!

# Chapter 2

# The Fast Fourier transformation

So far, we have defined the DFT and had a look at some of the many interesting properties that it possesses. In terms of actual application, however, that is, in terms of performing the actual computations, the DFT is rather slow and inefficient. This is where the *fast Fourier transformation* [4, p. 151], or FFT comes into big use. The FFT is an algorithm that minimizes the amount of operations needed to be performed to obtain the DFT of some function $z$. To be more clear, using the FFT we can reduce the amount of complex-multiplication by a significant amount. An interesting fact about the FFT is that it was actually initially formulated by Gauss [2, p. 57] as a means to approximate the orbits of the asteroids Pallas and Juno in 1805. Meaning it predated even Fourier's announcement of the Fourier series expansion. However, Gauss did not find the FFT to be a major breakthrough and his formulation only appeared much later, in 1866, in his compiled notes.

## 2.1 Change of Basis

Recall in Equation (1.15), for a vector $z \in \ell^2(\mathbb{Z}_N)$ expressed in some basis $B$ for $\ell^2(\mathbb{Z}_N)$ we denote it as $[z]_B$. Similarly, $z$ with respect to the standard basis is denoted as $[z]_E$. If we wish to obtain $[z]_B$, having $z$ with respect to $E$, we can obtain it by multiplying $[z]_E$ with the $E$ to $B$ change of basis matrix, which we can denote as $A$.

The standard change-of-basis calculation from linear algebra is done by

$$[z]_B = A[z]_E = Az.$$

To obtain each component of $Az$, we need to perform $N$ complex multiplications since component $m$ is computed by

$$\sum_{n=0}^{N-1} a_{m,n} z(n).$$

This means since $Az$ has a total of $N$ components we need to perform $N^2$ complex multiplications in order to compute the entire vector. The situation is the same in the case of the Fourier basis. Thus, if we wish to compute the DFT

$$\widehat{z} = W_N z.$$

Whereas in Definition 13, $W_N$ is the change of base matrix from the standard basis to the Fourier basis. The number of complex computations needed is the same as in the general case above. It is worth pointing out that the literature has varying definitions for the Fourier matrix. What usually differs is a scalar of $\frac{1}{\sqrt{N}}$. We defined the Fourier matrix as a matrix multiplied by the scalar $\frac{1}{\sqrt{N}}$, but the scalar $\frac{1}{\sqrt{N}}$ has a rather small impact on the overall time needed to perform the DFT computations. And in terms of computational complexity, it is of less importance. What is important to emphasize is the actual change of basis computation. This is because, as we recall, complex multiplication requires a total of 4 real multiplications. For two complex numbers $a_{m,n} = a + bi$ and $z(n) = c + di$, we compute

$$a_{m,n} z(n) = (a + bi)(c + di).$$

This is where the FFT comes in handy. We will see that the FFT can reduce the total amount of computations needed from $N^2$ to at most $\frac{1}{2}(N \log_2(N))$. It turns out the FFT maximizes the reduction in the number of computations when $N$ is a power of 2. That is, when $N = 2^n$, for some $n \in \mathbb{N}$. As an example, if $N = 2^{12} = 4096$, then the amount of complex multiplications needed is reduced from about 16 million to a mere 24 567.

## 2.2 The Simplest Case

In cases where $N$ is not a power of 2, we will see the FFT is still very useful. We have a look at a simple case where $N$ is an even integer. That is, when $N = 2M$, for some $M \in \mathbb{N}$.

**Lemma 14.** *Let $N = 2M$, $M \in \mathbb{N}$, $z \in \ell^2(\mathbb{Z}_N)$ and $u, v \in \ell^2(\mathbb{Z}_M)$. Furthermore define $u, v$ by*

$$\begin{aligned} u(k) &= z(2k), \quad k = 0, 1, \dots, M-1, \\ v(k) &= z(2k+1), \quad k = 0, 1, \dots, M-1. \end{aligned}$$

*That is, $u$ would consist of all even indices of $z$, and $v$ all the odd ones. Then, if $m = 0, 1, \dots, M-1$,*

$$\widehat{z}(m) = \widehat{u}(m) + e^{-\frac{2\pi i m}{N}} \widehat{v}(m). \tag{2.1}$$

*And, if $m = M, M+1, \dots, N-1$ we let $P = m - M$ such that $P = 0, 1, \dots, M-1$. Then*

$$\widehat{z}(m) = \widehat{u}(P) - e^{-\frac{2\pi i P}{N}} \widehat{v}(P). \tag{2.2}$$

*Proof.* Recall by definition, for any $m = 0, 1, \dots, N-1$,

$$\widehat{z}(m) = \sum_{n=0}^{N-1} z(n) e^{-\frac{2\pi i m n}{N}}.$$

This sum can be broken up into sums over the even values, $n = 2k$ and the odd values $n = 2k+1$, $k = 0, 1, \dots, M-1$. We end up with the two following sums:

$$\widehat{z}(m) = \sum_{k=0}^{M-1} \underbrace{z(2k)}_{=u(k)} e^{-\frac{2\pi i (2k) m}{N}} + \sum_{k=0}^{M-1} \underbrace{z(2k+1)}_{=v(k)} e^{-\frac{2\pi i (2k+1) m}{N}}.$$

In the first sum, we simply rewrite the exponent as $-\frac{2\pi i k m}{N/2}$, and for the second sum we first break out $e^{-\frac{2\pi i m}{N}}$, and then rewrite the exponent just as we did in the first sum, that is, as $-\frac{2\pi i k m}{N/2}$. Note however, since $N = 2M$, $N/2$ is just $M$, so what we end up with is

$$\widehat{z}(m) = \sum_{k=0}^{M-1} u(k) e^{-\frac{2\pi i k m}{M}} + e^{-\frac{2\pi i m}{N}} \sum_{k=0}^{M-1} v(k) e^{-\frac{2\pi i k m}{M}}. \tag{2.3}$$

If, in Equation (2.3), $m = 0, 1, \dots, M-1$, the two sums add up to $\widehat{u}(m) + e^{-\frac{2\pi i m}{N}} \widehat{v}(m)$, which is what we had in Equation (2.1). If, on the other hand,

$m = M, M+1, \ldots, N-1$ we as previously mentioned use $P = m - M$ or equivalently $m = P + M$. If we replace $m$ with $P + M$ in Equation (2.3) we get

$$\widehat{z}(m) = \sum_{k=0}^{M-1} u(k) e^{-\frac{2\pi i k(P+M)}{M}} + e^{-\frac{2\pi i(P+M)}{N}} \sum_{k=0}^{M-1} v(k) e^{-\frac{2\pi i k(P+M)}{M}}.$$

This simplifies to Equation (2.2) by firstly observing that $e^{-\frac{2\pi i k(P+M)}{M}} = e^{-\frac{2\pi i k P}{M}}$, due to the $M$-periodicity. Secondly, $e^{-\frac{2\pi i(P+M)}{N}} = e^{-\frac{2\pi i P}{N}} \cdot e^{-\frac{2\pi i M}{N}}$. But, since $M = N/2$, the exponent in the last term is $\pi i$, and $e^{\pi i} = -1$. Thus we end up with

$$\widehat{z}(m) = \sum_{k=0}^{M-1} u(k) e^{-\frac{2\pi i P}{M}} - e^{-\frac{2\pi i P}{N}} \sum_{k=0}^{M-1} v(k) e^{-\frac{2\pi i P}{M}} = \widehat{u}(P) - \widehat{v}(P) e^{-\frac{2\pi i P}{N}},$$

which was the desired form. □

This is the basic structure of the FFT algorithm. We obtain the values $\widehat{z}(0), \widehat{z}(1), \ldots, \widehat{z}(M-1)$ using $\widehat{u}(m)$ and $\widehat{v}(m)$ accordingly to Equation (2.1), and the values $\widehat{z}(M), \widehat{z}(M+1), \ldots, \widehat{z}(N-1)$ accordingly to Equation (2.2). This means we obtain $\widehat{z}$ by computing and using $\widehat{u}$ and $\widehat{v}$, both of which are of length $M$, thus requiring $M^2$ complex multiplications respectively. We then need to compute the products $e^{-\frac{2\pi i m}{N}} \widehat{v}(m)$ for $m = 0, 1, \ldots, M-1$, adding another $M$ multiplications. The rest of the computations are done using addition and subtraction, which usually is not taken into account when discussing computational complexity. This means in the above case, the most basic one that is, we at most require a number of complex multiplications totaling at

$$2M^2 + M = 2\left(\frac{N}{2}\right)^2 + \frac{N}{2} = \frac{1}{2}(N^2 + N).$$

When $N$ is large, this is essentially $\frac{N^2}{2}$. So in the most basic case, we have already cut the number of complex multiplications by half.

## 2.3 The Most Favorable Case

So what would happen if $N$ was not just divisible by 2, but by 4? Well, we could divide $N$ by 2 and define the vectors $u, v$ as above, but we could then take things one step further. If $N$ is divisible by 4, that would mean the order of $u$ and $v$ would have to be divisible by 2. Hence, we can apply the same algorithm on $u$ and $v$, reducing the computational complexity even further. This can be generalized and is the reason why the most favorable case is when $N$ is a power of 2. As an example, if $N$ was $2^9 = 512$, we could represent $F_{512}$ by $F_{256}$, which in return could be represented by $F_{128} \to F_{64} \to F_{32} \to \cdots \to F_2$. In fact, this case is so computationally efficient that if $N \neq 2^n$, vectors can be padded with zeroes until $N = 2^n$. Thus, the example above was more of a demonstrative case to get a better understanding of how the algorithm works.

**Definition 24.** Define $\#_N$ to be the least number of complex multiplications required to compute the DFT of a vector of length $N$.

Using this notation, the FFT algorithm reduces the number of complex multiplications to

$$\#_N \leq 2\#_M + M. \tag{2.4}$$

**Lemma 15.** *Suppose $N = 2^n$, $n \in \mathbb{N}$. Then*

$$\#_N \leq \frac{1}{2} N \log_2 N.$$

*Proof.* This proof is easily done by induction on $n$. When $n = 1$, $N = 2$ and $z$ is a vector consisting of only two elements, $z = (a, b)$. the Fourier matrix, $F$ is

$$W_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix},$$

$\hat{z} = W_2 z = (a + b, a - b)$. As we can see, for $n = 1$, the statement holds since $\#_2 = 0$ and $\frac{1}{2} \cdot 2 \log_2 2 = 1$. We suppose, by induction, that it also holds for $n = k - 1$. Then for $n = k$, we have by Equation (2.4) as well as the induction hypothesis that

$$\#_{2^k} \leq 2\#_{\frac{2^k}{2}} + \frac{2^k}{2} = \#_{2^{k-1}} + 2^{k-1} \leq 2 \cdot \frac{1}{2} 2^{k-1}(k-1) + 2^{k-1} = \frac{1}{2} k 2^k = \frac{1}{2} N \log_2 N.$$

□

# Chapter 3

# Applications

A natural way to finish this essay is to look at some applicative demonstrations of the FFT, as we shall in this chapter. We look at three applicative cases, namely noise filtering of a signal, computing the derivative of a function, and the integral of a function. We begin by using the FFT to denoise a signal, we then move on to differentiation and demonstrate how the FFT can be used to approximate derivatives, as well as primitives. The Python code used can be found in the Appendix.

## 3.1 Noise Filtering

A function, within the field of signal processing, is referred to as a *signal*. Thus, in accordance, we refer to our function as a signal in this chapter. These signals can be audio, imagery or video, however, we will only study the case of audio. The purpose of these sub-fields of signal processing is to analyze, process and modify signals. The term *noise* refers to an unwanted, often interfering part of the signal that one would like to either reduce or remove entirely, hence the term noise filtering.

The demonstrative example is in great resemblance to the one in [2, p. 60]. What we will do is begin with a rather simple signal, and to that signal add *Gaussian white noise*, which is white noise following a Gaussian distribution [7, p. 406]. Gaussian white noise is used in signal processing to mimic random noise effects that occur in nature. The process is rather simple. We begin by comparing our signal with the noisy signal, we will add a rather large amount of Gaussian white noise so that the difference will be more pronounced. We apply the FFT to both signals to get a better understanding of what the Gaussian white noise looks like and what its impact is on the original signal. Remember, applying the DFT, or FFT to a signal allows us to see each of the frequencies in the signal as well as their respective magnitudes. The idea is to identify the frequencies with lower magnitudes, hence lower overall weight in the signal. This is the noise. We then filter out the Gaussian white noise. Once filtered out, we apply the IFFT to obtain the time-domain signal and compare it with the original clean signal. Our signal, denoted as $f$ is

$$f(t) = \sin(2\pi \cdot 50t) + \sin(2\pi \cdot 120t).$$

Following the literature and common notation within the field, we denote our input variable as $t$. This is a natural notation as the discrete inputs of the signal, $t$ are in time.

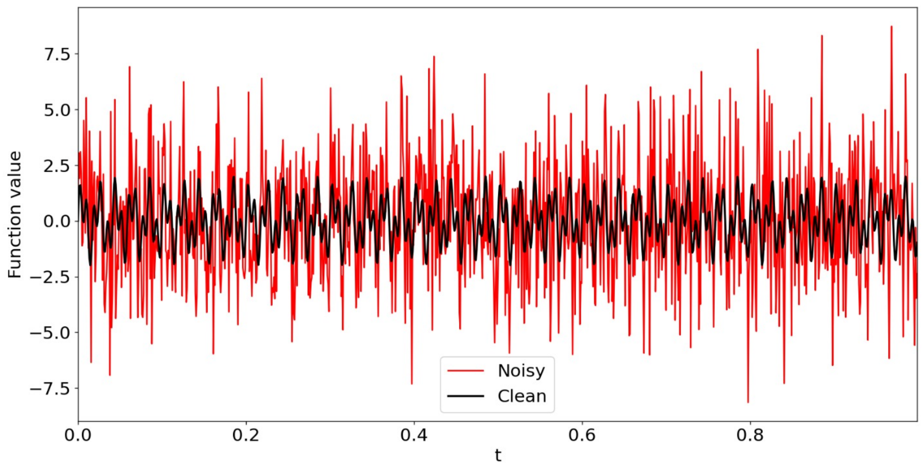


Figure 3: Comparing original signal with and without added noise.

As previously explained, we begin by comparing $f$ to itself, but with added noise. We plot the two functions, see Figure 3. Here the original signal is called "Clean", in black, and "Noisy", in red, is the noisy signal. We plot over the period of one second and sample $2^{10}$ discrete data points.

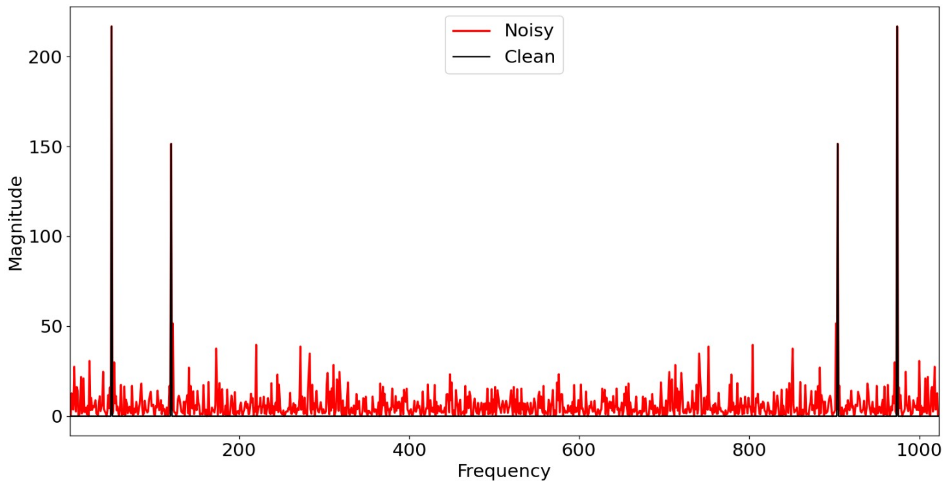


Figure 4: Comparing FFT of signal with and without added noise.

In Figure 4 we apply the FFT to the noisy signal as well as the original signal. The noise is in red, and the original is in blue. There are four big spikes in magnitude which are identical for both signals. The reason behind the symmetrical look is that the sinusoidal waves have equal positive and negative peaks in absolute terms. We filter out this noise by computing the so-called *Power Spectrum Density* (PSD) [2, p. 60], which is a way to measure the strength of each frequency component in the signal. We do this by squaring the magnitudes and then normalizing by the periodicity $N$, that is

$$PSD(m) = \frac{|\widehat{z}(m)|^2}{N}.$$

The normalization is used to get a power-measurement fit to the frequency interval we are using. It is clear the four frequencies have a higher magnitude than the noise, so we filter out any frequency with PSD under 100.

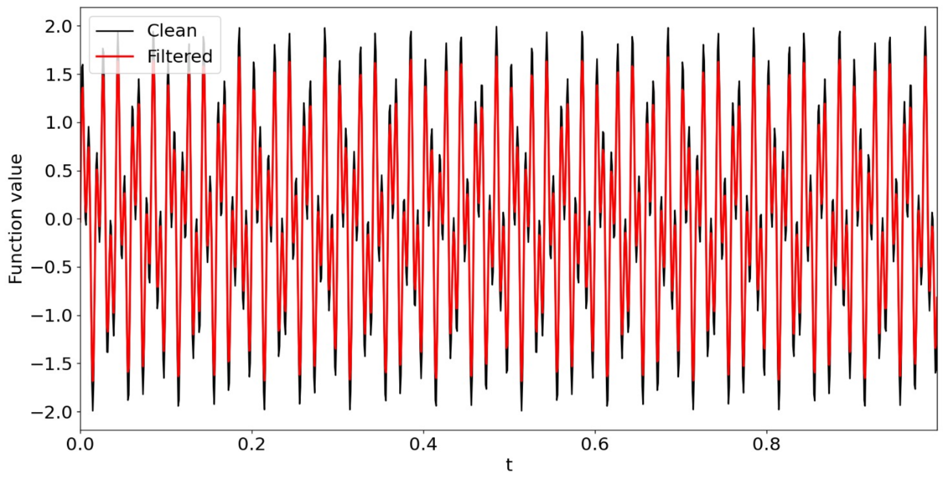


Figure 5: Comparing our original signal with the filtered one.

Figure 5 shows what the two signals look like in comparison after we have removed the noise and transformationed them back using the IFFT. As we can see, after filtering out the Gaussian white noise, the filtered signal looks almost identical to the original one. The results conclude the FFT is an excellent tool for denoising signals.

## 3.2 Spectral Derivative

Another interesting application of the FFT is within numerical computations of derivatives. The way it works is rather simple. Given a function $f(x)$, we can apply the FFT, once transformationed, we multiply each component by $i\frac{2\pi m}{N}$, applying the IFFT to this then gives us the derivative of the function but in the time domain. The term $\frac{2\pi m}{N}$ is often denoted as $\kappa(m)$. Again, we adjust our notations from $n$ to $x$ per the literature and common notation within the study.

It is often called the *spectral derivative* [2, p. 61] simply because the frequency domain can be referred to as the spectral domain. As in the case of noise filtering, this example coincides with an example one may find in [2, p. 62]. We look at the function $f(x) = \cos(x)e^{-\frac{x^2}{25}}$. We plot the analytical derivative and compare it with the numerical derivative obtained by two different numerical methods. Firstly using the FFT, secondly using

*Euler's forward method* or the *forward difference method* [10, p. 339], defined as

$$\frac{df}{dx}(x_k) \approx \frac{f(x_{k+1}) - f(x_k)}{\Delta x}.$$

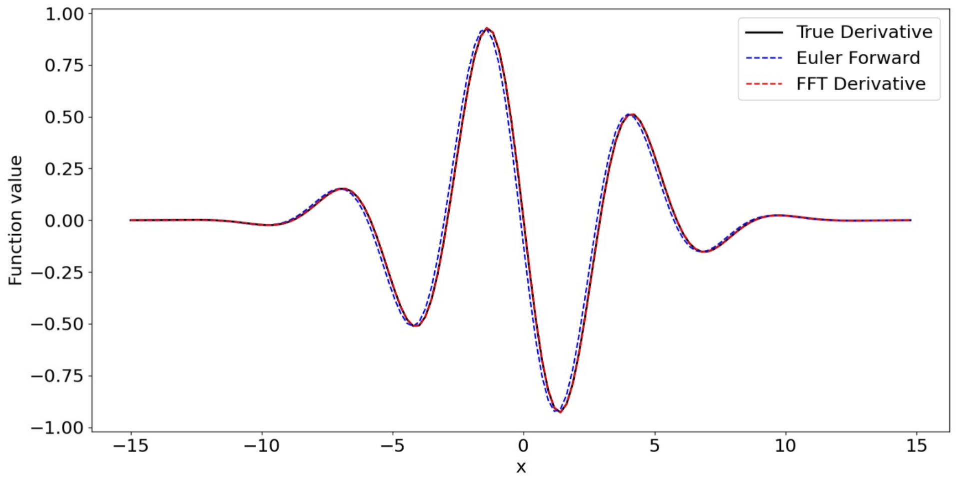


Figure 6: Derivative comparison.

In Figure 6, the black line denotes the true derivative or the analytical derivative. The red dotted line represents the numerical derivative obtained by the FFT, and the blue dotted line is the numerical derivative obtained by the Forward difference. As we can see, both numerical methods do a good job of approximating the analytic solution, however, the forward difference has a clear shift which is expected with the method, cf. [10, p. 343]. Thus the FFT is a powerful tool for differentiation.

## 3.3 Error Comparison on Numerical Derivatives

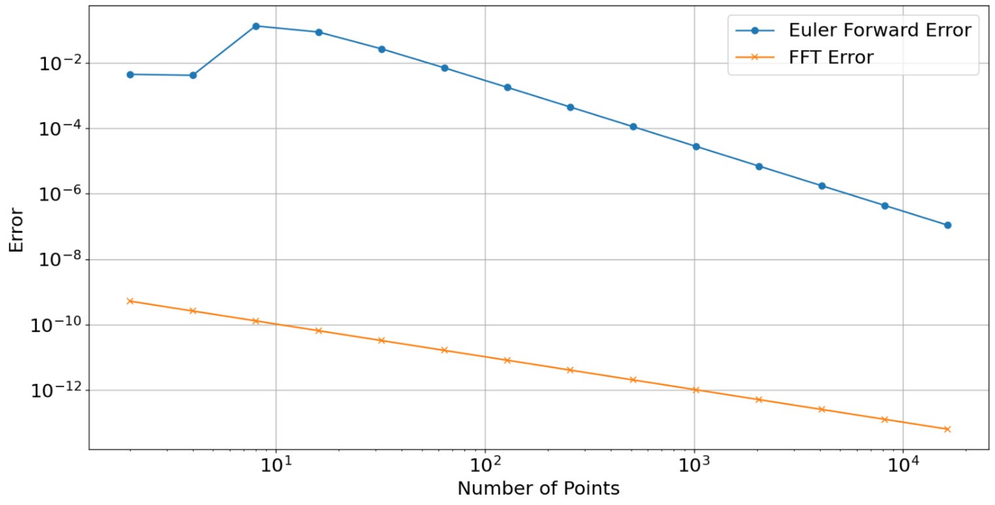


Figure 7: Error comparison.

To get a better understanding of how the errors compare, we compare them when increasing $N$ in powers of 2, ranging from 2 to $2^{14}$. We compare them using *mean square error* (MSE) [9, p. 28], where the analytical values are used as the true values. We denote $\widehat{\frac{df}{dx}}(x_i)$ as the values obtained by the numerical method in a discrete point $x_i$, and $\frac{df}{dx}(x_i)$ denote the analytically computed values,

$$\text{MSE} = \frac{1}{N}\sum_{i=0}^{N-1}\left(\widehat{\frac{df}{dx}}(x_i) - \frac{df}{dx}(x_i)\right)^2 .$$

Figure 7 shows that the FFT has a significantly smaller error for smaller $N$, measured by powers of 10. They are, however, both decreasing as $N$ increases. This once again establishes the FFT as a powerful tool for differentiating.

## 3.4 Integration

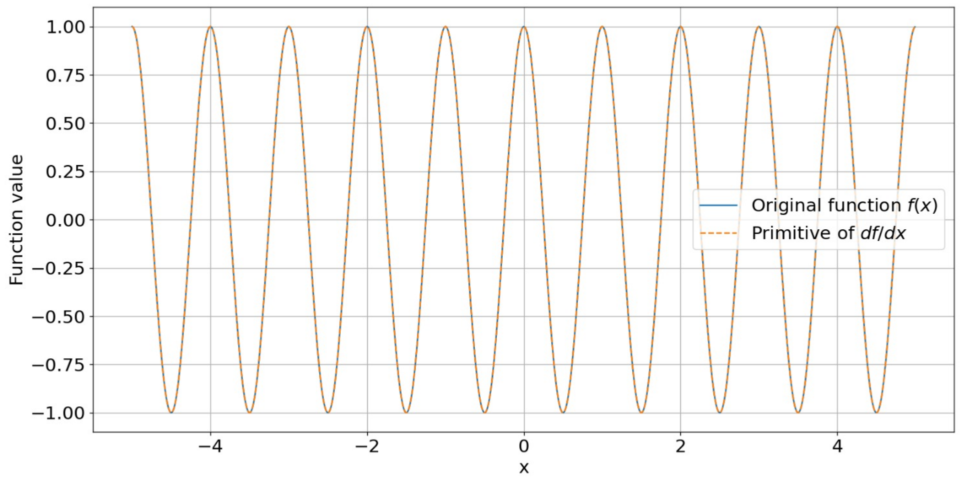


Figure 8: Original function compared with FFT integral.

As a final applicative demonstration, we look at how the FFT can be used to compute the integral, or primitive of a function. The process is almost identical to the one of differentiating, except once the FFT is applied to a given function, we multiply by $\frac{1}{i\kappa(m)}$ [1, p. 2], before applying the IFFT. In this example we integrated $f'(x) = -2\pi \sin(2\pi x)$ and compare it to $f(x) = \cos(2\pi x)$. Figure 8 shows our results, where the blue line is the function $\cos(2\pi x)$ and the orange dotted line is the primitive of $-2\pi \sin(2\pi x)$. As we can see, the method yields a near-perfect match. Thus, the FFT is a prominent method for integrating.

# Bibliography

[1] Rune Brincker and Anders Brandt, *FFT integration of time series using an overlapp-add technique*, Proceedings of the International Modal Analysis Conference (2010).

[2] Steven L. Brunton and J. Nathan Kutz, *Data-driven science and engineering*, Cambridge University Press, 2019.

[3] Rodica D. Costin, *An introduction to hilbert spaces*, The Ohio State University, 2019.

[4] Michael W. Frazier, *An introduction to wavelets through linear algebra*, Springer, 1999.

[5] Stephen H. Friedberg, Arnold J. Insel, and Lawrence E. Spence, *Linear algebra*, Pearson, 2002.

[6] Ivor Grattan-Guinness, *Joseph Fourier 1768–1830*, The MIT Press, 1972.

[7] John A. Gubner, *Probability and random process for electrical and computer engineering*, Cambridge University Press, 2006.

[8] Roger A. Horn, *Matrix analysis*, Cambridge University Press, 1994.

[9] Gareth James et al., *An introduction to statistical learning*, Springer, 2023.

[10] Qingkai Kong, Timmy Siauw, and Alexandre M. Bayen, *Python programming and numerical methods*, Academic Press, 2021.

[11] Oyvind Ryan, *Linear algebra, signal processing, and wavelets*, Springer, 2017.

[12] Edward B. Saff and Arthur D. Snider, *Fundamentals of complex analysis engineering, science and mathematics*, Pearson, 2014.

[13] Michael Sullivan, *Precalculus*, Pearson, 2020.

# Appendix

## Python Code

```
import numpy as np
import matplotlib

matplotlib.use('TkAgg')
import matplotlib.pyplot as plt

plt.rcParams['figure.figsize'] = [16, 12]
plt.rcParams.update({'font.size': 18})


""" Denoising a signal """

#  Defining our function, interval and adding noise.


# Create a simple signal with two frequencies
dt = 0.0009765625  # Our step-size, which is 1/2^10. So n = 0
                              ,1,..., 1024.
t = np.arange(0, 1, dt)
z = np.sin(2 * np.pi * 50 * t) + np.sin(2 * np.pi * 120 * t)
                               # Sum of 2 frequencies
z_clean = z
z_noised = z + 2.5 * np.random.randn(len(t))  # Add some
                              noise

# Compute the Fast Fourier transformation (FFT)

n = len(t)  # This is our N, we sample a total of
# 1024 samples in the interval [0,1]
z_hat = np.fft.fft(z_noised, n)  # Compute the FFT on our
                              function with noise.
```

```
PSD = z_hat * np.conj(z_hat) / n  # Computing the Power
                               Spectrum Density
freq = (1 / (dt * n)) * np.arange(n)  # Create x-axis of
                               frequencies in Hz
L = np.arange(1, np.floor(n),dtype='int')

# Use the PSD to filter out noise

indices = PSD > 100  # Find all frequencies with large power
PSD_clean = PSD * indices  # Zero out all others
z_hat = indices * z_hat  # Zero out small Fourier
                               coefficients
z_filtered = np.fft.ifft(z_hat)  # Inverse FFT for filtered
                               time signal

# Plotting our results

plt.plot(t, z_noised, color='r', linewidth=1.5, label='Noisy'
                               )
plt.plot(t, z_clean, color='k', linewidth=2, label='Clean')
plt.xlim(t[0], t[-1])
plt.xlabel('t')
plt.ylabel('Function value')
plt.legend()
plt.show()


plt.plot(t, z_clean, color='k', linewidth=1.5, label='Clean')
plt.plot(t, z_filtered, color='r', linewidth=2, label='
                               Filtered')
plt.xlim(t[0], t[-1])
plt.xlabel('t')
plt.ylabel('Function value')
plt.legend()
plt.show()


plt.plot(freq[L], PSD[L], color='r', linewidth=2, label='
                               Noisy')
plt.plot(freq[L], PSD_clean[L], color='k', linewidth=1.5,
                               label='Clean')
plt.xlim(freq[L[0]], freq[L[-1]])
plt.xlabel('Frequency')
plt.ylabel('Magnitude')
plt.legend()
plt.show()

plt.show()

""" Spectral derivative """

n = 128
L = 30
dx = L / n
x = np.arange(-L / 2, L / 2, dx, dtype='complex_')
f = np.cos(x) * np.exp(-np.power(x, 2) / 25)  # Function
df = -(np.sin(x) * np.exp(-np.power(x, 2) / 25) + (2 / 25) *
                               x * f)  # Derivative

# Euler's Forward

dfFD = np.zeros(len(df), dtype='complex_')
for kappa in range(len(df) - 1):
    dfFD[kappa] = (f[kappa + 1] - f[kappa]) / dx  # Array,
                                   Euler's Forward

dfFD[-1] = dfFD[-2]

# Derivative using FFT (spectral derivative)
fhat = np.fft.fft(f)  # FFT on f
kappa = (2 * np.pi / L) * np.arange(-n / 2, n / 2)
kappa = np.fft.fftshift(kappa)  # Re-order fft frequencies
dfhat = kappa * fhat * (1j)  # Array, ikappa * FFT
dfFFT = np.real(np.fft.ifft(dfhat))  # inverse of dfhat,
                               derivative
# Plots
plt.plot(x, df.real, color='k', linewidth=2, label="True
                               Derivative")
plt.plot(x, dfFD.real, '--', color='b', linewidth=1.5, label=
                               "Euler Forward")
plt.plot(x, dfFFT.real, '--', color='r', linewidth=1.5, label
                               ="FFT Derivative")
plt.xlabel('x')
plt.ylabel('Function value')
plt.legend()
plt.show()

""" Error comparison, numerical derivative """

n = 131072
```

```
L = 30
dx = L / n
x = np.arange(-L / 2, L / 2, dx, dtype='complex_')
f = np.cos(x) * np.exp(-np.power(x, 2) / 25)  # Function
df = -(np.sin(x) * np.exp(-np.power(x, 2) / 25) + (2 / 25) *
                               x * f)  # Derivative

n_list = [2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048,
                               4096, 8192, 16384]

error_forward_list = []
error_FFT_list = []

for i in n_list:
    dx = L / i
    x = np.arange(-L / 2, L / 2, dx, dtype='complex_')
    f = np.cos(x) * np.exp(-np.power(x, 2) / 25)  # Function
    df = -(np.sin(x) * np.exp(-np.power(x, 2) / 25) + (2 / 25
                                   ) * x * f)  # Derivative

    # Euler's Forward Difference
    df_FD = np.zeros(len(df), dtype='complex_')
    for kappa in range(len(df) - 1):
        df_FD[kappa] = (f[kappa + 1] - f[kappa]) / dx  #
                                       Array, Euler's Forward

    df_FD[-1] = df_FD[-2]

    # Spectral Derivative, FFT
    f_hat = np.fft.fft(f)  # FFT on f
    kappa = (2 * np.pi / L) * np.arange(-i / 2, i / 2)
    kappa = np.fft.fftshift(kappa)  # Re-order fft
                                   frequencies
    df_hat = kappa * f_hat * (1j)  # Array, ikappa * FFT
    df_FFT = np.real(np.fft.ifft(df_hat))  # inverse of dfhat
                                   , derivative

    error_forward = (np.sum((df - df_FD) ** 2)) / i
    error_FFT = (np.sum(df - df_FFT)) ** 2 / i

    error_forward_list.append(error_forward)
    error_FFT_list.append(error_FFT)

plt.figure(figsize=(10, 6))
plt.plot(n_list, error_forward_list, label='Euler Forward
                                Error', marker='o')
plt.plot(n_list, error_FFT_list, label='FFT Error', marker='x
                                ')
plt.xlabel('Number of Points')
plt.ylabel('Error')
#plt.title('Error Comparison between Numerical Methods')
plt.xscale('log')
plt.yscale('log')
plt.legend()
plt.grid(True)
plt.show()



""" Integration """

# Original setup
n = 1024
L = 10
dx = L / n
x = np.arange(-L / 2, L / 2, dx, dtype='complex_')
f = np.cos(2 * np.pi * x)
df = -2 * np.pi * np.sin(2 * np.pi * x)
# FFT of the derivative
dfhat = np.fft.fft(df)
kappa = (2 * np.pi / L) * np.arange(-n / 2, n / 2)
kappa = np.fft.fftshift(kappa)  # Re-order fft frequencies

# Handling the zero frequency to avoid division by zero
kappa[0] = 1  # Avoid division by zero
fhat = dfhat / (1j * kappa)
fhat[0] = 0  # Set the zero frequency component to 0 (or the
                                integral constant if known)

# Inverse FFT to get the primitive
f_primitive = np.real(np.fft.ifft(fhat))

# Plotting the original function and its primitive
plt.figure(figsize=(10, 6))
plt.plot(x, f, label='Original function $f(x)$')
plt.plot(x, f_primitive, label='Primitive of $df/dx$',
                                linestyle='dashed')
plt.xlabel('x')
plt.ylabel('Function value')
```

```
#plt.title('Original Function and Its Primitive')
plt.legend()
plt.grid(True)
plt.show()
```