Queen Mary, University of London

School of Mathematical Sciences

Felipe Puiggari Medici

Fourier Transforms

Final Project for BSc Mathematics

Supervisor: Dr. Arick Shao

Spring Semester 2024

Abstract

We introduce the mathematical foundations of Fourier Transforms, and explore applications of these transforms in partial differential equations and signal processing. The primary objective is to provide a thorough mathematical support for implementations of important algorithms, that can later be extended to other scenarios. The later part of the project specializes in audio denoising and computational solutions of PDEs with worked examples.

Contents

0 Introduction and Preliminary Material **2**
0.1 Introduction: . . . 2
0.2 Preliminary material . . . 2

1 Foundations of Fourier Transforms **4**
1.1 Definition and properties . . . 4
1.1.1 Definition . . . 4
1.1.2 Properties . . . 4
1.2 Gaussian Family . . . 5
1.3 Fourier Inversion Theorem . . . 6
1.3.1 Image of Fourier Transforms . . . 6
1.3.2 Fourier Inversion Theorem . . . 7
1.4 Heat Equation . . . 9
1.5 Parseval and Plancherel Theorems . . . 10

2 Signal Processing, DFT and FFT **12**
2.1 Signal Analysis . . . 12
2.1.1 Brief mention of Anti-Aliasing and Uncertainty Principle . . . 13
2.2 Discrete Fourier Transform . . . 13
2.2.1 Definition and Inverse Fourier Transform . . . 14
2.3 Fast Fourier Transform . . . 16
2.3.1 Fast Fourier Transform Algorithm: . . . 16
2.4 Denoising Signals . . . 19
2.5 Partial Differential Equations and the FFT . . . 21

Conclusion and Acknowledgements **24**

Chapter 0: Introduction and Preliminary Material

0.1 Introduction:

Fourier Transforms were first introduced by the mathematician Joseph Fourier in his study of the Heat Equation. Since then, the Fourier Transform has been studied in many fields of Mathematics and Physics, such as in Functional Analysis, Quantum Mechanics, Computer Science and Electrical Engineering. The purpose of this project is to give a mathematically rigorous foundation of Fourier Transforms, and explore two of its current most important applications: signal processing and partial differential equations.
Fourier Transforms are rarely studied on their own. Most sources will include the study of such transforms in one of the areas of applications or in a pure mathematics context such as in Fourier Analysis or Functional Analysis. However, Fourier Transforms are used in a wide variety of fields, and this project attempts to utilize a diverse set of sources from areas such as Pure Mathematics, Electrical Engineering and Data Science in order to show some applications and why these work. The main focus of this project will be the algorithms and computational applications of Fourier Transforms in signal processing and partial differential equations, together with their respective mathematical foundations.
Another reason for studying Fourier Transforms which will be given importance during this project is the complexity analysis of the Fast Fourier Transform, and the applications of this algorithm considered one of the current most important algorithms. In the last part of this project, I will give two examples of the applications of the Fast Fourier Transform: audio denoising and computational solutions for PDEs. Such methods could be extended upon further research in areas of Scientific Computing and other fields. For instance, the Sampling Theorem can be paired with psycho-acoustic modelling in order to create MP3 files, or computational solutions of PDEs can be paired with geological data in order to model more complicated systems such as earthquakes or tsunamis. This is why, the topic of Fourier Transforms is so relevant today.

0.2 Preliminary material

Throughout this project I will make use of the Schwartz Space, which is also called the space of rapidly decreasing functions. I will use a definition similar to [6][Chapter 6, Def. 6.3, p. 342]. I will also use that a function of several variables is in Schwartz space with respect to a variable x if we hold the other variables constant.

Definition 0.2.1. *A function $f : \mathbb{R} \to \mathbb{C}$ is rapidly decreasing if f is infinitely differentiable and for all integers $m, n \geq 0$:*

$$||f||_{m,n} := \sup\{|x^m f^{(n)}(x)| : x \in \mathbb{R}\} < \infty \tag{1}$$

Let the Schwartz space denoted by $S(\mathbb{R}, \mathbb{C})$ be the set of all rapidly decreasing functions from $\mathbb{R}$ to $\mathbb{C}$.

Any function in Schwartz Space will be infinitely differentiable everywhere, and the product of any of the derivatives with a polynomial of arbitrary degree is bounded. Another family of function spaces that are relevant to the topic of Fourier Transforms are the $L^p(\mathbb{R})$ spaces which are defined as follows:

Definition 0.2.2. *For some $p \in [1, \infty)$ and $f : \mathbb{R} \to \mathbb{C}$, we have that $f \in L^p(\mathbb{R})$ if*

$$||f||_p := (\int_{-\infty}^{\infty} |f(x)|^p \, dx)^{\frac{1}{p}} < \infty \tag{2}$$

We also have that $S(\mathbb{R},\mathbb{C}) \subset L^p(\mathbb{R})$ for any $p \in [1,\infty)$. Furthermore, any smooth function with compact support is in Schwartz Space, and the product of two functions in Schwartz Space will be in Schwartz Space. This will be useful when we need to use Fubini's Theorem for instance. However, in order to keep focus on Fourier Transforms, these statements won't be proven in this project.
On the other hand, convolution will be an important operation throughout this project. The definition of convolution is from Asadzadeh's [1][4.3, def 13, p.115]

Definition 0.2.3. *If f, g are functions in $S(\mathbb{R},\mathbb{C})$, their convolution is the function $f * g$ defined by*

$$f * g(x) = \int_{-\infty}^{\infty} f(x-y)g(y)\,dy \quad \forall x \in \mathbb{R} \tag{3}$$

Some of its properties as per Asadzadeh's [1][4.3, Thm. 16] are associativity, commutativity and distributive law (proofs omitted).

We will also need Lebesgue Dominated Convergence Theorem which I will not prove in this project.

Theorem 0.2.1 (Lebesgue Dominated Convergence Theorem)**.** *Let $(f_n)_{n\geq 0}$ be a sequence of functions in $L^1(\mathbb{R})$, and suppose (f_n) converges pointwise to $f : \mathbb{R} \to \mathbb{C}$. Moreover, assume that there is some $g \in L^1(\mathbb{R})$ such that*

$$|f_n(x)| \leq g(x), \quad x \in \mathbb{R} \tag{4}$$

Then, $f \in L^1(\mathbb{R})$ and

$$\lim_{n\to\infty} \int_{-\infty}^{\infty} f_n(x)\,dx = \int_{-\infty}^{\infty} f(x)\,dx \tag{5}$$

I will also use some background from Complexity Theory of algorithms in this project. I will use the following definition from MTH6105: Algorithmic Graph Theory [8].

Definition 0.2.4. *Suppose an algorithm has input size N. Then the algorithm $f(N)$ is in $O(g(N))$ iff there exists some $c \in \mathbb{R}$ and some $n_0 \in \mathbb{N}$ such that for any natural number $n : n \geq n_0$, then $f(n) \leq cg(n)$*

I will also use that the composition of algorithms that have complexity $O(f(N))$ and $O(g(N))$, that is to say, performing algorithms inside algorithms will be of complexity $O(f(N)g(N))$. Proof of this is outside the scope of this project.
Further required background includes basic knowledge of complex analysis and of methods for ODEs.

Chapter 1: Foundations of Fourier Transforms

1.1 Definition and properties

1.1.1 Definition

The definition I will use throughout the project are from Ko's lecture notes [11][week 12, 1.2],

Definition 1.1.1 (Fourier Transform)**.** *Let $f \in S(\mathbb{R}, \mathbb{C})$. Then the Fourier transform of f is the function*

$$\mathcal{F}[f(x)] = \hat{f} : \mathbb{R} \to \mathbb{C}, \quad \hat{f}(\xi) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} f(x) e^{-i\xi x} dx \tag{1.1}$$

1.1.2 Properties

The following properties are important properties of Fourier Transforms, as they will help to solve PDEs of different types analytically. I will give an example of this at a later stage. The first four are from Ko's[11][1.3, Theorem 3.1, 3.2, 3.3, 3.6, p.3]) and the latter is from Gu[9][Chapt. 2, Prop. 2, p.5].

Theorem 1.1.1 (Basic Properties)**.** *Suppose $f \in S(\mathbb{R}, \mathbb{C})$ then*

(a) For any $a \in \mathbb{R}$, we have

(1) $\mathcal{F}[f(x-a)](\xi) = e^{-ia\xi}\hat{f}(\xi)$
(2) $\mathcal{F}[e^{iax}f(x)](\xi) = \hat{f}(\xi - a)$

(b) For $\lambda \in \mathbb{R} \setminus \{0\}$ then the scaling formula is as follows:

$$\mathcal{F}[f(\lambda x)](\xi) = |\lambda|^{-1}\hat{f}(\lambda^{-1}\xi) \tag{1.2}$$

(c) $\mathcal{F}[xf(x)](\xi) = i\hat{f}'(\xi)$

(d) Suppose $g \in S(\mathbb{R}, \mathbb{C})$

$$\int_{-\infty}^{\infty} \hat{f}(x)g(x)\,dx = \int_{-\infty}^{\infty} f(x)\hat{g}(x)\,dx \tag{1.3}$$

(a) and (b) will follow from a change of variables. (c) will follow from $\frac{\partial}{\partial\xi}e^{-i\xi x} = \frac{e^{-i\xi x}}{ix}$ and from applying Leibniz Integral Rule. Proof of (d) is as follows:

Proof. We first apply the definition 1.1.1 and later as $f, g \in S(\mathbb{R}, \mathbb{C})$ we can apply Fubini's theorem as follows:

$$\int_{-\infty}^{\infty} \hat{f}(x)g(x)\,dx = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(x)f(y)e^{-iyx}\,dy\,dx \tag{1.4}$$

$$= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} f(y) \int_{-\infty}^{\infty} g(x)e^{-iyx}\,dx\,dy = \int_{-\infty}^{\infty} f(y)\hat{g}(y)\,dy \qquad \text{Fubini's Theorem}$$

□

The following theorem is also an important theorem for computing Fourier Transforms, and it is useful in the analytic solutions of PDEs.

Theorem 1.1.2 (Convolution Theorem)**.** *Suppose $f, g \in S(\mathbb{R}, \mathbb{C})$ then*

$$\mathcal{F}[f * g] = (f \hat{*} g) = \sqrt{2\pi}\hat{f}\hat{g} \tag{1.5}$$

Proof is similar to Asadzadeh's [1][4.3, Thm. 16]

Proof. On the one hand as $f, g \in S(\mathbb{R}, \mathbb{C})$ we will be able to use Fubini's Theorem. Furthermore, let $z = x - y$. As x is independent of y, we can treat y constant in this context when deriving to get $\frac{dz}{dx} = 1$. Therefore

$$(f \hat{*} g)(\xi) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x-y)g(y)e^{-i\xi x}\, dx\, dy = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(z)g(y)e^{-i\xi(z+y)}\, dz\, dy \tag{1.6}$$

$$= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} g(y)e^{-i\xi y}\left(\int_{-\infty}^{\infty} f(z)e^{-i\xi z}\, dz\right) dy = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} g(y)e^{-i\xi y}\, dy \int_{-\infty}^{\infty} f(z)e^{-i\xi z}\, dz = \sqrt{2\pi}\hat{f}\hat{g}(\xi) \tag{1.7}$$

as required □

1.2 Gaussian Family

We will now explore a family of important functions, as a way of providing an example of applying Fourier Transforms, and also as this family of functions will be important later on in the project. The following definition is from Gu[9][Chapt. 2, def.3, p.4]

Definition 1.2.1 (Gaussian Family)**.** *The function $G(x) = e^{-x^2}$ is called the Gaussian function. For any $\alpha > 0$, denote*

$$G_\alpha(x) = \frac{1}{2\sqrt{\pi\alpha}} e^{\frac{-x^2}{4\alpha}} \tag{1.8}$$

Where G_α is the Gaussian family.

Any function in the Gaussian family will be in Schwartz Space as it will be bounded when multiplied by any polynomial.

Lemma 1.2.0.1.

$$\int_{-\infty}^{\infty} G(x)\, dx = \int_{-\infty}^{\infty} e^{-x^2}\, dx = \sqrt{\pi} \tag{1.9}$$

The proof uses Calculus 2 techniques and it is omitted in this project. Furthermore, via change of variables, it can be shown that for any $\alpha > 0$ then $||G_\alpha||_1 = 1$. Furthermore, the statement of the following theorem is from Gu's [9][Chapter 2, Lemma 2, p.4] and the proof requires some properties of Complex Integration which I will mention briefly.

Proposition 1.2.0.1. *For any $\alpha \in \mathbb{R} : \alpha > 0$ we have*

$$\mathcal{F}[G_\alpha](\xi) = \hat{G_\alpha}(\xi) = \frac{e^{-\alpha\xi^2}}{\sqrt{2\pi}} \tag{1.10}$$

Proof. Firstly, as $-(\frac{x}{2\sqrt{\alpha}} + i\xi\sqrt{\alpha})^2 - \alpha\xi^2 = -\frac{x^2}{4\alpha} - i\xi x$ we can re-write $\hat{G_\alpha}(\xi)$ in the following way:

$$\hat{G_\alpha}(\xi) = \frac{1}{2\pi\sqrt{2\alpha}} \int_{-\infty}^{\infty} e^{-\frac{x^2}{4\alpha}} e^{-i\xi x}\, dx = \frac{1}{2\pi\sqrt{2\alpha}} \int_{-\infty}^{\infty} e^{-(\frac{x}{2\sqrt{\alpha}} + i\xi\sqrt{\alpha})^2 - \alpha\xi^2}\, dx \tag{1.11}$$

According to fundamental properties of complex integration, as G_α is smooth in any open region of $\mathbb{C}$, its integral is path independent and it only depends on the two endpoints. Therefore we can make the following change of variables to evaluate the integral of $f(w) = e^{-w^2 - \alpha\xi^2}$ along the real line: $z = \frac{x}{2\sqrt{\alpha}} + i\xi\sqrt{\alpha} \implies \frac{dz}{dx} = \frac{1}{2\sqrt{\alpha}}$. Thus, we get

$$\hat{G_\alpha}(\xi) = \frac{e^{-\alpha\xi^2}}{\sqrt{2}\pi} \int_{-\infty}^{\infty} e^{-z^2}\, dz \tag{1.12}$$

Using Lemma 1.2.0.1:

$$\hat{G_\alpha}(\xi) = \frac{e^{-\alpha\xi^2}}{\sqrt{2\pi}} \tag{1.13}$$

□

The following lemma is straightforward to prove and does not give further insight. It will, however, be used extensively later and it will be clearer to have it written in this way.

Lemma 1.2.0.2. *For any $\alpha \in \mathbb{R} : \alpha > 0$ we have*

$$\mathcal{F}[e^{-\alpha x^2}](\xi) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} e^{-\alpha x^2} e^{-i\xi x}\, dx = \frac{1}{\sqrt{2\alpha}} e^{-\frac{\xi^2}{4\alpha}} \tag{1.14}$$

Proof. We can write from definition 1.2.1, $\frac{\sqrt{\pi}}{\sqrt{\alpha}} G_{\frac{1}{4\alpha}}(x) = e^{-\alpha x^2}$. Hence, using Proposition 1.2.0.1,

$$\mathcal{F}[e^{-\alpha x^2}](\xi) = \frac{\sqrt{\pi}}{\sqrt{\alpha}} \mathcal{F}[G_{\frac{1}{4\alpha}}(x)](\xi) = \frac{1}{\sqrt{2\alpha}} e^{-\frac{\xi^2}{4\alpha}} \tag{1.15}$$

□

1.3 Fourier Inversion Theorem

In this section, I will prove the Fourier Inversion Theorem which is the most important theorem of this project. The existence of an inverse allows us to use the Fourier Transform to compute PDEs both analytically and computationally, prove the sampling theorem and it will give way to making sense of the Frequency Spectrum as you will see in Chapter 2.

1.3.1 Image of Fourier Transforms

It is important to prove that $\mathcal{F}[S(\mathbb{R}, \mathcal{C})] \subseteq S(\mathbb{R}, \mathcal{C})$ as otherwise, the inverse of the Fourier Transform may not exist, as the integral in the theorem could diverge. It will also be helpful to show some important corollaries later on, as we will be able to use Fubini's theorem in a way that is similar to the forward case. Furthermore, a transform is a map from a set to itself. For this reason, the following theorem shows that Fourier Transforms are indeed transformations in Schwartz Space.

Proposition 1.3.0.1. *For any $f \in S(\mathbb{R}, \mathbb{C})$ we have that $\hat{f}(\xi) \in S(\mathbb{R}, \mathbb{C})$.*

Proof is adapted from Conway's [6][Theorem 6.7, 344]

Proof. By applying Leibniz Integral rule we have that

$$\frac{d}{d\xi}\hat{f}(\xi) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty}(-ix)e^{-i\xi x}f(x)\,dx = \mathcal{F}[(-ix)f(x)](\xi) \tag{1.16}$$

and by induction it is not difficult to prove that

$$\frac{d^{(n)}}{d\xi^{(n)}}\hat{f}(\xi) = \mathcal{F}[(-ix)^n f(x)](\xi) \tag{1.17}$$

Let $m, n \in \mathbb{N}$ and let $I(x) = \int_{-\infty}^{\infty}(-it)^n f(t)x^m e^{-itx}\,dt$. Then by integration by parts we have

$$I(x) = \frac{-1}{i}\Big|(-it)^n f(t)x^{m-1}e^{-ixt}\Big|_{t\to-\infty}^{t\to\infty} + \frac{1}{i}\int_{-\infty}^{\infty}\frac{d}{dt}((-it)^n f(t))x^{m-1}e^{-ixt}\,dt \tag{1.18}$$

However as $f(t)$ is decreasing faster than any polynomial, then $I(x) = -i\int_{-\infty}^{\infty}\frac{d}{dt}((-it)^n f(t))x^{m-1}e^{-ixt}\,dt$. Recursively, we can repeat this process and we get

$$I(x) = (-i)^m\int_{-\infty}^{\infty}\frac{d^m}{dt^m}((-it)^n f(t))e^{-ixt}\,dt \tag{1.19}$$

Note that this is possible because $f(t)$, as it is in Schwartz space, it tends to 0 for all its derivatives even when multiplied by a polynomial as $t \to \pm\infty$. Hence we get

$$\begin{aligned}||\hat{f}||_{m,n} &= \sup\{|\xi^m\frac{d^n}{d\xi^n}\hat{f}(\xi)| : \xi\in\mathbb{R}\} = \sup\{\frac{1}{\sqrt{2\pi}}|\int_{-\infty}^{\infty}\xi^m(-it)^n f(t)e^{-i\xi t}\,dt| : \xi\in\mathbb{R}\} \quad \text{Equation (1.17)}\\ &= \sup\{|\frac{1}{\sqrt{2\pi}}I(\xi)| : \xi\in\mathbb{R}\} \le \sup\{\frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty}|\frac{d^m}{dt^m}((t)^n f(t))|\,dt : \xi\in\mathbb{R}\} < \infty \end{aligned} \tag{1.20}$$

As $\frac{d^m}{dt^m}((t)^n f(t)) \in S(\mathbb{R},\mathbb{C}) \subset L^1(\mathbb{R})$ □

Similar to how we got to equation (1.17), we have the following Corollary:

Corollary 1.3.0.1. *Suppose $f \in S(\mathbb{R},\mathbb{C})$ then*

$$\mathcal{F}[\frac{d^n}{dx^n}f(x)](\xi) = (i\xi)^n\hat{f}(\xi) \tag{1.21}$$

This corollary will be fundamental to write PDEs in terms of ODEs.

1.3.2 Fourier Inversion Theorem

The following theorem will prove that $\mathcal{F}$ is a bijection from $S(\mathbb{R},\mathbb{C})$ to itself and it will prove that the Fourier Inversion formula is indeed the inverse of f when $f \in S(\mathbb{R},\mathbb{C})$.

Theorem 1.3.1 (Fourier Inversion Theorem)**.** *Suppose $f \in S(\mathbb{R},\mathbb{C})$. Then*

$$f(x) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty}\hat{f}(\xi)e^{i\xi x}d\xi \tag{1.22}$$

To prove this I am going to use Asadzadeh's[1] [Chapt. 4, Theorem 19, p.120]

Proof. Firstly note that, as $f \in S(\mathbb{R},\mathbb{C})$, then as per the previous proposition, $\hat{f} \in S(\mathbb{R},\mathbb{C})$, and therefore the integral converges. Therefore, it will be possible to use Fubini's Theorem as follows:

$$\frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty}\hat{f}(\xi)e^{i\xi x}e^{-\frac{\xi^2\epsilon^2}{2}}\,d\xi = \frac{1}{2\pi}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}f(y)e^{i\xi x}e^{-i\xi y}e^{-\frac{\xi^2\epsilon^2}{2}}\,dy\,d\xi \tag{1.23}$$

$$= \frac{1}{2\pi}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}f(y)e^{i\xi x}e^{-i\xi y}e^{-\frac{\xi^2\epsilon^2}{2}}\,d\xi\,dy = \frac{1}{2\pi}\int_{-\infty}^{\infty}f(y)\int_{-\infty}^{\infty}e^{-\frac{\xi^2\epsilon^2}{2}}e^{-i\xi(y-x)}\,d\xi\,dy \tag{1.24}$$

$$= \frac{\sqrt{2\pi}}{2\pi}\int_{-\infty}^{\infty}f(y)\mathcal{F}[e^{-\frac{\epsilon^2\xi^2}{2}}](y-x)\,dy \tag{1.25}$$

Now simply applying lemma 1.2.0.2 with $\alpha = \frac{\epsilon^2}{2}$ we get

$$\frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty}f(y)\mathcal{F}[e^{-\frac{\epsilon^2\xi^2}{2}}](y-x)\,dy = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty}f(y)\frac{e^{-\frac{(y-x)^2}{2\epsilon^2}}}{\epsilon}\,dy \tag{1.26}$$

Substituting $z = \frac{y-x}{\sqrt{2}\epsilon} \implies \frac{dz}{dy} = \frac{1}{\sqrt{2}\epsilon}$. Thus

$$\frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty}\hat{f}(\xi)e^{i\xi x}e^{-\frac{\xi^2\epsilon^2}{2}}\,d\xi = \frac{1}{\sqrt{\pi}}\int_{-\infty}^{\infty}f(x+\sqrt{2}\epsilon z)e^{-z^2}\,dz \tag{1.27}$$

Now as $||\hat{f}(\xi)e^{i\xi x}e^{-\frac{\xi^2\epsilon^2}{2}}||_1 \leq ||\hat{f}||_1$ and $||f(x+\sqrt{2}\epsilon)e^{-z^2}||_1 \leq Me^{-z^2}$ as f is bounded. Therefore, as $\hat{f}, Me^{-z^2} \in S(\mathbb{R},\mathbb{C}) \subset L^1(\mathbb{R})$ we can use Lebesgue Dominated Convergence Theorem in the following way: As f is continuous, then $f(x+\gamma) = f(x)$ when $\gamma \to 0$. Therefore

$$\lim_{\epsilon\to 0}\frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty}\hat{f}(\xi)e^{i\xi x}e^{-\frac{\xi^2\epsilon^2}{2}}\,d\xi = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty}\hat{f}(\xi)e^{i\xi x}\,d\xi \tag{1.28}$$

$$= \lim_{\epsilon\to 0}\frac{1}{\sqrt{\pi}}\int_{-\infty}^{\infty}f(x+\sqrt{2}\epsilon z)e^{-z^2}\,dz = \frac{f(x)}{\sqrt{\pi}}\int_{-\infty}^{\infty}e^{-z^2}\,dz = f(x) \qquad \text{Lemma 2.1.2}$$

□

The Fourier Inversion Theorem can be used to prove important corollaries. The following Corollary is an important statement in distribution theory. It demonstrates that the Gaussian Family, as α tends to 0^+, behaves like the Dirac's Delta Functional, an important functional both in Pure Mathematics as well as in Applied Mathematics: for instance in Quantum Mechanics and in Probability Theory. A more in depth explanation goes beyond the scope of this project as it requires material from Functional Analysis and Distribution Theory. Statement of the theorem is from Gu's[9][Chapter 2, p.4. Prop.1].

Corollary 1.3.1.1. *For any $f \in S(\mathbb{R},\mathbb{C})$ then*

$$\lim_{\alpha\to 0^+} f * G_\alpha(x) = f(x) \tag{1.29}$$

Proof. Using the proof of Fourier's Inversion theorem, from equation 1.24, we have

$$f(x) = \lim_{\epsilon\to 0}\frac{1}{2\pi}\int_{-\infty}^{\infty}f(y)\int_{-\infty}^{\infty}e^{-\frac{\xi^2\epsilon^2}{2}}e^{-i\xi(y-x)}\,d\xi\,dy \tag{1.30}$$

And so as $\sqrt{2\pi}\hat{G}_{\frac{\epsilon^2}{2}}(\xi) = e^{-\frac{\epsilon^2\xi^2}{2}}$ via Proposition 1.2.0.1 then

$$f(x) = \frac{1}{\sqrt{2\pi}}\lim_{\epsilon\to 0}\int_{-\infty}^{\infty}f(y)\int_{-\infty}^{\infty}\hat{G}_{\frac{\epsilon^2}{2}}(\xi)e^{-i\xi(y-x)}\,d\xi\,dy \tag{1.31}$$

Now simply using Fourier Inversion's Theorem we have

$$f(x) = \lim_{\epsilon \to 0} \int_{-\infty}^{\infty} f(y) G_{\frac{\epsilon^2}{2}}(x-y)\, dy \tag{1.32}$$

Replacing $\alpha = \frac{\epsilon^2}{2}$ we get

$$f(x) = \lim_{\alpha \to 0^+} f * G_\alpha(x) \tag{1.33}$$

□

The following Corollary may be helpful to compute Fourier Transforms and inverses of Fourier Transforms, and it is symmetric to the Convolution Theorem. Statment and proof is from Ko's[11][1.3, Theorem 3]

Corollary 1.3.1.2. *Suppose $f \in S(\mathbb{R}, \mathbb{C})$:*
If $g \in S(\mathbb{R}, \mathbb{C})$ and f and g are continuous at x then

$$\mathcal{F}[f(x)g(x)] = \frac{1}{\sqrt{2\pi}}(\hat{f} * \hat{g})(\xi) \tag{1.34}$$

Proof. As f, g are continuous at x we can use Fourier Inversion Theorem and we can use Fubini's theorem as $f, g \in S(\mathbb{R}, \mathbb{C})$.

$$\begin{aligned}
\mathcal{F}[f(x)g(x)] &= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} f(x)g(x)e^{-i\xi x}\, dx && (1.35)\\
&= \frac{1}{2\pi} \int_{-\infty}^{\infty} f(x)e^{-i\xi x} \int_{-\infty}^{\infty} \hat{g}(l)e^{ixl}\, dl\, dx && \text{Fourier Inversion Theorem}\\
&= \frac{1}{2\pi} \int_{-\infty}^{\infty} \hat{g}(l) \int_{-\infty}^{\infty} f(x)e^{-ix(\xi - l)}\, dx\, dl && \text{Fubini's Theorem}\\
= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \hat{g}(l)\hat{f}(\xi - l)\, dl &= \frac{1}{\sqrt{2\pi}} \hat{f} * \hat{g}(\xi) && (1.36)
\end{aligned}$$

□

1.4 Heat Equation

In order to show an example of an application of the Fourier Transform in analytic solutions of PDEs, I will demonstrate the solution of the inhomogeneous heat equation. Many of the techniques used in the proofs can be used in other PDEs, such as in the wave equation and in the Laplace equation.

Theorem 1.4.1. *Suppose $u : \mathbb{R} \times (0, \infty) \to \mathbb{R}$ decays fast with respect to x (i.e it is in Schwartz Space), and suppose $u_0 : \mathbb{R} \to \mathbb{R}$ and $f : \mathbb{R}^2 \to \mathbb{R}$ are in Schwartz Space with respect to x, then*

$$\begin{cases} u_t - k u_{xx} = f(x,t) \\ u(x,t) = u_0(x) \text{ as } t \to 0^+ \end{cases}$$

$$\implies u(x,t) = \int_0^t \frac{1}{\sqrt{4\pi k(t-s)}} \int_{-\infty}^{\infty} f(x-y,s) e^{-\frac{y^2}{4k(t-s)}}\, dy\, ds + \frac{1}{\sqrt{4kt}} \int_{-\infty}^{\infty} u_0(x-y) e^{-\frac{y^2}{4kt}}\, dy \tag{1.37}$$

Proof. We will first write the PDE in terms of an ODE, by first applying the Fourier Transforms on both sides with respect to x and using Corollary 1.3.0.1. This method for writing the PDE as an ODE was presented by Ko [11][Problem 2.2, p. 11] for the homogeneous case of the heat equation.

$$\hat{u}_t + k\xi^2\hat{u}(\xi,t) = \hat{F}(\xi,t) \implies \frac{\partial}{\partial t}(\hat{u}(\xi,t)e^{\xi^2 kt}) = \hat{F}(\xi,t)e^{\xi^2 kt} \tag{1.38}$$

Applying the fundamental theorem of calculus, we have

$$\hat{u}(\xi,t)e^{kt\xi^2} = \int_0^t \hat{F}(\xi,s)e^{ks\xi^2}\,ds + A(\xi) \tag{1.39}$$

For some function $A(\xi)$. Applying initial conditions when $t \to 0^+$, we have

$$\lim_{t\to 0^+} \hat{u}(\xi,t)e^{\xi^2 kt} = \lim_{t\to 0^+} \hat{u}(\xi,t) = \hat{u}_0(\xi) = A(\xi) \tag{1.40}$$

Therefore, we are left with

$$\hat{u}(\xi,t) = \int_0^t \hat{F}(\xi,s)e^{k\xi^2(s-t)}\,ds + \hat{u}_0(\xi)e^{-kt\xi^2} \tag{1.41}$$

Taking the Inverse Fourier Transform on both sides, Fubini's theorem will allow us to exchange the limits of integration, and we are left with the following

$$u(x,t) = \int_0^t \mathcal{F}^{-1}[\hat{F}(\xi,s)e^{k\xi^2(s-t)}]\,ds + \mathcal{F}^{-1}[\hat{u}_0(\xi)e^{-kt\xi^2}] \tag{1.42}$$

We can now apply the convolution theorem ($\mathcal{F}^{-1}[\hat{f}\hat{g}] = \frac{1}{\sqrt{2\pi}} f * g$) together with $\mathcal{F}^{-1}[e^{-\frac{\xi^2}{4\alpha}}](x) = \sqrt{2\alpha}e^{-\alpha x^2}$ from Lemma 1.2.0.2 with α to be $\frac{1}{4k(t-s)}$ and $\frac{1}{4kt}$ respectively for each term. Therefore

$$u(x,t) = \int_0^t \frac{1}{\sqrt{4\pi k(t-s)}} \int_{-\infty}^{\infty} f(x-y,s)e^{-\frac{y^2}{4k(t-s)}}\,dy\,ds + \frac{1}{\sqrt{4kt}} \int_{-\infty}^{\infty} u_0(x-y)e^{-\frac{y^2}{4kt}}\,dy \tag{1.43}$$

□

We could later apply Leibniz Integral rule to differentiate the given integral and show that it satisfies $u_t - ku_{xx} = f(x,t)$. To show that as $t \to 0^+$, initial conditions are satisfied we would apply the limit and use Corollary 1.3.1.1 where α is chosen to be kt. Furthermore, the solution of the Heat equation exists for any u_0 and f in Schwartz Space with respect to x, as the multiplication of two functions in Schwartz Space is also in Schwartz Space.

1.5 Parseval and Plancherel Theorems

The following two theorems are important in mathematics. For instance, we will need them to show that after denoising and computational approximations, little information is lost. On the other hand, in Functional Analysis we can approximate functions from spaces such as $L^2(\mathbb{R})$ to Schwartz Space, and subsequently use Plancherel's Theorem to extend Fourier Transforms to these spaces. In order to maintain the focus of the project on signal analysis and applications to PDE solutions, I will keep using Schwartz Space.
To prove Parseval's Theorem, first we need the following lemma:

Lemma 1.5.0.1. *For $f, g \in S(\mathbb{R}, \mathbb{C})$ we have*

$$\overline{\int_{-\infty}^{\infty} g(x)f(x)\,dx} = \int_{-\infty}^{\infty} \overline{f(x)}\,\overline{g(x)}\,dx = \tag{1.44}$$

Proof follows from definition of complex integration and is omitted as it only involves applying distributive law with the real and complex parts of the integrals.

The following proofs and theorems are from Rudin's[17]'s Functional Analysis [7.9, p.188].

Theorem 1.5.1 (Parseval Theorem)**.** *Suppose $f, g \in S(\mathbb{R}, \mathbb{C})$. Then*

$$\int_{-\infty}^{\infty} f(x)\overline{g(x)}\,dx = \int_{-\infty}^{\infty} \hat{f}(\xi)\overline{\hat{g}(\xi)}d\xi \tag{1.45}$$

Proof. Using Fourier Inversion Theorem,

$$\int_{-\infty}^{\infty} f(x)\overline{g(x)}\,dx = \int_{-\infty}^{\infty} \overline{g(x)}\frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty} \hat{f}(\xi)e^{i\xi x}d\xi\,dx \tag{1.46}$$

As functions are in Schwartz Space then we can use Fubini's theorem as follows

$$= \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty} \hat{f}(\xi)\int_{-\infty}^{\infty} \overline{g(x)}e^{i\xi x}d\xi\,dx = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty} \hat{f}(\xi)\overline{\int_{-\infty}^{\infty} g(x)e^{-ix\xi}\,dx}d\xi \qquad \text{Lemma 3.0.1}$$

$$= \int_{-\infty}^{\infty} \hat{f}(\xi)\overline{\hat{g}(\xi)}d\xi \tag{1.47}$$

□

Theorem 1.5.2 (Plancherel Theorem)**.** *Let $f \in S(\mathbb{R}, \mathbb{C})$ then*

$$||f||_2 = ||\hat{f}||_2 \tag{1.48}$$

Proof. Proof is straightforward from Parseval's identity as

$$||f||_2^2 = \int_{-\infty}^{\infty} |f(x)|^2 dx = \int_{-\infty}^{\infty} f(x)\overline{f(x)}\,dx = \int_{-\infty}^{\infty} \hat{f}(x)\overline{\hat{f}(x)}\,dx = ||\hat{f}||_2^2 \tag{1.49}$$

□

Chapter 2: Signal Processing, DFT and FFT

2.1 Signal Analysis

We now turn into Fourier Transforms representing signals in the physical world. As explained by Orfanadis [16] [1.2.2, p.2] ω will represent the angular frequency (or radian frequency), which is the number of occurrences of a repeating event per unit of time (frequency) times a factor of 2π. Therefore, we have that

$$\omega = \frac{2\pi}{T} = 2\pi \boldsymbol{f} \tag{2.1}$$

Where T is the period, $\boldsymbol{f}$ is the frequency, $f(t)$ is the amplitude of a periodic signal and $\hat{f}$ is defined to be the frequency spectrum.
Therefore, similar to Orfanadis Introduction to Signal Analysis [16][1.2.3, p.2], where $f(t)$ is a continuous signal, from the Fourier Inversion theorem we will have:

$$f(t) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \hat{f}(\omega) e^{i\omega t}\, d\omega \tag{2.2}$$

Hence the amplitude of a signal will be the sum of the impulses of a signal superposed with the wave $e^{i\omega t}$. The following definition is from Ulaby and Yagle [19]:

Definition 2.1.1. *The energy of a signal is defined to be*

$$E[f] = ||f||_2^2 \tag{2.3}$$

Via the Plancherel Theorem, we have that $E[f] = E[\hat{f}]$. This gives way to the implementation of algorithms in the frequency spectrum that will leave the energy practically unchanged, thus preventing the loss of information.

Furthermore, the following theorem will be important for compression and sampling algorithms. The statement and proof is from Lerman's [12][1.2, p.2]. It states that, when Ω is large enough such that $\hat{f}$ is supported, then $f(t)$ can be uniquely determined by samples $\frac{\pi}{\Omega}$ distance apart. In such cases it also details how the continuous signal can be reconstructed.

Theorem 2.1.1 (Nyquist Shannon sampling theorem)**.** *If $f \in S(\mathbb{R}, \mathbb{C})$ and $\hat{f}$ is supported on the interval $[-\Omega, \Omega] = [-2\pi B, 2\pi B]$ then*

$$f(t) = \sum_{n=-\infty}^{\infty} f(\frac{\pi n}{\Omega}) \frac{sin(\Omega t - n\pi)}{\Omega t - n\pi} \tag{2.4}$$

Note: The minimal value of B such that $\hat{f}$ is supported on $[-2\pi B, 2\pi B]$ is called the Nyquist frequency and it is written as B_{Nyq}. The Nyquist rate is defined to be $2B_{\text{Nyq}}$.

Proof. We start by expanding $\hat{f}$ into its Fourier series. We can do this because $\hat{f}$ will be continuous and bounded as it is in Schwartz Space as per Theorem 1.3.1. We can thus extend $\hat{f}$ outside the interval $[-2\pi B, 2\pi B]$ in order to make it periodic and write it in the following way:

$$\hat{f}(\omega) = \sum_{n=-\infty}^{\infty} c_n e^{\frac{in\omega}{2B}} \tag{2.5}$$

where, using the formula for computing the Fourier coefficients and the Inverse Fourier Transform, we have

$$c_n = \frac{1}{4\pi B}\int_{-\infty}^{\infty} \hat{f}(\omega)e^{-\frac{i\omega n}{2B}}\,d\omega = \frac{\sqrt{2\pi}}{4\pi B} f(\frac{-n}{2B}) \tag{2.6}$$

And therefore

$$\hat{f}(\omega) = \frac{\sqrt{2\pi}}{4\pi B}\sum_{n=-\infty}^{\infty} f(\frac{-n}{2B})e^{\frac{in\omega}{2B}} = \frac{\sqrt{2\pi}}{4\pi B}\sum_{n=-\infty}^{\infty} f(\frac{n}{2B})e^{\frac{-in\omega}{2B}} \tag{2.7}$$

Therefore using the support of $\hat{f}$ and the Inverse Fourier Transform we have

$$f(t) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty} \hat{f}(\omega)e^{i\omega t}\,d\omega = \frac{1}{4\pi B}\sum_{n=-\infty}^{\infty} f(\frac{n}{2B})\int_{-2\pi B}^{2\pi B} e^{i\omega(t-\frac{n}{2B})}\,d\omega \tag{2.8}$$

Simply computing the integral and re-arranging terms, when $\Omega = 2\pi B$

$$f(t) = \frac{1}{4\pi B}\sum_{n=-\infty}^{\infty} f(\frac{n}{2B})\Big|\frac{e^{i\omega(t-\frac{n}{2B})}}{i(t-\frac{n}{2B})}\Big|_{\omega=-2\pi B}^{\omega=2\pi B} = \frac{1}{2\pi B}\sum_{n=-\infty}^{\infty} f(\frac{n}{2B})\frac{1}{t-\frac{n}{2B}}\frac{e^{2i\pi B(t-\frac{n}{2B})} - e^{-2i\pi B(t-\frac{n}{2B})}}{2i} \tag{2.9}$$

$$f(t) = \sum_{n=-\infty}^{\infty} f(\frac{n}{2B})\frac{sin(2\pi B(t-\frac{n}{2B})}{2\pi B(t-\frac{n}{2B})} = \sum_{n=-\infty}^{\infty} f(\frac{\pi n}{\Omega})\frac{sin(\Omega t - n\pi)}{\Omega t - n\pi} \tag{2.10}$$

□

Computers store signals discretely, which makes the Sampling Theorem so important as it ensures signal data can be stored in such a way that it uniquely identifies the continuous physical signal.

2.1.1 Brief mention of Anti-Aliasing and Uncertainty Principle

As explained by Lerman[12][Prop. 4.1], if B is chosen below the maximum value where $\hat{f}$ is supported, we say f is undersampled, and the sampling formula will generate a function $g(t)$ that approximates with distortion. Such g is called an alias of f. To avoid this it is usually needed to oversample and filter frequencies that are above certain thresholds.
Having said this, in practice, the series in (2.4) will be from 0 to a real value N as the signal will be measured on a finite interval, starting from $t = 0$ and outside the interval the signal will be assumed to be 0. While researching quantum mechanics Werner Heisenberg proved in his uncertainty principles states that such functions can not be band-limited, nor can they be arbitrarily approximated. Therefore, in practice, reconstructing the continuous signal will only yield approximations of such signal. The impact of uncertainty principles in signal analysis is an area of current research that combines areas such as quantum mechanics and stochastic processes. This is explained in research such as Feng, Li and Rassias's [7][1. Introduction] where they investigate the Linear Canonical Transform that can lower uncertainty bounds and where methods such as sampling and filtering can be extended.

2.2 Discrete Fourier Transform

In this section, the following notation will be used:

Definition 2.2.1. *We define $\boldsymbol{C}_N$, for any $N \in \mathbb{N}$, as $\boldsymbol{C}_N := \{a = (a_0, a_1, ..., a_{N-1}) : a_i \in \mathbb{C},\ i \in [0, N-1] \cap \mathbb{Z}\}$. For any $x = (x_0, x_1, ..., x_{N-1}) \in \boldsymbol{C}_N$ and $y = (y_0, y_1, ..., y_{N-1}) \in \boldsymbol{C}_N$ we have*

- $x[: j] := (x_0, x_1, ..., x_{j-1})$ *and* $x[j :] := (x_j, x_{j+1}, ..., x_{N-1})$.
- *The Hadamard product is an operation in $\boldsymbol{C}_N$ which will be denoted by*

$$x \odot y := (x_0 y_0, x_1 y_1, ..., x_{N-1} y_{N-1}) \tag{2.11}$$

- $\oplus : \boldsymbol{C}_N \times \boldsymbol{C}_M \to \boldsymbol{C}_{N+M}$ *will be the concatenation of two vectors. If $w \in \boldsymbol{C}_M$ then*

$$x \oplus y = (x_0, x_1, .., x_{N-1}, w_0, w_1, ..., w_{M-1}) \tag{2.12}$$

- *The following definition is from Calder [5][6.3, p.95]. If N is even then*

$$x_e := (x_0, x_2, ..., x_{N-2}) \in \boldsymbol{C}_{\frac{N}{2}} \tag{2.13}$$

$$x_o := (x_1, x_3, ..., x_{N-1}) \in \boldsymbol{C}_{\frac{N}{2}} \tag{2.14}$$

Furthermore, I will use the following inner product operation and the following norm

Definition 2.2.2. *For any $x \in \boldsymbol{C}_N : x = (x_0, x_1, .., x_{N-1})$ and $y \in \boldsymbol{C}_N : y = (y_0, y_1, .., y_{N-1})$, the inner product operation is*

$$\langle x, y \rangle := \sum_{n=0}^{N-1} x_i \overline{y}_i \tag{2.15}$$

and the norm of such vector x will be defined to be:

$$||x|| := \sqrt{\langle x, x \rangle} \tag{2.16}$$

2.2.1 Definition and Inverse Fourier Transform

The Discrete Fourier Transform's definition is from Asadzadeh [1][4.12, def. 20, p.149]. It is an approximation of the continuous Fourier Transform using the sampling points $\frac{n\Omega}{N}$ of a signal in $[0, N]$ by the Riemann Sum over this period.

Definition 2.2.3 (Discrete Fourier Transform)**.** *The N-point Discrete Fourier Transform $\mathcal{F}_N : \boldsymbol{C}_N \to \boldsymbol{C}_N$ is defined by*

$$\mathcal{F}_N[\underline{\mathbf{a}}] = \underline{\hat{\mathbf{a}}} : \ \hat{a}_m = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} a_n e^{\frac{-2\pi i n m}{N}} \tag{2.17}$$

where $\underline{\hat{\mathbf{a}}} = (\hat{a}_0, \hat{a}_1, \hat{a}_2, ..., \hat{a}_{N-1})$

The following lemma is also from Asadzadeh [1][4.12, lemma 11, p.149]

Lemma 2.2.0.1. *For $m = 0, 1, 2, ..., N-1$ define*

$$\underline{\mathbf{e}}_m = (1, e^{\frac{2\pi i m}{N}}, e^{\frac{4\pi i m}{N}} ... e^{\frac{2\pi i (N-1) m}{N}}) \tag{2.18}$$

Then $\{\underline{\mathbf{e}}_m\}_{m=0}^{N-1}$ is an orthogonal basis for $\boldsymbol{C}_N$ and $\forall m\ ||\underline{\mathbf{e}}_m||^2 = N$

Proof. For $l \neq m$ we have

$$\langle \underline{\mathbf{e}}_l, \underline{\mathbf{e}}_m \rangle = \sum_{n=0}^{N-1} e^{\frac{2\pi i l n}{N}} \overline{e^{\frac{2\pi i m n}{N}}} = \sum_{n=0}^{N-1} e^{\frac{2\pi i n (l-m)}{N}} = \frac{e^{2\pi i (l-m)} - 1}{e^{\frac{2\pi i (l-m)}{N}} - 1} = 0 \tag{2.19}$$

And for $l = m$ then

$$\langle \underline{\mathbf{e}}_m, \underline{\mathbf{e}}_m \rangle = ||\underline{\mathbf{e}}_m||^2 = \sum_{n=0}^{N-1} e^0 = N \tag{2.20}$$

$\{\underline{\mathbf{e}}_m\}_{m=0}^{N-1}$ is a basis as all vectors are linearly independent (vectors are orthogonal with respect to each other) and there are $N = dim(\mathbf{C}_N)$ vectors. □

Theorem 2.2.1 (Discrete Fourier Transform Inverse Formula)**.** *Let* $\mathcal{F}_N[\underline{\mathbf{a}}] = \mathcal{F}_N[(a_0, a_1, ..., a_{N-1})] = (\hat{a}_0, \hat{a}_1, ..., \hat{a}_{N-1}) = \underline{\hat{\mathbf{a}}}$*. The inverse formula for the Discrete Fourier Transform is as follows*

$$\underline{\mathbf{a}} = \frac{1}{\sqrt{N}} \sum_{m=0}^{N-1} \hat{a}_m \underline{\mathbf{e}}_m \iff a_n = \frac{1}{\sqrt{N}} \sum_{m=0}^{N-1} \hat{a}_m e^{\frac{2\pi i m n}{N}} \tag{2.21}$$

Where we will express $\mathcal{F}_N^{-1}[\underline{\hat{\mathbf{a}}}]$ *to be the sum* $\frac{1}{\sqrt{N}} \sum_{m=0}^{N-1} \hat{a}_m \underline{\mathbf{e}}_m$

Proof is similar to Asadzadeh's[1][4, Theorem 30, p.150]

Proof. From the previous lemma, we can write each term a_i in the following way

$$a_j = \frac{1}{N} \sum_{m=0}^{N-1} \sum_{l=0}^{N-1} a_l e^{\frac{2\pi i m l}{N}} e^{\frac{-2\pi i m j}{N}} \tag{2.22}$$

As each term will be 0 whenever $j \neq m$ and for $j = m$ we will have that $e^{\frac{2\pi i m l}{N}} e^{\frac{-2\pi i m j}{N}} = 1$ (the sum of a_j N times will give Na_j which will then be divided with the coefficient.). Simply taking the definition of the Fourier transform we have

$$a_j = \frac{1}{N} \sum_{m=0}^{N-1} e^{\frac{2\pi i m j}{N}} \sum_{l=0}^{N-1} a_l e^{\frac{-2\pi i m l}{N}} = \frac{\sqrt{N}}{N} \sum_{m=0}^{N-1} e^{\frac{2\pi i m j}{N}} \hat{a}_m = \frac{1}{\sqrt{N}} \sum_{m=0}^{N-1} e^{\frac{2\pi i m j}{N}} \hat{a}_m \tag{2.23}$$

□

The following theorem gives way to the analysis of the energy conservation of a signal in the discrete sense, which will be important for the explanation of the denoising algorithm which follows.

Corollary 2.2.1.1 (Plancherel Formula for Discrete Fourier Transform)**.**

$$||\underline{\mathbf{a}}||^2 = ||\underline{\hat{\mathbf{a}}}||^2 \tag{2.24}$$

Proof. From Fourier Inversion Formula and using Inner Product properties

$$||\underline{\mathbf{a}}||^2 = \langle \frac{1}{\sqrt{N}} \sum_{m=0}^{N-1} \hat{a}_m \underline{\mathbf{e}}_m, \frac{1}{\sqrt{N}} \sum_{m=0}^{N-1} \hat{a}_m \underline{\mathbf{e}}_m \rangle = \frac{1}{N} \sum_{m=0}^{N-1} \langle \hat{a}_m \underline{\mathbf{e}}_m, \overline{\hat{a}_m} \underline{\mathbf{e}}_m \rangle = \frac{1}{N} \sum_{m=0}^{N-1} \hat{a}_m \overline{\hat{a}_m} \langle \underline{\mathbf{e}}_m, \underline{\mathbf{e}}_m \rangle \tag{2.25}$$

However from the previous lemma we have that $\langle \underline{\mathbf{e}}_m, \underline{\mathbf{e}}_m \rangle = N$ and so

$$||\underline{\mathbf{a}}||^2 = \sum_{m=0}^{N-1} \hat{a}_m \overline{\hat{a}_m} = \sum_{m=0}^{N-1} |\hat{a}_m|^2 = ||\underline{\hat{\mathbf{a}}}||^2 \tag{2.26}$$

□

2.3 Fast Fourier Transform

2.3.1 Fast Fourier Transform Algorithm:

If one were to make an algorithm from the definition of the DFT, such algorithm would be in $O(N^2)$. The algorithm depends on the dimension of the vectors being used, as each value a_i and $\hat{a}_i$ will be bounded as the original continuous function is in Schwartz Space. Recall that the sequences $\underline{\mathbf{a}}$ and $\underline{\hat{\mathbf{a}}}$ approximate the continuous $f(t)$ and $\hat{f}(\xi)$ respectively. Furthermore, the algorithm will consist on summing over N bounded values N times, which is why the DFT algorithm would be in $O(N^2)$. Considering how important the DFT is in the computing and engineering world, a faster algorithm has significant importance.

As explained by Calder [5][6.3, p.95], the Fast Fourier Transform will be based on subsampling a signal into its even and odd samples and applying the Discrete Fourier Transform to the two components separately. The process is repeated until the whole signal is sampled. The following lemma and proof is adapted from Calder [5][6.3, Lemma 6.3.1, p.96], and it uses this idea:

Lemma 2.3.0.1. *Suppose $N \in \mathbb{N}$ such that N is even, and suppose $l \in \mathbb{Z}$, $l \in [0, N-1]$. Let $\gamma = l \pmod{\frac{N}{2}}$ then*

$$\mathcal{F}_N[\underline{\mathbf{a}}]_l = \frac{1}{\sqrt{2}}\mathcal{F}_{\frac{N}{2}}[\underline{\mathbf{a}}_e]_\gamma + \frac{e^{\frac{-2\pi i l}{N}}}{\sqrt{2}}\mathcal{F}_{\frac{N}{2}}[\underline{\mathbf{a}}_o]_\gamma \tag{2.27}$$

Where $\mathcal{F}_{\frac{N}{2}}[\underline{\mathbf{a}}_e]_\gamma$ and $\mathcal{F}_{\frac{N}{2}}[\underline{\mathbf{a}}_o]_\gamma$ will be the γ-th components of $\mathcal{F}_{\frac{N}{2}}[\underline{\mathbf{a}}_e]$ and $\mathcal{F}_{\frac{N}{2}}[\underline{\mathbf{a}}_o]$ respectively, and where $\mathcal{F}_N[\underline{\mathbf{a}}]_l$ will be the l-th element of $\mathcal{F}_N[\underline{\mathbf{a}}]$.

Proof. Proof follows from simple computations. From the definition we have

$$\mathcal{F}_N[\underline{\mathbf{a}}]_l = \frac{1}{\sqrt{N}}\langle \underline{\mathbf{a}}, \underline{\mathbf{e}}_l \rangle = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} a_n e^{\frac{-2\pi i n l}{N}} \tag{2.28}$$

We can break the sum into two parts, where n is even and where n is odd

$$\mathcal{F}_N[\underline{\mathbf{a}}]_l = \frac{1}{\sqrt{N}}\Big(\sum_{n=0}^{\frac{N}{2}-1} a_{2n} e^{\frac{-4\pi i n l}{N}} + e^{\frac{-2\pi i l}{N}} \sum_{n=0}^{\frac{N}{2}-1} a_{2n+1} e^{\frac{-4\pi i n l}{N}}\Big) = \frac{1}{\sqrt{N}}\Big(\sum_{n=0}^{\frac{N}{2}-1} a_{2n} e^{\frac{-2\pi i n l}{\frac{N}{2}}} + e^{\frac{-2\pi i l}{N}} \sum_{n=0}^{\frac{N}{2}-1} a_{2n+1} e^{\frac{-2\pi i n l}{\frac{N}{2}}}\Big) \tag{2.29}$$

By the cyclic property of the $\frac{N}{2}$ roots of unity, we have

$$\mathcal{F}_N[\underline{\mathbf{a}}]_l = \frac{1}{\sqrt{N}}\Big(\sum_{n=0}^{\frac{N}{2}-1} a_{2n} e^{\frac{-2\pi i n \gamma}{\frac{N}{2}}} + e^{\frac{-2\pi i l}{N}} \sum_{n=0}^{\frac{N}{2}-1} a_{2n+1} e^{\frac{-2\pi i n \gamma}{\frac{N}{2}}}\Big) \tag{2.30}$$

Therefore, we can use the definition of the Discrete Fourier Transform to write

$$\mathcal{F}_N[\underline{\mathbf{a}}]_l = \frac{1}{\sqrt{2}}\mathcal{F}_{\frac{N}{2}}[\underline{\mathbf{a}}_e]_\gamma + \frac{e^{\frac{-2\pi i l}{N}}}{\sqrt{2}}\mathcal{F}_{\frac{N}{2}}[\underline{\mathbf{a}}_o]_\gamma \tag{2.31}$$

□

With a similar proof, by splitting the sum, we have a similar lemma for the Inverse Discrete Fourier Transform:

Lemma 2.3.0.2. *For any $N \in \mathbb{N}$ such that N is even, and any $l \in \mathbb{Z}$, $l \in [0, N-1]$, let $\gamma = l \pmod{\frac{N}{2}}$, then*

$$\mathcal{F}_N^{-1}[\underline{\hat{\mathbf{a}}}]_l = \frac{1}{\sqrt{2}}\mathcal{F}_{\frac{N}{2}}^{-1}[\underline{\hat{\mathbf{a}}_e}]_\gamma + \frac{e^{\frac{2\pi i l}{N}}}{\sqrt{2}}\mathcal{F}_{\frac{N}{2}}^{-1}[\underline{\hat{\mathbf{a}}_o}]_\gamma \tag{2.32}$$

Where $\mathcal{F}_N^{-1}[\underline{\hat{\mathbf{a}}}]_l$ is the l component of the vector $\mathcal{F}_N^{-1}[\underline{\hat{\mathbf{a}}}] \in \boldsymbol{C}^N$. For the two terms in the right hand side, these are the γ-th components of the vectors $\mathcal{F}_{\frac{N}{2}}^{-1}[\underline{\hat{\mathbf{a}}_e}]$ and $\mathcal{F}_{\frac{N}{2}}^{-1}[\underline{\hat{\mathbf{a}}_o}]$ in $\boldsymbol{C}^{\frac{N}{2}}$.

Using these two lemmas, by simply writing each element of $\underline{\mathbf{a}}$ and $\underline{\hat{\mathbf{a}}}$ in terms of its even and odd parts, we get the following theorem that will be essential for computing the Fast Fourier Transform, and writing it this way will make each step in the algorithm clearer.

Theorem 2.3.1. *For any $N \in \mathbb{N}$ such that N is even and for some $\underline{\mathbf{a}} \in \boldsymbol{C}_N$, for which DFT is given by $\mathcal{F}_N[\underline{\mathbf{a}}] = \underline{\hat{\mathbf{a}}}$, we have*

$$\underline{\hat{\mathbf{a}}} = \frac{1}{\sqrt{2}}((\underline{\hat{\mathbf{a}}}_e + \underline{\overline{\mathbf{e}}}_1[:\frac{N}{2}] \odot \underline{\hat{\mathbf{a}}}_o) \oplus (\underline{\hat{\mathbf{a}}}_e + \underline{\overline{\mathbf{e}}}_1[\frac{N}{2}:] \odot \underline{\hat{\mathbf{a}}}_o)) \tag{2.33}$$

where $\mathcal{F}_N[\underline{\mathbf{a}}_e] = \underline{\hat{\mathbf{a}}}_e$ and $\mathcal{F}_N[\underline{\mathbf{a}}_e] = \underline{\hat{\mathbf{a}}}_e$

Proof. Suppose $N \in \mathbb{N}$ is even and $\underline{\mathbf{a}} \in \mathbf{C}_N$. Simply rewriting in terms of its elements, we have:

$$\underline{\hat{\mathbf{a}}} = (\hat{a}_0, \hat{a}_1, ..., \hat{a}_{N-1}) \tag{2.34}$$

and using Lemma 2.3.0.1, then

$$\underline{\hat{\mathbf{a}}} = \frac{1}{\sqrt{2}}(\hat{a}_{e_0} + \hat{a}_{o_0}, \hat{a}_{e_1} + e^{\frac{-2\pi i}{N}}\hat{a}_{o_1}, ..., \hat{a}_{e_{\frac{N}{2}-1}} + e^{\frac{-2\pi i(\frac{N}{2}-1)}{N}}\hat{a}_{o_{\frac{N}{2}-1}}, \tag{2.35}$$

$$\hat{a}_{e_0} + e^{\frac{-2\pi i(\frac{N}{2})}{N}}\hat{a}_{o_0}, ..., \hat{a}_{e_{\frac{N}{2}-1}} + e^{\frac{-2\pi i(N-1)}{N}}\hat{a}_{o_{\frac{N}{2}-1}}) \tag{2.36}$$

where $\hat{a}_{e_m} = \mathcal{F}_{\frac{N}{2}}[\underline{\mathbf{a}}_e]_m$, and $\hat{a}_{o_m}$ will be similar but for the DFT of the odd part.

$$= \frac{1}{\sqrt{2}}((\hat{a}_{e_0} + \hat{a}_{o_0}, ..., \hat{a}_{e_{\frac{N}{2}-1}} + e^{\frac{-2\pi i(\frac{N}{2}-1)}{N}}\hat{a}_{o_{\frac{N}{2}-1}}) \oplus (\hat{a}_{e_0} + e^{-\pi i}\hat{a}_{o_0}, ..., \hat{a}_{e_{\frac{N}{2}-1}} + e^{\frac{-2\pi i(N-1)}{N}}\hat{a}_{o_{\frac{N}{2}-1}})) \tag{2.37}$$

And so, from the definition of each of the vectors, together with the definition of the operators, the theorem is proved. □

It is easy to see that an equivalent formula for the inverse discrete Fourier transform can be proved by using the conjugate of $\underline{\overline{\mathbf{e}}}_1$. Using this we can derive the Fast Fourier Transform algorithm for any $N = 2^k$, where $k \in \mathbb{N}$. The Fast Fourier Transform will be called with a sequence $\underline{\mathbf{a}}$ as parameter, and it will be defined to be as follows:

- Compute $\underline{\overline{\mathbf{e}}}_1 = (1, e^{\frac{-2\pi i}{N}}, e^{\frac{-4\pi i}{N}}, e^{\frac{-6\pi i}{N}}, ..., e^{\frac{-2\pi(N-1)i}{N}}) \in \mathbf{C}_N$ for later reference.
- Define FFT to receive N and $\underline{\mathbf{a}}$ as input. FFT will then make use of the previous theorem to recursively compute the Fourier Transform as follows:

  - Base Case: If $N = 1$ then return $\underline{\mathbf{a}}$.
  - Recursive Call 1: $A = FFT(\underline{\mathbf{a}}_e, \frac{N}{2})$
  - Recursive Call 2: $B = FFT(\underline{\mathbf{a}}_o, \frac{N}{2})$
  - Replace the first $\frac{N}{2} - 1$ elements of $\underline{\mathbf{a}}$ to be $\frac{A}{\sqrt{2}} + \frac{\overline{\underline{\mathbf{e}}}_1[:\frac{N}{2}]}{\sqrt{2}} \odot B$ and the elements with indices greater than or equal to $\frac{N}{2}$ to be $\frac{A}{\sqrt{2}} + \frac{\overline{\underline{\mathbf{e}}}_1[\frac{N}{2}:]}{\sqrt{2}} \odot B$
  - Return $\underline{\mathbf{a}}$.

- Return $FFT(N, \underline{\mathbf{a}})$ with N being the length of the initial sequence $\underline{\mathbf{a}}$.

Note two things:

- The base case makes use of the fact that, directly from the definition of the Discrete Fourier Transform, when $N = 1$, then $\hat{\underline{\mathbf{a}}} = (\hat{a}_0) = (\frac{1}{\sqrt{N}} \sum_{m=0}^{N-1} a_m e^0) = (a_0)$
- the Inverse Fast Fourier Transform is computed exactly the same way by replacing $\overline{\underline{\mathbf{e}}}_1$ with $\underline{\mathbf{e}}_1$.

Theorem 2.3.2. *The Fast Fourier Transform and Inverse Fast Fourier Transform for $N = 2^k$ will be $O(Nlog(N))$*

Proof uses techniques from Calder [5][Lemma 6.3.3]. I will prove the theorem for the Fast Fourier Transform as the Inverse Fast Fourier Transform proof will be very similar.

Proof. $N = 2^k \implies log_2(N) = k$ for some $k \in \mathbb{N}$. Computing $\overline{\underline{\mathbf{e}}}_1$ is done once, and it is a $O(N)$ algorithm. Furthermore, for the recursive function FFT, let its complexity be A_N, and so we have that for some constant c, $A_N = 2A_{\frac{N}{2}} + cN$. This is because we need to call the function FFT twice, and afterwards compute each of the new terms of $\underline{\mathbf{a}}$ which involves iterating over the length of the parameter $\underline{\mathbf{a}}$ (of size at most N). The constant term c is due to the internal operations inside the iteration which don't depend on input size N. We iterate this process k times, as we need $\frac{N}{2^k} = 1$. For this reason we have, by applying the formula k times, we get

$$A_N = 2A_{\frac{N}{2}} + cN = A_{2^k} = 2A_{2^{k-1}} + c2^k \tag{2.38}$$

$$A_{2^k} = 2(2A_{2^{k-2}} + c2^{k-1}) + c2^k = 4A_{2^{k-2}} + (2c)2^k \tag{2.39}$$

$$= 8A_{2^{k-3}} + (3c)2^k, ..., \tag{2.40}$$

$$A_{2^k} = 2^k A_1 + (kc)2^k \tag{2.41}$$

As the base case has constant time complexity then $A_1 = 1$. Hence

$$A_N = 2^k + ck(2^k) = N + cN \log_2(N) \tag{2.42}$$

An algorithm in $O(cNlog_2(N) + 2N)$ for some constant c is in $O(Nlog_2(N))$ as a straightforward consequence of definition 0.2.4 by choosing appropriate bounds. Therefore, the Fast Fourier Transform which we showed to be in $O(A_N + N)$ (recall the first N from computing $\overline{\underline{\mathbf{e}}}_1$ at the beginning) will be in $O(Nlog_2(N))$ □

Using this algorithm, we can also simplify further the multiplication over $\frac{1}{\sqrt{2}}$, as this multiplication is repeated k times, and therefore multiplying the whole vector $\frac{1}{\sqrt{2^k}}$ is equivalent to multiplying it once by $\frac{1}{\sqrt{2^k}} = \frac{1}{\sqrt{2^{log_2(N)}}} = \frac{1}{\sqrt{N}}$.
The following piece of python code applies this algorithm in Python, and taking the second parameter equal to True would compute the Inverse Fourier Transform of the sequence a:

```python
import numpy as np
def fft(a, inverse = False):
    '''
    Given a sequence a, returns the dft. If inverse parameter is true, return
       the inverse of the dft.
    '''
    conj = 1 if inverse else -1
    originalLength = len(a)
    conjRootsOfUnity = [np.exp(conj*2*np.pi*1j*l/originalLength) for l in range
       (originalLength)]
    def recursiveFFT(FourierVector, N, conjugateFactor):
        if N == 1:
            return FourierVector.copy()
        else:
            NHalf = int(N/2)
            aEven = recursiveFFT(FourierVector[::2], NHalf, 2*conjugateFactor)
            aOdd = recursiveFFT(FourierVector[1::2], NHalf, 2*conjugateFactor)
            for i in range(N):
                FourierVector[i] = aEven[i % NHalf] +
                   conjRootsOfUnity[i*conjugateFactor]*aOdd[i % NHalf]
            return FourierVector.copy()
    return recursiveFFT(a, originalLength, 1)/np.sqrt(originalLength)
```

Note: 2.1: I used numpy library as little as possible in order to make the algorithm as independent as possible with respect to individual programming languages or libraries. However I used np library to compute the roots of unity and the square root, as it contributed to the accuracy.

There is a generalized version of the Fast Fourier Transform for any N that is highly composite. However, going over this algorithm is technical and would require to change the focus of the project, as they rely on splitting the sum in different ways, using theorems number theory. Instead, I will take samples of length $N = 2^k$ for some k or append zeroes to the samples in the next part of the project.

2.4 Denoising Signals

As explained by Calder [5][6.6], noise is defined as random vibrations in a signal that does not belong to the original sampled signal. This noise arises, for instance, from the random motion of electrons inside a conductor, compression artifacts, or corruption in transmission. This noise can affect the resulting signal. The study of electronic systems classifies these noises and studies the origin of such noise, however in this project we will focus on the implementation of algorithms that mitigate the impact of such noise. We will use the fact that noise follows normal distribution in general, as per the central limit theorem, as noise relies on many variables.

There are a number of denoising techniques, but one of the simpler ones to implement is Spectral subtraction, explained by Brunton and Kutz [4][Chapter 2, page. 70]. The assumption will be that noise will usually have considerably different power from the signal. For instance, electrons inside a conductor or other types of distortion will have low pressure, thus having smaller power and frequency.

Definition 2.4.1 (Power Spectral Density)**.** *For* $(\hat{a}_0, \hat{a}_1, ...\hat{a}_{N-1}) \in \mathbf{C}_N$ *representing the frequencies of a signal, the Power Spectral Density will be as follows*

$$PSD = \frac{1}{N}(|\hat{a}_0|^2, |\hat{a}_1|^2, ..., |\hat{a}_{N-1}|^2) \tag{2.43}$$

The value at each index will be the normalized magnitude of each frequency, which indicates the power of the signal at each frequency.

In order to implement what we know, we need the following code. Firstly we need to import libraries we will need. These will be NumPy[15] for array processing and other maths methods, Librosa[14] for signal processing and Matplotlib[13] to plot our results. We will also use Scipy[21] to reconstruct the wav files, the denoised one and the one with noise.

```
    import numpy as np
    import librosa
    from matplotlib import pyplot as plt
    from scipy.io import wavfile
```

And using what was explained in previous sections, we need the following function to set up the audio signal properly.

```
def setUpAudioData(nameOfWavFile, noise = False):
    '''
    Receives the name of the wav file to be imported. If noise is to be added, second parameter is true
    Returns the audio data, with padding, the length of the audio and the original samples without padding.
    '''
    audioData, sampleRate = librosa.load(nameOfWavFile, sr=None)
    ##nyquist rate assuming frequency is band limited by what the human ear can hear, which is 20kHz
    nyquistRate = 40000
    lengthOfAudio = len(audioData)/sampleRate
    originalNumberOfSamples = len(audioData)
    if noise:
        #add noise. Low amplitude as noise will be "lower" than original sampled sound, and it will follow normal distribution. In this case, the mean amplitude of added noise is of 0.005
        audioData = audioData + 0.005*np.random.randn(len(audioData))
    ##As per Nyquist-Shannon theorem, we need a sample rate of 40kHz
    samplesNeededFromShannon = int(np.ceil(nyquistRate*originalNumberOfSamples/sampleRate))
    # Calculate the indices of the evenly spaced samples, such that there are the sufficient samples to satisfy the sampling theorem, and we remove extra samples
    indices = np.round(np.linspace(0, originalNumberOfSamples- 1, int(samplesNeededFromShannon))).astype(int)
    # Select the samples using the calculated indices
    audioData = audioData[indices]
    #We need the number of samples to satisfy Nyquist-Shannon theorem's criteria but also be of the form 2^N for our FFT function
    zeroesNeededForPwtwo = int(2**np.ceil(np.log2(samplesNeededFromShannon)) - samplesNeededFromShannon)
    #Add trailing zeroes so signal is of the form 2^N, without affecting the result
    audioData = np.pad(audioData, (0, int(zeroesNeededForPwtwo)), mode='constant', constant_values=(0, 0)).tolist()
    return (audioData, lengthOfAudio, samplesNeededFromShannon)
```

The algorithm for signal denoising via Spectral subtraction is from Brunton's[4][Code 2.3]. The lowpass filter will depend on physical conditions, and what the expected signal looks like.

```
def denoiseSignal(frequencyData, lowPassFilter):
    frequencyData = frequencyData.copy()
    psd = [aHat*np.conj(aHat)/len(frequencyData) for aHat in frequencyData]
    for i in range(len(psd)):
        power = psd[i]
        if power < lowPassFilter:
            frequencyData[i] = 0
    return frequencyData
```

Afterwards we can use the Fast Fourier Transform function from before in order to test this method, while adding noise to a song. The full code can be found in the github repository: Fast-Fourier-Transform. Using a 27 seconds extract from Beethoven 5th Symphony[2], adding noise to the code and then denoising, we get the following result. The result can be heard in the above repository, with the original, the noisy version, and the denoised version. Note that when denoising, resolution will decrease as well, and this can be heard in the example I provided. This is due to some energy being lost in the subtraction process, but at the same time the similarity of the clean audio and the denoised audio is due to energy being generally conserved following Plancherel Theorem. Furthermore, energy from noise will be cleared out.

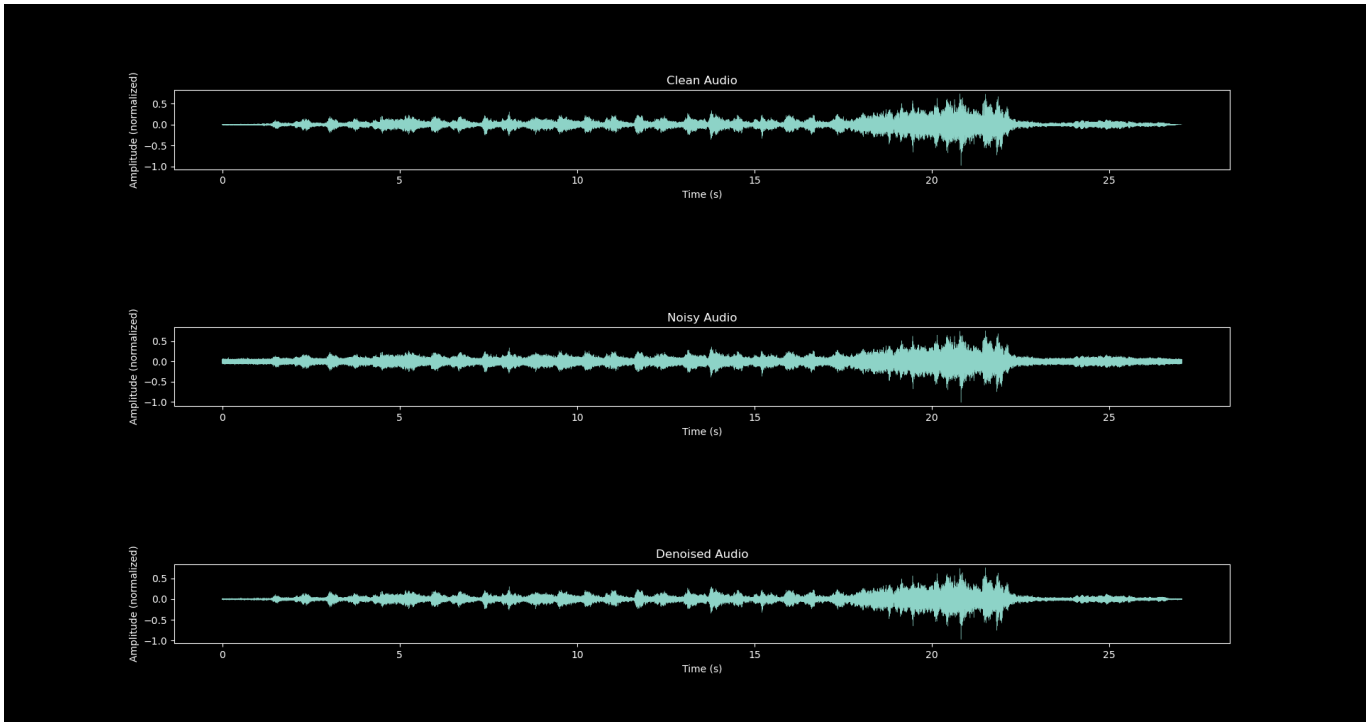

Figure 2.1: You can see that, particularly in quieter sections (sections with lower amplitude), the noisy signal has more effect. When denoised, the denoised signal is similar to the clean signal.

2.5 Partial Differential Equations and the FFT

We can also use the Fast Fourier Transform algorithm in order to solve Partial Differential Equations computationally. We first need the notion of a Spectral Derivative as explained by Brunton [4][2.2, p.77].

It is the numerical approximation of a derivative of a function $f(x)$, it involves taking a discrete sample, computing the DFT (recall that this is an approximation of the continuous FT), applying Corollary 1.3.0.1 and multiplying each kth term by ik, computing the inverse and reconstructing the original function.
Then, as explained by Brunton [4][2.3], solutions to partial differential equations can be approximated numerically in two ways:

- In the frequency domain by applying Corollary 1.3.0.1 in order to convert a PDE into an ODE, solve the ODE in the frequency domain, and compute the inverse DFT to get the original solution of the PDE,
- In the spatial domain by computing the spectral derivative of the FFT, computing the inverse and then integrating in the spatial domain.

Numerical solutions are useful for cases in which the initial value function u_0 and the initial force f are given by discrete samples instead of continuous functions, and also when the integrals such as the ones in Theorem 1.4.1 are not easy to compute. The example I am going to show now in Python is for the McKendrick–von Foerster equation taken from Keyfitz's study[10] which is used in population density estimation with uses for instance in epidemiology. The pde is as follows: suppose $P(a,t)$ is the population density at a given age a and time t, where $m(a)$ is the instantaneous death rate.

$$P(a,t)_t + P(a,t)_a + m(a)P(a,t) = 0 \tag{2.44}$$

For the example I am going to use that $P(a,0)$ follows normal distribution with mean 40 and deviation 15. Computing via the first method I mentioned would be difficult, as the Fourier Transform in a of the term $m(a)P(a,t)$ is not easy to approximate for an arbitrary $m(a)$. This algorithm is similar to Brunton's [4][2.3, Code 2.12] which he used for the Burger equation.

```
kappa = 2*np.fft.fftfreq(N, dx)
p_0 = 1/(dev*np.sqrt(2*np.pi))*np.exp(-(x-mean)**2/(2*dev**2))

def dp_dt(p, t, kappa, m):
    ## This is how we compute the spectral derivative in Python
    p_aHat = -1j*kappa*fft(p + 0j)
    p_a = fft(p_aHat, inverse=True)
    dp_dt = -(p_a + m*p)
    return dp_dt.real
#odeint solves ODEs computationally. How this is done was omitted in this
    project
p = odeint(dp_dt, p_0, t, args=(kappa,m))
```

I used the following initial values

```
L = 100
N = 2**8
dx = L/N
x = np.arange(0,L, dx)
a = 0
b = 0.001
dev = 15
mean = 40
dt = 0.25
t = np.arange(0, 100*dt, dt)
m = a*x**2+b*x
```

and the following code to plot the solutions, (the next snippet of code is exactly as Brunton's [3][min: 11]):

```
plt.style.use('dark_background')
fig = plt.figure()
ax = fig.add_subplot(111, projection="3d")
for j in range(0,len(t), 5):
    ax.plot(np.full_like(x, t[j]), x, p[j])
ax.set_xlabel('t')
ax.set_ylabel('x')
ax.set_zlabel('Population density')
ax.set_title('3D Line Plot')
plt.show()
```

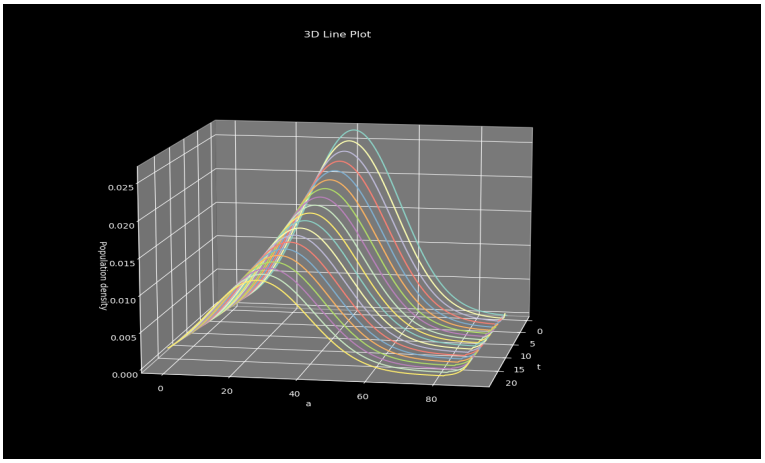

Figure 2.2: This is a fictitious example of a rapidly decaying population where instantaneous death rate is linear. a and t can be thought to be in years but for generality purposes this wasn't specified.

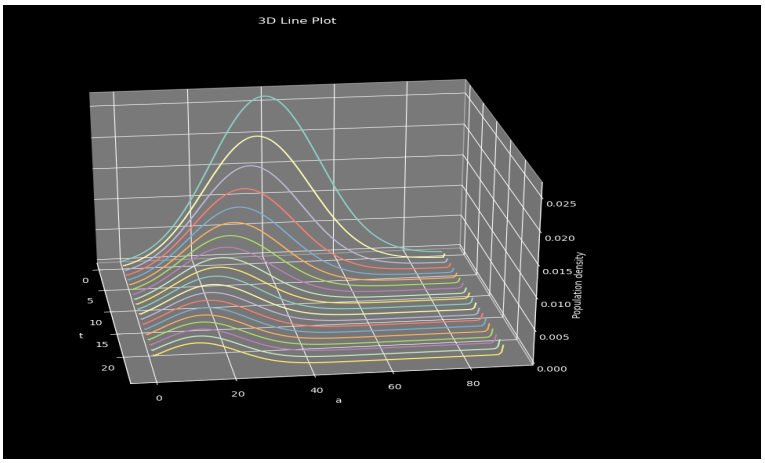

Figure 2.3: Case for m being quadratic ($m = 0.0001a^2 + 0.001a$). Higher ages die off faster.

Code can be found on the github repository: Fast-Fourier-Transform.

Conclusion:

In this project, we delved into the mathematical foundations of Fourier Transforms and later progressed to the study of Fourier Transforms in Signal Analysis. The Fast Fourier Transform played a fundamental role in this second chapter, and two important applications were shown: signal denoising and computational solutions of partial differential equations. For these two applications, we showed an example for each and demonstrated the results. We also analysed the algorithmic complexity of the Fast Fourier Transform. The algorithm in $O(N \log_2(N))$ provides for a more efficient way of computing DFTs, reducing for instance the time spent by patients undergoing MRI scans. Throughout the project, a wide variety of mathematical and computational techniques were used, aligning with the broad spectrum of fields where Fourier Transforms can be applied.

Furthermore, new research and more interesting applications can be developed with Fourier Transforms. For example, the FFT and signal analysis implementations are used in MP3 file conversion. Or, for instance, denoising implementations can also be applied to image denoising, where noise can be seen as artifacts in the image. On the other hand, Fourier Transforms can be used to derive other transforms such as the Elzaki transform[18] which was developed only in 2011 and which is being used to model Tsunami Propagation in research such as Varsoliwala and Singh's [20] in 2021. Also, recall the research being done to lower uncertainty bounds with other transforms which use the Fourier Transform such as the Linear Canonical Transform. All of these are important areas of further research, where the foundational knowledge of Fourier Transforms is needed.

Acknowledgements

I want to thank my supervisor and advisor Dr. Arick Shao, who taught me a different way to study and think about mathematics and supported me during the last years of my degree. I also want to thank Hernán Münch, who introduced me to the world of mathematics and encouraged me to pursue it further, and Samuel Hatt, without whose support I wouldn't have been able to study in the UK. In particular, I want to thank my parents to whom I owe everything for their support, trust, encouragement and patience during my degree and while moving to the UK.

Bibliography

[1] Mohammad Asadzadeh. *Lecture Notes in Fourier Analysis*. Charmer University of Technology, 2008. URL: http://www.math.chalmers.se/~mohammad/teaching/Fourier/LectureNotes_A3/draft_1.pdf.

[2] Ludwig van Beethoven. *Symphony No. 5, Op. 67: II. Andante con moto*. Recorded live on March 10, 2002. 2002. URL: http://imslp.org/wiki/Symphony_No.5,_Op.67_(Beethoven,_Ludwig_van).

[3] Steve Brunton. *Computing Derivatives with FFT [Python]*. https://www.youtube.com/watch?v=y8SqkjoKV4k. Accessed: Date accessed. Year video was published.

[4] Steven L. Brunton and J. Nathan Kutz. *Data Driven Science & Engineering: Machine Learning, Dynamical Systems, and Control*. 1st. Cambridge: Cambridge University Press, 2017. ISBN: 9781108422093. URL: http://databookuw.com/databook.pdf.

[5] Jeff Calder. *Math5467: Introduction to the Mathematics of Image and Data Analysis*. May 10, 2022. URL: https://www-users.cse.umn.edu/~jwcalder/5467Notes.pdf.

[6] John B. Conway. *A Course in Functional Analysis*. Springer, 1990. URL: https://www.datascienceassn.org/sites/default/files/A%20Course%20in%20Functional%20Analysis%20-%20Conway.pdf.

[7] Qiang Feng, Bing-Zhao Li, and John-Michael Rassias. "Weighted Heisenberg–Pauli–Weyl uncertainty principles for the linear canonical transform". In: *Signal Processing* 165 (2019), pp. 209–221. ISSN: 0165-1684. DOI: https://doi.org/10.1016/j.sigpro.2019.07.008. URL: https://www.sciencedirect.com/science/article/pii/S0165168419302610.

[8] Felix Fischer. *Lecture Notes MTH6105 - Algorithmic Graph Theory*. Queen Mary, University of London, 2024. URL: https://qmplus.qmul.ac.uk/mod/resource/view.php?id=2410144.

[9] Qing Gu. *Fourier Transform and Wavelets Course Notes*. East China Normal University. URL: https://math.ecnu.edu.cn/~qgu/lecture.htm.

[10] B.L. Keyfitz and N. Keyfitz. "The McKendrick partial differential equation and its uses in epidemiology and population study". In: *Mathematical and Computer Modelling* 26.6 (1997), pp. 1–9. ISSN: 0895-7177. DOI: https://doi.org/10.1016/S0895-7177(97)00165-9. URL: https://www.sciencedirect.com/science/article/pii/S0895717797001659.

[11] Justin Ko. *Lecture Notes APM 346: Partial Differential Equations*. Week 12. University of Toronto, 2020. URL: https://www.math.toronto.edu/jko/APM346_summary_12_2020.pdf.

[12] Gilad Lerman. *The Shannon Sampling Theorem and Its Implications*. Notes for Math5467: Introduction to the Mathematics of Image and Data Analysis, University of Minnesota Twin Cities. URL: https://www-users.cse.umn.edu/~lerman/math5467/shannon_aliasing.pdf.

[13] Matplotlib Development Team. *Matplotlib*. 2022. URL: https://matplotlib.org/.

[14] Brian McFee et al. *LibROSA*. 2022. URL: https://librosa.org/.

[15] NumPy Developers. *NumPy*. 2022. URL: https://numpy.org.

[16] Sophocles J. Orfanidis. *Introduction to Signal Processing*. Prentice Hall, 2010. ISBN: 978-0-13-209172-5. URL: https://www.ece.rutgers.edu/~orfanidi/intro2sp/orfanidis-i2sp.pdf.

[17] Walter Rudin. *Functional Analysis*. 2nd. New York, NY: McGraw-Hill, 1991. ISBN: 978-0070542365.

[18] Elzaki Tarig. "The new integral transform "Elzaki transform"". In: *Global Journal of Pure and Applied Mathematics* 7 (Jan. 2011), pp. 57–64.

[19] Fawwaz T. Ulaby and Andrew E. Yagle. *Signals and Systems: Theory and Applications*. Chapter 1, Concept Question 12. University of Michigan, Ann Arbor, 2018. ISBN: 978-1-60785-486-9. URL: https://ss2.eecs.umich.edu/concept_questions/ch1/cq12.pdf.

[20] Archana C. Varsoliwala and Twinkle R. Singh. "Mathematical modeling of tsunami wave propagation at mid ocean and its amplification and run-up on shore". In: *Journal of Ocean Engineering and Science* 6.4 (2021), pp. 367–375. ISSN: 2468-0133. DOI: https://doi.org/10.1016/j.joes.2021.03.003. URL: https://www.sciencedirect.com/science/article/pii/S2468013321000255.

[21] Pauli Virtanen et al. "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python". In: *Nature Methods* 17 (2020), pp. 261–272. DOI: 10.1038/s41592-019-0686-2.