OXFORD JOURNAL
OF STUDENT
SCHOLARSHIP

A Mathematical Miracle: Shazam

Urvang Jetly
urvangjetly@gmail.com

ABSTRACT

This paper investigates and explains the already established foundational mathematical system behind the application Shazam, focusing on the Fourier Transform and its applications in audio signal processing. It explains how the Fourier Transform helps in the decomposition of the complex audio input signals into their constituent frequencies, while the Fast Fourier Transform (FFT) is examined as an efficient computational algorithm. This paper also discusses the practical limitations of the Fourier Transform including, time truncation, spectral leakage, frequency resolution, and noise sensitivity.

To investigate the effect of noise sensitivity in input signals, an experimental study was conducted in which audio signals with varying signal-to-noise ratios (SNRs) were tested using Shazam. The results showed that decreasing the SNR increased identification time, and the accuracy had a sudden decrease when the noise power was set equal to the signal power and Shazam was not able to identify the song when the noise overpowered the input signal. Finally, the paper examines how Shazam overcomes these challenges through spectral peak extraction and audio fingerprinting, demonstrating the practical power of Fourier-based signal analysis.

INTRODUCTION

There have been many instances in my life where a song playing in the background catches my attention, and I want to add it to my playlist. I simply take out my phone, tap the Shazam button and allow the application to identify the song.

Many readers will likely have used Shazam once, but do you know how it works? Is there a musical genius listening to your phone's audio when you press the Shazam button and instantly identifying the song? In reality, no such person exists; instead Shazam works because of a mathematical genius, the Fourier Transform.

The mathematical foundation of audio identification systems has already been documented in literature, especially through Avery Wang's 2003 paper and also through other studies. Hence this paper does not claim any originality in the derivation of the Fourier Transform, the Fast Fourier Transform, or fingerprinting algorithms. Instead, this paper's aim is to provide an integrated and accessible mathematical explanation of these concepts while also examining the practical limitations of the Fourier Transform through an original experimental study on noise sensitivity. In particular, the experimental investigation

explores how varying signal-to-noise ratios affect Shazam's ability to quickly and accurately identify the song and relates the observed behaviour to Fourier-domain analysis.

THE MATHEMATICS OF SOUND

Sound Waves as a Function of Time

We hear sounds after plucking a guitar string or singing a note because these produce waves in air. The waves cause disturbances in the air pressure that travels through the medium, i.e., the air, into our ear, enabling us to hear. One may wonder how this relates to mathematics. In fact, the most simple sound wave can be represented by,

$$y(t) = Asin(2\pi ft)$$

In this equation each term has a special and important purpose.
y(t): Tells the displacement of a particle in the wave passing through the air at any given time instant 't'
A: Refers to the amplitude of the wave, which is perceived as loudness, this loudness depends on the intensity of the wave, which in turn depends on its amplitude, mathematically speaking, $I \propto A^2$ where I represents the intensity of the wave and A represents the amplitude of the wave

f**:** Refers to the frequency of the wave. Our ears perceive the frequency as the pitch of the sound.

2π: This is the conversion factor which converts the waves frequency from 'cycles per second' to 'radians per second'

Multiplying f with 2π stretches or squeezes the graph along the x-axis in the xy plane s.t. that wave completes exactly 'f' cycles per second.

Unlike a pure wave, the music we listen to has a very complex waveform and is made up of thousands of sine waves added together.

Mathematically, this is known as the superposition of waves.

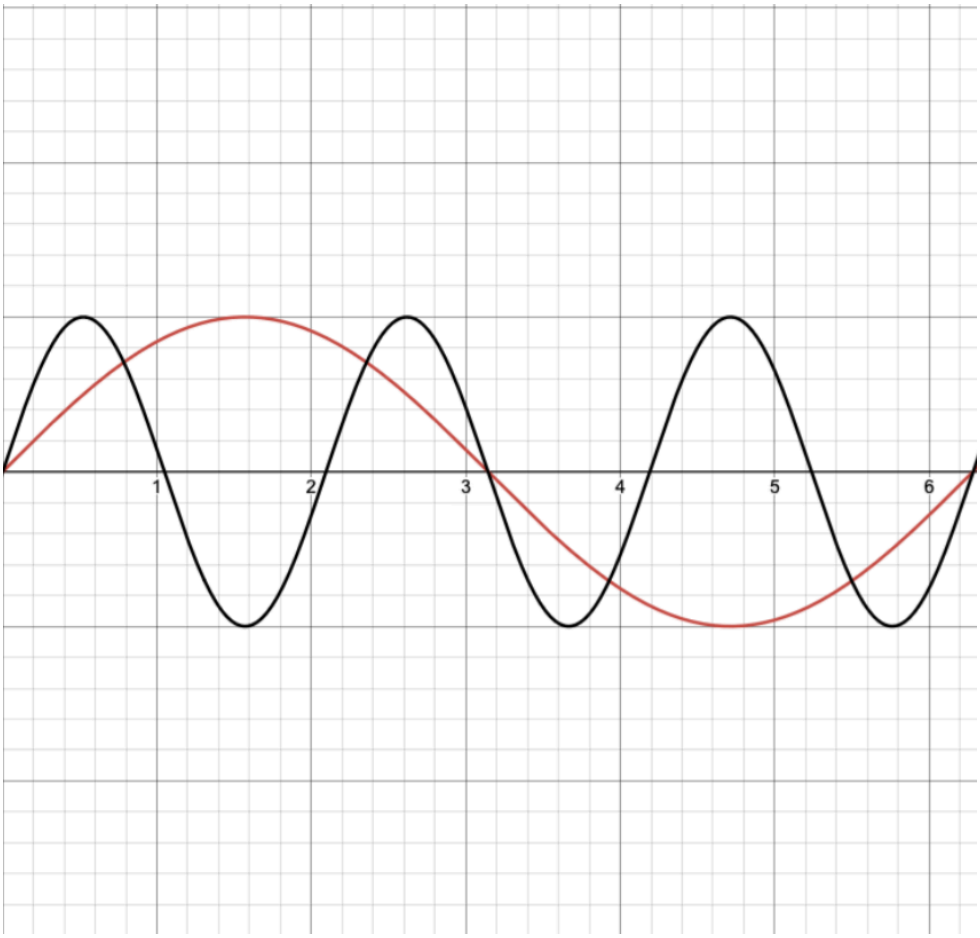

Figure 1: $sin\ 3x$ (BLACK) completes 3 whole cycles while $sin\ x$ (GREEN) completes 1 in the same period.

PRINCIPLE OF SUPERPOSITION

The Principle of Superposition states that when two or more waves overlap in a medium, the resultant displacement at any point is the algebraic or vector sum of the individual displacements of each wave at that point and it is a fundamental principle of wave physics; Mathematics provides the tools needed to represent these waves as a **sum of sinusoidal waves.**

The following example illustrates the mathematics underlying this principle.

Consider, $y_1(t) = Asin(2\pi f_1 t),\ y_2(t) = Asin(2\pi f_2 t)$:

$y(t) = Asin(2\pi f_1 t) + Asin(2\pi f_2 t)$

Using the identity,

$$\sin x + \sin y = 2\sin\left(\frac{x+y}{2}\right)\cos\left(\frac{x-y}{2}\right)$$

Let $x = 2\pi f_1 t,\ y = 2\pi f_2 t$,

$$\Rightarrow y(t) = 2A\sin\left(\frac{2\pi f_1 t + 2\pi f_2 t}{2}\right)\cos\left(\frac{2\pi f_1 t - 2\pi f_2 t}{2}\right)$$

$$\Rightarrow y(t) = 2A\sin\left(2\pi\frac{f_1+f_2}{2}t\right)\cos\left(2\pi\frac{f_1-f_2}{2}t\right)$$

$$\Rightarrow f_{avg} = \frac{f_1+f_2}{2},\quad f_{mod} = \frac{|f_1-f_2|}{2}$$

$$A_{env}(t) = 2A\cos(2\pi f_{mod} t)$$

$$\therefore y(t) = A_{env}(t)\sin(2\pi f_{avg} t).$$

Hence, the resultant wave may be viewed as a sinusoidal carrier wave of frequency f_{avg}, whose amplitude is controlled by the envelope $A_{env}(t)$ defined above. Since $A_{env}(t)$ depends on time, the amplitude of the resultant wave varies periodically, giving rise to beats.

Generally speaking,

$$y_{total}(t) = y_1(t) + y_2(t) + y_3(t) + \ldots = \sum_{i=1}^{n} A_i \sin(2\pi t f_i + \phi_i)$$

From this, we can conclude that the mathematical core behind all music is actually the superposition of many thousands of individual sinusoidal components.
After this, the question arises: how does Shazam know what waves a certain song is made up of?

This leads us to the Fourier Transform.

THE FOURIER TRANSFORM

This is where one of ***Jean-Baptiste Joseph Fourier's*** most significant contributions to mathematics becomes pertinent.

The Fourier Transform;

$$X(f) = \int_{-\infty}^{\infty} x(t)e^{-2\pi ift}\,dt$$

First, the practical use of this mathematical tool will be illustrated through an example, then each term of the equation will be explained.

Frequency-Domain Representation of Audio Signals

A sample audio segment from the song *Time* by Pink Floyd was selected.

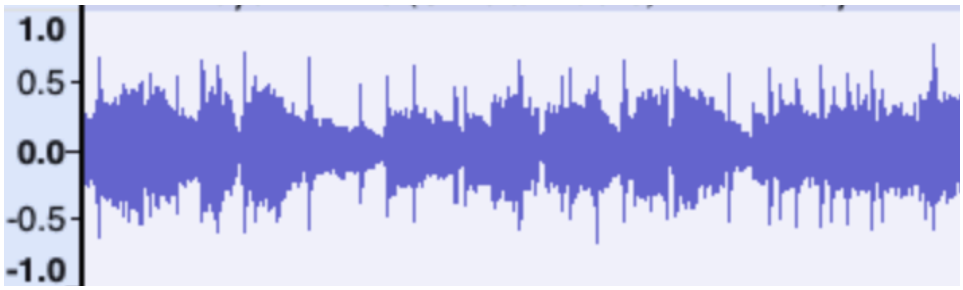

Figure 2: *Time Domain Graph of 'Time' by Pink Floyd*

Figure 2 is complex and difficult to interpret, hence it is very difficult for Shazam to directly identify the song from it.

We also have to consider the fact that in the actual audio there will be background noises in addition to the song itself, hence the graph will be even more complex and it will become almost impossible for Shazam to interpret it.

Thus, Shazam does not rely directly on the raw audio signal and this is where the Fourier Transform comes into play,

The Fourier Transform converts this time-domain graph into a frequency-domain graph which in practical uses is further processed to produce a fingerprint.

Note: The fingerprint is produced with the help of a spectrogram in which both time and frequency are accounted for along with the intensity of the signal. From this 3 variable spectrogram dominant spectral peaks are chosen to form a fingerprint.

After obtaining the time-domain graph, its Discrete Fourier Transform (DFT) was computed using the Fast Fourier Transform (FFT) algorithm;

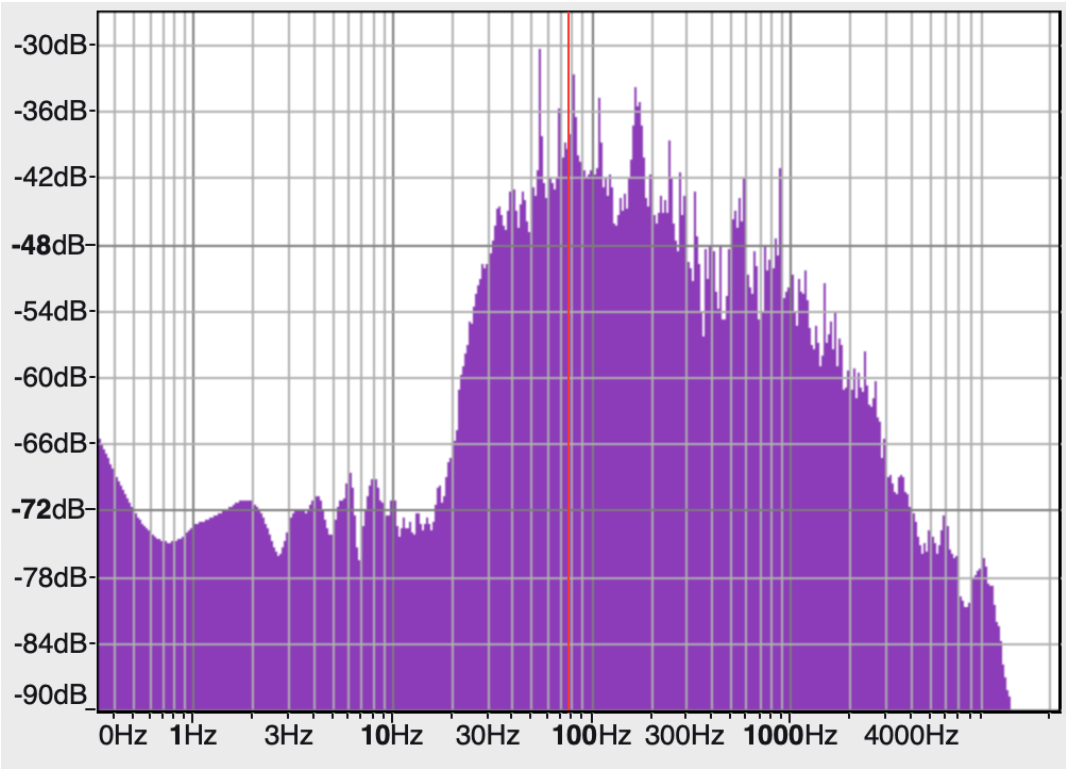

Figure 3: *Frequency Domain Graph of 'Time' by Pink Floyd obtained by computing the DFT of the sampled audio using the FFT algorithm*

Notice how in ***Figure 3*** the x-axis is representing frequency instead of time.
We observe that this is much cleaner and easier to interpret than the corresponding time-domain representation.
This is the purpose of the **Fourier Transform**: converting a signal from the **time domain to the frequency domain.**

Fourier Transform Explained

$$X(f) = \int_{-\infty}^{\infty} x(t)e^{-2\pi ift}\,dt$$

x(t): It is the input signal which is represented in the time-domain.
A periodic signal satisfying the Dirichlet conditions can be represented by a convergent Fourier series, they are:

- The signal should be absolutely integrable over one period, i.e.,

$\int_0^{T_0} |x(t)|dt < \infty$, where T_0 denotes the fundamental period of a signal.

- The input signal must have a **finite** number of **maxima and minima** in any given **finite interval**.
- The input signal must have a **finite number** of **discontinuities** within any given bounded interval, and each **discontinuity must itself be finite**

$e^{-2\pi fit}$: This term may be interpreted as a test wave that oscillates at a frequency 'f'.

In ***Figure 4***, a sinusoidal wave (blue) is shown which when multiplied by $e^{-2\pi fit}$, maps each value of the signal to a rotating complex vector in a clockwise direction at a rate proportional to the test frequency 'f'. This allows the signal to be compared against the test frequency 'f' where matching frequencies will accumulate (constructive interference) and non-matching frequencies will cancel out on an orthogonal basis.

To explain what "comparison on an orthogonal basis" means consider the following example:

Recall that, $cos\, x = \frac{e^{ix}+e^{-ix}}{2}$.

Let us say $x(t) = cos(2\pi f_1 t)$,

$$\Longrightarrow X(f) = \int x(t)e^{-2\pi ift}\,dt$$

$$\Longrightarrow X(f) = \int cos(2\pi f_1 t)e^{-2\pi ift}\,dt$$

$$\Longrightarrow X(f) = \frac{1}{2}\int (e^{2\pi if_1 t} + e^{-2\pi if_1 t})e^{-2\pi ift}\,dt$$

$$= \frac{1}{2}\int (e^{2\pi i(f_1 - f)t} + e^{-2\pi i(f_1+f)t})\,dt$$

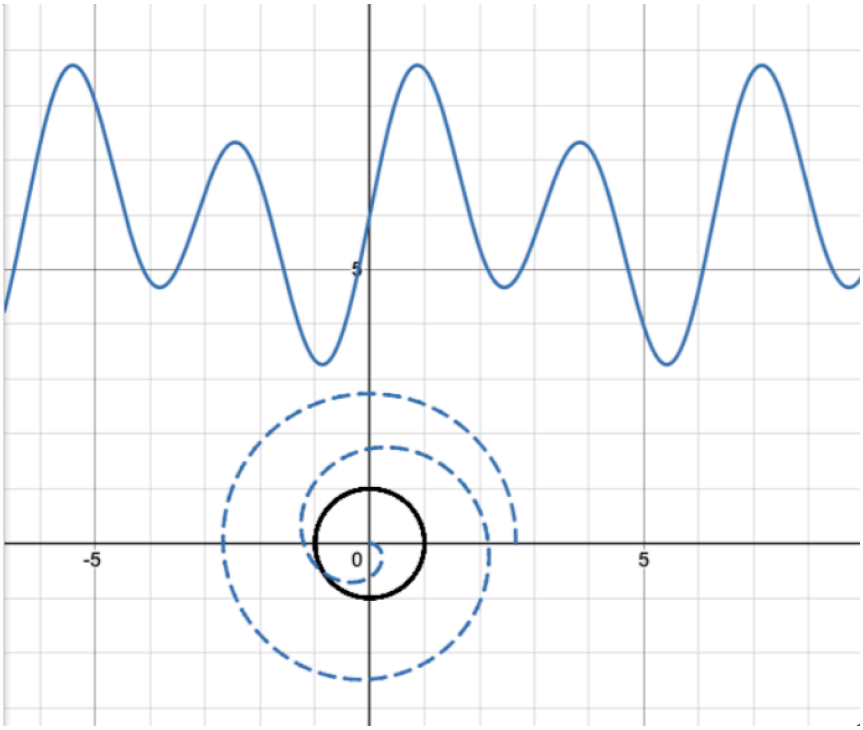

Figure 4: *Shows the graph (dotted line) obtained when an input signal is multiplied by* $e^{-2\pi fit}$

Let the net frequency = α
Now the orthogonality of complex exponentials is seen as,

$$\int_{-\infty}^{\infty} e^{2\pi\alpha it}\, dt = \delta(\alpha)$$

Where δ(α) denotes the Dirac delta distribution. Applying the above result gives,

$$X(f) = \frac{1}{2}[\delta(f - f_1) + \delta(f + f_1)]$$

From this result we can conclude that when a signal is passed through the Fourier Transform, it shifts from the time domain to the frequency domain and the **output is non-zero ONLY** at the frequencies that **match the component of the original signal**. This occurs as the **matching frequencies accumulate constructively under integration**, whereas the **non-matching frequencies cancel out to 0 under integration** due to orthogonality.

COMPUTING THE DFT EFFICIENTLY: THE FFT ALGORITHM

In the section above, the Fourier Transform was introduced theoretically. In practical digital systems, this transformation is computed using the DFT. The *Discrete Fourier Transform* operates only on sampled signals.

The Discrete Fourier Transform,

$$X_k = \sum_{n=0}^{N-1} x_n e^{-2\pi kin/N}, \qquad k = 0, 1, \ldots, N-1$$

$where$:

x_n is the n^{th} sample of the signal,

N is the total number of samples

k indexes the discrete frequencies,

X_k gives the contribution of the k^{th} frequency-component.

The Discrete Fourier Transform computes the contribution of each discrete frequency component present in the signal while the Fast Fourier Transform performs the same computation more efficiently.
While the Discrete Fourier Transform calculates the contribution of each discrete frequency component, it multiplies each point in the signal then sums up everything. Hence, this process is being done 'N' times and each time there are 'N' operations making the total work $O(N) \times O(N) = O(N^2)$

This becomes computationally expensive when N is large. For example, if $N = 10^6$, then $N^2 = 10^{12}$ operations, which is computationally expensive.
So this is where the Fast Fourier Transform comes in. By exploiting symmetries in the complex exponential terms, it decomposes a DFT of size N into 2 DFTs of size $\frac{N}{2}$, hence reducing the total number of computations required.
The mathematical basis of the FFT is as follows ,

X_k = (even part) + (odd part)

$X_k = E_k + W^k{}_N \, O_k$,

where E_k = Discrete Fourier Transform of even-indexed points, O_k = Discrete Fourier Transform of odd-indexed points and $W_N = e^{-2\pi i/N}$.

This recursive decomposition continues until subproblems of size 1 are reached, producing a recursion tree of depth $\log_2 N$, At each level of the recursion tree, the total work required to combine the subproblems is $O(N)$. As there are $\log_2 N$ levels and each level requires $O(N)$ operations, the total running time is $O(N) \times O(\log_2 N) = O(N \log N)$

From the time complexity difference we can conclude that the Fast Fourier Transform Algorithm is *more efficient* than the Discrete Fourier Transform.

THEORETICAL LIMITATIONS

In the earlier sections we are assuming the fact that all examples have taken place in perfect conditions, following all mathematical & physical conditions required. However in the case of practical applications of the Fourier Transform these assumptions do not hold.
In this section, we will investigate how - Time Truncation, Spectral Leakage and Frequency Resolution affect the frequency-domain representation.

Effect of a Finite Signal (Time Truncation)

The signals we input into the Fourier Transform are for a finite interval **T** and are a truncated version s.t.,

$x_T(t) = x(t)w_T(t)$, where $w_T(t) = 1_{[0,T]}(t) = \{1;\ 0 \leq t \leq T,\ 0: otherwise\}$.

Where $1_{[0,T]}(t)$ is the indicator function on the interval $[0, T]$,
and by the convolution property of the Fourier Transform,

$X_T(f) = X(f) * W_T(f)$ where * denotes convolution.

When we put $w_T(t)$ through the Fourier Transform the output is,

$W_T(f) = T\,sinc(\pi fT)\,e^{-i\pi fT}$, where $sinc(x) = \frac{sinx}{x}$

As an example if we take $x(t) = cos(2\pi f_1 t)$ then, $X(f) = \frac{1}{2}\left[\delta(f - f_1) + \delta(f + f_1)\right]$
hence the truncated spectrum becomes,

$X_T(f) = \frac{T}{2}\left[sinc(\pi(f - f_1)T) + sinc(\pi(f + f_1)T)\right]$

Due to this truncated spectrum, instead of the sharp frequency peaks which are given when an infinite signal is passed we get sinc-shaped peaks which have a width proportional to $\frac{1}{T}$, which means that

shorter signals will have reduced frequency precision.
∴ The effect of a finite signal is that it makes the frequency-domain graph less accurate.

Spectral Leakage

The loss of accuracy that we learnt about in the above section not only leads to a visual effect but it also leads to the energy spreading into the neighbouring frequencies. As we saw before,

$W_T(f) = T\,sinc(\pi fT)\,e^{-i\pi fT}$, where $sinc(x) = \frac{sinx}{x}$

Due to this, each ideal frequency component present in $X(f)$ is convolved with a sinc function which produces a main lobe (of width $\frac{1}{T}$) and side lobes which are non-zero $\forall\, f$.

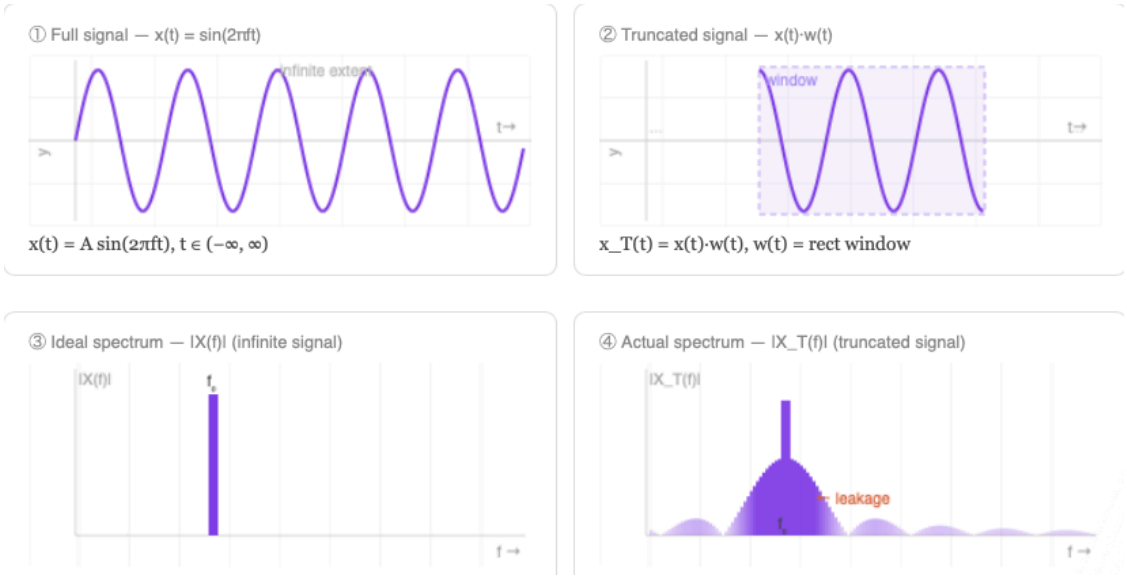

Figure 5: *Infinite Signal v/s Truncated Signal*

This means that the energy from a single frequency spreads into its adjacent frequencies (as shown in ***Figure 5***) instead of remaining perfectly localized, this is known as **Spectral Leakage**.

Frequency Resolution

Time Truncation imposes a limit on how closely spaced frequencies can be distinguished and this is known as **Frequency Resolution.**
The truncated spectrum that we have arrived at contains sinc-shaped peaks of the form,

$sin(\pi(f - f_1)T)$

Let us say the first root of this function occurs at,
$f = f_1 \pm \frac{1}{T}$,

So 2 frequencies f_1 and f_2 can only be uniquely resolved iff their separation satisfies
$|f_1 - f_2| \gtrsim \frac{1}{T} \Longrightarrow \Delta f \approx \frac{1}{T}$

This means that if T is short then the nearby frequencies will overlap which will lower the frequency resolution. If T is longer then there is a finer frequency discrimination.

RELATION TO EXISTING SCHOLARSHIP

Avery Wang's 2003 paper 'An Industrial-Strength Audio Search Algorithm' introduced the landmark-based audio fingerprinting system that formed the basis of Shazam. Many recent studies and papers have focused on improving the audio fingerprinting robustness (such as the 'Enhancing Neural Audio Fingerprint Robustness to Audio Degradation for Music Identification') to noise. These studies primarily examine and explore the algorithmic design, whereas the following investigation focuses on the mathematical consequences of noise from the perspective of Fourier-domain analysis and experimentally examines how increasing noise levels affect practical song recognition.

RESEARCH ON NOISE SENSITIVITY

In this section, the effect of Noise Sensitivity in input signals passed through the Fourier Transform will be explored.

Mathematical Modelling of Noise in Signals

Let us assume an input signal $y(t)$ is passed through the Fourier Transform and that it has some noise component. Then we can write $y(t)$ as:

$y(t) = x(t) + \epsilon(t)$,
where $x(t)$ represents the pure signal and $\epsilon(t)$ is the noise added to it.

Now due to the ***linearity*** of the Fourier Transform,

$Y(f) = X(f) + \varepsilon(f)$

where,

- $X(f)$ is the Fourier Transform of the pure signal
- $\varepsilon(f)$ is the Fourier Transform of the noise
- $Y(f)$ is the Fourier Transform of the input signal

This demonstrates that any noise present in the input signal will not be discarded, instead the noise will also be shifted from the time-domain to the frequency-domain.
∴ Any noise present in the input signal will affect the frequency representation.

Effect and Characteristics of Noise in the Frequency Domain

As we have seen above, the Fourier Transform of the input signal consists of the Fourier Transform of the pure signal and that of the noise. $X(f)$ has a definite structure and distinct peaks but because $\varepsilon(f)$ has a broad and distributed frequency-domain representation; the transform of the input signal has additional frequency components due to the noise present.
As $\varepsilon(f)$ is spread across all frequencies due to its random nature, it creates a **baseline level** in the transform of $y(t)$. This baseline level is known as the **noise floor**.
A clean spectrum that is free of noise has values close to 0 except the frequency peaks, whereas in a noisy signal's spectrum everything will be slightly above 0 which reduces the difference between the actual frequency peaks and the frequency due to the noise. This makes it difficult to distinguish between the actual frequency peaks and the high frequencies present due to the noise.

The effect of noise in the frequency-domain depends on its characteristics:

1. ***White Noise*** - It is the type of noise which has equal power at all frequencies. Mathematically, $S_n(f) = N_o\ (a\,constant)$, where $S_n(f)$ is the power spectral density (PSD) of the noise and f is the frequency. This means that the noise is uniform and will be distributed across the frequency domain in an even manner making the noise floor flat (i.e. parallel to the frequency axis).

2. ***Gaussian Noise*** - It is the type of noise whose amplitude follows normal distribution, i.e., $\epsilon(t) \sim N(0, \sigma^2)$
   In this type of noise most noise values are small, large deviations rarely occur and the noise fluctuates in a random manner near 0.

3. ***Environmental Noise*** - Unlike ideal noise, real-world noise can be modelled as, $\epsilon(t) = \Sigma_i a_i cos(2\pi f_i t) + \eta(t)$, where $\eta(t) =$ random component (often Gaussian) In contrast to the theoretical white noise, real-world noise can induce both wideband disruptions and specific frequency elements, which may interfere with the true signal and alter its spectral representation.

Signal-to-Noise Ratio (SNR)

Even though the Fourier Transform separates frequencies, distinguishing the original signal from the noise depends on how large the signal is relative to the surrounding noise.

For a continuous signal,

$$P_s = \frac{1}{T}\int_0^T |x(t)|^2\, dt$$

where, P_s = average signal power, $x(t)$ = original signal and

$P_n = \frac{1}{T}\int_0^T |\epsilon(t)|^2 \, dt$

where, P_n = average noise power, $\epsilon(t)$ = noise signal

Mathematically, this means that Power measures the average energy content of a signal over time.

$Linear\ SNR = \frac{P_s}{P_n}$

Experimental Investigation of Noise Sensitivity in Shazam

The objective of this experiment was to investigate how decreasing the signal-to-noise ratio (SNR) affects the ability of Shazam to correctly identify an audio signal in as little time as possible.

For this experiment **Shazam Version 26.12.0** was used on the iPhone 16e.
Controlled levels of white noise were added to a clean audio sample of the song *Self Care by Mac Miller* using the ***Audacity*** application. Then the time taken to identify the song at different SNR levels was measured using a stopwatch with a **least count of 0.01 seconds.** The signal-to-noise ratio (SNR) was measured both as a linear power ratio and in decibel form. The sampling rate of the audio was **44.1kHz** and all trials were conducted in a room with **no external noise** and the playback volume was constant for all trials. The two quantities are related by

$\text{SNR}_{dB} = 10\log_{10}(\frac{P_s}{P_n}) = 10\log_{10}(Linear\ SNR)$

where P_s denotes the signal power and P_n denotes the noise power.

Different SNR levels were generated by adjusting the RMS (Root Mean Square) amplitude of the noise signal relative to the original audio signal using the Audacity application.

The readings from the experiment are shown below:

Effect of SNR on Recognition Time and Accuracy:

SNR (dB)	**Power Ratio** $\frac{P_s}{P_n}$	**Trial 1 (s)**	**Trial 2 (s)**	**Trial 3 (s)**	**Avg Time (s) ± S.D.**	**Correct Match?**
20	100:1	4.62	4.95	4.93	4.83±0.19	Y/Y/Y
10	10:1	4.68	4.99	5.04	4.90±0.16	Y/Y/Y
5	3.16:1	9.45	9.93	11.98	10.45±1.10	Y/Y/Y
0	1:1	12.23	-	-	-	Y/N/N
-5	1:3.16	-	-	-	-	N/N/N
-10	1:10	-	-	-	-	N/N/N

(In the table above: Y-Yes & N-No Result)

(The mean was calculated using the formula ***Avg Time*** $= \frac{(\sum_{i=1}^{3} Trial_i)}{3}$*, and the Standard Deviation was calculated using the formula* ***S.D.***$=\sqrt{(\frac{\sum_{i=1}^{3}(Trial_i-\mu)^2}{3})}$ *where* μ *= Avg Time)*

(Recognition time was recorded only for successful identifications. Trials in which Shazam failed to identify the song are denoted by "–")

Observation

The results from the experimental investigation show that as the signal-to-noise ratio (SNR) decreases it causes a significant increase in the time taken by Shazam to correctly identify the song.

At an SNR of 20 dB, corresponding to the power ratio of $\frac{P_s}{P_n} = 100:1$, The song was correctly identified in **4.83±0.19** seconds. However at an SNR of 0 dB, $\frac{P_s}{P_n} = 1:1$, where the noise power was equal to the signal power the identification time of the song increased to **12.23** seconds in the 1st trial and trials 2 & 3 were unsuccessful leading to an accuracy rate of only **33.33%**.

Overall, this experiment demonstrates that decreasing SNR both increases recognition time and reduces identification accuracy, with the transition from reliable recognition to consistent failure occurring near an SNR of 0 dB.

Mathematical Interpretation

The impact of noise on the signal's Fourier-domain representation can account for the rise in recognition time. The effective noise floor in the frequency spectrum rises in parallel with the noise level, making dominating spectral peaks harder to identify. As a result, it becomes harder to extract spectral peaks with confidence.

However, the accurate identification of the song at SNR values as low as 5 dB suggests that Shazam does not use the entire Fourier spectrum. Rather, it derives strong correlations between dominating spectral peaks that are stable enough even in the presence of substantial background noise.

The decline in recognition accuracy at 0 dB and the **complete failure of identification at −5 dB and −10 dB** indicate the existence of a **practical noise threshold** beyond which reliable spectral fingerprint extraction becomes increasingly difficult.

As a result, the *experiment reveals the practical limitations as well as the advantages of the Fourier signal analysis*. While increased noise significantly reduces the efficiency and reliability of audio identification, strong spectral fingerprinting techniques allow accurate song recognition even when a significant amount of noise is present in the signal (at SNR = 5 dB in the experiment).

APPLICATION TO SHAZAM

From the results of Time Truncation, Spectral Leakage, Frequency Resolution, and Noise Sensitivity we can conclude that there is a significant difference between the theoretical and practical use of the Fourier Transform.

Due to these factors, it would be difficult for Shazam to identify the correct song solely from the obtained frequency-domain graph.

Consequently, Shazam applies several additional processing stages in order to identify the song with high accuracy in noisy environments:

1. **Conversion into a spectrogram**
   When the Fourier Transform is applied to the full audio signal, it provides the frequencies present in the signal, but it cannot determine at what time those frequencies occur. To overcome this problem, Shazam applies the FT repeatedly over short intervals of time using the Short-Time Fourier Transform (STFT). Mathematically, the STFT is defined as:

   $$X(t,f) = \int_{-\infty}^{\infty} x(\tau)w(\tau - t)e^{-i2\pi f\tau}\, d\tau$$

   where: $x(\tau)$ represents the audio signal,
   τ Is the integration variable,
   $w(\tau - t)$ is a window function centred around time t and,
   $X(t,f)$ gives the frequency content at a **specific time interval.**

   So, instead of producing a frequency-domain graph, the STFT produces a time-frequency graph known as a spectrogram. In this,

   - the horizontal axis represents time,
   - the vertical axis represents frequency,
   - and the brightness represents the magnitude of the Fourier coefficients.

   This lets Shazam analyse how frequencies change during the song rather than the full signal at once.

2. **Extraction of Spectral Peaks**

   After obtaining the spectrogram of the audio signal, storing all information contained in the spectrogram would be computationally inefficient, hence Shazam only extracts the dominant spectral peaks from the spectrogram.
   Mathematically, these peaks correspond to the maxima of,

   $$|X(t,f)|$$

   Finding the strongest peaks is very important because noise usually spreads small amounts of energy across many frequencies, while meaningful musical elements create much more prominent peaks. As a result, even when the signal is affected by noise, the main spectral peaks can often still be clearly identified above the surrounding noise floor.

This helps explain the experimental observation that Shazam remained capable of identifying songs at low SNR values.

Hence, Shazam only retains the strongest and most stable frequencies from the obtained spectrogram.

3. **Formation of Audio fingerprints**

Spectral Peaks alone are not sufficient to identify a song. To increase reliability, Shazam forms relationships between nearby peaks in the spectrogram. Suppose 2 dominant peaks occur at: (f_1, t_1) and (f_2, t_2)
Shazam then constructs a fingerprint of the form:

$H = (f_1, f_2, \Delta t)$
where Δt represents the time separation between the peaks

The combination of dominant spectral peaks with their time separation (Δt) helps maintain accuracy when noise is present. At times noise distorts the weaker parts of the spectrum. However, the strongest peaks and their timing usually stay the same. This explains why the song was successfully identified at SNR values of 20 dB, 10 dB, and 5 dB in the experimental investigation. The accuracy of recognition of the song decreased as the noise levels increased which made it harder to maintain these spectral relationships.

This process creates highly unique audio fingerprints. While there may be many similar high frequencies in many songs, the relative relation between multiple dominant spectral peaks are far more unique and stable.

4. **Database Matching**

After these fingerprints are generated, they are compared against a large database of previously stored fingerprints. Suppose that,

t_i represents the timestamp of a fingerprint in the input signal, and
t_d represents the timestamp of a matching fingerprint in the database.

For the correct song, the quantity, $t_d - t_i$ remains approximately constant across many fingerprint matches.

∴ Correct matches form strong clusters while random matches appear to be scattered. Shazam identifies the song with the greatest concentration of matching fingerprints and is able to accurately identify the song, even in noisy environments.

CONCLUSION

The aim of this paper was to explore the already established mathematics behind Shazam, focusing on the role of the Fourier Transform in audio signal processing. The discussion began with the application of the Fourier Transform in converting an input signal from the time domain to the frequency domain.

However, we saw that the practical application of the Fourier Transform significantly differed from the theoretical formulation. By examining time truncation, spectral leakage, frequency resolution, and noise sensitivity, it was shown that real-world audio signals introduce several challenges that are not present in ideal mathematical models.

Besides explaining the established mathematics underlying audio identification, through the original experimental investigation in noise sensitivity, it was observed that as the signal-to-noise ratio decreased, the time required for Shazam to identify the song increased substantially. After a significant increase in noise levels in the input signal, Shazam was not able to identify the song, highlighting the robustness and limitations of modern audio identification systems.

Overall, this investigation has shown that the Fourier Transform is not just a mathematical tool for changing the domain from time to frequency, but it is also a powerful model for extracting useful information from signals. Although there may be practical limitations of its direct application, with the help of techniques such as time-frequency analysis and audio fingerprinting we are able to overcome these problems.

Ultimately, this study illustrates that the genuine strength of the Fourier Transform is not found in the calculation of frequencies as such, but in the uncovering of fundamental structure via mathematical decomposition.

REFERENCES

- Wang, Avery Li-Chun. "An Industrial Strength Audio Search Algorithm." *Proceedings of the 4th International Society for Music Information Retrieval Conference (ISMIR)*, 2003. https://www.ee.columbia.edu/~dpwe/papers/Wang03-shazam.pdf.
- Freeman, Dennis. "Lecture 15: Fourier Series." *MIT OpenCourseWare*, Massachusetts Institute of Technology, 2011. https://ocw.mit.edu/courses/6-003-signals-and-systems-fall-2011/resources/lecture-15-Fourier-series/.
- Freeman, Dennis. "Lecture 16: Fourier Transform." *MIT OpenCourseWare*, Massachusetts Institute of Technology, 2011. https://ocw.mit.edu/courses/6-003-signals-and-systems-fall-2011/resources/lecture-16-Fourier-transform/.
- Oppenheim, Alan V. "Continuous-Time Fourier Transform." *MIT OpenCourseWare*, Massachusetts Institute of Technology, 2011. https://ocw.mit.edu/courses/res-6-007-signals-and-systems-spring-2011/resources/lecture-8-continuous-time-Fourier-transform/.
- Oppenheim, Alan V. "Fourier Transform Properties." *MIT OpenCourseWare*, Massachusetts Institute of Technology, 2011. https://ocw.mit.edu/courses/res-6-007-signals-and-systems-spring-2011/resources/lecture-9-Fourier-transform-properties/.
- Massachusetts Institute of Technology. "Lecture Notes: Signals and Systems." *MIT OpenCourseWare*, 2011. https://ocw.mit.edu/courses/6-003-signals-and-systems-fall-2011/pages/lecture-notes/.
- Massachusetts Institute of Technology. *Signals and Systems*. MIT OpenCourseWare, 2011. https://ocw.mit.edu/courses/res-6-007-signals-and-systems-spring-2011/.
- Jerison, David. *Fourier Analysis*. MIT OpenCourseWare, Massachusetts Institute of Technology, 2013. https://ocw.mit.edu/courses/18-103-Fourier-analysis-fall-2013/.
- Smith, Steven W. *The Scientist and Engineer's Guide to Digital Signal Processing*. California Technical Publishing, 1997. https://www.dspguide.com/pdfbook.htm.

- Smith, Steven W. "Chapter 8: The Fourier Transform." *The Scientist and Engineer's Guide to Digital Signal Processing*, California Technical Publishing, 1997. https://www.dspguide.com/CH8.PDF.
- Smith, Steven W. "Chapter 12: The Fast Fourier Transform." *The Scientist and Engineer's Guide to Digital Signal Processing*, California Technical Publishing, 1997. https://www.dspguide.com/ch12.htm.
- Ellis, Daniel P. W. "Robust Landmark-Based Audio fingerprinting." *Columbia University LabROSA*. https://www.ee.columbia.edu/~dpwe/LabROSA/matlab/fingerprint/.
- Serrano, Sergio, et al. "A New fingerprint Definition for Effective Song Recognition." *Pattern Recognition Letters*, vol. 165, 2022, pp. 88–95. https://www.sciencedirect.com/science/article/abs/pii/S0167865522002033.
- Kamuni, Navin, et al. "Advancing Audio fingerprinting Accuracy: Addressing Background Noise and Distortion Challenges." *arXiv*, 2024. https://arxiv.org/abs/2402.13957.
- Altwlkany, Kemal, et al. "Application of Audio fingerprinting Techniques for Real-Time Scalable Speech Retrieval and Speech Clusterization." *arXiv*, 2024. https://arxiv.org/abs/2410.21876.
- Sanderson, Grant. "But What Is the Fourier Transform? A Visual Introduction." *3Blue1Brown*, 25 Jan. 2018. https://www.3blue1brown.com/lessons/Fourier-transforms.
- Sanderson, Grant. "But What Is a Fourier Series? From Heat Flow to Circle Drawings." *3Blue1Brown*, 30 June 2019. https://www.3blue1brown.com/lessons/Fourier-series.
- Sanderson, Grant. *Essence of Linear Algebra*. *3Blue1Brown*, 2016. https://www.youtube.com/playlist?list=PLZHQObOWTQDPD3MizzM2xVFitgF8hE_ab.
- Sanderson, Grant. "Vectors – Chapter 1, Essence of Linear Algebra." *3Blue1Brown*, 5 Aug. 2016. https://www.youtube.com/watch?v=fNk_zzaMoSs.
- Sanderson, Grant. "Linear Transformations and Matrices – Chapter 3, Essence of Linear Algebra." *3Blue1Brown*, 7 Aug. 2016. https://www.youtube.com/watch?v=kYB8IZa5AuE.