**Discrete Fourier Transform**

[[Kamil Budagov](https://medium.com/@kamil2000budaqov?source=post_page-----75d9a9f37821--------------------------------)](https://medium.com/@kamil2000budaqov?source=post_page-----75d9a9f37821--------------------------------)

[Kamil Budagov](https://medium.com/@kamil2000budaqov?source=post_page-----75d9a9f37821--------------------------------)

·

[Follow](https://medium.com/m/signin?actionUrl=https%3A%2F%2Fmedium.com%2F_%2Fsubscribe%2Fuser%2F74baf91e26f&operation=register&redirect=https%3A%2F%2Fmedium.com%2F%40kamil2000budaqov%2Fdiscrete-fourier-transform-75d9a9f37821&user=Kamil+Budagov&userId=74baf91e26f&source=post_page-74baf91e26f----75d9a9f37821---------------------post_header-----------)

8 min read

·

Mar 3

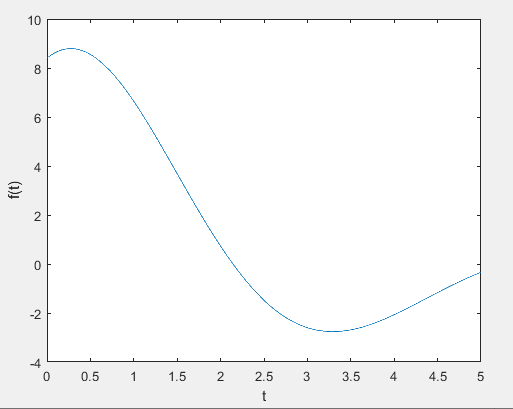
21

**Introduction**

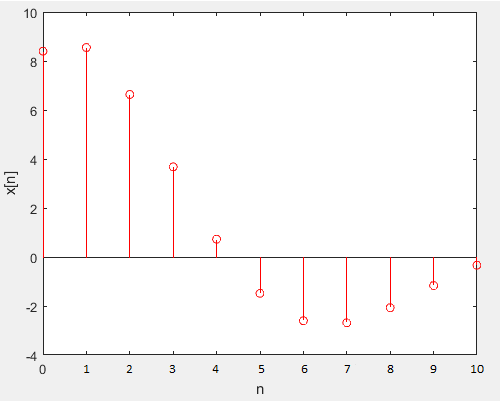
The Discrete Fourier Transform (DFT) is considered one of the most influential algorithms of all time. It is utilized in a variety of fields, such as Digital Communication, Image and Audio Compression, Signal Processing, and others. I previously discussed the [Fourier Transform](https://medium.com/@kamil2000budaqov/fourier-series-fourier-transform-ae748fcd424a) for continuous time signals and showed how it can be derived from the Fourier Series for continuous periodic functions. In this story, I will explain the same concept, but with regards to discrete time signals. To begin, we will examine discrete time signals and systems, analyze their frequency response, and then delve into a detailed explanation of DFT.

**Discrete-Time Signals**

Discrete time signals are obtained by taking measurements of continuous time signals at evenly spaced points in time and expressing those measurements as a series of numerical values. In order to be processed by a computer, all continuous signals must be converted into a discrete form.



**Continuous-time signal f(t)**

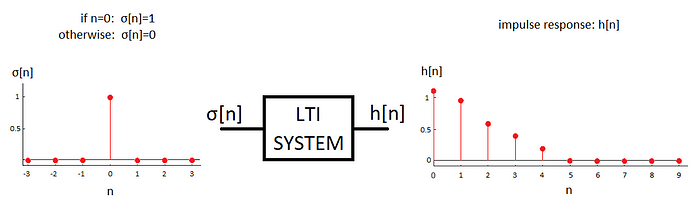


**Discrete-time signal x[n]**

In the diagrams presented above, both continuous and discrete time signals are shown. The discrete time signal is created by taking measurements of the continuous time signal at regular intervals, with a sampling frequency of 2 samples per second, or a sample interval of 0.5 seconds.

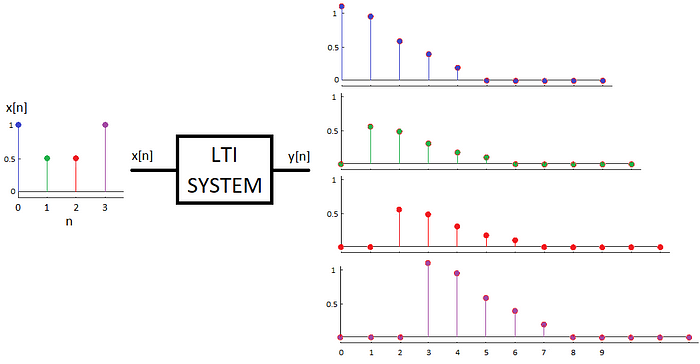
**Discrete-Time Systems**

In contrast to continuous time systems, a discrete time system processes a sequence of numerical values as its input and produces another sequence of numerical values as its output. If the system is considered linear and time-invariant (LTI), we can examine how it responds to any input signal. The behavior of an LTI system can be determined by analyzing its response to a unit impulse signal. By knowing the response of the LTI system to a unit impulse function, we can calculate its response to any other input signal by performing a convolution operation that utilizes the linearity and time-invariance characteristics of the LTI system.



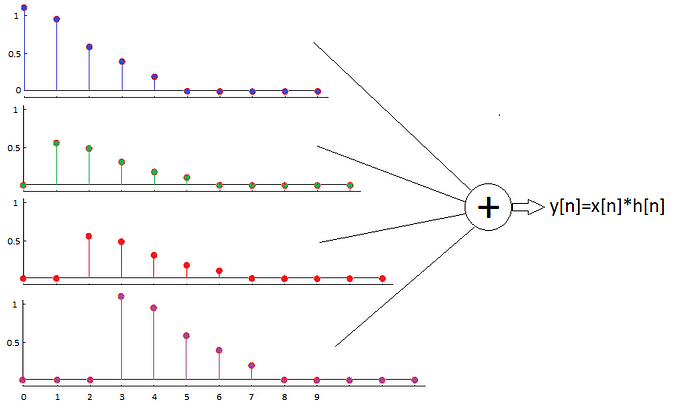
**Impulse response of LTI system**

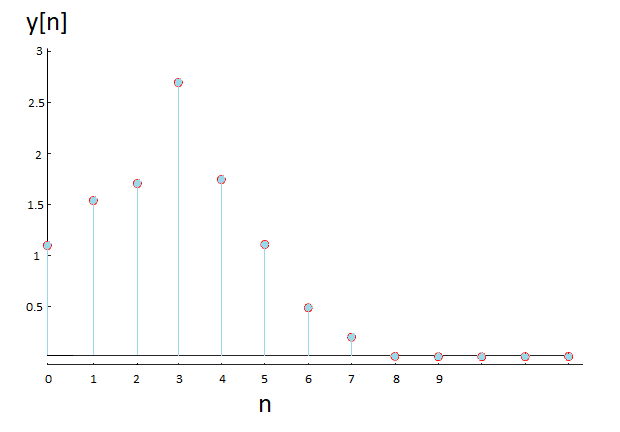
If we know h[n] (the impulse response of the system), then output for any x[n] can be calculated by a convolution operation as follows:



**Outputs for each impulse in x[n]**

According to the principles of linearity and time-invariance, when a signal x[n] is passed through a linear time-invariant system, each impulse in x[n] will result in a scaled and delayed version of the system’s impulse response h[n]. The overall output of the system is obtained by combining these scaled and delayed responses through the principle of superposition.





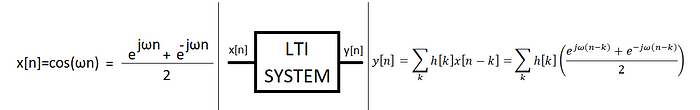
**Output of LTI system**

The output of a linear time-invariant system can be mathematically represented as the convolution of the input signal x[n] with the system’s impulse response h[n], which is expressed as follows: y[n] = x[n] \* h[n]. This results in the same output y [n] as shown in the above diagram.

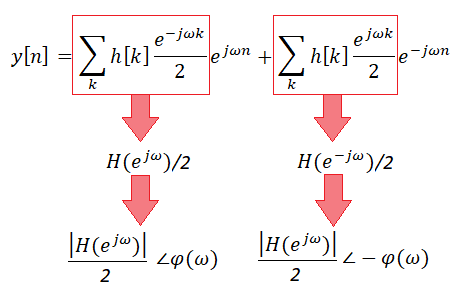


**Frequency response of LTI system**

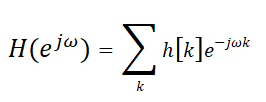
So far, we have discovered that by determining the impulse response of a system, we can determine its output for any input signal x[n] through a convolution operation. Now, let’s examine how a linear time-invariant system reacts to a sinusoidal input. Using Euler’s formula, we can express cos(ωn) as a combination of exponentials. By substituting the given input for x [n], we obtain the following outcome:



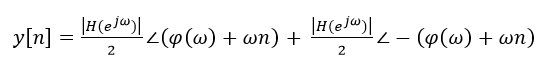
With a simple rearrangement, we can demonstrate that the output can be expressed as the sum of two conjugate complex terms.



where:



By multiplying two complex numbers in phasor form (where the magnitudes are multiplied and the phases are added), we will obtain the following result:



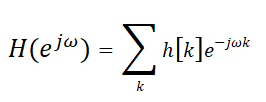
If we further extend the phasor representation, the sine terms will cancel each other out, resulting in the final output being:



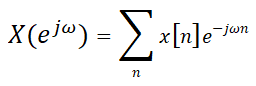
What is noteworthy in the result is that only the magnitude and phase of the input ( x [n]=cos ωn) change, but the frequency remains constant. This property of linear time-invariant systems is referred to as "sinusoidal fidelity."

**Discrete-time Fourier Transform**

So far, we have derived the following formula, which is calculated using the impulse response h[n] and represents how the different frequencies are affected by the system.



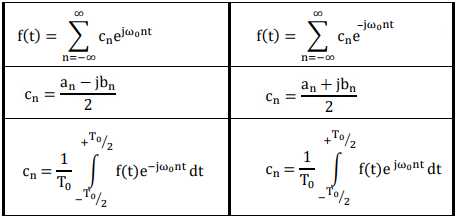
Since the impulse response of the system can be any series (x[n]) of numeric values, we can write this formula for any discrete time signal as follows:



This formula is also referred to as the Discrete-time Fourier Transform of x[n]. What is important to note in the result is that the Discrete-time Fourier Transform of a finite-length signal x[n] is periodic with a period of 2π. This periodicity is due to the following fact:

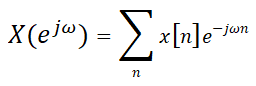


Let’s review the [previous story](https://medium.com/@kamil2000budaqov/fourier-series-fourier-transform-ae748fcd424a) and rewrite the equations for the complex Fourier series of the periodic signal f(t) with angular frequency 𝜔0 or period 𝑇0.

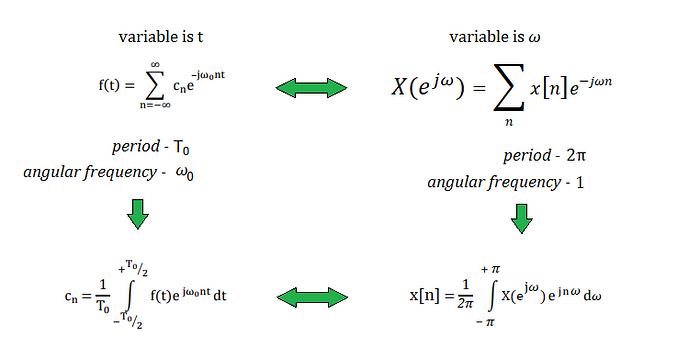


**Fourier Series representation of f(t)**

Following that, let us present a Fourier transform of the discrete-time signal x[n].



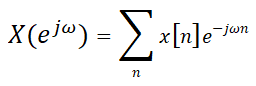
As mentioned before, the function X(e^j𝜔) exhibits periodicity in 𝜔 with a period of 2π. Therefore, it is possible to express this signal as linear combinations of complex exponentials. By comparing X and f, we can conclude that while X is a function of 𝜔 with a period of 2π, f is a function of t with a period of T0. Consequently, it seems that x[n] represents the Fourier series coefficients of the periodic function X and can be defined as follows:



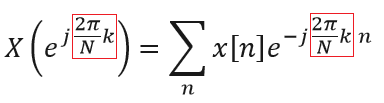
**Fourier Series Coefficients for f(t) and X(e^j𝜔)**

**Discrete Fourier Series**

So far, we have defined the discrete-time Fourier Transform for the finite-length signal x[n], which is calculated according to the following formula:

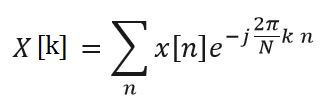


It was also demonstrated that X has a period of 2π and is periodic in 𝜔. Since X is a continuous function of 𝜔, we can take samples of it using a sample interval of 2π/N, where N is the length of x[n].



**Sampled X**

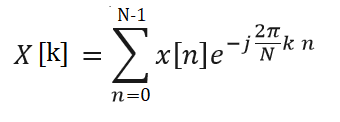
We can further simplify the above equation since X is defined only for integer multiplies of 2π/N:



It is quite easy to see that X[k] is a periodic sequence with a period of N, which is because of the periodicity of a complex exponential. Therefore, we can say that X[k] = X[k+N] or X[k] = X[k+rN], where r can be any integer.

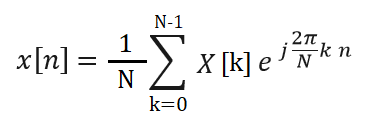
**Discrete Fourier Transform**

It should be emphasized that x[n] is a signal that is nonzero only for a limited interval. (n=0,1,….,N-1). Therefore, let us write the above formula more explicitly as follows:



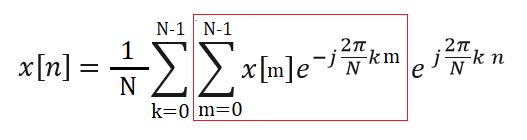
**DFT**

The above equation enables us to compute the Discrete Fourier Transform of a signal x[n], which has a length of N. *By using N samples of X[k] over one period [k = 0:N-1], it is possible to reconstruct the original signal x[n] through a process called Inverse Discrete Fourier Transform (IDFT)*. The formula for IDFT is as follows:

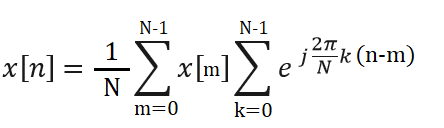


**IDFT**

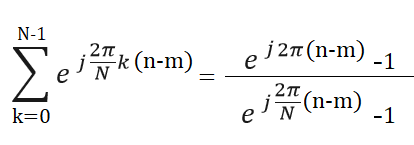
One approach to validate the IDFT formula involves substituting X[k] directly into the IDFT equation, which can be executed as follows: (it should be noted that replacing n with another letter (m) is completely acceptable in this case)



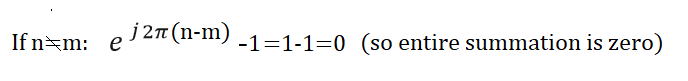
We can get the following formula for x [n] by rearranging different terms:



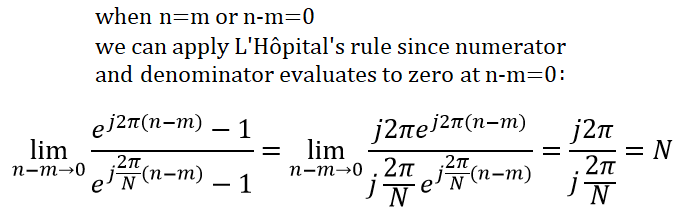
By applying the formula for the n-th sum of a geometric series, we can show that the second sum:



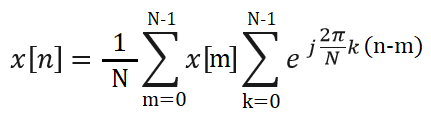
will be equal to zero when m! = n:



and will only evaluate to N when m = n:

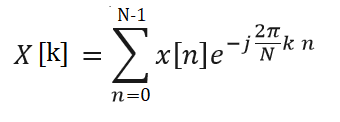


We have demonstrated that the summation will only yield a non-zero result when m equals n, thereby confirming that the right-hand side is indeed equivalent to x [n].



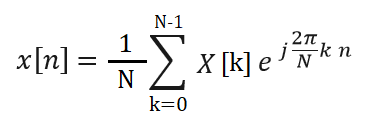
Before considering an example, let us summarize two important equations that govern the Discrete Fourier Transform:

***For k=0,….N-1***



**DFT**

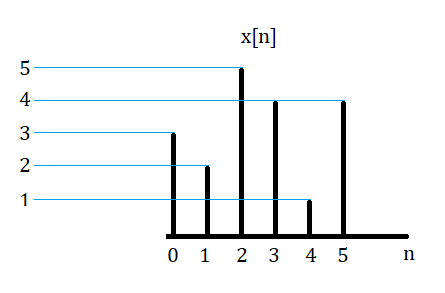
***For n=0,….N-1***



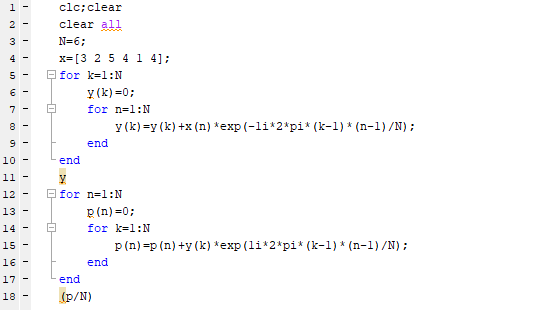
**IDFT**

**Example:**

Let us calculate Discrete Fourier coefficients for the following sequence, and then use these coefficients to rebuild the signal x[n]:



Using the following Matlab code, we can calculate coefficients and reconstruct x[n] from them as follows: It should be noted that the first two for loops serve to compute frequency coefficients, while the last two help us to construct x[n] from those coefficients.



Here are the obtained DFT coefficients:



and reconstructed x[n]:



Since imaginary parts of the reconstructed vector are zero, we can say that it is the same signal as the original x [n].

**Conclusion**

In this story, we clarified how the DFT equation can be derived from the [Fourier Series](https://medium.com/@kamil2000budaqov/fourier-series-fourier-transform-ae748fcd424a), as well as the proof of the IDFT equation, which allows for the representation of x[n] as a sum of different frequencies. To sum up, DFT is a very useful tool in digital signal processing because it enables us to analyze signals in the frequency domain. By applying DFT, we obtain the frequency spectrum of a digital signal and can extract useful information about the signal and perform operations such as filtering, modulation, and compression. It has many applications that range from audio and image processing to telecommunication and finance. Therefore, it is necessary to understand DFT for anyone working with digital signals.