

Basics of Noise and Its Measurements
Prof. Nachiketa Tiwari
Department of Mechanical Engineering
Indian Institute of Technology, Kanpur

Lecture - 35
FFT and Inverse DFT

Welcome to Basics of Noise and its Measurements. Today is the 5th day of this particular week, and we have been discussing over this entire week is the concept of Discrete Fourier Transform and we will continue this discussion today.

What we will cover today are 2 specific topics; the first one is FFT - Fast Fourier Transform and then the second concept we are going to discuss today is Inverse DFT. So we will start with FFT.

(Refer Slide Time: 00:37)

The image shows a whiteboard with handwritten mathematical definitions and notes. At the top, it says 'FFT = FAST FOURIER TRANSFORM'. Below that, the Discrete Fourier Transform (DFT) is defined as $X_k = \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi n k}{N}}$. This is then expanded using Euler's formula to $X_k = \sum_{n=0}^{N-1} x_n \left[\cos\left(\frac{2\pi n k}{N}\right) - i \sin\left(\frac{2\pi n k}{N}\right) \right]$. A note indicates 'N OUTPUTS $\rightarrow X_0, X_1, \dots, X_{N-1}$ '. Another note states 'Each output is a sum of N terms.' and finally, it concludes with ' $O(N^2)$ computations'.

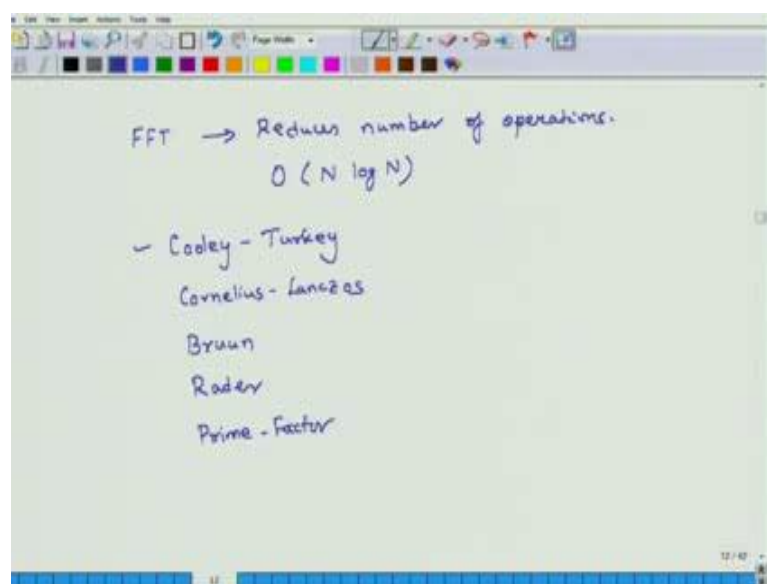
So, DFT was Discrete Fourier Transform and FFT is transformed for Fast Fourier Transform. Most of the times, when you hear in industry or in signal analysis you do not hear DFT rather you hear FFT. We had seen that the DFT of any signal $x(t)$ is given by this relation $X(k)$ equals or we had also explained it as expanded it as

Student: (Refer Time: 02:15)

You are right $2 \pi n k$ over N times i . This was $x_n \cos$ of $2 \pi n k$ over capital N plus i sine of $2 \pi n k$ over capital N . And there is a negative here and there is i here and this entire thing is in parentheses. This is the discrete fourier transform for a function x_n . In this function we have N outputs. What are those N outputs? X_0, x_1, \dots, x_{N-1} and then each output, what is it? In each output I have N sums, is a sum of N terms. It is a sum of N terms. Then to compute each term you have to find out its cosine and sin and so on and so forth.

So, essentially, when you do this whole computation for N different terms you have to do something like N^2 computations. And it is not an exact N^2 thing, so it is something whose order is something like N^2 computations. It is a very intensive process. So, that is what FFT is all about. FFT is basically same. You do using this regular approach and you will end up by having a number of computations will be something with an order of N^2 . If you do it in a faster way and a smarter way, the formula is the same that mathematics does not change, then you call it a fast fourier transform this is the FFT, because it is faster.

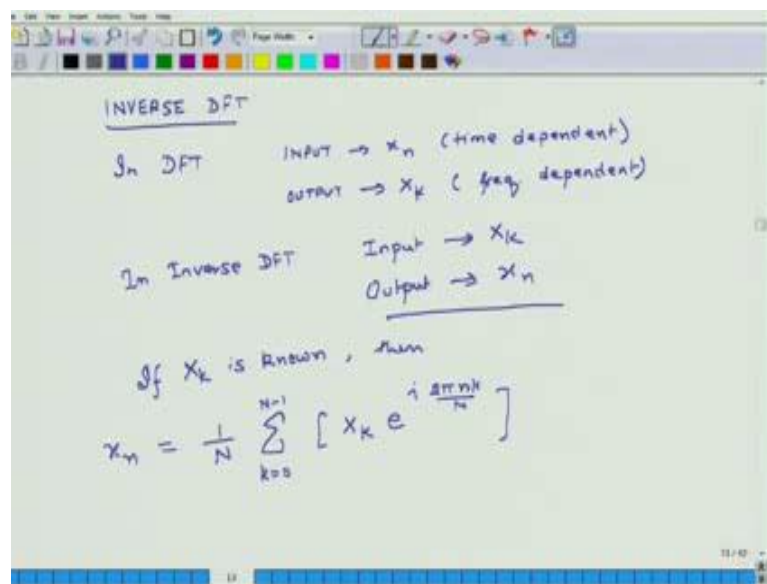
(Refer Slide Time: 05:10)



So, FFT or fast fourier transform it reduces number of operations. And typically it requires not the computations is not N square, but order is something like N log of N, if it is somewhere closer to this then you say it is fast, it is fast fourier transform. There are several algorithms to do that, and one very popular algorithm is by called Cooley-Turkey algorithm. This is this is very popular Cooley- Turkey algorithm. There is another algorithm not that much popular Cornelius-Lanczos. And then there are other algorithms Bruun, Rader, Prime- Factor and so on and so forth.

But the point is that fast fourier transform is essentially same as DFT. It is a faster way to do these computations and if you do it faster we call it fast fourier transform. Most of the algorithms which are being used by standard course like matlab, lathmatica and things like that I think they use FFT approach. So, that covers the topic of FFT.

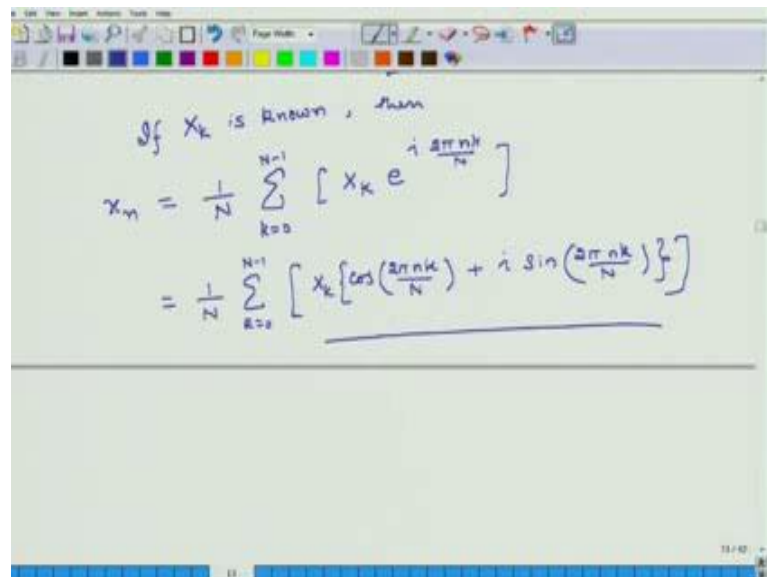
(Refer Slide Time: 07:04)



Next topic I am going to cover is inverse FFT actually inverse DFT, but inverse FFT is also some name very similar. When we do DFT our aim is to get X k from x n. In DFT input is x n which is time dependent and output is X k which is frequency dependent. In inverse DFT our input is X k and output is x n. And how do we do that? If I know if X k is known then x n equals 1 over capital N, so excuse me, this time I will be adding upon the (Refer Time: 08:55) k because I have X k's. X k e to the power of i 2 pi n k over N.

So, formula is pretty much the same except instead of minus i you put plus i and instead of adding on the (Refer Time: 09:22) of n you here are adding over k. But k and n, I mean you can interchange them it does not matter.

(Refer Slide Time: 09:40)



The image shows a handwritten derivation of the inverse Discrete Fourier Transform (IDFT) formula. At the top, it says "If X_k is known, then". Below this, the formula is written as:

$$x_n = \frac{1}{N} \sum_{k=0}^{N-1} [X_k e^{i \frac{2\pi nk}{N}}]$$

The second line of the derivation shows the exponential term expanded using Euler's formula:

$$= \frac{1}{N} \sum_{k=0}^{N-1} [X_k \{ \cos(\frac{2\pi nk}{N}) + i \sin(\frac{2\pi nk}{N}) \}]$$

The entire derivation is written in blue ink on a white background, with a toolbar visible at the top and a blue bar at the bottom.

I will just make it available in trigonometric format. So, it is 1 over N, k is equal to 0 to N minus 1 X k cosine of 2 pi n k over N plus i sin of 2 pi and k over N. So that is my inverse transform. What does it tell me, that if X_k is known - if I know all the frequency components, and then using those frequency components I can get back my time series data. And if I have time series data using DFT I can get frequency data, from frequency data I can back calculate time series data using this thing.

So, that is all what I wanted to cover in this lecture. In the next lecture we will change tracks and we will go back to instrumentation and about what kind of considerations do we have to think about as we select different types of instruments for noise related measurements. So, thank you very much and we will meet once again tomorrow.

Thanks.