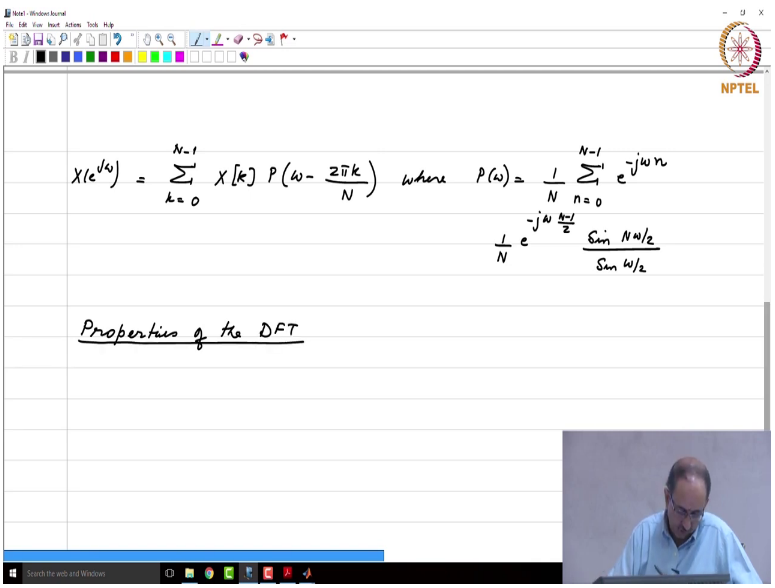
Digital Signal Processing

Prof. C.S. Ramalingam
Department Electrical Engineering
Indian Institute of Technology, Madras

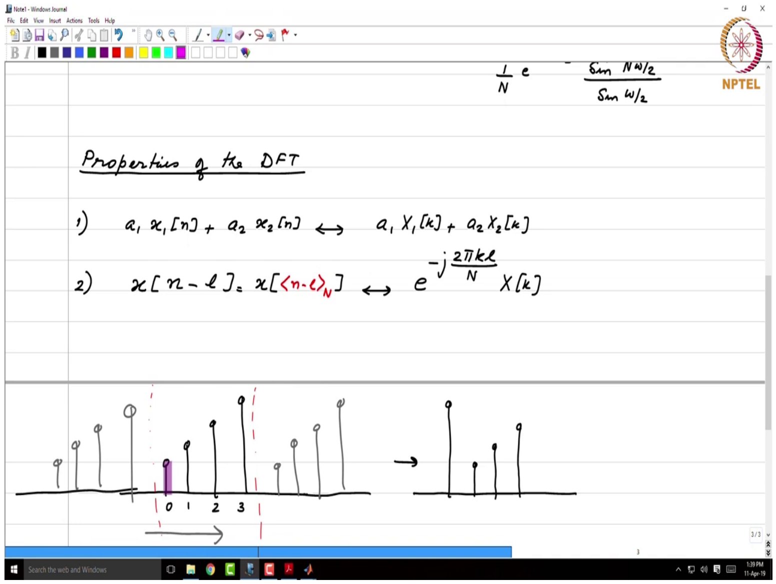
Lecture 79:
The Discrete Fourier Transform (3)
- Properties of the DFT

(Refer Slide Time: 00:16)



Now, we can look at the Properties of the DFT. So, these are very simple. Once you know the properties of the DTFT; properties of the DFT are very similar. Some of the differences will point out if there are any differences.

(Refer Slide Time: 00:48)

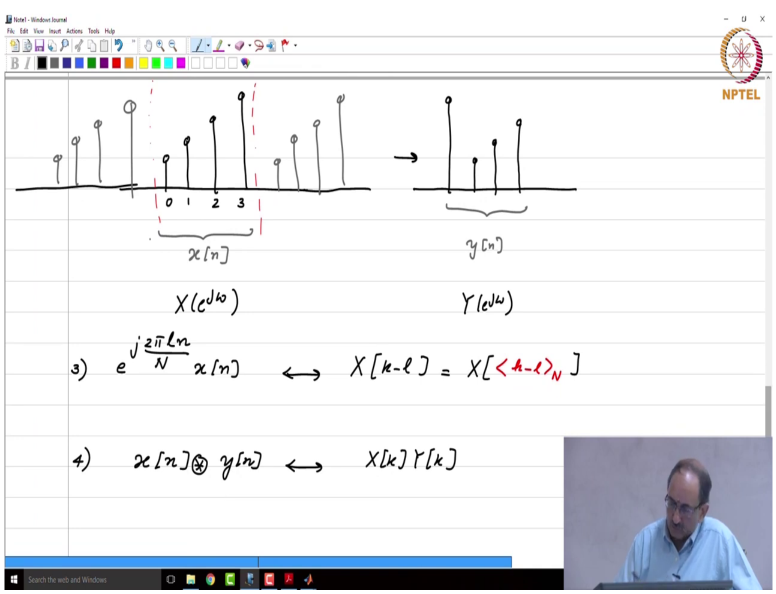


So, linearity follows from the linearity of the transform. So, this is $x[n-l] \longleftrightarrow e^{-j2\pi kl/N}X[k]$. But note that, remember, I had mentioned because both the time domain signal as well as the transform they are periodic, you need to concern yourself only to indices in the range 0 to $N-1$, where N is a periodicity. Therefore, any index is really index mod N therefore, this is really $x[< n-l >_N]$.

So, what does this mean? So, if we had a sequence like this. If you shift it by one sample to the right, it will come like this because this is really periodically period N. Therefore, even though you have samples like this, for a given sequence from 0 to $N-1$, really the implied sequence overall n is the periodic version. Therefore, you have this; this is the periodic repetition on this side, also you have the periodic repetition right.

Therefore, when you move this by one sample to the left; this samples comes here, this sample comes here and so on; this sample will come here.

(Refer Slide Time: 04:45)



So, which is what you are seeing here. And right shift by 1 in this case is same as left shift by 3. You can shift this by $3, 1, 2$; this sample will come here. If you call this as $x[n]$ and if you call this as $y[n]$, there is no connection between $X(e^{j\omega})$ and $Y(e^{j\omega})$.

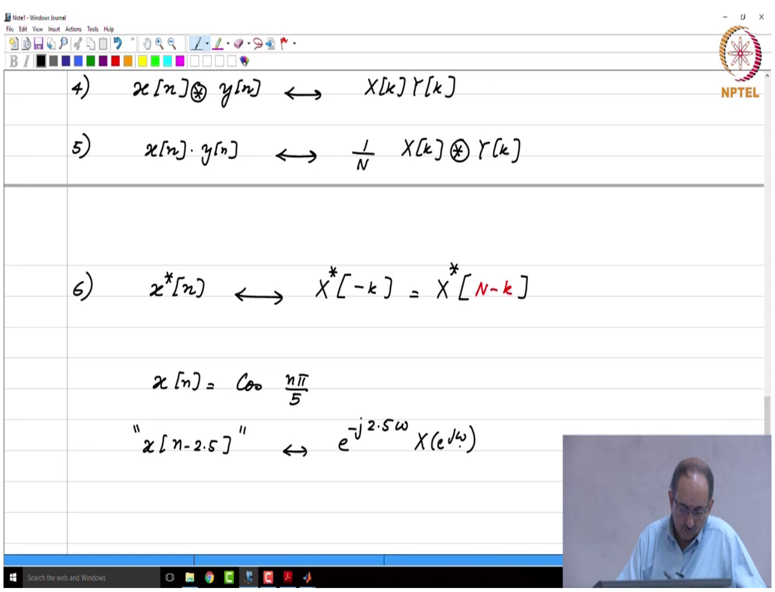
However, the DTFTs are related like this. When I say this is $x[n]$, I mean $x[n]$ is are these samples from 0 to 3 and they are 0 otherwise. Similarly, when I say $y[n]$ and $Y(e^{j\omega})$, I mean $y[n]$ is this sequence and it is 0 outside. Therefore, X and Y the transforms; $X(e^{j\omega})$ and $Y(e^{j\omega})$ are unrelated, but their DFTs are, because moment you bring this into the DFT framework; you have to make them periodic and then the relationship immediately follows.

The counterpart to the shift in the time domain is modulation; we have $e^{j2\pi ln/N}$ this is nothing, but $X[< k-l >_N]$. So, this is the counter part of, you can multiply by $e^{j\omega_0 n}$; the DTFT will be $X(\omega - \omega_0)$. So, that is exactly similar to what is happening here, after all you are going to sample the modulated signals spectrum.

So, this is what you will get. But again, the transform is also periodic and hence this is really $< k-l >_N$. If you convolve in the time domain, you will multiply in the frequency domain except that remember, both X and Y are periodic sequences because here we are talking about the DFT framework. And when we talked about multiplication in the time domain; in the DTFT context, if you recall, in the transform domain, it was convolution. But, we mentioned that that convolution is not your usual convolution, but circular convolution.

Because, it is both the transforms are periodic and there I made the statement that we will talk more about circular convolution later and I had briefly mentioned circular convolution both for continuous time periodic signals as well as discrete time. So, when I said, we will discuss it later, we have reached that point. So, this is really circular convolution. Once you are done with the properties, we will talk about circular convolution in more detail.

(Refer Slide Time: 08:34)

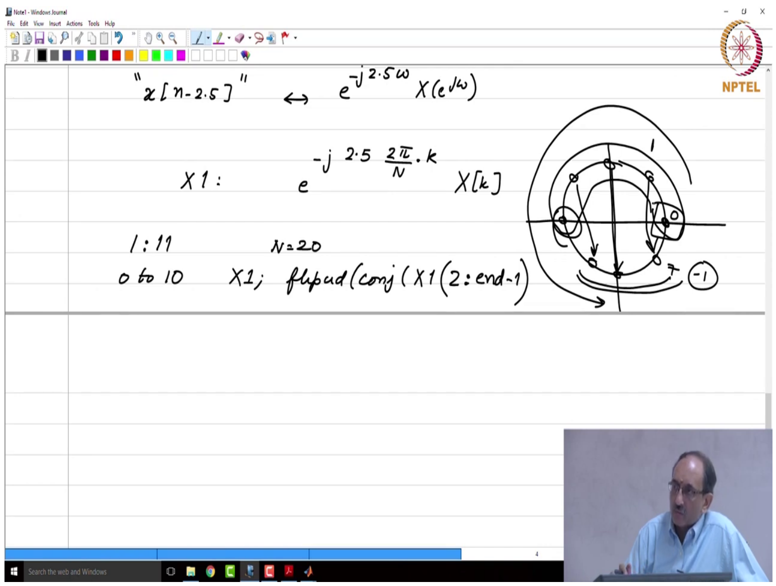


So, if you convolve in the time domain, you multiply their transforms, convolution is now circular rather than linear. Similarly, if you multiply in the time domain, you convolve in the other domain. The other domain also is periodic and hence the convolution also has to be periodic convolution or circular convolution. $x^*[n]$ remember, the corresponding DTFT property was $x^*[n]$ had DTFT $X^*(e^{-j\omega})$.

Therefore, you can expect this to be $X^*[-k]$, but immediately $X^*[-k]$ is nothing but $X^*[\,]$, remember, every index is index mod N. Therefore, $X^*[-k]$ is really $X^*[N-k]$. And if you recall what I had done in the matlab code for fractional shift, this is exactly what had done. I had a sequence $x[n]$ and this was $\cos(n\pi/5)$ and we needed to shift this by 2.5 samples which means in the transform domain, you had to do $e^{-j2.5\omega}X(e^{j\omega})$.

And if you go back and look at the code, on a machine implementation, you would work with the DFT because you cannot work with the DTFT and the DFT is nothing but samples of the DTFT.

(Refer Slide Time: 11:08)



Therefore, $e^{-j2.5(\;)}$ stays as it is. For ω, you have to discretize it. You replace ω by ω_k; ω_k is $k2\pi/N$. Therefore, this becomes $(2\pi/N)k$. So, this was exactly the code that was implemented in matlab and then you replace this with $X[k]$ and then what we did was, we restricted this only to the first half.

Remember, we computed this over 20 samples. Therefore, we computed the 20 point DFT and we applied this only to the upper half of the transform. Therefore, we went from 1 to 11 because N was 20 in this case. Therefore, you went from 0 to 10. In matlab index, you have to go from 1 to 11; therefore, we restricted this equation to the first half of the top half of the unit circle. And then when we form the inverse transform, we form this vector $X1$ and then what we did was, we took $X1$ as it is.

To compute the time domain signal, you needed to compute the inverse DFT. For inverse DFT, you needed to form the 20 point DFT and then compute the inverse DFT using the $ifft$ command. To complete the whole transform, this was what was there based on this equation. You needed to supply the missing half which is the bottom part corresponds to sampling of this unit circle in the bottom part. There we used this property; $X^*[N-k]$. So, remember, when you sample this on the unit circle, you will take samples like this.

So, we have computed these samples. You needed to, these are the ones that are missing. Sample at 0, we need not repeat because that should occur only once. Sample at $N/2$ also should not be repeated. Therefore, what we did was, we took $X1$ and then we did flip u d; $flipud$ will upside down. So, if it is a column vector, you will do flipud. If it is a row vector, you will do $fliplr$, left to right. Then, we did conjugate.

So, this $flipud$ is, because we are dealing with $N - k$. Remember, this index is 0, this index is 1. For the 7 point case, this index is 7, but really this is also the same as -1. And remember, so we are going like this therefore, we have to take this vector and then this has to come here, this has to come here and this has to come here. So, that is why you need this $flipud$ and then you have to conjugate because this $X^*[N - k]$. And then, you need to take the vector 1 and then you need to start off with the second sample.

Because, the first sample corresponds to index $k = 0$, which you do not want to repeat. The last sample also, you do not want to repeat. Remember, $X1$ is the first half; it is not the 20 point, it is the 11 point sequence. That is, we have taken the 20 samples and then we have gone from 0 to 10 or index 1 to 11. Therefore, this is actually 2 to $end - 1$.

So, 2 gets rid of the $k = 0$ index which we do not want. You do not want the end index also, because end index would correspond to $N/2$. So, this now explains the matlab code that was there in your notes for reproducing this fractional 2.5 delay. So, all the sequence of steps comes from this; $flipud(conj(X1(2 : end - 1)))$. If this were an odd length sequence, instead of $end - 1$, you would go to up to end because you would not have a sample at $N/2$.

So, there is no question of repeating this. Only when N is even, would you go to $end - 1$ because you do not want to repeat this guy. Any case, you never want to repeat this, because that is $k = 0$ sample. So, after having now learnt DFT, now you can appreciate the matlab code that was part of this example. The main things are, you replace ω by its sampled value and then you only need to compute this for the first part, namely in the range 0 to π. From π to 2π, you can repeat the information that you form in the range 0 to π.