Digital Signal Processing

Prof. S. C. Dutta Roy

Department of Electrical Engineering

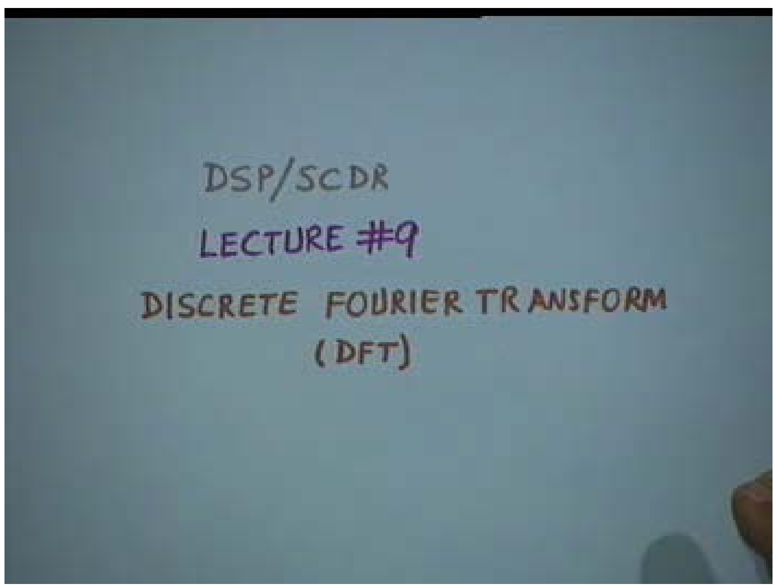
Indian Institute of Technology, Delhi

Lecture - 9

Discrete Fourier Transform (DFT)

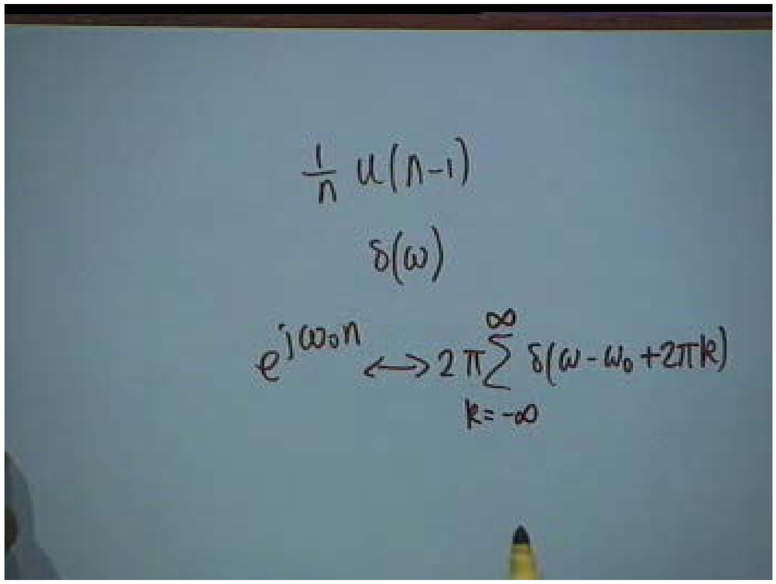
In the 9th lecture on DSP, our topic today is Discrete Fourier Transform which we abbreviate as DFT.

(Refer Slide Time: 01:07 – 01:09)



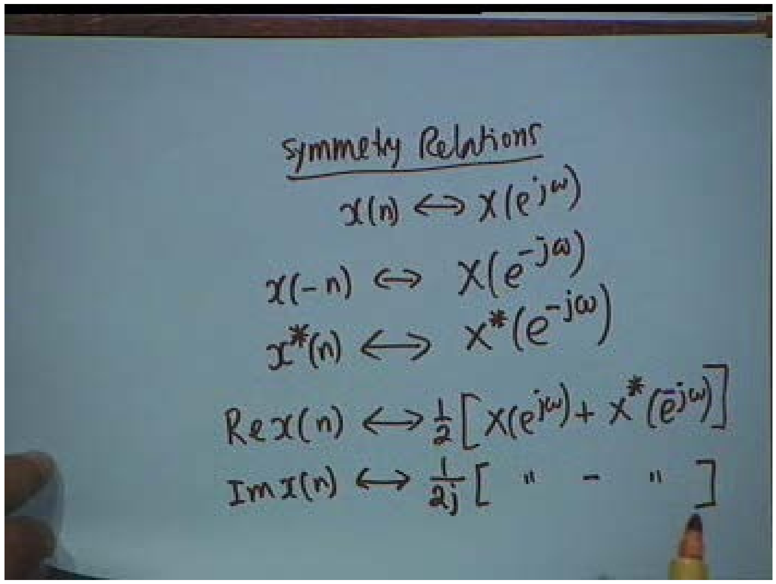
We have abbreviated the Discrete Time Fourier Transform simply as FT in order to avoid confusion. In the 8th lecture, we considered the question of convergence or existence of Fourier Transform and we said that absolute summability of the sequence is a sufficient condition but not a necessary condition. Similarly square summability of a sequence is also a sufficient condition. There are sequences which are square summable but not absolutely summable and the example was (1/n) u(n – 1).

(Refer Slide Time: 01:55 – 02:38)



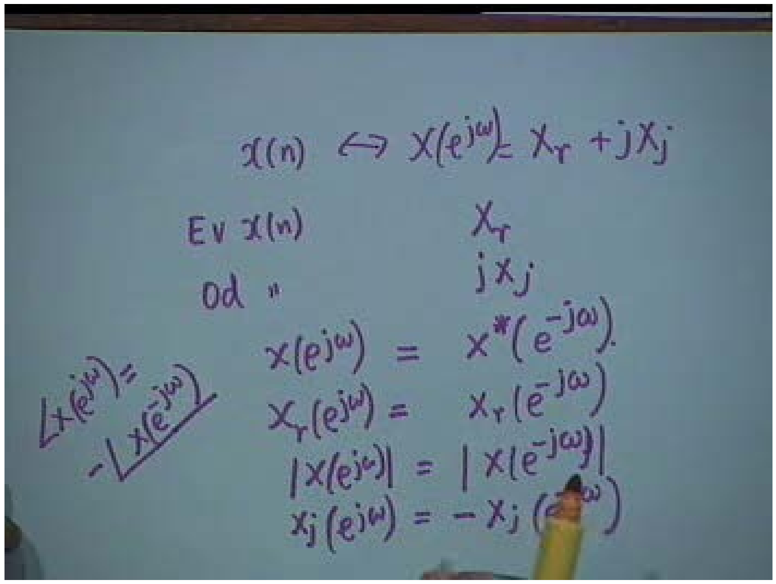
Then we said that there are sequences which are neither square summable nor absolutely summable but their Fourier Transform can be defined in terms of the analog delta function. And as an example, we took the Fourier Transform of exp (j omega$_0$ n) and we showed that it is 2pi summation delta (omega – omega0 + 2pik) where k goes from – infinity to + infinity. We also gave you the basic table of Fourier Transforms, where the basic functions are delta n, u(n), ($e^{j\ n\ omega0}$), and alpha^n times u(n). If you know these four transforms, then you can find almost all Fourier Transforms of all familiar sequences. Then we took the example of the up sampler and we showed that the spectrum gets compressed. I would suggest that you carry out the example of down sampler and find out what happens to its spectrum. Next, we took an example of an inverse Fourier Transform which does not require integration. A bit of thought reveals that it can be done very simply by expanding cosine and sine in terms of the Euler relations. We discussed some properties of Fourier Transform, such as Linearity, Time Shift, Frequency Shift, Convolution, Modulation, which corresponds to multiplication in the time domain, and finally the Parseval’s Relation which has an interpretation in terms of energy. That is, energy in the time domain is equal to the energy in the frequency domain. And then we talked about the symmetry relations.

(Refer Slide Time: 04:22 – 05:59)



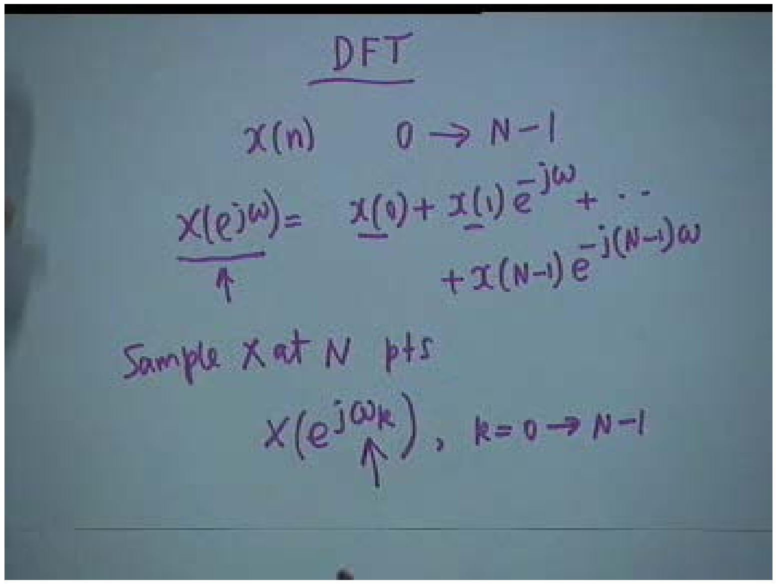
One symmetry relation is that is if x (n) has the Fourier transform $X(e^{jomega})$, then x(– n) has the FT $X(e^{-jomega})$. Also, x*(n) is the pair of $X^*(e^{-j\ omega})$. And then if x(n) is complex, then real part of x(n) is the Fourier pair of ½ [X (e^{jomega}) + $X^*(e^{-jomega})$] and the imaginary part of x(n) is the pair of [1/(2j)] [$X(e^{jomega})$ – $X^*(e^{-jomega})$]. There are other symmetry relations we want to mention casually because they are obvious.

(Refer Slide Time: 06:16 – 08:28)



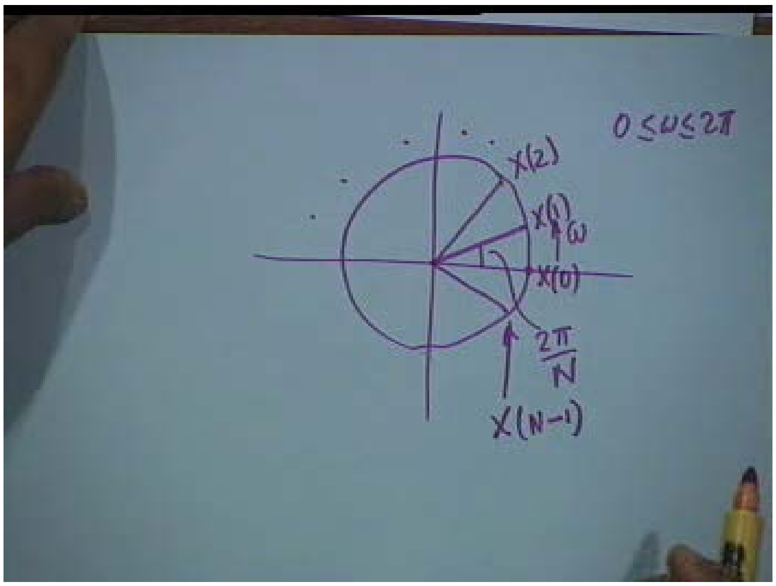
And it is that if x(n) has the Fourier Transform of $X(e^{jomega})$ which is $X_r + jX_j$, then for the even part of x(n), the Fourier Transform would be the real part X_r. The odd part of x(n) has the Fourier Transform j times X_j. And then $X(e^{jomega})$ is the same as $X^*(e^{-jomega})$. And if I take the real part of $X(e^{j\omega})$, which is an even function, it will be the same as $X_r(e^{-jomega})$. That is, if omega changes to – omega in the real part, it does not matter. The magnitude of $X(e^{jomega})$ is also an even function and therefore this is the same as the magnitude of $X(e^{-jomega})$. Whereas, the imaginary part of $X(e^{j\omega})$ is an odd function and therefore this is $-X_j(e^{-jomega})$. And similarly, the angle of $X(e^{j\omega})$ is also an odd function and therefore if I change omega to – omega, I shall get this as equal to – angle of $X(e^{-j\,omega})$. These are the basic relations. To repeat, the real part is even, the imaginary part is odd, the magnitude is even and the angle is odd. Now let us see the Discrete Fourier Transform.

(Refer Slide Time: 08:49 – 13:09)



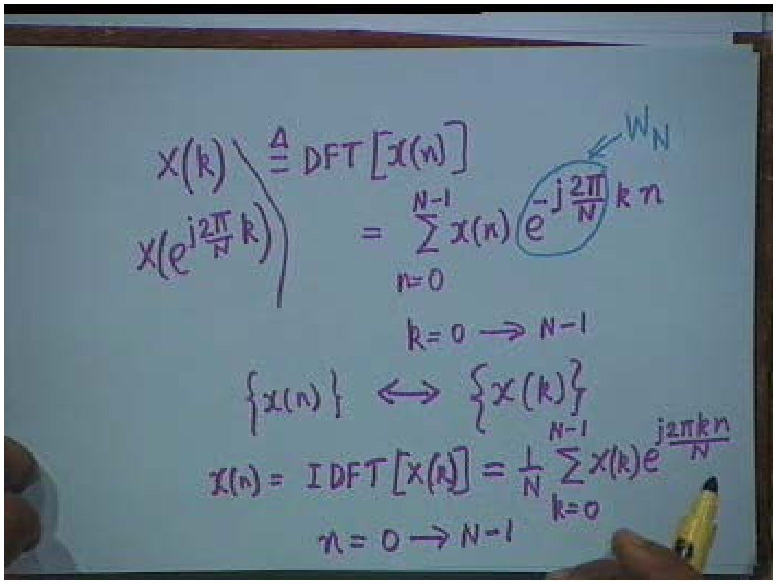
To understand the concept of discrete Fourier Transform, let us consider a sequence of finite length. Discrete Fourier Transform holds only for sequences of finite length. Suppose x(n) is of length N; then $X[e^{jomega}]$ is simply $x(0) + x(1)e^{-jomega} + etc + x(N-1)e^{-j(N-1)omega}$ where I have expanded the summation. It is a polynomial in $e^{-jomega}$ with N number of coefficients. So $X(e^{jomega})$ is completely specified if these N number of coefficients are specified. A polynomial is specified in terms of its coefficients. So, for $X(e^{jomega})$, which is a continuous function of omega, it suffices to specify N pieces of information. In the frequency domain, since FT is a one to one transformation, if we take $X(e^{jomega})$ and sample it at N number of distinct points, that should be good enough. N pieces of information are required to specify $X(e^{jomega})$. This information can be in the time domain, that is x_0, x_1,… up to x(N – 1) or it can also be in the frequency domain. In other words if I sample $X(e^{jomega})$ at N number distinct points, then we have N independent pieces of information and that should suffice to specify x(n). So we take $X(e^{jomegak})$ where k goes from 0 to N – 1. There must be N number of distinct points in the omega domain. That should completely specify $X(e^{jomega})$. That is, a part of the information is adequate to reconstruct the full information. Usually this is not possible. From only a few parts, one cannot construct the whole. If I give you 2 or 3 pieces of furniture, it requires a lot of research to find out what the total furniture is. If I simply give you two legs of a chair, they could also be parts of a cot or some other furniture. But it is possible in this instance because the function itself has only N pieces of independent information in terms of the constants x(0), x(1)… x(N – 1). So this is the basis of defining a DFT. This sampling at N number of omega points can be uniform or non uniform. Uniform sampling is preferred because this keeps life simple.

(Refer Slide Time: 13:15 – 14:45)



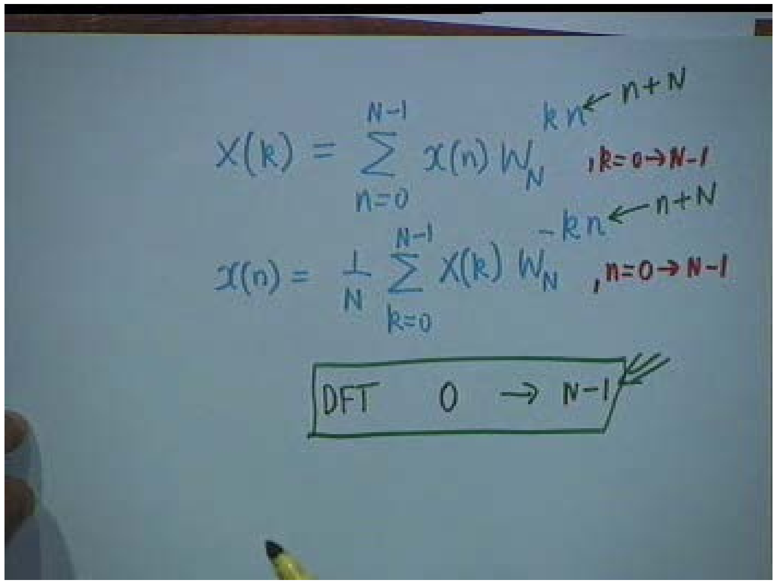
To understand this sampling, let us construct a circle where omega is 0 on the right horizontal axis and omega increases in the anticlockwise direction from 0 to 2pi. Why do we consider omega between 0 and 2pi? It is because $X(e^{jomega})$ is periodic with the period of 2pi. So within one period, if I take N number of samples, then I should take samples at every 2pi/N. The range 2pi is divided into N equal intervals. So X(0) will be my first sample, followed by X(1) at the angle 2pi/N. Then I get X(2) and so on, till I come (2pi/N) (N – 1), because the sample at X(2π) is simply X(0), because of periodicity. This picture defines what Discrete Fourier Transform is.

(Refer Slide Time: 14:50 – 18:25)



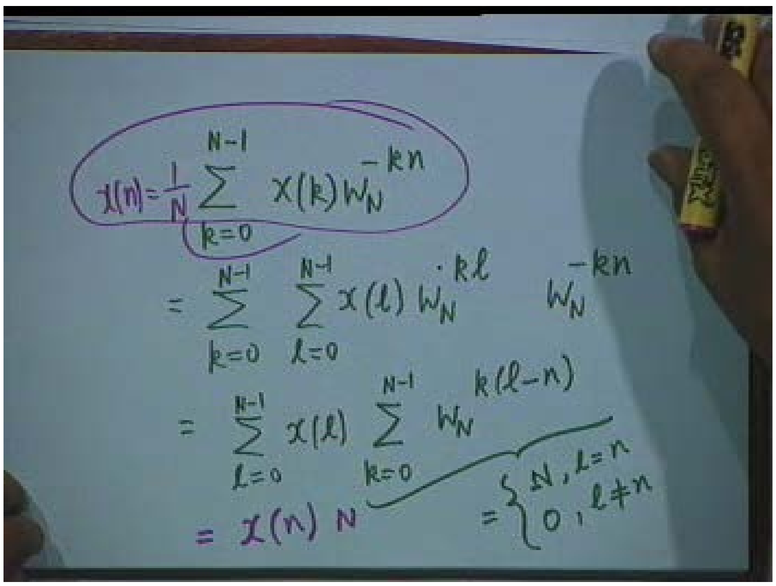
In other words, our definition of Discrete Fourier Transform is X(k), which is the k^{th} sample and this symbol is used for $X(e^{j2\pi k/N})$; it is summation (n = 0 to N – 1) x(n) $e^{-j(2\pi/N)kn}$. Here 2pi/N is the basic interval. It is the FT sampled at $\omega_k = (2\pi/N)k$. This has to be computed for k = 0 to N – 1. So the sequence x(n) which is of finite length and the sequence X(k) which is also of the same length, are one to one transformation. One to one can be shown by taking expression for the inverse DFT of X(k); this expression is not a matter of definition. It is given by (1/N) summation $X(k)e^{+j2pikn/N}$ where k now goes from 0 to N – 1, and it has to be computed for n = 0 to N – 1. This is the inverse DFT, and I repeat, it is not a matter of definition. It follows from DFT and vice versa. This is a one to one transformation and therefore going either in the forward direction or in the reverse direction should suffice. In the literature, for historical reasons, the quantity $e^{-j2pi/N}$ is usually given the symbol W_N. It is a bit unfortunate that negative sign is associated with this but this is how it has gone in the literature starting with the days of Cooley and Tukey. And therefore, we can rewrite our definition of Discrete Fourier Transform as summation n = 0 to N – 1 x(n) W_N raised to the power kn.

(Refer Slide Time: 18:33 – 22:30)



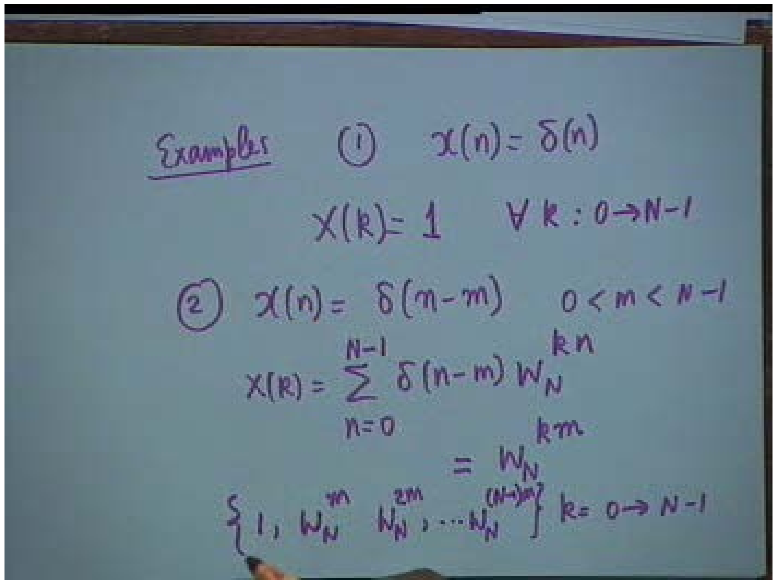
In a similar manner, the inverse DFT x(n) is (1/N) summation (k = 0 to N – 1) X (k) W_N^{-kn}. [Please distinguish between capital X and small x. Small x I write with curves, capital X is two straight lines crossing each other]. The values of k in X(k) are 0 to N – 1 and the values of n in x(n) are 0 to N – 1. This is a complete description of a finite length sequence, either in the time domain or in the frequency domain. Points to notice: in X(k), if I increment k by N, does it change? It does not change, therefore X(k) is periodic with a period of N. This is expected because X(e^{jomega}) is periodic with a period of 2pi. And what we have done is that the interval 2pi has been broken up into N equal intervals, and therefore X(k) should repeat after N number of samples. But what is surprising is that in IDFT expression also, if I change n to n + N, it does not change. So x(n) is also forced to be periodic although, originally, it was not. We did not start with a periodic x(n); we started with a finite length x(n) which does not repeat. That is, x(n) exists only from n = 0 to N – 1, but if you continue to carry out the computation, it will exist beyond N – 1. This is an advantage as well as a disadvantage. Disadvantage is that it has changed the character of x(n); it has made it periodic. But if you confine your attention to n = 0 to N – 1, there is no aliasing, there is no distortion. Therefore it is important that in DFT, our range of vision, whether in the frequency domain or in the time domain, should be from 0 to N – 1 and we cannot go beyond that. Whatever operation you do on DFT or the sequence x(n), i.e. Delay, Multiplication, Convolution etc., we cannot go beyond N – 1, and we cannot go below 0. We must confine ourselves to these limits. It is a very important point which will come up in a few seconds. The DFT inversion expression can be very easily proved. In the previous case of Fourier Transform, I did not prove it, but here let me indicate the proof.

(Refer Slide Time: 22:43 – 26:15)



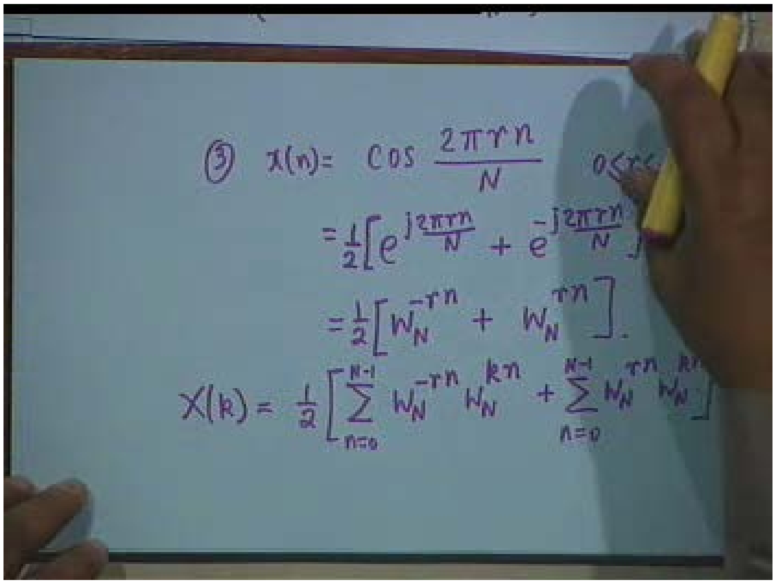
The trick is the same, as in Laplace Transform, Fourier Transform, or the Discrete Time Fourier Transform. What we do is to replace the X(k) by its value in terms of the definition of DFT. That is, we write this as summation (ℓ = 0 to N – 1) $x(\ell)W_N^{+k\ell}$. Since n was already there in the IDFT expression, therefore we use another dummy variable ℓ. Now these two summations are independent of each other and therefore we can interchange. So if we first do the summation ℓ = 0 to N – 1, then we combine the two W_N's and get summation (k = 0 to N – 1) $W_n^{k(\ell - n)}$. If you take the interpretation of W_N as $e^{-j2pi/N}$ and sum this up then you will see very simply that it is equal to 1 if $\ell = n$, summed up N times; if $\ell \neq n$, then the summation is 0. This is the outline of the proof and you can carry out the algebra. So, this is a proof of the inversion formula. Let us take some examples.

(Refer Slide Time: 26:33 – 29:19)



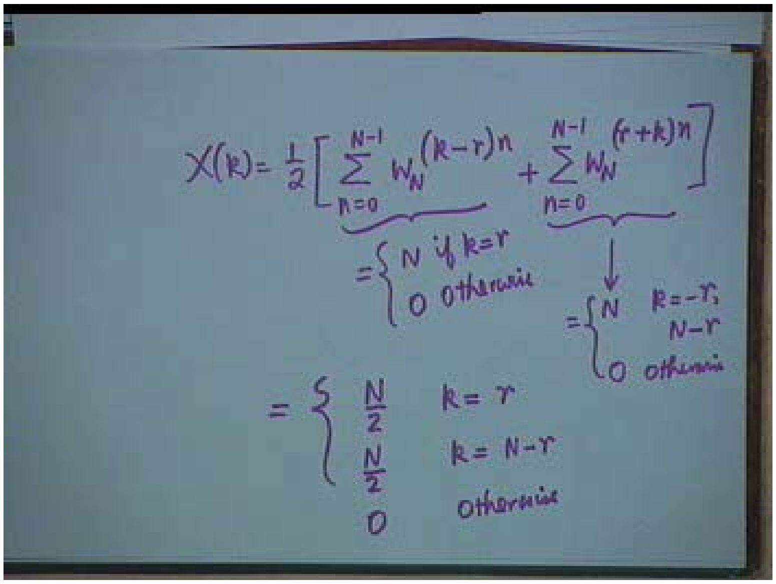
The first example we take is the simplest sequence, delta (n). This is a one point sequence. Suppose I want to find out its N-point DFT; then what do we do? We add N – 1 0s. And then if you put this x(n) in this summation, since delta n exists only at n = 0, therefore X(k) will be simply equal to 1. It is true for all k between 0 to N – 1. So N point DFT is simply 1, 1, 1, 1,… up to N – 1 samples. Second example: suppose our impulse is delayed i.e. our x(n) is delta (n – m), and we want to compute its N point DFT, then obviously the delay m has to be between 0 and N – 1. If it goes out then we are not concerned. We cannot see anything beyond N – 1. So if we apply the definition, then you get summation delta (n – m) W_N raised to the power kn, n = 0 to N – 1. But this delta function exists only at n = m and at every other point it is 0. Therefore the summation reduces to a single term W_N raised to the power km for all k between 0 and N – 1. So the sequence X(k) is W_N raised to the power m, W_N raised to the power 2m, and so on, up to W_N raised to the power (N – 1)m; this is the sequence. Put m = 0 here and you get 1, 1, 1, 1,…

(Refer Slide Time: 29:30 – 32:38)



The third example is interesting.

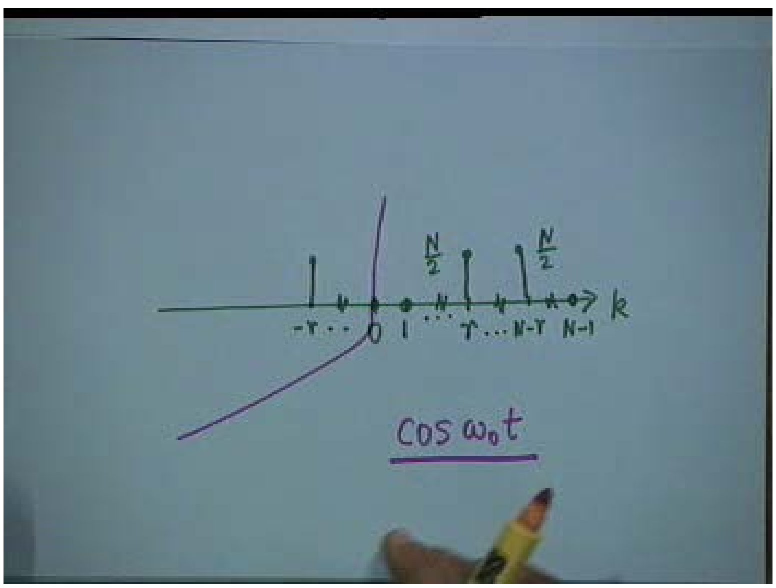
(Refer Slide Time: 32:42 – 34:37)



Third example is x(n) = cosine of 2pirn/N, where r is a given integer, which has to be between 0 and N – 1. Now I find X(k) by Euler's Relation where we put x(n) as half of (e to the power j

2pirn/N + e to the power – j 2pirn/N), that is half of [W_N raised to the power – rn plus W_N raised to the power rn]. Once we have done this, finding X(k) becomes very easy; we are able to do this because DFT operation is linear. We broke this sequence into two and I find the X(k) as the sum of the two transforms. So this becomes X(k) = (1/2) [summation (n = 0 to N – 1) W_N raised to the power (k – r)n + summation (n = 0 to N – 1) W_N raised to the power (r + k)n. The first summation is equal to N if k = r, or 0 otherwise. In a similar manner the second summation is equal to N if k = – r; now that creates a problem. We should not go into the forbidden region. But what value exists at k = – r should also exist at k = N – r because it is periodic. Therefore in the range of vision, X(k) = N/2 for k = r and N/2 for k = N – r, and 0 otherwise. In other words, the digital signal cosine 2pirn/N has only two non zero samples in its DFT. Let us see the plot.

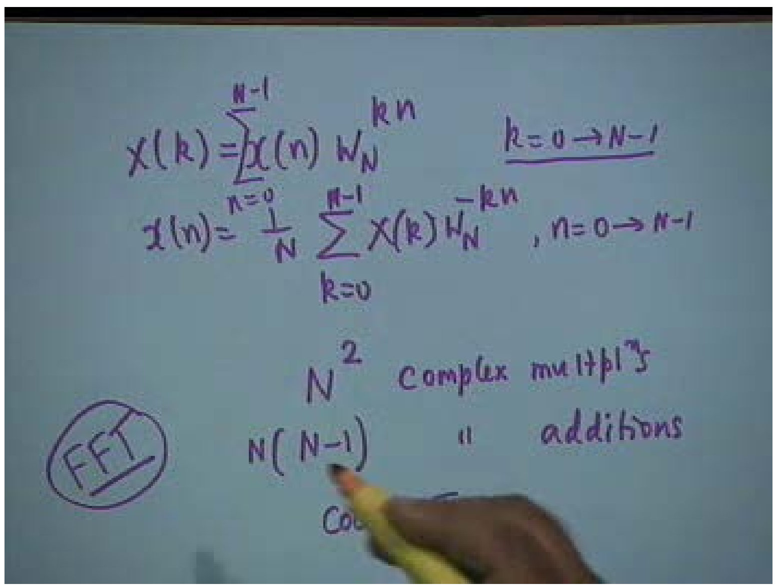
(Refer Slide Time: 34:48 – 37:08)



I have only two samples at k = r and k = N – r, and all others are 0. At k = r the value is N/2 and it is N/2 at k = N – r also. Had you been allowed to look beyond, or to cross the forbidden region, then we should have another sample N/2 here at k = – r. So the basic DFT consists of two samples of value N/2 at k = r at – r. Although the latter is a forbidden region, I brought this into view for two reasons. One is that X(k) is periodic. And the other thing is that cosine signal has only two samples of non zero value and these are at r and – r. If you remember, cosine of omega_0

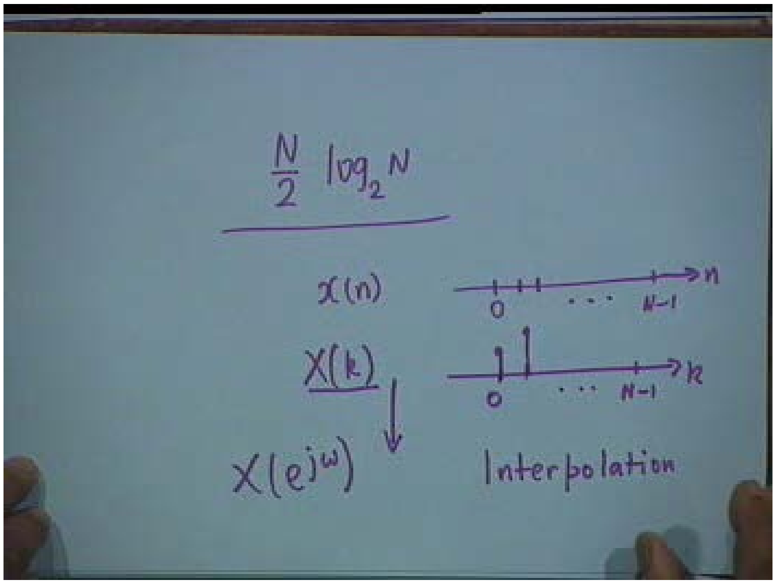
t, spectrum has only two non – zero samples and both of them are delta functions at + omega 0 and – omega 0. This is a reflection of the same fact. What we have done is that we have taken the discrete equivalent of this analog signal and instead of impulses of infinite height, we get finite height. This is the simplification afforded by discretizing the process. Our next concern would be the number of operations.

(Refer Slide Time: 37:23 – 41:51)



If you want to compute X(k), each X(k) requires N number of multiplications, although $x(0)W_N^0$ is a trivial multiplication. Let us consider all of them and see later how this simplifies. Each X(k) requires N number of multiplications and each multiplication in general is complex (x(n) can be a complex signal). Each complex multiplication requires four real multiplications. To compute all the N number of points, you shall require N times N or N square complex multiplications. How many additions are required for each X(k)? It is N – 1. For computing all X(k)'s, the total number of additions would be N(N – 1). And if N is something like 1024, then the number of operations acquire astronomical dimensions. Therefore DFT, although it was known from Gauss' times, was not used till about early 70s. Only after Cooley and Tukey came out in 1965 or 1966 with the so called Fast Fourier Transform (FFT, which is not a Transform) that DFT began to be used extensively. The Fast Fourier Transform is an unfortunate nomenclature used by Cooley and Tukey and has gone into the literature like that. It is not a new transform. It is an algorithm for computing DFT or IDFT. DFT and IDFT are of the same form, except for the change in sign of powers of W_N. Since they are of the same form, the same algorithm can be used for computation of DFT and IDFT. FFT reduces the number of arithmetic operations drastically.

(Refer Slide Time: 42:01 – 46:04)

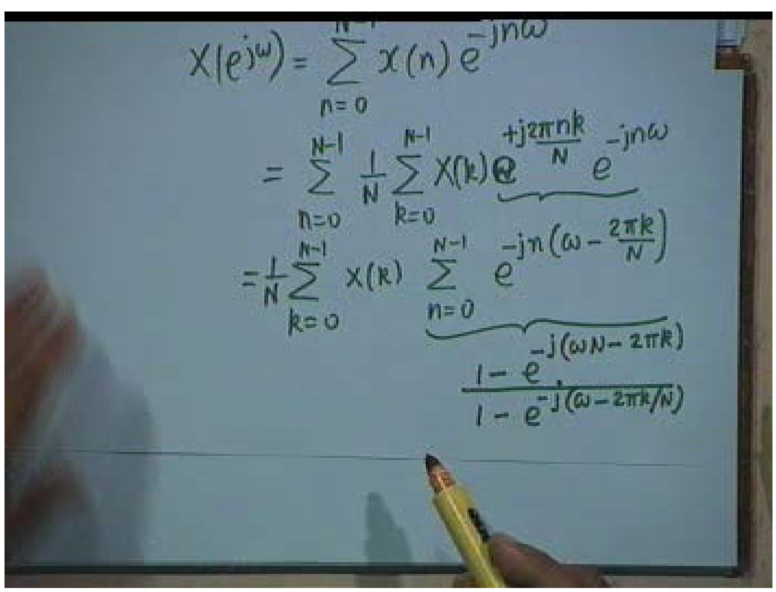


In fact we will show that instead of N square complex multiplications and N(N – 1) complex additions, what you require with FFT is of the order of N/2 $\log_2$ n and this reduces the computations drastically.

In fact, DSP was considered, till FFT came into the picture as a very complicated matter due to two reasons. One is the lack of appropriate hardware. Hardware was very costly those days. But since 70s, there has been revolutionary progress in hardware in integrated circuits. And the second one is that nobody used DFT because of the complexity of computation. You have to do it fast and with FFT, the horizon of application of DSP widened like anything and wherever you go, be it Power Quality Assessment, Sonar or Radar, you get FFT. They use DSP in the form of FFT. FFT is one of the main applications of DSP and it has revolutionized technology, facilitating DSP even where analog processing was done earlier. So this is a monumental contribution by Cooley and Turkey. Some people say they rediscovered Gauss algorithm, but they were the first ones to point out that DFT can be used profitably in Digital Signal Processing.

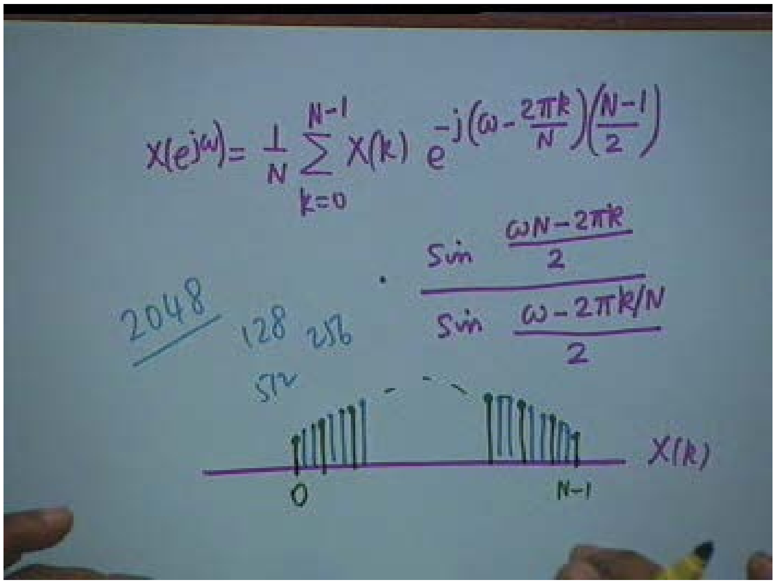
We shall discuss FFT at a later date. There have been many variations of FFT. But one thing you should remember is that FFT is not a new transform. The Fast Fourier Transform is simply an algorithm for computing DFT. FFT is not different from DFT but it is the way you compute the DFT. The next question that we address is the following. You have x(n) and the samples are from 0 to N – 1. You have X(k) which is the DFT and the samples are from k = 0 to N – 1. We said that X(k) is a sufficient description of $X(e^{j\ omega})$, because it contains N independent pieces of information. The question now is, given X(k), how do you go to $X(e^{j\ omega})$? That is, if the parts are given, how will you reconstruct the whole? In effect what we are trying to do is that given values at discrete points, we are trying to fill in between and this process is known as Interpolation. The question therefore is, given X(k), the DFT, how do you interpolate to $X(e^{j\ omega})$? Once again, I shall outline the Mathematics. The detailed steps and other things are very simple and they can be done by you.

(Refer Slide Time: 46:21 – 50:24)



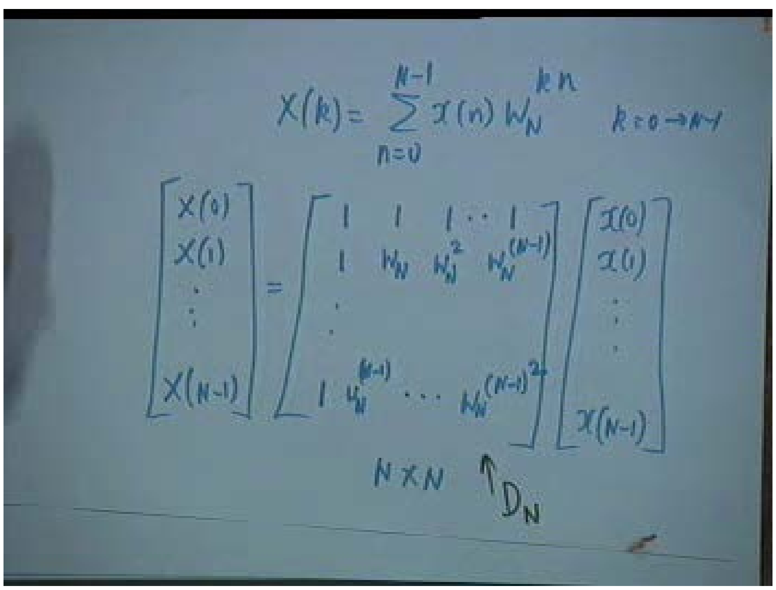
Here $X(e^{j\,omega})$ is by definition, summation x(n)e to the power – jnomega, where n goes from 0 to N – 1 (because x(n) is of finite length). Now we substitute x(n) by the inverse DFT and then we interchange the limits of summation. We interchange the summations n = 0 to N – 1, and k = 0 to N – 1. Combining the powers of e, we get a geometric series summation. The result is shown under the curly bracket in the figure. This quantity now can be put in a slightly different form by taking exponential factors out from numerator and denominator. You can then combine the two exponentials, one in the numerator and one in the denominator, and the final result is the following:

(Refer Slide Time: 50:36 – 54:58)



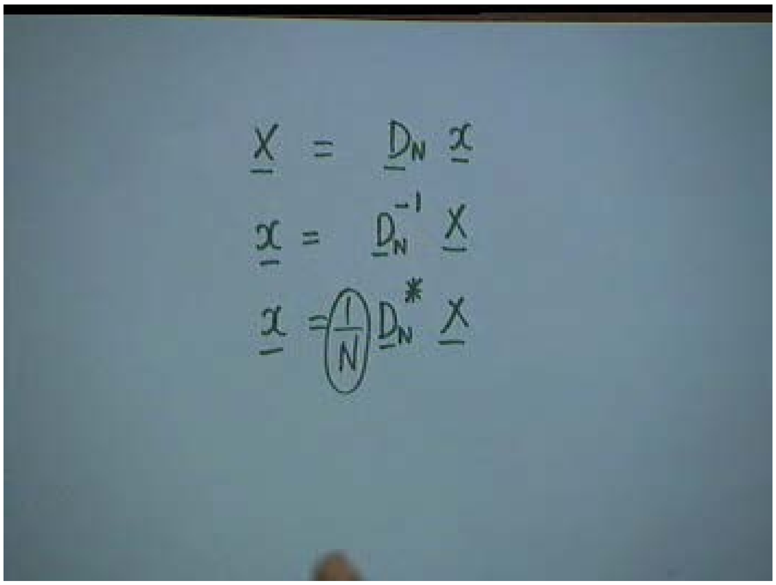
It is $X(e^{jomega})$ = (1/N) summation X(k), where k goes from 0 to N – 1, e to the power – j(omega – (2πik/N))[(N – 1)/2]), multiplied by sine of (omega N – 2pik)/2 and divided by sine of [omega – (2pik/N]/2. This is the formula for computation of X(e to the jomega) at any frequency. The formula is useful if you want to compute $X(e^{jomega})$, at a dense set of frequencies. The formula can of course be programmed in the computer. You can find its magnitude, you can find its angle and so on. Basically, this is a process of interpolation. When you compute at a sufficiently large number of frequencies, then on the CRO, it will look like a continuous spectrum. This is what is done in a Sonar Application for example. In a Sonar Application, to find the signature of an enemy submarine, you take the DFT of x(n) and you use a very large number of points in the DFT so that these samples are very close together. That is very costly and it also takes time. Suppose you want to compute 2048 points, the computation requires a certain number of operations and each operation takes time. By the time you finish the computation, the enemy submarine may shoot you and sink your submarine. And therefore what one may do is to start with a smaller number of points, for example 128, which can be done very quickly. Then you get an idea of this spectrum of the signature of the enemy submarines. And if this fits the known data about enemy submarine collected through other confidential means, then you shoot at the enemy. But if you are in doubt because of reflection from a nearby big fish like a shark or may be a submerged ship somewhere or some other object or even because of one of your friendly submarine, you increase it to 256 then you increase it to 512 till this spectrum makes you confident that it is the enemy or the friendly submarine. This is one of the major applications of DFT computed by FFT.

(Refer Slide Time: 55:19 – 58:24)



The relationship X(k) equal to summation x(n)W_N raised to the power kn n = 0 to N – 1 contains N number of equations X(0), X(1), …X(N – 1). There is a compact way of writing the same equations. If uses matrices and if I write the set X(0), X(1),… X(N – 1) as a column vector and the set x(0), x(1),…X(N – 1) as another column vector, then obviously the multiplying matrix shall be of dimension N x N. It has to be a square matrix. And the first row which relates X(0) to x (0), x (1), x (N – 1) obviously will have the elements 1, 1, 1, 1,…; irrespective of the value of k, when n = 0, the coefficient will be 1 and therefore, the first column should also have all elements equal to 1. The second row shall be 1, W_N, W_N squared and so on up to W_N raised to the power N – 1, and you can write the rest. The last row for example shall be 1, W_N raised to the power N – 1 and so on up to W_N raised to the power (N – 1) squared. This square matrix is represented by the symbol D subscript N, where N indicates the dimension.

(Refer Slide Time: 58:31 – 1:00:41)



And therefore in matrix form, we write this as $\underline{X} = \underline{D}_N$ multiplied by $\underline{x}$. And it is obvious that while the direct DFT can be written in matrix form, the inverse DFT can also be written in matrix form. It would be $\underline{x}$ = ($\underline{D}_N$ inverse) times $\underline{X}$. And if you take $\underline{D}_N$ inverse, the only change is that 1/N factor comes and instead of plus sign in powers of W_N, the negative sign comes. And W_N^{-1} is W_N^* = e to the power +j2pi/N. Therefore it follows that $\underline{D}_N$ inverse is nothing but D_N complex conjugate multiplied by 1/N. So, if you know D_N, then you can do either DFT or IDFT. In other words, the same algorithm which is used for computing DFT can also be used for computing IDFT and this is one of the versatility of the DFT operation. You know in FT operation, one is a summation the other is integration; so you have to have two different algorithms. Integration is also done numerically (a digital computer cannot perform a continuous integration), but the algorithm is different. In DFT and IDFT, the algorithms are the same. The only thing you have to do is to introduce a scaling factor and change the power of W_N to negative, instead of positive.

This is where I should stop today.