Digital Signal Processing

Prof. S. C. Dutta Roy

Department of Electrical Engineering

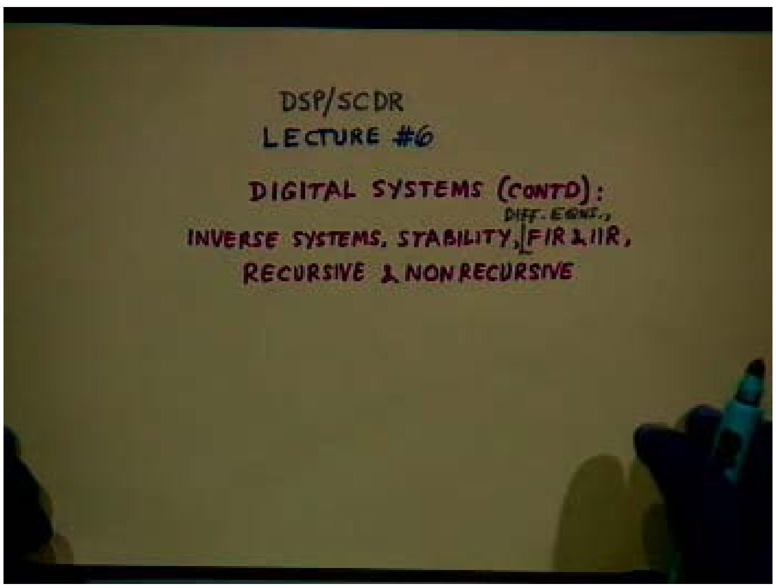
Indian Institute of Technology, Delhi

Lecture - 6

Digital systems (contd.); inverse systems, stability, FIR and IIR, recursive and non recursive

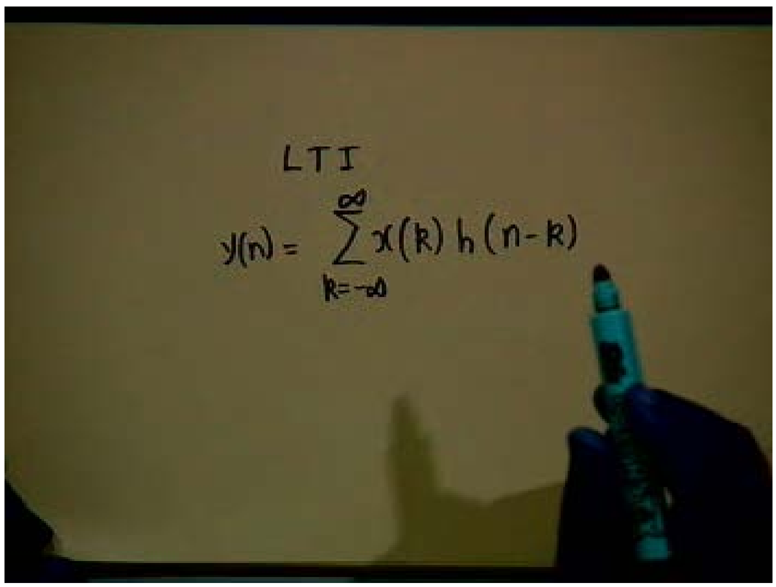
We continue our discussion on digital systems. Today's topics are inverse systems, stability criterion, difference equations, FIR and IIR, recursive and non recursive.

(Refer Slide Time: 01:11 - 01:27)



Although I have put a large canvas, we shall uncover only that part which we can do comfortably. In the last lecture, we had first clarified a question about time invariance and the example in question was an up-sampler. We showed that even if the waveform is preserved, the number of samples by which the output is delayed is not the same as the number of samples by which the input was delayed. Then we introduced the terms impulse response and step response and commented that the two responses are intimately related to each other. We illustrated the impulse response and step response by various kinds of accumulators. We also took an example of up-sampler and an interpolator. Then by simple reasoning, we derived a convolution equation for an LTI system and we said that if y(n) is the output to the input x(n), then all have to do is x(k) × h(n – k) and sum it up from k = – infinity to + infinity. For special cases, the limits change. For example, if x(n) is casual, then the lower limit becomes 0. If x (n) is casual and h (k) is also casual, then the lower limit is 0 the upper limit becomes n.

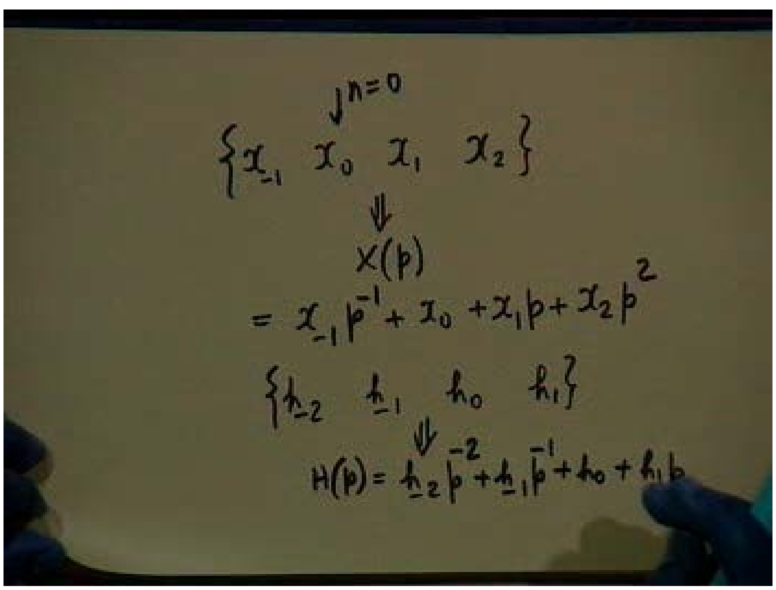
(Refer Slide Time: 02:38 - 03:26)



Then we took an example of calculation of the convolution sum by the usual graphical method, that is shifting one of the sequences one step at a time either to the right or to the left depending on which particular sample of the output you wish to compute. I also introduced a trick by which this graphical step by step procedure is not needed. That was simply the same process as that of arithmetic multiplication except that there is no carry and except for the fact that you must keep track of your reference, that is the pivot y (0). That occurs when the sum of indices of x and h equals zero. If it is $x_{-1,}$ the corresponding h term must be h_1 and so on. Then you go to the right one step and you get y(1); and two steps to the right gives you y(2) and so on. I would like to introduce you to a third trick which is equivalent to the trick that I mentioned earlier. But it is interesting from the point of view of history of DSP.

The Geophysicists were the first ones to use DSP without knowing that they were doing digital signal processing. Let us take the same example: $x(n) = \{x_{-1}, x_0, x_1,$ and $x_2\}$. The geophysicist said that instead of writing the sequence, let us write it in the form of a polynomial. They introduced an arbitrary variable; let us call this variable as p. The polynomial form of x(n) is then $X(p) = x_{-1}$(p to the power minus 1) plus x_0 (p to the power zero) plus x_1p plus x_2(p to the power 2). Please follow this carefully and appreciate how intelligently they hit upon what is now known as Z- transform. They did not know the Z-transform. Z–transform came much later, in 50's, whereas geophysicists did this in 40's. They said that we will convert x(n) into a polynomial X(p). The power of the variable indicates the position of the sequence. Since p^0 is 1, we have $X(p) = x_{-1} p^{-1} + x_0 + x_1 p + x_2 p^2$. They were also required to do convolution for finding some geophysical parameters of interest and importance. Similarly, if our h sequence was: $\{h_{-2}, h_{-1}, h_0,$ and $h_1\}$, the corresponding polynomial will be $H(p) = h_{-2} p^{-2} + h_{-1} p^{-1} + h_0 + h_1p$. The power of the variable indicates the position of the number in sequence. They said that, in order to find the convolution summation, you multiply X(p) by H(p).

(Refer Slide Time: 05:24 - 07:41)

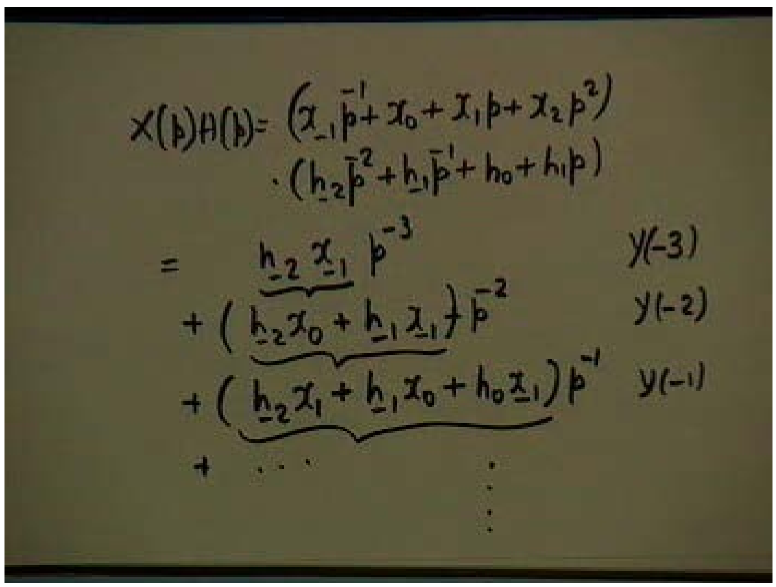


Then you get $X(p)H(p) = (x_{-1} p^{-1} + x_0 + x_1 p + x_2 p^2) \times (h_{-2} p^{-2} + h_{-1} p^{-1} + h_0 + h_1p)$; expanding this, you will also get a polynomial in terms of powers of p. Precisely, $X(p)H(p) = h_{-2}$

$x_{-1}\, p^{-3} + (h_{-2}\, x_0 + h_{-1}\, x_{-1})p^{-2} + (h_{-2}\, x_1 + h_{-1}\, x_0 + h_0\, x_{-1})p^{-1} + \dots$ Don't you see that the coefficient of p to the power minus 3 is y(–3), that of p to the power minus 2 is y(-2), that of p to the power minus 1 is y(-1) and so on? Thus X(p) H(p) = Y(p) and the output sequence is obvious.

Is the process clear? What we are doing exactly is arithmetic multiplication except that we are not putting in the form of a table, but we are doing this by algebra.

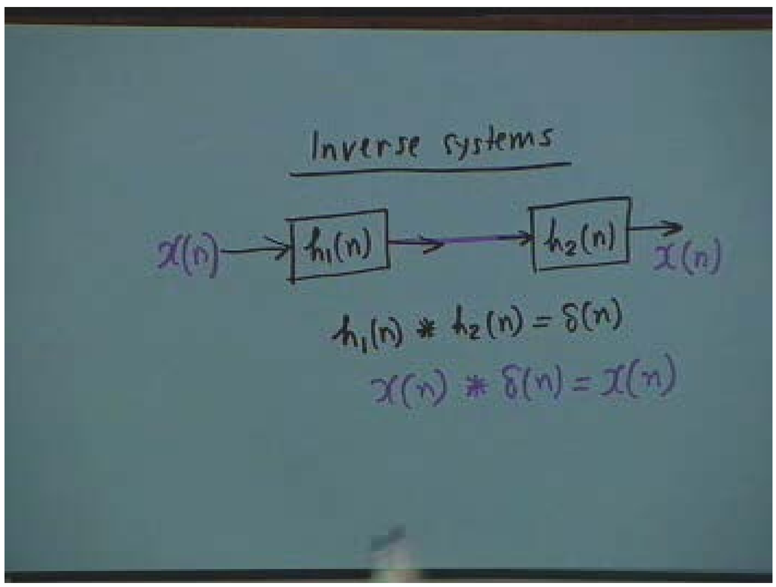
(Refer Slide Time: 07:57 - 10:20)



This is how the geophysicists used to compute convolution and this is also a trick. In graphical shifting, you make one mistake and your total result is shattered. As those of you, who have some familiarity with Z-transforms, will recognize, this is nothing but application of Z-transform, except that instead of the variable z^{-1}, we have used the variable p. So you can compute convolution by at least three techniques viz graphical, arithmetic multiplication or multiplication of two polynomials. If you are in doubt, check it by one more method. I have not completed this example which I hope you would be able to do. You show that the result is exactly the same as that obtained by any other method.

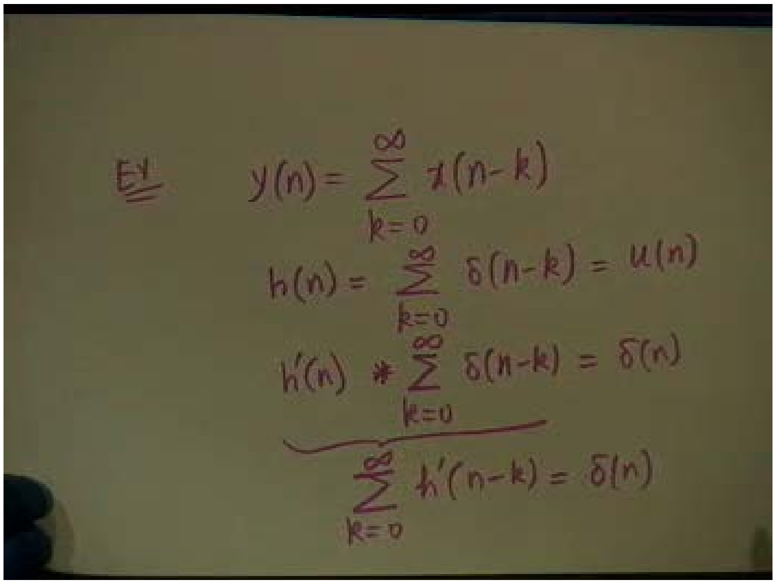
Next, we will talk about inverse digital systems. A system with impulse response $h_1(n)$ is said to be inverse of another system whose impulse response is $h_2(n)$ if and only if the convolution $h_1(n)* h_2(n) = \delta(n)$. If the convolution gives you a unit impulse, then the two systems are said to be inverses of each other. The reason for the nomenclature is very simple; if the output of $h_1(n)$ is connected to the input of $h_2(n)$, and input to $h_1(n)$ is x(n), then the output shall also be x(n) because x(n) convolved with δ(n) is simply equal to x(n). Two systems h_1 and h_2 are inverses of each other if their convolution gives rise to a unit impulse function δ(n).

(Refer Slide Time: 11: 35 -13:07)



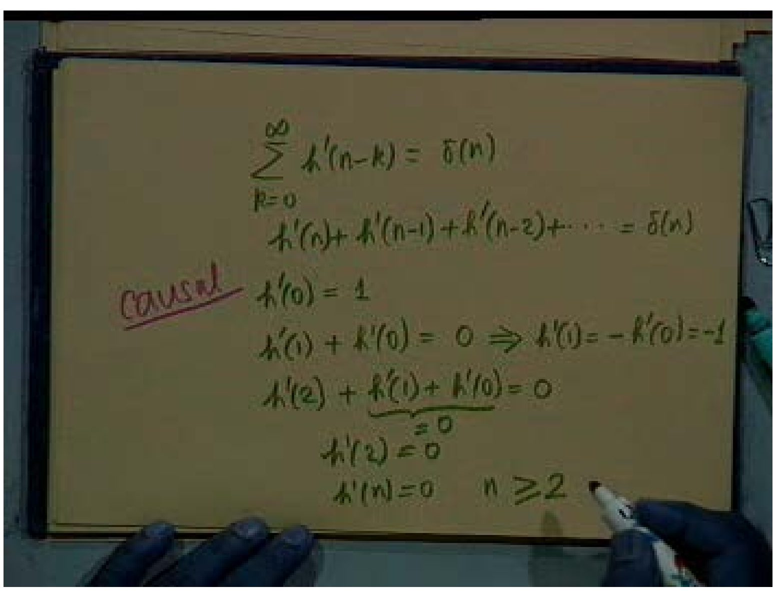
Let us take an example. Suppose we take the accumulator y(n) = sigma (k = 0 to infinity) of x(n – k). We wish to find out its inverse. So the first thing we do is to find h(n) which is sigma (k = 0 to infinity) δ(n – k). Isn't this precisely u(n), the unit step function? If an inverse system exists and its impulse response is h prime (n), then h(n) convolved with h prime (n) should δ(n). In other words, h prime (n) convolved with u(n) should be equal to δ(n). Now δ(n) can be written as u(n) – u(n – 1). Thus we need u(n)*h prime(n) = u(n) – u(n – 1). Clearly, h prime(n) = δ(n) – δ(n – 1).

(Refer Slide Time: 13:15 - 14:55)



Thus h prime (n) consists of only two samples, one at n = 0 of amplitude 1 and the other at n = 1 of amplitude – 1.

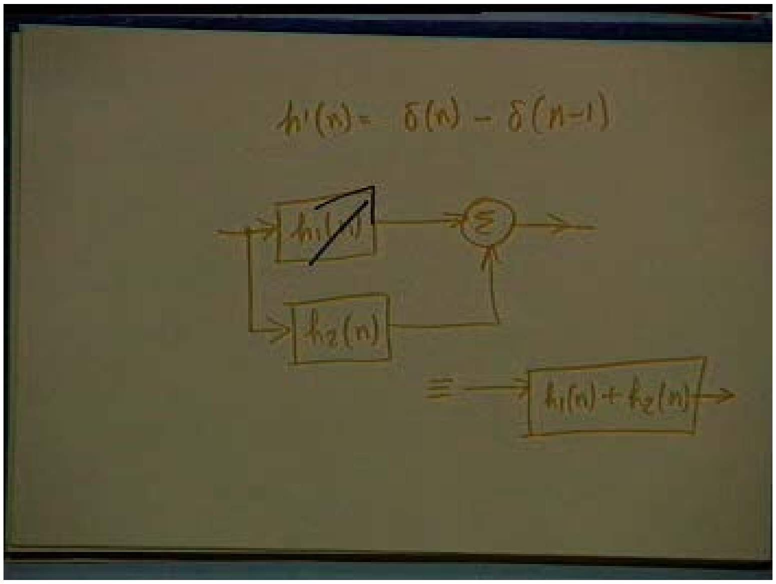
(Refer Slide Time: 15:01 - 17:13)



The inverse of a system may or may not exist. It does not exist, if the inverse system is unstable.

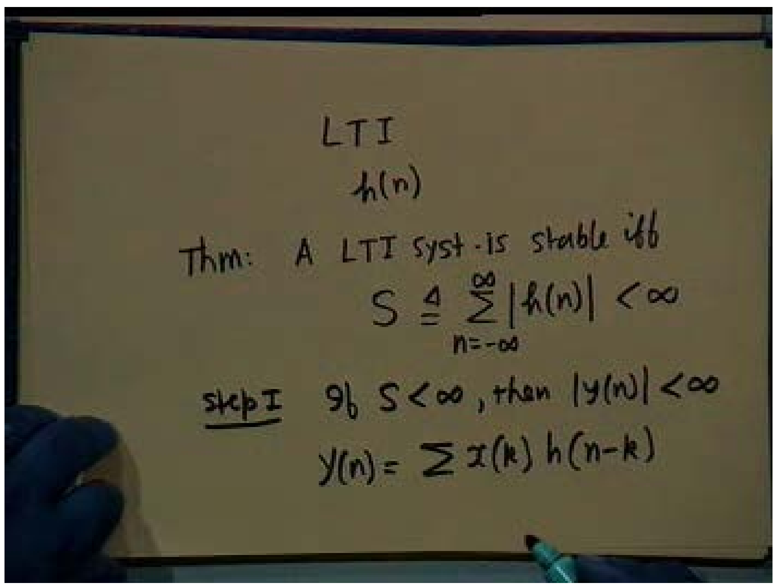
Next, consider a parallel interconnection: A system $h_1(n)$ is in parallel with $h_2(n)$; this is equivalent to a single system in which the impulse response is $h_1(n) + h_2(n)$ because there is parallel processing. This system is very important in practice because it enhances the speed of processing. When $h_1(n)$ delivers its output signal, simultaneously $h_2(n)$ also delivers its output signal. Therefore if a complicated high order system is broken up into simpler systems and connected in parallel, then the speed of processing goes up. Parallel processing is important for digital systems and wherever possible, instead of serial or sequential processing, one resorts to parallel processing. What can you say about the stability of the system? That is a very good question. In a parallel system if $h_1(n)$ and $h_2(n)$ are individually stable is there a possibility that the system becomes unstable? No. Suppose one of them is unstable the other is not then the whole system is unstable because the unstable subsystem processes the signal independently of the stable one. If one system goes wild, the whole system goes wild.

(Refer Slide Time: 18:05 - 20:43)



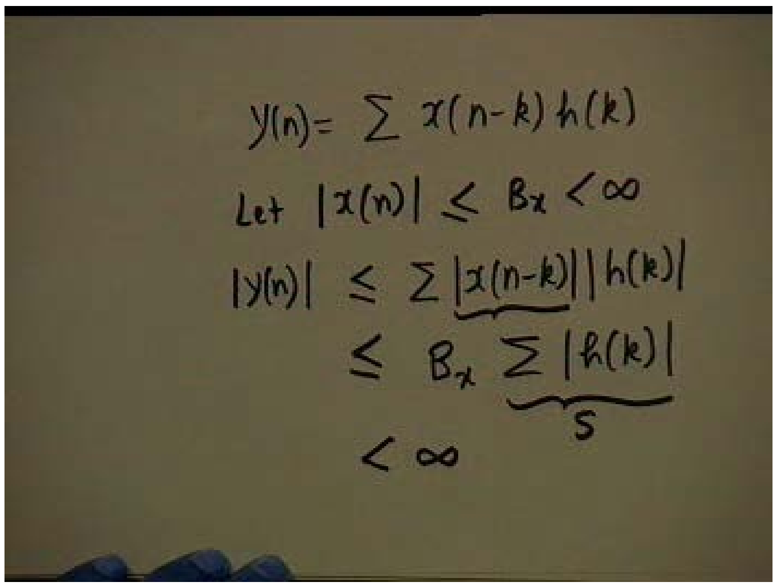
Stability, as I have defined, is in the sense of bounded input leading to bounded output. For an LTI system, stability can be characterized by its impulse response h(n). I will first state a theorem relating to stability and h(n), and prove the necessity as well as sufficiency. It is a very simple theorem. It simply says that a linear time invariant (LTI) system is stable if and only if the quantity S, defined as summation of absolute value of h(n), is bounded where n goes from – infinity to + infinity. Bounded means less than infinity. Another way of saying the same thing is that the impulse response is absolutely summable. Thus absolute summability of the impulse response ensures stability. Now we prove it in the forward direction and also in the reverse direction. First, we prove that if S is less than infinity, then the absolute value of y(n) is also less than infinity. It is very simple to prove because your y(n) is nothing but summation[x(k) h(n – k)]. In general, the limits of k are – infinity and + infinity.

(Refer Slide Time: 20:59 - 22:49)



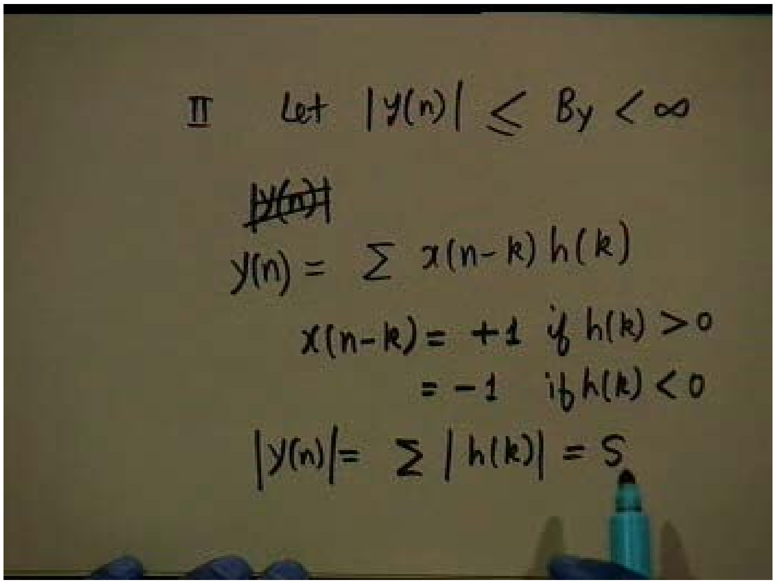
As the operation of convolution is commutative, so we can also write: y(n) = summation [x (n – k) h(k)]. We are talking of BIBO stability. So let x(n) can be bounded by some quantity B_x, which is less than infinity. Then magnitude y(n) shall be less than equal to the summation [magnitude (x (n – k)) × magnitude h(k)]. If x(n) and h (n) are all real numbers and all of the products are positive, then we do not have to use the absolute sign. We are using the absolute sign, because x (n – k) as well as h(k) may be either positive or negative. Using magnitudes gives the upper bound and therefore |y(n)| is less than equal to the summation. Suppose we replace each x (n – k) by B_x that is by its upper bound. Then I can take B_x outside and then I get summation |h(k)|, which is equal to S; therefore if S is less than infinity then SB_x is also less than infinity. In other words we have proved that if h(n) is absolutely summable, then the system is BIBO stable. Now we have to prove it conversely. What is the converse statement? If the output is bounded i.e. if the absolute value of y(n) is less than infinity with x(n) bounded, then S should be less than infinity. In the previous step, we assumed S as less than infinity; now we assume that the output is bounded and prove that S is less than infinity. So let the output be bounded.

(Refer Slide Time: 22:54 - 24:53)



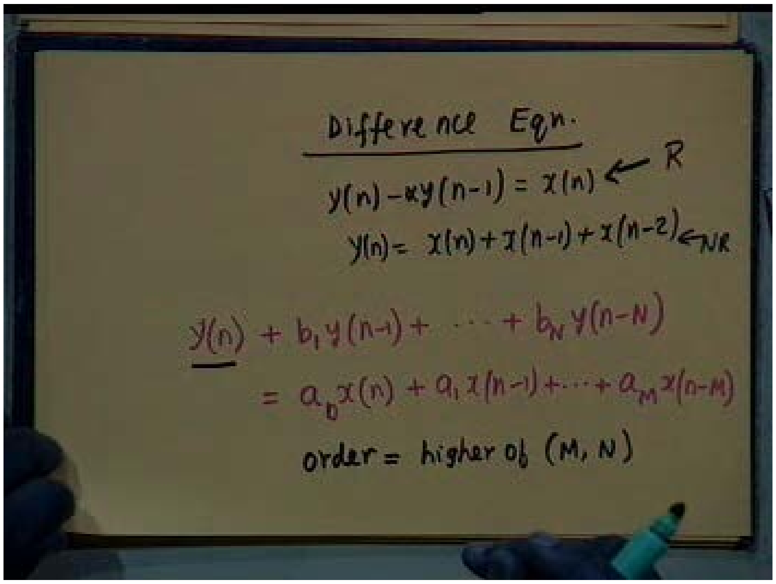
Let the output be bounded by some bound B_y less than infinity. Let us take y(n) = summation [x (n – k) h(k)]. Let us take x (n – k) = + 1, when h(k) is positive, and x (n – k) = – 1 if h(k) is negative. Depending on the sign of h(k), we choose our x(n – k) as either + 1 or – 1. Then what does the sum become? Under this condition, y(n) = summation |h(k)| = S. Therefore if y(n) is bounded, S should also be bounded. So it is a proof of the theorem that an LTI system is BIBO stable if and only if h(n) is absolutely summable. Absolute summation simply means that summation of the absolute value over all values of n is less than infinity. We shall use this a little while later to prove something else.

(Refer Slide Time: 25:30 - 27:31)



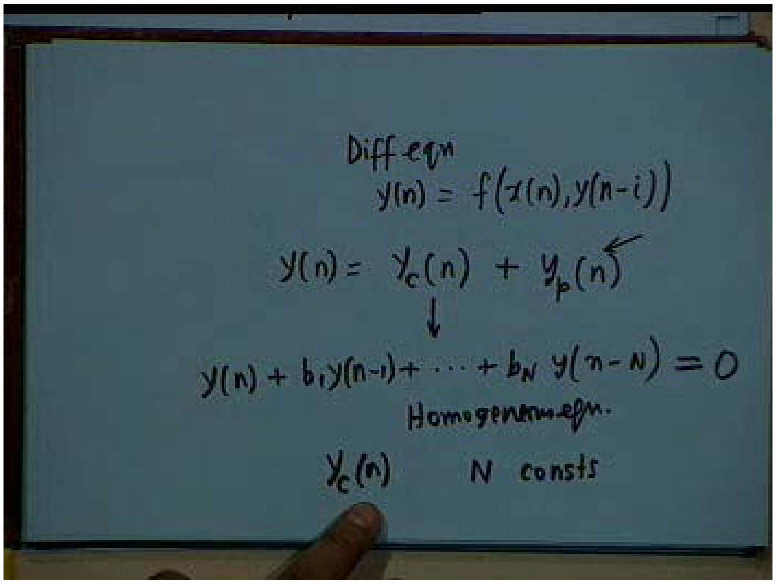
A digital system in general is described by a difference equation. For example, $y(n) – ά y(n – 1) = x(n)$ is a difference equation; $y(n) = x(n) + x(n – 1) + x(n – 2)$ is also a difference equation. The first one is a recursive difference equation while the second one is non recursive. Let me explain the terms recursive and non recursive. The first equation uses a past value of output; the second one does not. The first equation is recursive while the second is non recursive. In general, an LTI digital system is described by an equation of this form: $y(n) + b_1 y(n – 1) + b_2 y(n – 2) + \ldots\ldots +b_N y(n – N) = a_0 x(n) + a_1 x(n – 1) + \ldots\ldots. a_M x(n – M)$. There is no obligation for M and N to be equal. M may be greater than, less than or equal to N. I have written present output y(n) with a coefficient of 1. If it is not, if it comes with a coefficient b_0 we shall divide throughout by b_0. So it is always possible to write it in this form and we shall always do it. It is a discipline which we shall follow. Now the order of the system in such a difference equation is higher of the two quantities M and N, i.e. order = higher (M, N).

(Refer Slide Time: 28:00 - 31:16)



Given this difference equation description of a system and given the fact that it is causal you can compute y(n) for any n greater than or equal to some quantity n_0 provided initial conditions on the system are given prior to $n = n_0$. On this topic of solution of difference equation, we shall spend a little time.

(Refer Slide Time: 32:29 - 36:30)

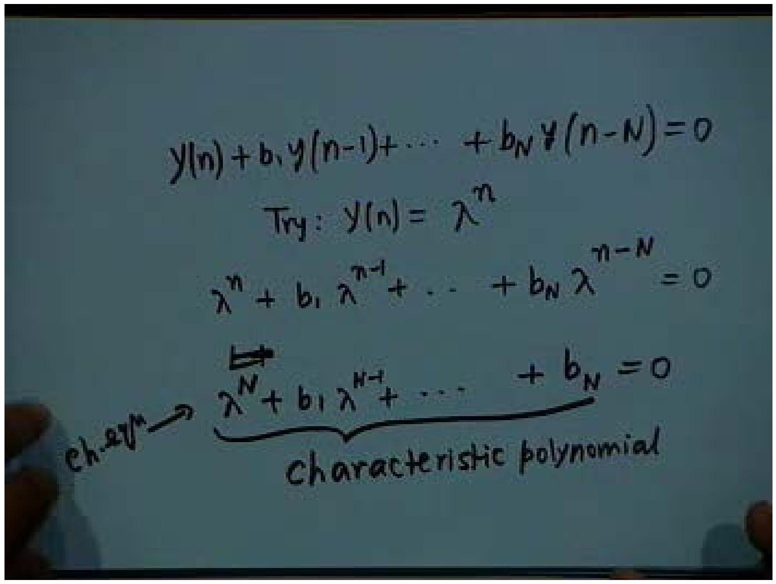


Given a difference equation for $y(n)$ as a linear combination of $x(n)$, $x(n – j)$ and $y(n – i)$, that is, given a linear difference equation, the solution is exactly like that of a differential equation; it consists of two parts, namely a part which is called the complementary function, and the other called a particular solution. The complementary function is the solution to the given equation with $x(n) = 0$. That is $x(n)$, $x(n – 1)$ and $x(n – M)$ are all equal to 0. In the differential equation, to find the complementary function, you put right hand side equal to 0. It is exactly the same. Then you add it to a particular solution. Particular solution is the solution with $x(n) \neq 0$.

Now, whenever you solve an equation with right hand side equal to 0, there would be some unknown constants. These constants have to be found out from the initial conditions. If it is a second order system, then you require two initial conditions. If it is of third order, then you require three initial conditions. From the initial conditions, you can find out the constants in the complementary function $y_c(n)$. $y_p(n)$, the particular solution shall have no constants; $y_p(n)$ shall be completely determined from the difference equation. The constants have to be found out not before adding the particular solution; it is to be done after adding the particular solution. This point must be remembered. The equation obtained by putting $x(n) = 0$ in the given difference equation is called the homogenous equation.

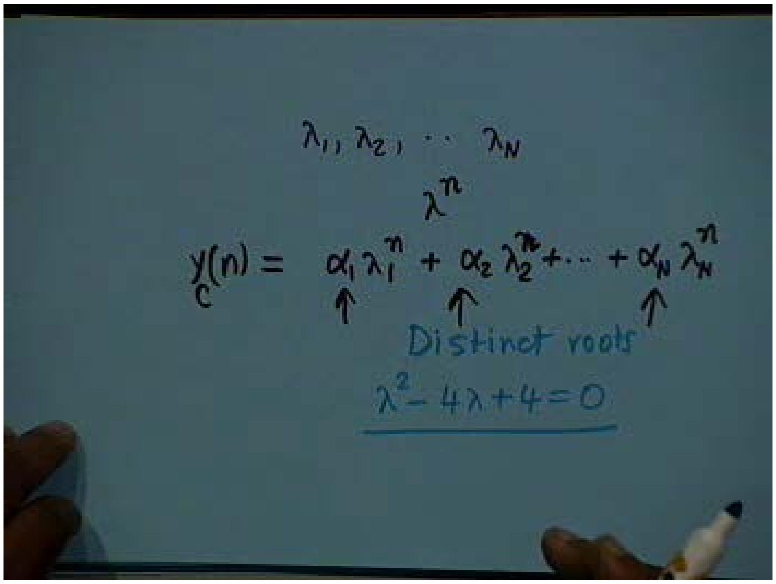
For example, the homogenous equation in our case shall be $y(n) + b_1\, y(n – 1) + \ldots\ldots + b_N\, y(n – N) = 0$. The solution to this equation shall be $y_c(n)$, called the complementary function, exactly like linear differential equation. $y_c(n)$ in general shall contain N number of constants. These constants have to be evaluated from initial conditions but not at this stage. You must first find out what is $y_p(n)$, add $y_p(n)$ to $y_c(n)$ and then put initial conditions. This step is extremely important. Initial conditions are called for only after you find the particular solution and add it to the complementary function containing N number of unknown constants. I repeat, do not evaluate the constants before adding the particular solution.

(Refer Slide Time: 36:57 - 40:05)



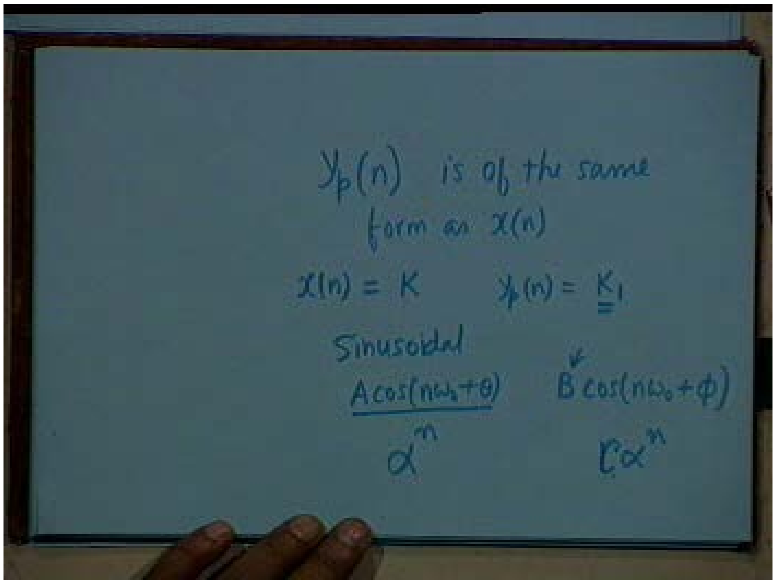
If you have an equation y(n) + b_1 y(n – 1) + … + b_n y(n – N) = 0, then the complementary function is obtained by trial solution of the form y(n) = λ to the power n (compare with differential equations, where the trial solution is y(t) = e to the power mt). λ to the power n is also an exponential, because λ can always be written as e to the power m. If you put y(n) = λ to the power n in the equation, then you get λ to the power n + b_1 λ to the power n – 1+ …+b_N λ to the power (n-N) = 0. From this, obviously you can cancel λ to the power n from both sides provided it is not 0. If λ to the power n = 0, then you have no equation and no solution. So λ to the power n can be cancelled from both sides; what will you get? If you cancel λ to the power n and multiply by λ to the power N, then the last term would be b_N and the first term would be λ to the power N. The second term would be b_1 λ to the power N – 1. So the equation becomes λ to the power N + b_1 λ to the power N – 1+…+b_N = 0. The left hand side is called the characteristic polynomial of the linear time invariant system. The roots of this characteristic polynomial, or the solution of this characteristic equation obtained by equating the characteristic polynomial to 0, are called eigenvalues. The fundamental theorem of algebra says that if you have a polynomial of degree N, then it has N number of roots. Therefore let the roots of this characteristic equation be λ_1, λ_2, and so on, up to λ_N.

(Refer Slide Time: 40:14 - 42:33)



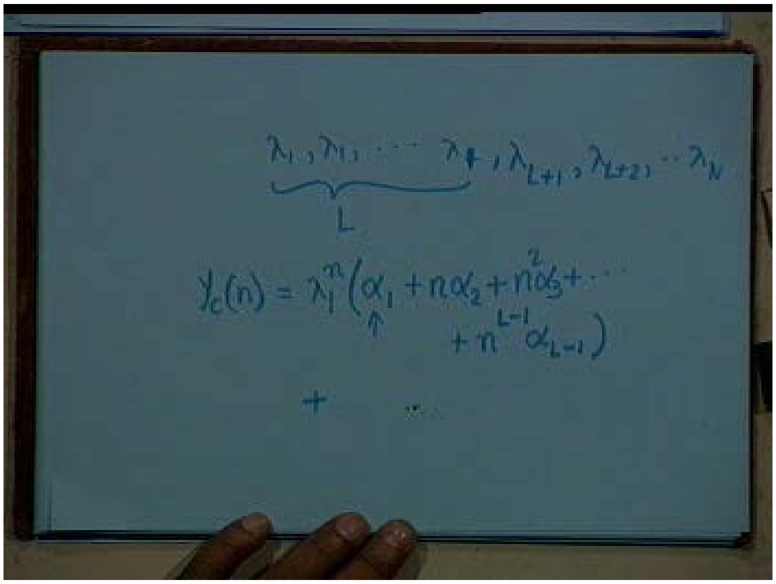
Since we had tried the solution λ to the power n and λ can take any of these values, our solution $y_c(n)$, which is the complementary function, would be of the form $ά_1$ $λ_1$ to the power n + $ά_2$ $λ_2$ to the power n+…+ $ά_N$ $λ_N$ to the power n. This will be the complementary function or the solution to the homogenous equation which contains the constants $ά_1$, $ά_2$… $ά_N$. It is very tempting at this stage to put the initial values such as: y(0), y(– 1), y(–2) and so on to find the constants, but this temptation must be done away with because you shall end up in a wrong result. Evaluation of the constants has to be done later after you have added the particular solution to the complementary function. In this formulation, we have assumed that none of the λ's are identical, that is we assumed distinct roots. If some of the roots are repeated, for example in the equation λ square-4 = 0, obviously $λ_1$ = $λ_2$ = 2; it is a case of repeated roots. We shall see a little later how to tackle the case of repeated roots.

(Refer Slide Time: 42:45 - 45:15)



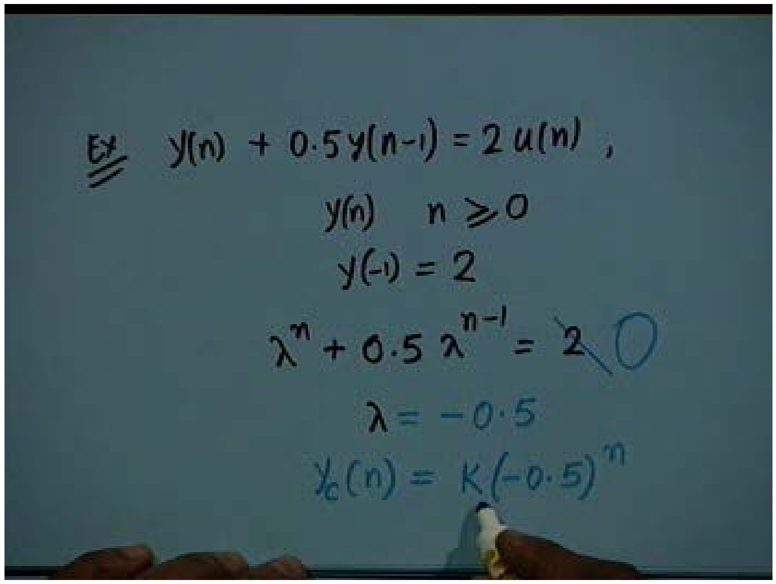
The particular solution $y_p(n)$ is very simple to find. In general, $y_p(n)$ is of the same form as the input x(n). In other words, if x(n) is a constant k, then $y_p(n)$ shall also be another constant k_1. If x(n) is sinusoidal then $y_p(n)$ is also sinusoidal. If x(n) is A cosine (n (omega) $_0$ + theta), then $y_p(n)$ would be some other constant B times cosine (n (omega)$_0$ + phi). If x(n) is $A(\alpha)^n$, an exponential, then $y_p(n)$ also be some other constant C times α to the power n. So the particular solution is very easy to find. We assume $y_p(n)$ to be of the same form as given x(n) with unknown constants. If x(n) is k then you assume it to be k_1. Similarly, if it is sinusoidal, then you assume two constants B and phi in the same form as the input. If x(n) is $(\alpha)^n$ then you assume $y_p(n) = C(\alpha)^n$. You substitute the assumed $y_p(n)$ in the given difference equations and find out the unknown constants. When the assumed particular solution is substituted in the given equation, you get $y_p(n)$ completely along with its constants. After finding $y_p(n)$, you come back to the complementary function. That is you add $y_p(n)$ to the complementary function and then you invoke the initial conditions. We shall take one or two examples, but before I do that, I would like to talk about the case of repeated roots.

(Refer Slide Time: 45:35 - 49:50)



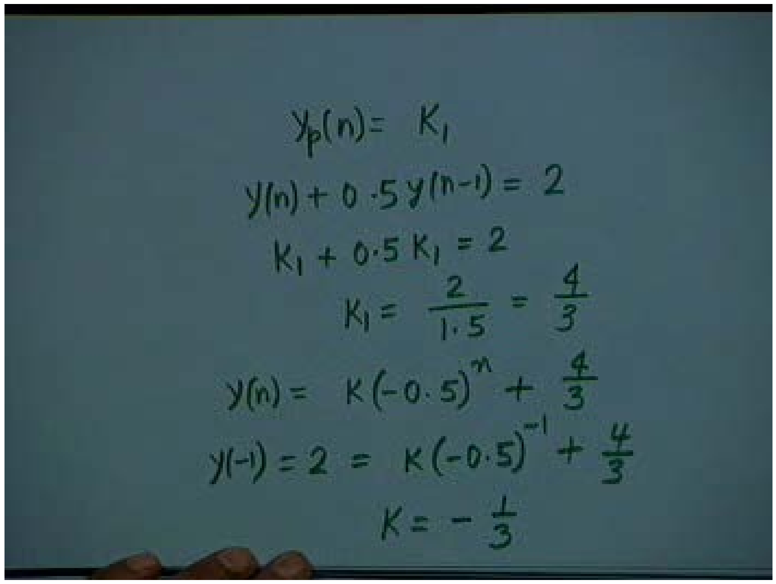
Suppose in the Nth order equation that we had, one of the roots λ_1 is repeated L times, then you have the other roots as λ_{L+1}, λ_{L+2},…λ_N. The total number is N. L of them is repeated and N-L are distinct. If that is the case, then your complementary function will be of the form $[(\alpha_1) + n(\alpha_2) + n^2 (\alpha_3)+\ldots+ n^{L-1} (\alpha_L)] \lambda_1^n + (\alpha_{L+1}) (\lambda_{L+1})^n +\ldots..+ (\alpha_N) (\lambda_N)^n$. How many constants have we used? It is exactly N of course. If a root is repeated twice, for example, then you simply use $(\alpha_1 + n\, \alpha_2)$ λ_1 to the power n. The highest power of n in the polynomial multiplying λ_1 to the power n is 1 less than the number of repetitions. This is in general the solution for the complementary function. For the particular solution, you simply put y(n) of the same form as x(n) and find out what the constant is. That would be the complete particular solution. Particular solution should have no constants. Add that to the complementary solution and then invoke N number of initial conditions to find out the N number of constants α_1, α_2 and so on. Let us take an example.

(Refer Slide Time: 50:34 – 52:42)



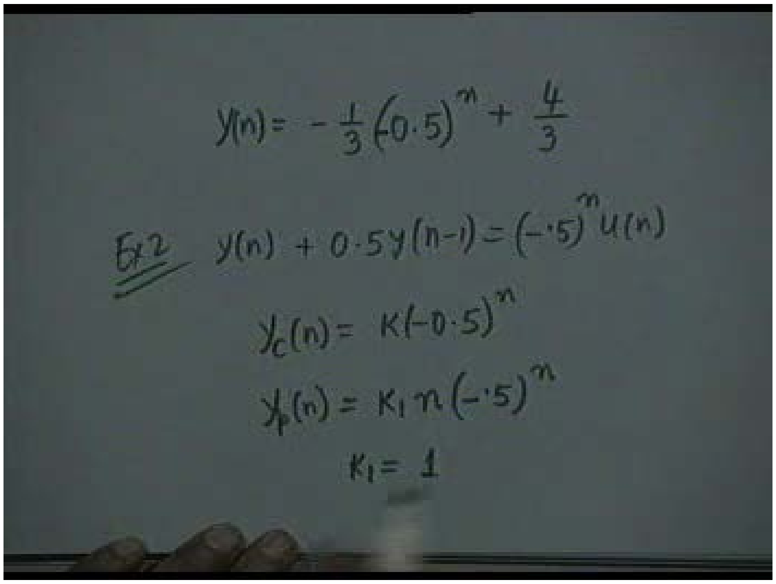
We take y(n) + 0.5 y(n – 1) = 2u(n) and the problem is to find y(n) for n greater than or equal to 0. It is a first order equation and you require one initial condition, and it is given that y(– 1) = 2. For the complementary function, the equation to be solved is $(\lambda)^n + 0.5(\lambda)^{n-1} = 0$. This gives you $\lambda = -0.5$. Therefore $y_c(n)$ is of the form k $(-0.5)^n$. I do not evaluate k by invoking initial conditions; rather I first find the particular solution. To find out the particular solution, I assume it to be of the same form as the given input which is a constant equal to 2.

(Refer Slide Time: 52:55 – 55:08)



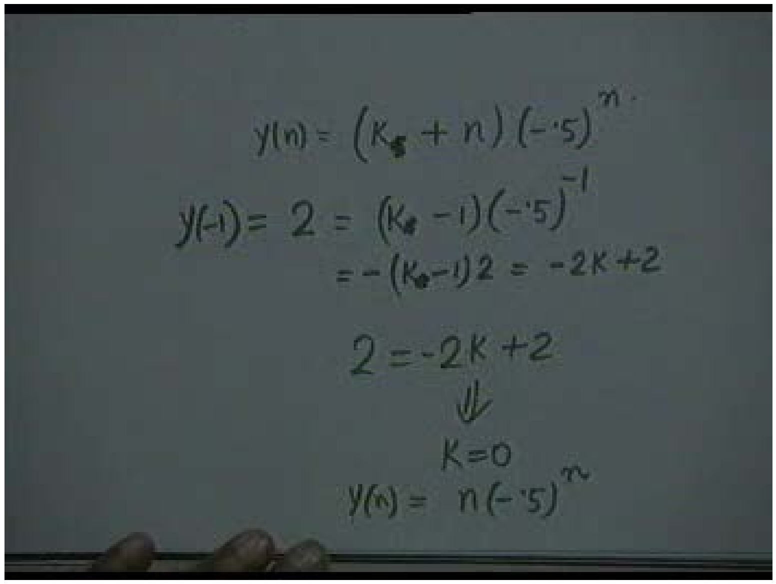
So I assume $y_p(n)$ = some constant k_1. Substitute it in the original equation. The equation is $y(n) + 0.5\,y(n-1) = 2$. So I get $k_1 + 0.5k_1 = 2$. $y(n-1)$ is also k_1 because $y_p(n)$ is a constant. So $k_1 = 2/1.5 = 4/3$. I do not write it is as 1.3333 recurring. In a DSP system, you must be careful; maintain a fraction till you are forced to change it because as soon as you make a truncation you make an error. Truncation will have to be done because of finite word length. 4/3 cannot be represented by a binary number, so you will make error there. Why introduce an error at this stage? In the calculation you keep the number as a fraction as long as you can. If it is 1.73 then you are permitted to do that. But if it is recurring then you do not. Pi: keep as pi; exponential e: keep as exponential e as long as you can. So what we do now is, $y_c(n) = k(-0.5)^n$, the complementary function and I add it to 4/3. Then I put the initial condition, that is $y(-1) = 2 = k(-0.5)^{-1} + 4/3$ and you can very easily show that k comes as – 1/3.

(Refer Slide Time: 55:16 – 57:36)



Therefore my total solution is: $y(n) = -(1/3)(-0.5)^n + 4/3$. Now I am tempted to take another complicated example in which the input is of the same form as one of the eigenvalues. Let us take x(n) as $(-0.5)^n u(n)$. Now you have a problem, input is of the same form as the eigenfunction of the system $(\lambda)^{n.}$ The complementary function is the same. What is the complementary function? $y_c(n) = k(-0.5)^n$ and $y_p(n)$ you assume as $k_1 n(-0.5)^n$ because it is an eigenfunction of the system. What you do is exactly like the case of repeated roots. Substitute it in this equation to find k_1, but I will omit the calculation. If you substitute that we get $k_1 = 1$.

(Refer Slide Time: 57:47 – 59:34)



Thus the total solution is y(n) = (k + n) $(-0.5)^n$. And at this point you introduce the initial condition y(– 1) = 2. If you put this you get (k – 1) $(-0.5)^{-1}$ = 2. Our equation now becomes 2 = – 2k + 2. What does this give for k? It gives k = 0. The complementary function therefore does not exist. That means there are no transients in this system for this input. It is a very interesting case. I think we should stop here.