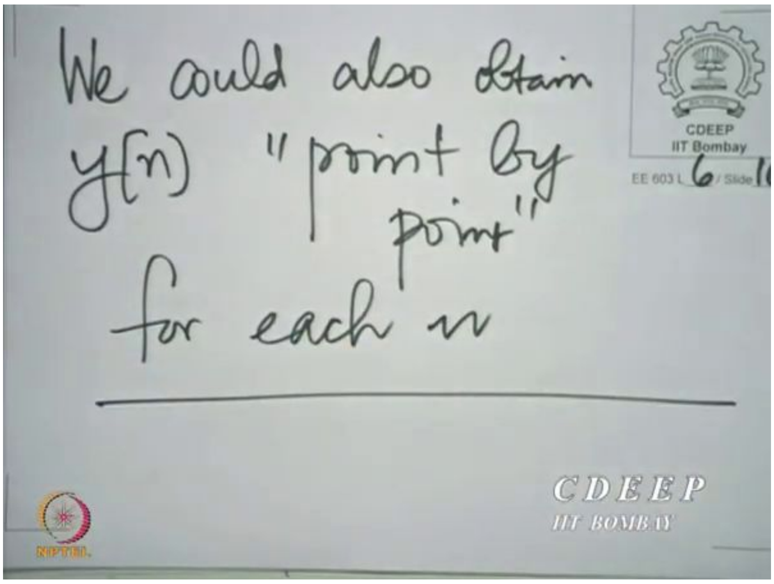
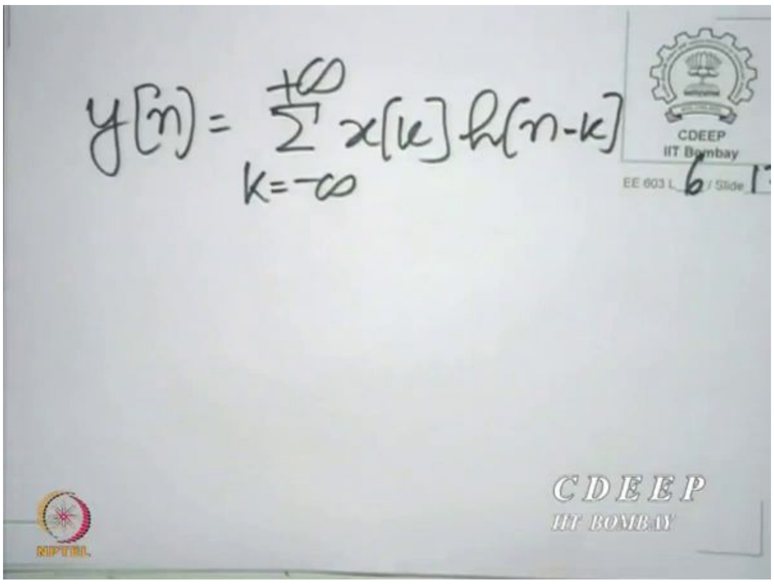
Digital Signal Processing & Its Applications
Professor Vikram M. Gadre
Department of Electrical Engineering
Indian Institute of Technology, Bombay
Lecture – 06 c
Convolution: Deeper Ideas and Understanding through Examples.

(Refer Slide Time: 00:17)



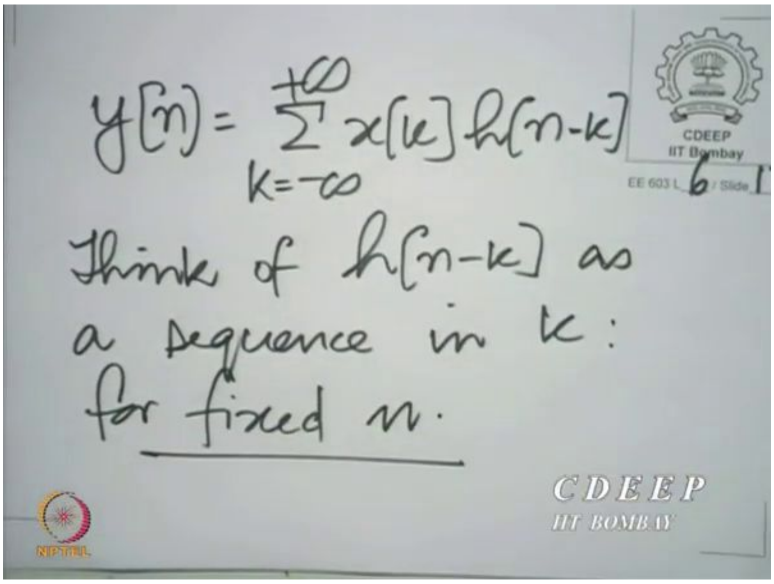
We could also obtain *y[n]* point by point for each *n* and here, all that we need to do is to reinterpret the expression for convolution. So what we are saying is, take note of the expression for convolution once again.

(Refer Slide Time: 00:43)



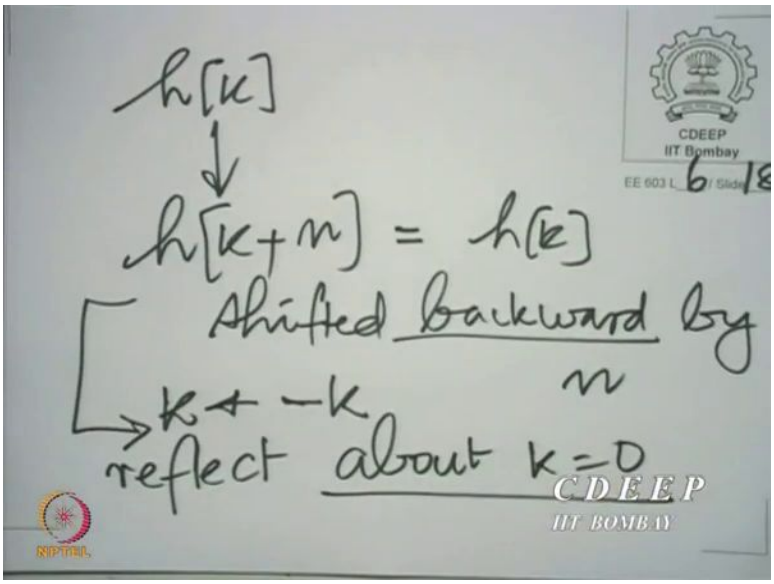
And instead of now thinking of *x[k]* as a constant, and *h[n-k]* as a sequence, I told you interpretation is important. We will think of both *x[k]* and *h[n-k]* without the summation as sequences indexed by *k*. So *x[k]* is the sequence indexed by *k*, that is easy to understand. *h[n-k]* is also sequence now not indexed by *n* but indexed by *k*.

(Refer Slide Time: 01:37)



Okay, so think of *h[n]*. So the difference in that approach is think of *h[n-k]* as a sequence in *k* not in *n* for fixed *n*, you see, because you are trying to calculate point by point. And how do we, how do we obtain that sequence? We need to interpret, so towards interpreting that we will follow two steps.

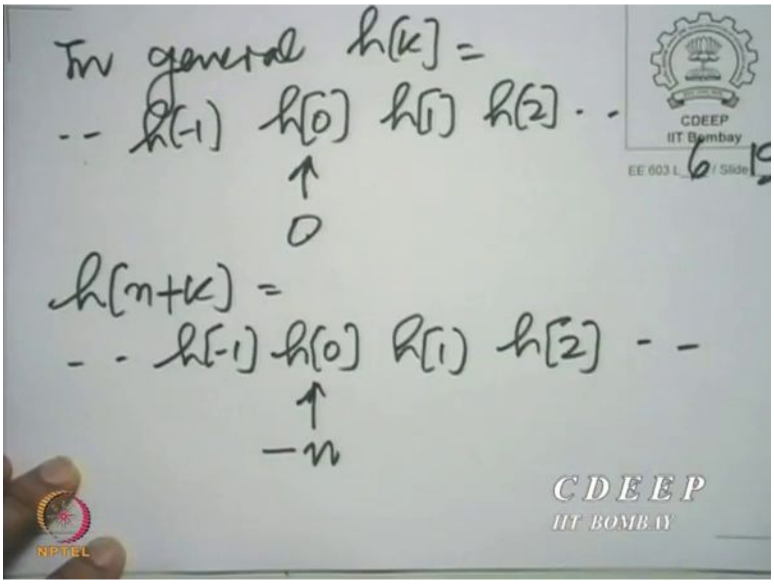
(Refer Slide Time: 02:05)



We will first go from *h[k]* to *h[k+n]* for a fixed *n*. And obviously, for a fixed *n*, *h[k+n]* is essentially the, in the sequence *h[k]* shifted backward by *n*. So *h[k+n]* is *h[k]* shifted backward by *n*. And the next step is of course, to replace *k* by (-*k*), so when we replace *k* by (-*k*), what we are doing is to reflect about *k* = *0*.

Replacing *k* by (-*k*) essentially means switching every pair of points, for example, switching *1* and (-*1*) switching *2* and (-*2*), switching *3* and (-*3*), keeping *0* where it is. So it really means you are making a mirror image of that sequence with the mirror placed at *k* = *0*. Remember, the sequence that you are mirroring is *h[k+n]*, all right? So we are mirroring *h[k+n]* and we can now visualize what to expect.

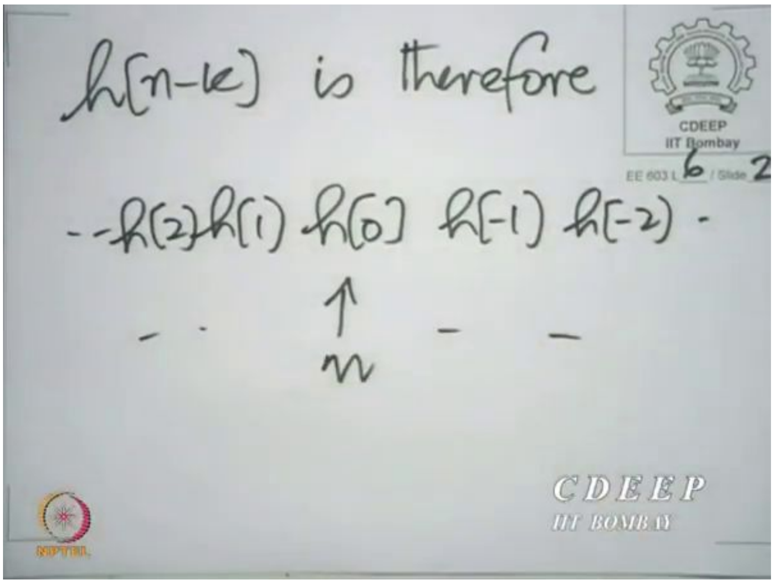
(Refer Slide Time: 03:36)



In fact, I do not need to take a specific example, we can write in general, in general, you know, when you have *0* or something, you might have *h[0]* and so on, is not it? *h[1]*, *h[2]* and so on, this is a sequence here, *h[-1]* and you can go on behind. If this is *h[k]*, then *h[n-k]* is going to be obtained by shifting this backwards to, maybe we will first write *h[n+k]* shifting this backwards.

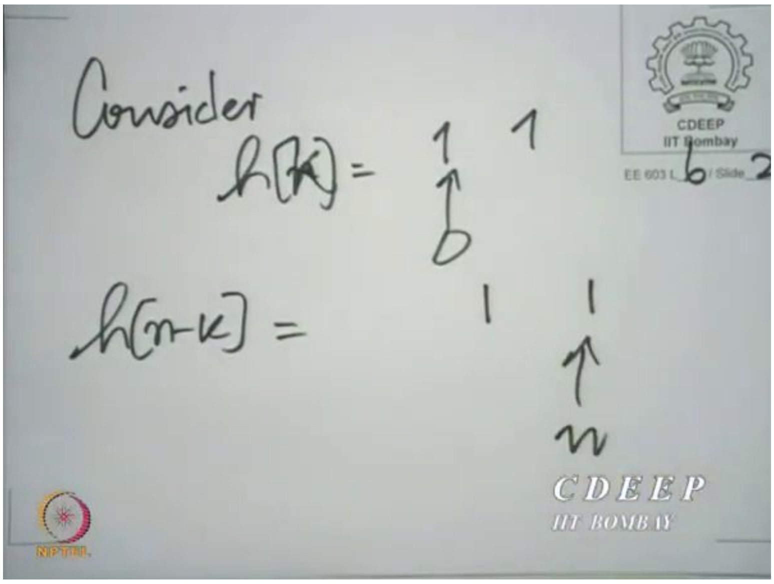
So, this appears at (-*n*) and this can go on and then reflection now, when you reflect (-*n*) goes to (+*n*), *h[1]* which appears here after (-*n*) will now appear before (+*n*). So, what we will do is, move this to the point *n* and then switch or mirror all the points around that point *n*, all right?

(Refer Slide Time: 05:01)



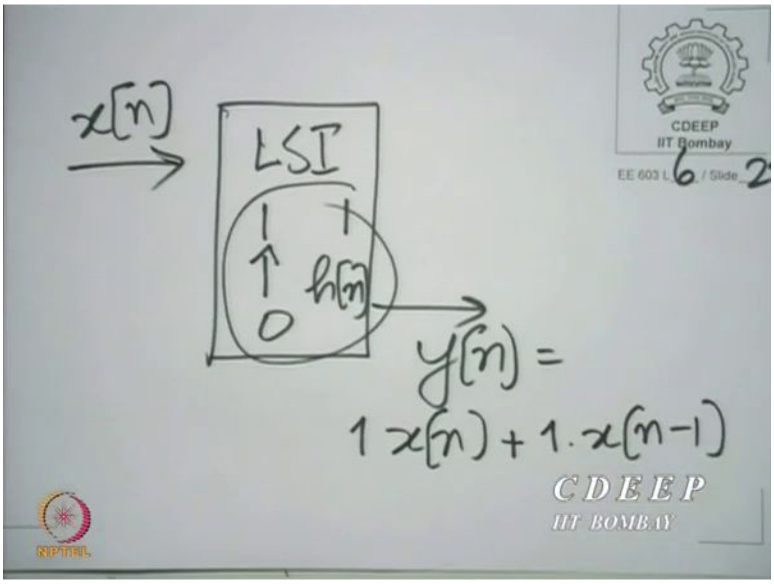
So what we have is as follows, *h[n-k]* is therefore, at *n* put *h[0]* at *n-1* put *h[1]*, *h[2]* and so on behind. And here you have *h[-1]*, *h[-2]* and this can go on. This was not too difficult to see generally.

(Refer Slide Time: 05:46)



But we can of course, fix our ideas by taking the same sequence that we did a minute ago for *h[n]*, namely, if we consider *h[n]* as we did there, or, you see, of course, here you can call it *h[k]*, if you like to be specific and *h[n-k]*, essentially, at *n* put *1* and then therefore, at *n-1* you will also have a *1*. Simple.

(Refer Slide Time: 06:37)



So, essentially what we are saying is that this *h[n]*, or *h[n-k]* more precisely has a *0* at the, has *1* at the point *n* and *1* at the point *n-1*. And therefore, if a, if a system had the impulse response given by this *h[n]*, if an LSI system had this *h[n]* as its impulse response. Remember, this tells me everything about the LSI system. Once I know this unit impulse response, I know everything about the LSI system.

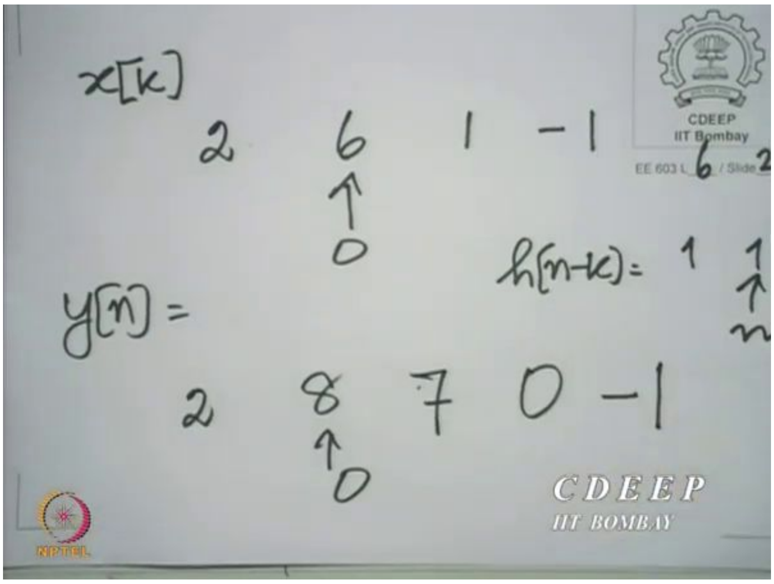
So, once I write this, I have specified it completely. I, if I have an *x[n]*, what I am saying is, *y[n]* is essentially $x[n]*1+1*x[n-1]$. So the impulse response also gives me an interpretation of what the system does. In fact, more particularly a finite length impulse response immediately gives us an interpretation.

The moment you have a finite length impulse response, a finite length impulse response means, an impulse response where the number of non-zero samples is finite. So here, for example, the number of non-zero samples in the impulse response is only *2*. So the beauty is that with this interpretation of the operation of convolution, I also have an interpretation for how the impulse response describes the system.

For a finite length impulse response it is very straightforward. It tells you that the output at any point becomes a linear combination of the input at that point and a few points around it. Now, the key issue is whether you involve only points from the so called past, or also points from the so called future, I am saying so called because past and future have a meaning when the independent variable is time, not otherwise.

But if the independent variable is time, it is very easy to see that if the impulse response has non-zero samples for negative n, then you are going to involve so called future samples. And if impulse response has non-zero samples only for non-negative *n*, that is *n = 0* and *positive n*, then the output at a given point depends only on the input at that point and on points before it, so to speak, points from the past, all right? We will come to that a little later.

(Refer Slide Time: 09:31)



But let us use this expression. Let us use this idea to recalculate the output that we did a few minutes ago. So there we have *x[k] = 2, 6, 1, -1,* is not it? And now you can visualize *h[n-k]*. So I will just write it down on the side. You see, when I want to calculate, so *h[n-k]* looked like this, *1, 1* at *n*.

So now I can, I can recalculate *y[n]* using this very easily. You see, I can visualize that when *n* is (-*1*), I can visualize this sequence being brought here. And the 1 comes in contact with 2, so you multiply them pointwise, so *1* gets multiplied by *2*, *2*, *2* plus this *1* gets multiplied by *0*, so the output is *2* here.

On the other hand, when *n* is *1*, I am sorry, when *n* is *0*, then this *1* clashes with a *6*, and this *1* clashes with a *2* and therefore you have *6+2*, that is *8*. Similarly, when *n* is *1*, you have this *1* colliding with this *1*, and this *1* with the *6*, so you have *7* there. And when *n* is *2*, you have this *1* colliding with (-*1*), so the product is (-*1*).

And this collides with *1*, so you have $1 * (-1) + 1 * 1$ that is *0*. And finally, when *n* is *3*, you have *1* colliding with a *0*, and *1* colliding with a (*-1*), and therefore, you have just a (*-1*) there. For *n*<*-1*, it is very easy to see that these two *1's* collide only with *0's* here. And for *n*>*3*, it is very easy to see again that these two *1's* collide only with *0's*. And therefore, there is no non-zero sum at all.

Now, it is convenient to give another mnemonic to this operation. And that mnemonic is to think of one sequence as being static. And the other sequence as being dynamic. You might think of this sequence as the passengers on a platform. And you might think of this sequence as the passengers inside the bogies of a train that moves.

And you might visualize that this train moves one step at a time. And there is a handshake between the passengers on the platform and the passengers on the train. At every point, the output is interpretable as the net effect that the train feels due to this handshake. So it is a combination, you know, some passengers are stronger, some passengers weaker, you might think of that, as the number put there.

Of course, some passengers make a handshake outwards, some passengers make a handshake inwards. That is the positive and negative part of it. So all together at every point, there is a net impact of all these handshakes, and the train moves one step at a time. And the net impact is interpreted as the product of the strength of the person inside with the strength of the person outside summed over all the passengers, which does make a handshake.

And of course, after sometime you run out of passengers. Now, this is easy to understand, when you have a finite number of passengers, a train has a finite number of passengers. And the platform also has a finite number of passengers. But it is not too difficult for us to extend the idea to a context where the train has an infinite length, and the number of passengers on the platform is also infinite.

It takes us a little more visualization to arrive at an interpretation for that context. Is that right? So this is another convenient mnemonic to interpret convolution. Is that right? Convolution, is a very fundamental operation. In the study of Linear Shift-Invariant systems. Convolution is an operation in its own right.

What I mean by that is, other than the fact that it occurs in the context of Linear Shift-Invariant systems in a very, very meaningful way. You can think of it as an operation between two sequences independent of the context, what I mean by that is, you could have taken two sequences multiplied them point by point to get an output sequence, that is also an operation between two sequences.

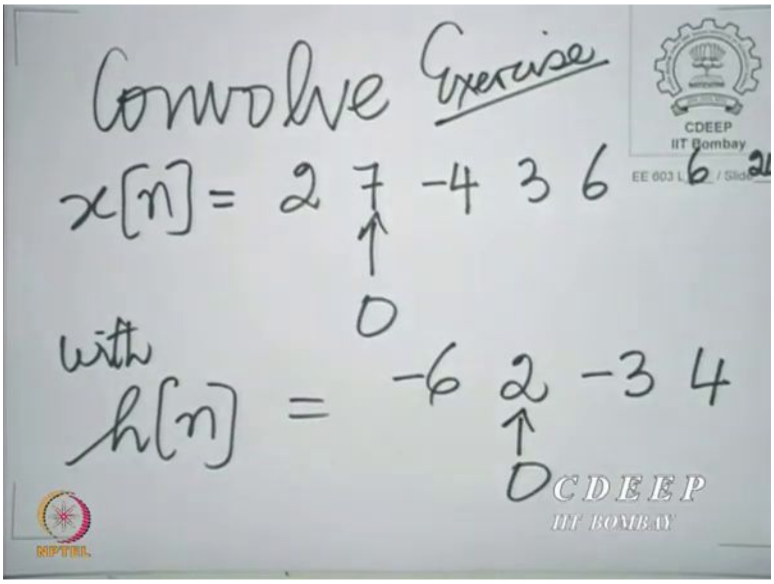
You could have taken two sequences, added them point by point and that gives me an operation between two sequences. Of course, those are what are called point operations. That means each point of the output involves only the particular point of the input and a particular, I mean, particular point of the two sequences.

If you add sequences point by point, then you are operating on a in a pointwise way, so the output is a pointwise function of the input and the impulse response or the output, if you do not want to call them impulse input an impulse response, if you have two sequences, the output is a pointwise function of the points of the two sequences. However, convolution is not a pointwise function.

In convolution, we must now see from this example, that in principle, all the points of the input, and all the points of the impulse response have come in to the picture, to create a point of the output, again, going back to the mnemonic of the train in the platform, the impact felt by the train at every move is dependent on all the passengers on the platform and all the passengers in the train. It is not passenger by passenger. Is that right?

So it is truly, convolution is truly an operation between two sequences and not pointwise at all. This must be emphasized and understood, very clearly. So are there any doubts before we proceed? It is a very important operation, and you must be absolutely clear how this is done.

(Refer Slide Time: 16:24)



Now, I put before you an exercise to do, and I also introduced the verb form of convolution, convolution is a noun, it is the name of the operation, the verb form is convolve. So convolve, *x[n] = 7, -4, 3, 6, 2,* with *h[n] = 2, -3, -6, 0,* and let me put another point there to make you work a little harder. So I will leave this for you as an exercise, obtain this convolution?

And we conclude this lecture with this exercise, and just give a feel of what we are going to do the next time as a trailer. The next time, we are going to take note of this relation between the input of the impulse response and the output, and we are going to fulfil our first promise. Namely, the whole reason why we want to look at linearity and shift invariance was that if I gave a complex exponential as an input, I would have expected a complex exponential to emerge.

But we said we need one more thing at stability. In the next lecture, we will begin by seeing what happens when we give a complex exponential as the input to an LSI system. And then see whether we are happy with the system being only LSI or we need something more, that would bring us to a few more properties that we desire of linear shift-invariances. Thank you.