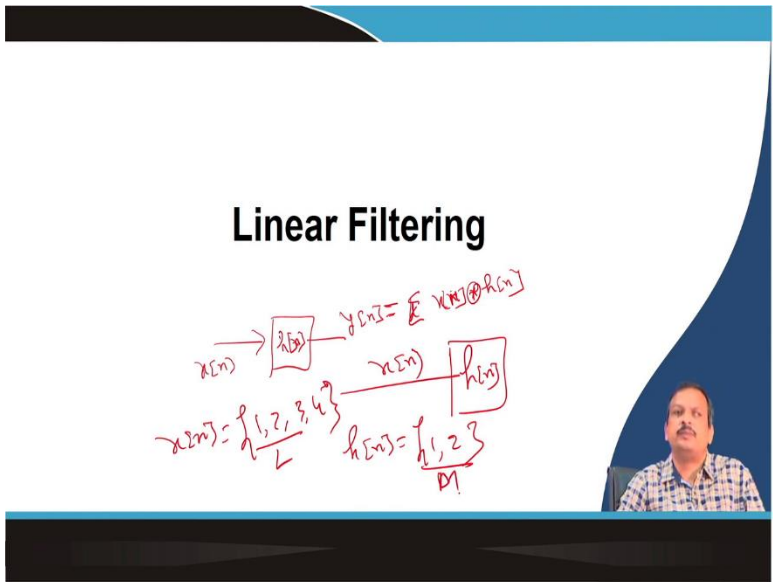
Signal Processing Techniques and Its Applications

Dr. Shyamal Kumar Das Mandal
Advanced Technology Development Centre
Indian Institute of Technology, Kharagpur

Lecture - 24
Linear Filtering

So, we have completed the properties of discrete Fourier transform. So, we have seen time reversal things, time shifting, frequency shifting, circular convolution, and all those things.

(Refer Slide Time: 00:36)

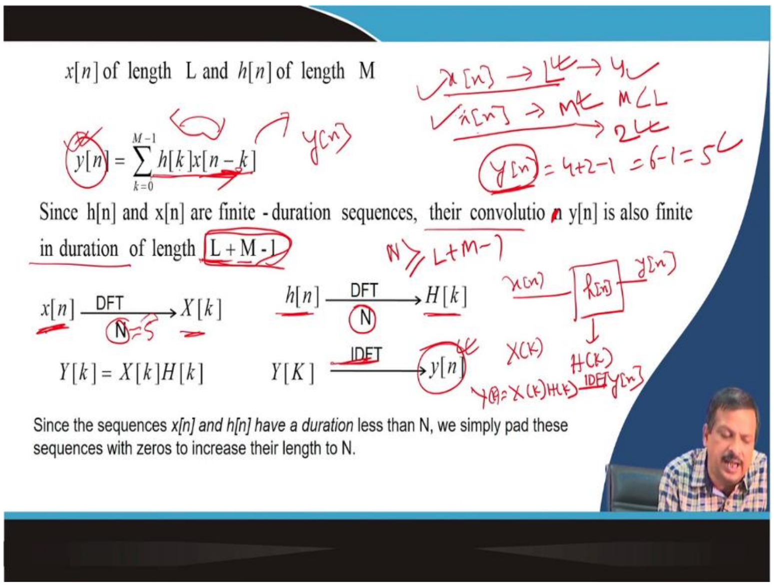


Linear Filtering

Today, we discuss the linear filtering view: What is linear filtering? Now, as you know, when I say linear filtering, that filter is nothing but a system. I said a filter is nothing but a system; if I pass a signal based on the response of the filter, the output will be created.

So, I know if x[n] is my input and the impulse response of the filter is h[n], then the output y[n] is nothing but a linear convolution of x[n], x[n] linear convolution of x[n] with impulse response of the filter, that I know ok.

(Refer Slide Time: 01:29)



$x[n]$ of length L and $h[n]$ of length M

$$y[n] = \sum_{k=0}^{M-1} h[k]x[n-k]$$

Since h[n] and x[n] are finite - duration sequences, their convolution y[n] is also finite in duration of length $\boxed{L+M-1}$

$$x[n] \xrightarrow[N]{DFT} X[k] \qquad h[n] \xrightarrow[N]{DFT} H[k]$$

$$Y[k] = X[k]H[k] \qquad Y[K] \xrightarrow{IDFT} y[n]$$

Since the sequences *x[n] and h[n] have a duration* less than N, we simply pad these sequences with zeros to increase their length to N.

Now, let us say I have an x[n] signal whose length is L, and I have a filter impulse response whose length is M or whose order is M. On the other hand, I can say that I have a system h[n] and I have a signal x[n], let us say x[n] is equal to 1, 2, 3, 4 and h[n] is equal to let us say 1 and 2. Then the length of the signal is L, and the length of the filter is M. So, I have a signal whose length is L, and I have a filter response whose length is M; let us say M is less than L.

Now I know the y[n] is nothing but a linear convolution of h[k] and x[n-k] or whatever x[n] h[n-k]. Now x[n] and h[n] both are the finite duration of the length of L and length of M. So, what is the length of this y[n]? If we calculate y[n], the length of the y[n] will be forgotten about their convolution. y[n] is also a finite duration whose length will be L plus M minus 1. So, suppose I have a signal length equal to 4, and I have a system M equal to 2. So, the length of y[n] will be 4 plus 2 minus 1. So, 6 minus 1’s means 5 or not.

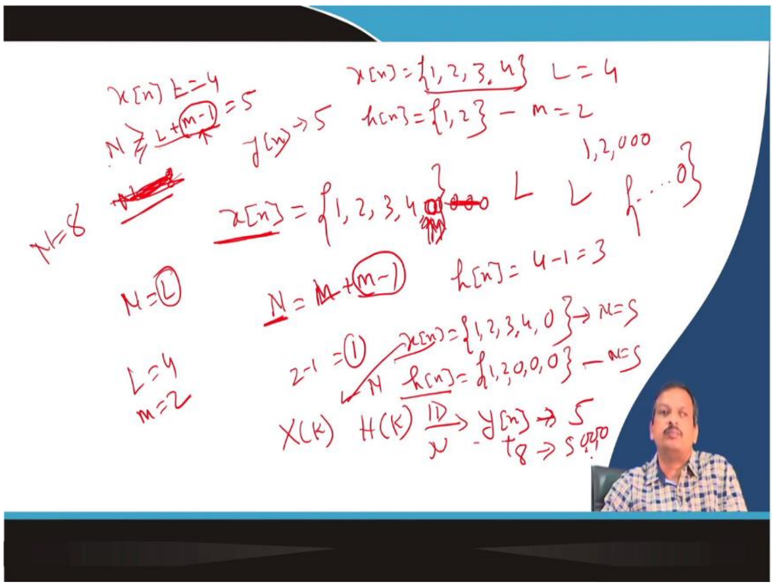
So, the length of y[n] you calculate, you calculate x[n] equal to 1, 2, 3, 4 and h[n] equal to this 1. You calculate the y[n] value of y[n], and you can get that it will be M plus n L plus M minus 1 order; just calculate the linear convolution. Take the example and calculate the linear combination convolution, and you get the order of y[n].

Now, let us say I want to do it instead of you knowing the time domain convolution is equal to the frequency domain multiplication. So, instead of having a filter, I have an application of x[n], and I get h[n] as the impulse response, so I get y[n]. Now, instead of doing that, let us say I calculate X(k), I calculate H[k]. So, X(k) is the DFT of x[n], and H[k] is the DFT of h[n] frequency response.

Then, in the frequency domain, I multiply both them X(k) and H[k]. Then, I take the inverse discrete Fourier transform, and I get y[n]. So, this is the Y(k) frequency response of y[n]. Now, if I apply I DFT, I get y[n]. So, let us know if the order is N, then what should be the value of N? I know the length of y[n] is equal to M minus L minus M minus 1. So, my N must be greater than equal to L plus M minus 1. So, my length of the DFT must always be greater than equal to L plus M minus 1.

So, when I say that, let us say this example, my N should be 5 in this case if the L is equal to 4 and M is equal to 2, then I know the length of the y[n] is equal to 4 plus 2 minus 1 which is nothing but a 5. So, my length of DFT is 5.

(Refer Slide Time: 06:23)



$x[n]\ L=4$

$N \geq L + \boxed{m-1} = 5$

$y[n] \to 5$

$x[n] = \{1, 2, 3, 4\}\quad L = 4$

$h[n] = \{1, 2\} - m = 2$

$N = 8$

$x[n] = \{1, 2, 3, 4, 0\ 0\ 0\ 0\}$ L L $1, 2, 0 0 0$ $\{\ldots 0\}$

$N = \boxed{L}$

$N = M + \boxed{m-1}$

$h[n] = 4 - 1 = 3$

$2 - 1 = \boxed{1}$

$x[n] = \{1, 2, 3, 4, 0\} \to N = 5$

$h[n] = \{1, 2, 0, 0, 0\} - N = 5$

$L = 4$

$m = 2$

$X(k)\ H(k) \xrightarrow[N]{ID} y[n] \to 5$

$8 \to 5\ 000$

So, I have a signal x[n] whose length is equal to 4 L is equal to 4 x[n] is equal to I said 1, 2, 3, 4 and I said h[n] is equal to 1 and 2 let us say. So, M is equal to 2, and L is equal to 4. So, I know my N is equal to L plus M minus 1, let us say 5. Let us say I take N as equal to 8, and I can take this as more significant than equal to. So, if I take N equal to 8, my length is 4.

So, I know that I have to create an x[n] whose length is equal to 8. So, what I have to do is 0 padding 1, 2, 3, 4. Let us first explain that n is equal to 5. Then, I explain that n is equal to 8. If it is n equal to 5, then I have to add 1 0 here.

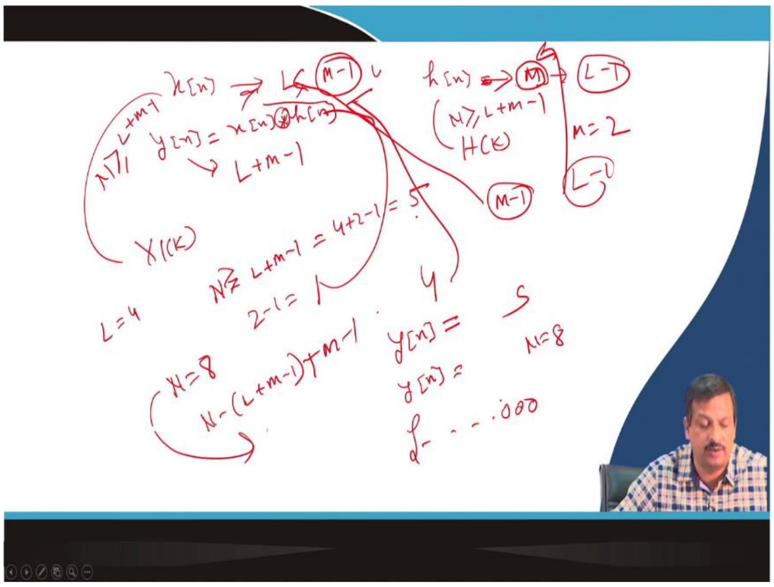
So, if I say how many 0 have to be added with x[n], if N is equal to L and if N is equal to L, then I know that 0 paddings are not required. Since it is nothing but an L plus M minus 1, then I have to M minus 1 number of 0 I have to add. Do you understand or not? Forget about N equal to 8. I said I have a signal whose length is L, I have to take DFT whose N is equal to L plus M minus 1, which is the length of the DFT, but I have a signal whose length is L.

So, I have to add M minus 1 number of 0 here. So, if L is equal to 4 and M is equal to 2, how many 0s do I have to add? M minus 1 2 minus 1 is equal to 1 0 I have to add. How many 0s do I have to add to h[n]? It is nothing but a L minus 1. So, it is nothing, but a 4 minus 1 is equal to 3. So, I have to do 1, 2, and 3. So, now, I have a sequence x[n] that is equal to 1, 2, 3, 4, 0, and I have an h[n] that is equal to 1, 2, 1, 0, 0, 0.

Now, I get this: N is equal to 5 here, and N is equal to 5. Now, I can calculate N point DFT. I get X(k), and I get H[k], then I take the N point inverse DFT, and I get y[n] whose length is equal to 5. But you get all the values if you computed y[n]. The last element will be 0; you can verify it. So, I said take x[n] is equal to 1, 2, 3, 4, take y of h[n] equal to 1 and 2, calculate the linear convolution, calculate y[n], which is the linear convolution.

Now you do 5 point DFT of x[n], calculate 5 point DFT of h[n], multiply their the out result and compute 5 point inverse DFT of that result; you get y[n]. You see, both y[n] are similar, except that the length of this y[n] is also 5, and this y[n] is also 5. If you take just N equal to L plus M minus 1. Now if you take N equal to 8 more than L plus M minus 1, that means this length will be 8, but you can get after 5 you can get 3 0 value you can calculate it.

(Refer Slide Time: 10:58)



So, in summary, again, I am saying if I have a signal x[n] whose length is L, I have a system h[n] whose length is M. Now, if I want to compute convolution if x[n] passes through h[n]. So, I get y[n], which is nothing but a convolution of x[n] convolved with h[n]; the length of y[n] will be L plus M minus 1.

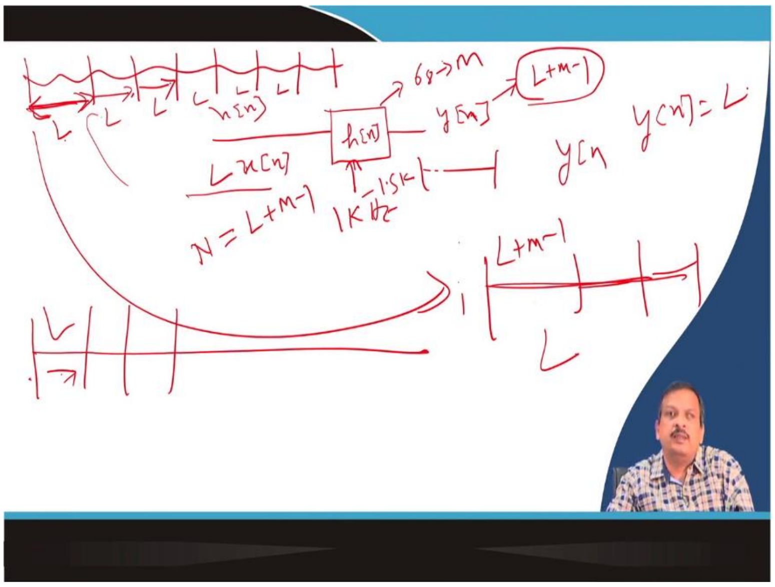
Instead of convolution, I want to calculate in the frequency domain. So, I calculate X(k), and then what should be N? The value of N should be greater than equal to L plus M minus 1. So, when I calculate H[k], the value of N should be greater than or equal to L plus M minus 1.

Since the M number of samples is only there, so I have to pad up L minus 1 0 here only L number of samples is there, I have to pad up M minus 1 is the minimum. But if it is N, it is more than that; also, I have to pad up. So, for example, suppose x[n] L is equal to 4, L is equal to 4, and M is equal to 2, then I know the minimum length of the DFT will be L plus M minus 1. So, 4 plus 2 minus 1 is nothing but a 5; it can be greater than that.

So, if N is equal to 5, then I have to add M minus 1, the number of 0 in x[n] and L minus 1, the number of 0 in h[n]. So, since M is equal to 2; that means, 2 minus 1 1 number of 0 I have to padded in x[n]. Now, if my N is equal to 8, then I know I have to pad up; how many? So, N minus L plus M minus 1 that plus M minus 1. So, I can say 1 plus 3 4 number of 0 I have to padded up.

Now, if you compute y[n] by linear convolution, the length of the y[n] will be 5. Now, let us say I have taken N equal to 8, and then if I compute in frequency domain method, So, after inverse DFT I get y[n] whose length is equal to N equal to 8. You can see that you get 5 values, and the last 3 will be 0; you can verify it, you can do it very verify it. So, this is a linear filtering view.

(Refer Slide Time: 14:19)



Another problem is the long data sequence. So, what is a long data sequence? Suppose I have a filter here. So, for the long data sequence, I have a filter here, h[n], whose order is, let us say, 68, and I have a signal x[n], which is a huge signal is there. So, if I took the whole signal at a time and convolved with this huge process.

So, instead of doing that, let us say I divided this long signal into a small block L. First, I am not initially discussing what will happen in the frequency response; I am doing how do I do it first? How do I recover that? I will discuss the problem first, and then I will discuss what will happen in a long data sequence frequency problem. So, I have divided the input signal into a small frame, and the L number of data points is ok. So, this is the L number of data. There is an L number of again data, and there is an L number of again data. So, I apply an L number of x[n] here, and I get y[n].

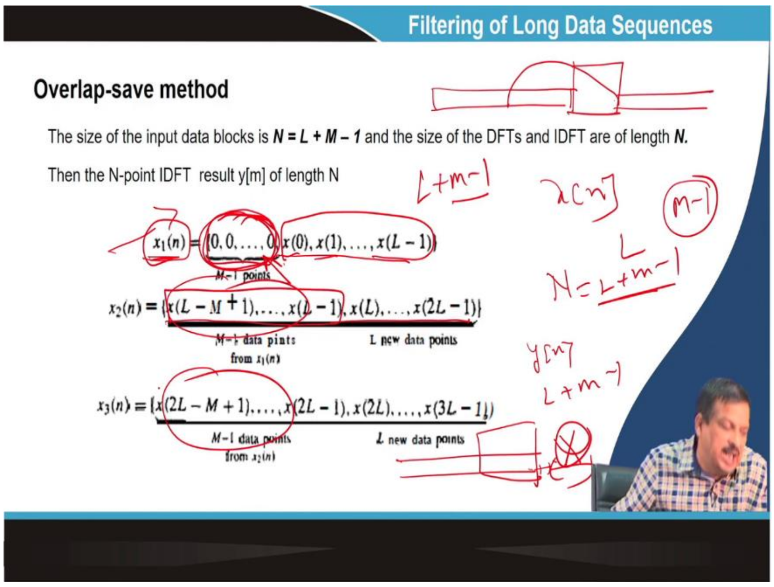
So, if this is my M and this is my L, then I know that if I produce that N point DFT. So, N is equal to the minimum L plus M minus 1, so the length of y[n] is L plus M minus 1. But for 1 frame, the length of data is L. I should get the L number of data because if I want to reproduce the signal after the signal passes through the filter, the signal length should not be very high.

Suppose I said I had designed this filter to cut off that 1-kilohertz signal 1 kilohertz to 1.5-kilohertz signal. So, the output data length and input data length must be equal because I have applied a signal. How do I know if the data length has changed?

So, in that case, I want to get back to this: Do you understand the physical problem? I have a lot of data. I take an L length of data at a time and pass it through a filter. Now, what is happening is that L length of data I pass through a filter, but the output I get is data whose length is L plus M minus 1, where M is the length of the filter.

Now, if I frame by frame, I analyze this as a frame. Now, the length of the frame is changed to L plus M minus 1. So, if I add them, then my signal length is different, and the output signal will be different length. So, how do I recover? Although I frame length by length, I want the L number of data in y[n], and the length of the y[n] should be L. So, how do I do that? That method is called the overlap save or overlap add method.

(Refer Slide Time: 17:56)



Filtering of Long Data Sequences

Overlap-save method

The size of the input data blocks is $N = L + M - 1$ and the size of the DFTs and IDFT are of length N.

Then the N-point IDFT result y[m] of length N

$x_1(n) = \{\underbrace{0, 0, \ldots, 0}_{M-1 \text{ points}}, x(0), x(1), \ldots, x(L-1)\}$

$x_2(n) = \{\underbrace{x(L-M+1), \ldots, x(L-1)}_{\substack{M-1 \text{ data pints} \\ \text{from } x_1(n)}}, \underbrace{x(L), \ldots, x(2L-1)}_{L \text{ new data points}}\}$

$x_3(n) = \{\underbrace{x(2L-M+1), \ldots, x(2L-1)}_{\substack{M-1 \text{ data points} \\ \text{from } x_2(n)}}, \underbrace{x(2L), \ldots, x(3L-1)}_{L \text{ new data points}}\}$

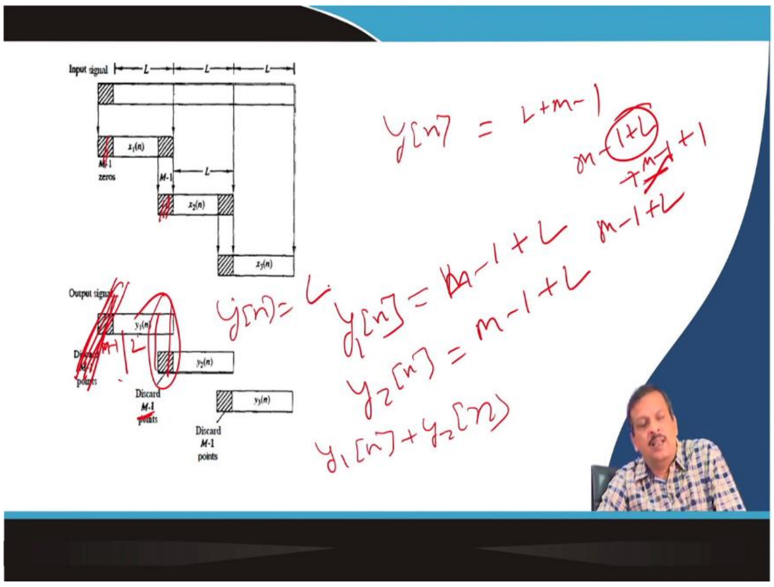
So, what is overlap save? Let us say I have a signal x1[n]. So, in the first frame, I have a signal x[n]. So, the first frame is x1[n] of length L. Now, if I want to do it, I have to do a frequency transform N point DFT, and the length should be L plus M minus 1. So, I am making it L plus M minus 1; that means the M minus 1 number of 0 has to be added to the signal.

So, I have a x(0), x(1), x(L – 1), and initially, I added M minus 1 number of 0. Why? Because why this y[n] is the length of L plus M minus 1; that means, suppose this is your painting station, and you apply a bamboo here if a pipe is here to paint. So, at the end, the pipe will come out, and then the length will be. So, initially, what will happen? How do we paint it? The convolution means flap it here. So, if the pipe is here, I flap the painting station in here first, and then I convolve the then I shift the pipe inside this thing.

So, initially, when you shift it then, you get this portion, but the rest of the portion is 0; the rest of the portion is 0. So, that is why, to avoid that initial aliasing part, what we will do is we put n number of 0 M minus 1 number of 0 at the initiation of the frame. Then, in the second frame, I put again M minus 1 number of 0 here initiation of the frame; then I took the L number of the data point.

So, each frame's first M number of 1 data point will be 0, understand or not? So, I have a x of. So, each frame's length is L plus M minus 1, L number of data points will be there, and M minus 1 number of 0 will be added up at the initiation of the frame.

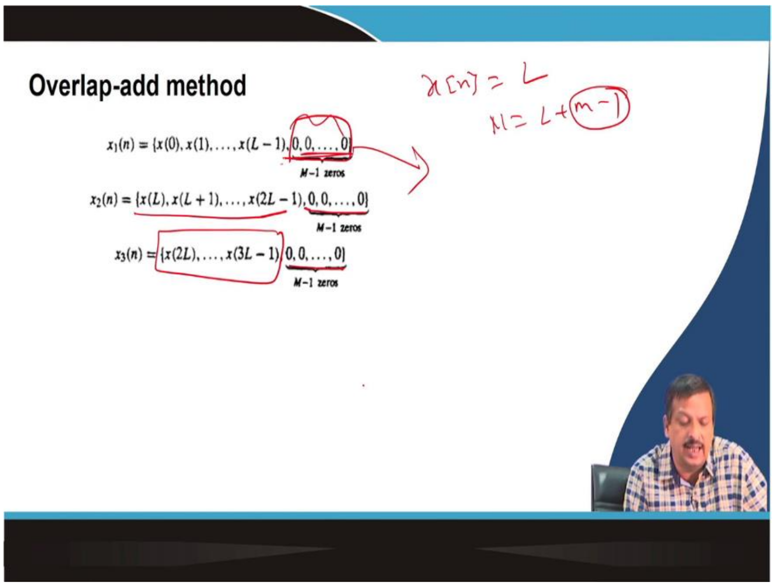
(Refer Slide Time: 20:34)



Then I compute y[n], and then what if I see that this portion is M minus 1 number of 0 x1[n] and this portion is 0. So, if I say that, then I calculate y[n], which is the length of L plus M minus 1, and then I add them. So, M minus of 1 point will be added together. So, overlap, so M minus 1 point overlap. So, the initial M minus 1 point will be discarded, then I add y 1. So, y1[n] and y2[n] will overlap by M, M minus 1 number of points. So, basically, y1[n] is the length of L. This is M minus 1, and this is L.

So, M minus 1 plus L and y 2 n is also the length is M minus 1 plus L. Now I said y1[n] plus y 2 n when I do it, so instead of saying that M minus 1 plus L plus M minus 1 plus L. So, M minus of 1 point will overlap with this length. So, basically, I will get M minus 1 plus L length. So, this first M minus 1, if I discarded, then I get the signal y[n] whose length is L. This is called the overlap save method.

(Refer Slide Time: 22:25)



Overlap-add method

$$x_1(n) = \{x(0), x(1), \ldots, x(L-1), \boxed{\underbrace{0, 0, \ldots, 0}_{M-1 \text{ zeros}}}\}$$

$$x_2(n) = \{\underline{x(L), x(L+1), \ldots, x(2L-1)}, \underbrace{0, 0, \ldots, 0}_{M-1 \text{ zeros}}\}$$

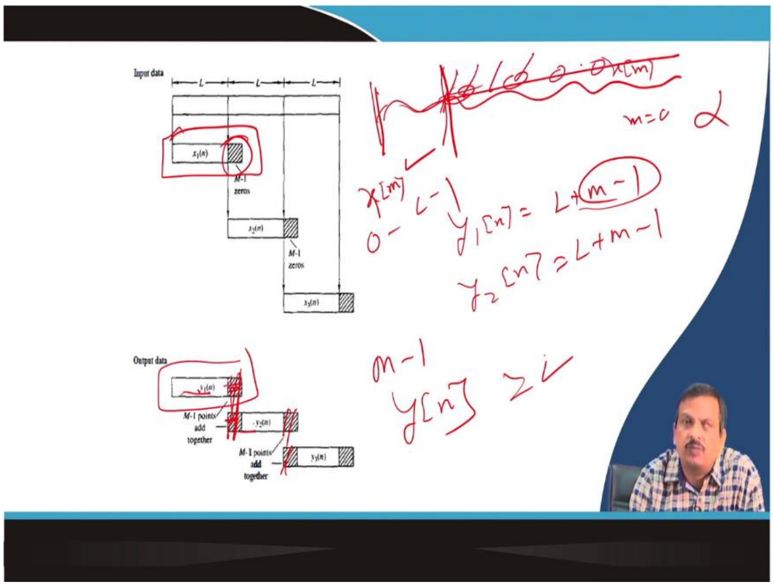
$$x_3(n) = \boxed{\{x(2L), \ldots, x(3L-1)}, \underbrace{0, 0, \ldots, 0}_{M-1 \text{ zeros}}\}$$

$x[n] = L$

$N = L + \boxed{m-1}$

Similarly, I can go for the overlap add method. So, instead of putting the 0 at the initial point, I put the 0 at the final point. So, I have a x[n] whose length is L, I have to make N is equal to L plus M minus 1. So, M minus of 1 number of 0 I added in here. So, in the next frame, I took an L data point and M minus 1 number of 0. Again L data point next frame and M minus 1 number of 0.

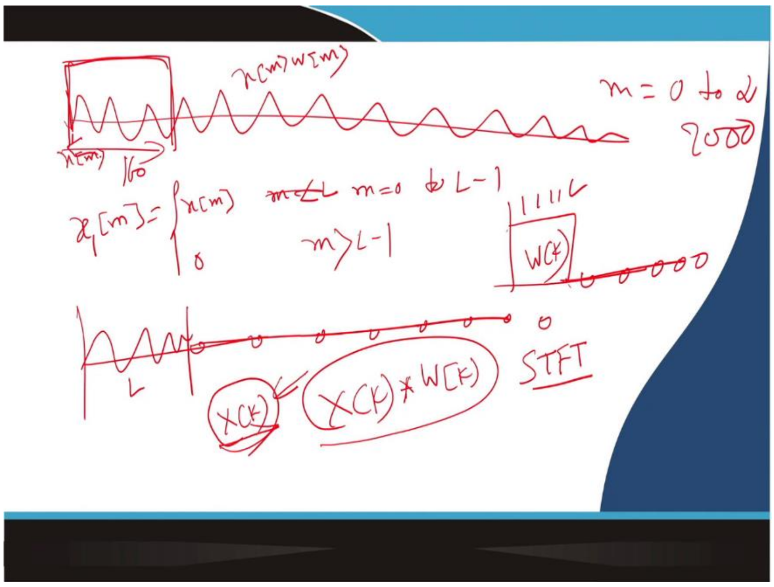
(Refer Slide Time: 23:02)



So, when I analyze x1[n] I get y1[n]. So, x1[n] M minus 1 number of 0 is there. I get y1[n], whose length is also L plus M minus 1, where M is minus. So, M minus 1 point, I will overlap with the y 2, which also has a length L plus M minus 1. So, I just overlap this M minus 1 point with L y 1 to y 2. So, this overlaps M minus 1 will be cancelled. So, I just overlap and add them overlap and add them; I get y whose length is L; is it clear; is it clear to everybody? So this is called the overlap-add method and overlap save method; overlap adds method and overlap-save method.

But there is another point. So, I have a long sequence of data when I say I cut the data with L number of points, what is the meaning? This means that I have an x[m] whose M varies from 0 to infinite. Now I take data which is, let us say, first frame x 1 M whose length varies from 0 to L minus 1. So, how do we do that? So, by default, I am saying that up to this, the signal exists. After this, the outside window signal is 0.

(Refer Slide Time: 25:00)


x[m]w[m]
m = 0 to ∞
2000
x[m]
160
$x_1[m] = \begin{cases} x[m] & m=0 \text{ to } L-1 \\ 0 & m > L-1 \end{cases}$
W(k)
L
X(k)
X(k) * W(k)
STFT

So, what I am saying is that I have a long signal. There is a long signal. I M M varies from signal x. Let us say the signal is x[m], m varies from 0 to infinity, let us say. Now I said I take L number of samples of this long number of samples. So, let us say I have a 2000 sample signal. Now I said 160 samples I have taken. So, how do I take 160 samples? That means, fast frame number 1 x[m] I have taken is equal to x[m], if m is less than L or equal to up to m is greater than m is 0; m is 0 to m equal to 0 to L minus 1 else where it is 0. So, if m is greater than L minus 1, then it is 0.

So, that means I have only one signal in this portion, and the rest of the side signal is 0. Now, when I say that, what is happening? The problem is that x 1 m has a length L. If I analyze the frequency domain, I get X(k); if I take less than L point DFT, I have taken. But X(k) when I take X(k) is not only X(k) but also I transform the frequency domain response of this cutting point. What do you mean by cutting point? Cutting point means a rectangle that has a 1 up to L 0 elsewhere.

So, the frequency response after DFT that I will get I will get not only X k, but also X k is convolved with, let us say, that is called W k; that is the window function. Because in the time domain, it is x[m] multiplied by w of m, in the frequency domain, it will be convolution. So, I am not getting the exact signal representation in the frequency domain that is called STFT short-term Fourier transform, short-time Fourier transform or short-term Fourier transform, whatever short-time Fourier transform. So, time is short, and that is why I said short-time Fourier transform.

So, next week we discuss short-time Fourier transform details we discuss short-time Fourier transform ok.

Thank you.