Principles of Digital Communications

Prof. Shabbir N. Merchant
Department of Electrical Engineering
Indian Institute of Technology, Bombay

Lecture - 30
Differential Pulse Code Modulation DPCM - I

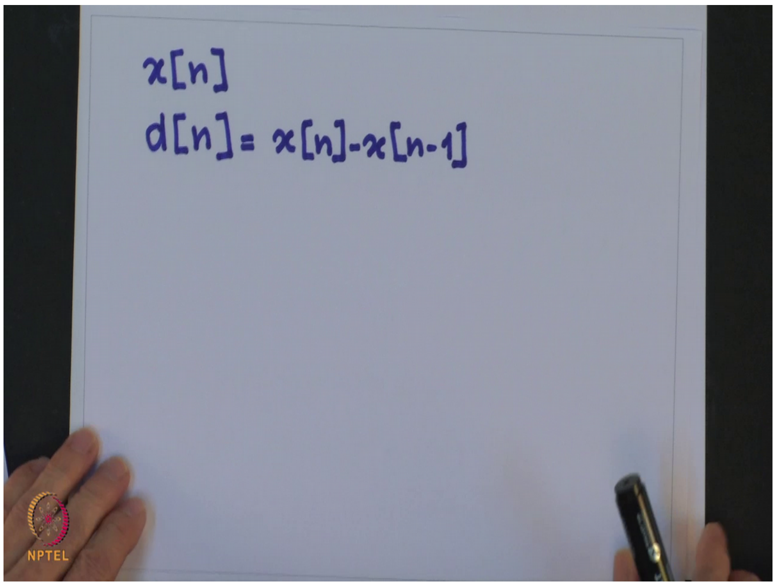
The different schemes of quantization which we have studied so far fall in the category of what is known as scalar quantization. In scalar quantization the samples are quantized independent of other samples. There is another category of quantization which is known as vector quantization. In vector quantization a group of samples are quantized together.

Now, a vector quantizer will provide a lower mean squared quantization noise compared to the scalar quantizer for the same number of average bits used to represent a sample. But implementation of vector quantizer is more complex than the scalar quantizer. The reason for better performance of vector quantizer compared to a scalar quantizer is that vector quantizer exploits the statistical properties of the samples which are being quantized. To be more specific it exploits correlation among the samples to be quantized. There is a form of scalar quantization which also exploits correlation among the samples to be quantized and that scheme is known as differential encoding.

Let us look at the motivation for this differential encoding. For any quantizer the quantization noise is directly related to the quantization interval and this quantization interval is directly related to the dynamic range of the input for the given number of quantization levels. Now, for a uniform pdf the dynamic range will affect the variance. So, the quantization noise is directly related to the variance of the input to the quantizer. This is the basic philosophy which is used in differential encoding.

So, in many sources of interest the sample source output which we will represent as x n, n denotes the time index.

(Refer Slide Time: 03:26)

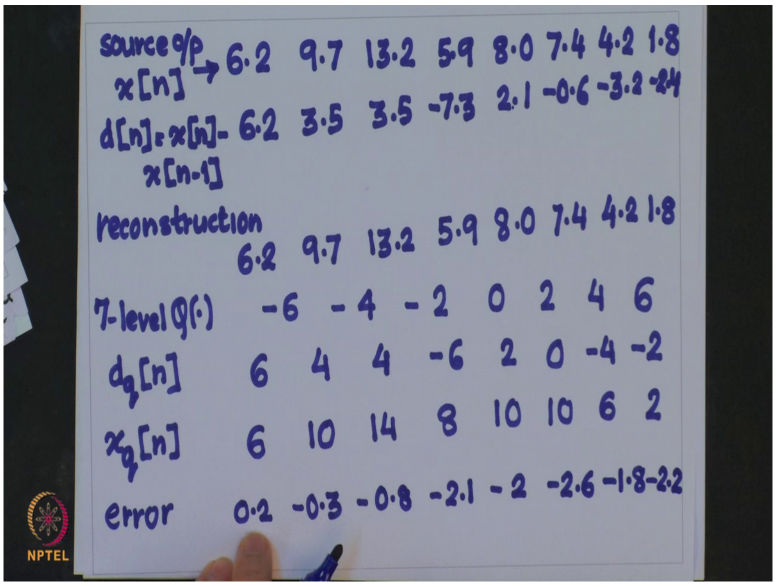


$x[n]$

$d[n] = x[n] - x[n-1]$

This sample source output does not change a great deal from one sample to the next. What this means that both the dynamic range and the variance of the sequence of the differences defined as follows are significantly smaller than that of the source output sequence itself. Furthermore for correlated sources the distribution of this differences that is d is highly peaked at 0. To understand this in a better way let us take one example.

(Refer Slide Time: 04:37)



source o/p $x[n] \rightarrow$	6.2	9.7	13.2	5.9	8.0	7.4	4.2	1.8
$d[n] = x[n] - x[n-1]$	6.2	3.5	3.5	-7.3	2.1	-0.6	-3.2	-2.4
reconstruction	6.2	9.7	13.2	5.9	8.0	7.4	4.2	1.8
7-level $Q(\cdot)$	-6	-4	-2	0	2	4	6	
$d_q[n]$	6	4	4	-6	2	0	-4	-2
$x_q[n]$	6	10	14	8	10	10	6	2
error	0.2	-0.3	-0.8	-2.1	-2	-2.6	-1.8	-2.2

Let us say I have source output which I denote by x n. So, this x n denotes both the sequence and also a sample of that sequence depending on the contexts. Let us assume these are the values which are being generated by this source.

Now, if you compute the difference of the sequence obtained by taking the difference as current sample minus the previous sample the sequence which we will generate is 6.2, I presume that x 0 is known in this case we will assume it to be 0 without loss of generality then I take the difference between this sample and the previous sample, so this turns out to be 3.5. Similarly this sample and this turns out to be 3.5, this would be minus 7.3 2.1 minus 0.6 minus 3.2 and minus 2.4.

Now, given this different sequence we can see that it is possible for me to reconstruct back in the lossless manner my original sequence by taking the current sample from this different sequence and adding to the previous reconstructed value of the sample. So, in this case its simple. So, reconstruction will give me. So, I start this will be 6.2 itself because the initial value is 0 then I take this sample different sample and add it to the previous value which I have reconstructed. So, this turns out to be 9.7, this 3.5 gets added to 9.7 so 13.2, this gets added to 13 point know 13.2, so this is 5.9, this gets added to this so 8.0, this will get added to this so I will get 7.4, 4.2 and 1.8. So, what I get is the perfect reconstruction.

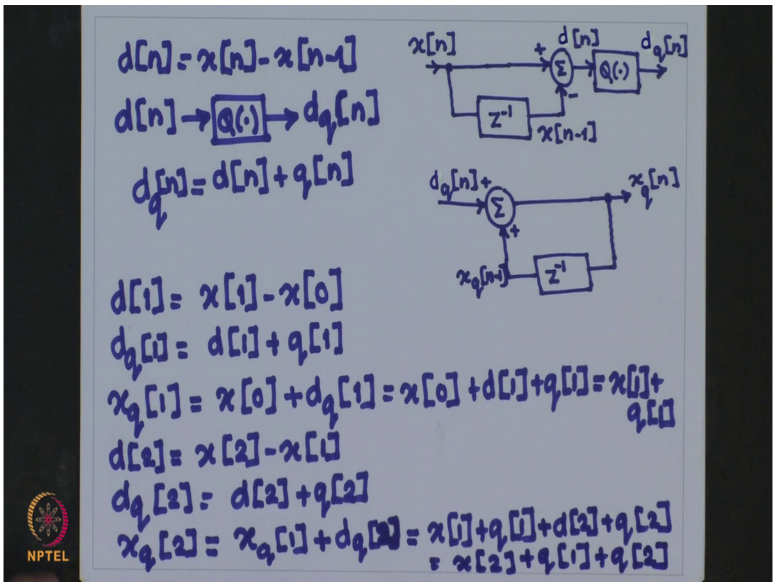
But there is little flaw in the argument that in practical scenario when I want to transmit the different sequence I will not be able to transmit with infinite precision we are using digital transmission. So, what I will have to do is that I will have to quantize this different sequence. So, let us assume a 7 level uniform quantizer and I presume that my reconstruction levels are minus 6, minus 4, minus 2, 0, 2, 4, 6. Using this reconstruction level let us try to see what is going to happen to the different signal when I transmit. So, the different signal which I am going to transmit. Now, is going to be quantized and that value will be 6.2, this will be equal to 6, this will be equal to 4, this will be equal to 4, this would be equal to minus 6, this would be equal to 2, this would be equal to 0, this would be equal to minus 4, and this would be equal to minus 2.

Now, what I do at the receiver is basically I follow the same strategy of reconstruction take my error signal or different signal. So, at any particular instant I will take the different sample and add it to the previously reconstructed value. So, if I do this I get my reconstruction I denote that reconstruction as x q, it is not going to be the same as x n you will see very soon. So, the first case is going to be 6 then I take this and add it to this, so it becomes n, next addition of this 14 this gets added to this, so 8, 10, 10, 6 and 2. Now, let us calculate the error which I am going to get between this sequence and the original sequence, correct.

So, if I calculate the error I have here 0.2 the difference between this and this. So, 9.7 minus 10 is minus 0.3, this is 0.8, minus 2.1, minus 2, minus 2.6, minus 1.8 and minus 2.2. Something interesting thing has happened. Initially I will find that the error terms are quite low, but as I keep on progressing the error keeps on building up, correct.

So, let us try to understand this process little more in depth and see what is the what is happening exactly. So, in our case now, what is really happening is something like this.

(Refer Slide Time: 11:22)



x[n]
+
d[n]
Σ
Q(·)
d_q[n]
−
z⁻¹
x[n-1]
d_q[n] +
Σ
x_q[n]
+
x_q[n-1]
z⁻¹

$d[n] = x[n] - x[n-1]$

$d[n] \rightarrow \boxed{Q(\cdot)} \rightarrow d_q[n]$

$d_q[n] = d[n] + q[n]$

$d[1] = x[1] - x[0]$

$d_q[1] = d[1] + q[1]$

$x_q[1] = x[0] + d_q[1] = x[0] + d[1] + q[1] = x[1] + q[1]$

$d[2] = x[2] - x[1]$

$d_q[2] = d[2] + q[2]$

$x_q[2] = x_q[1] + d_q[2] = x[1] + q[1] + d[2] + q[2]$
$= x[2] + q[1] + q[2]$

We have an x n sequence and we obtain their different sequence by taking the difference between the current value of the sample and the previous value of the sample which is denoted by this block. So, this z minus 1 denotes the delay of 1 unit. So, I get my d n then I pass it through a quantizer and I get my quantized value of the difference. So, this is what happens at the transmitting end, correct.

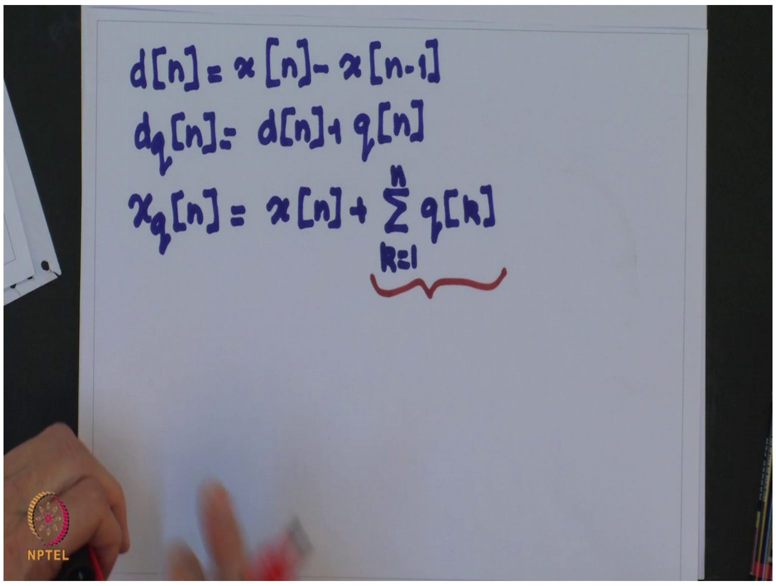
Now, at the receiving end what I am doing is that I take my different sequence and to that sequence I add the previous reconstructed value correct and how do I get the reconstructed value at any particular instant is the different sample to that I add the previous reconstructed value I get this. So, I delay this I get this point, ok. So, what is happening is that I am getting my d n is equal to x n minus x of n minus 1, and this d n is passing through a quantizer to give me d q n, and I model this process of quantization as a additive noise. So, in that case my d n plus q n which is a quantization noise this should be equal to d q n.

Now, let us look at the first different sample which I will generate is a d 1 which is equal to x 1 minus x 0, as I have said earlier x 0 is known at the receiver we can assume also to be 0 it is not going to make any difference. So, this d one is going to pass through the quantizer to get me d q which is nothing but the original value plus the quantization error. Now, at the receiver I am going to reconstruct my x 1 which I denote by x cube this would be equal to my previous reconstructed value in this case is x 0 it could be 0 that will make a difference and the force different sample this I can rewrite it as.

Now, this plus this is equal to x 1. So, the first reconstructed value has suffered the quantization error q 1. Let us take the second different sample in the sequence which would be equal to. So, your d q would be equal to, so your the reconstructed sample at the receiver would be equal to, the previous reconstructed sample plus there are difference this is 2, sorry this is 2. So, this is equal to this is equal to x 1 plus q 1, and d q 2 is d 2 plus q 2.

Now, from this we get this is equal to x 2 plus q 1 plus q 2. So, if we continue this process and try to find out all the reconstructed samples what we would get is something like this, d n will obtain like this, my d q n is equal to d n plus q n and my reconstructed sample at the receiver would be equal to.

(Refer Slide Time: 16:29)

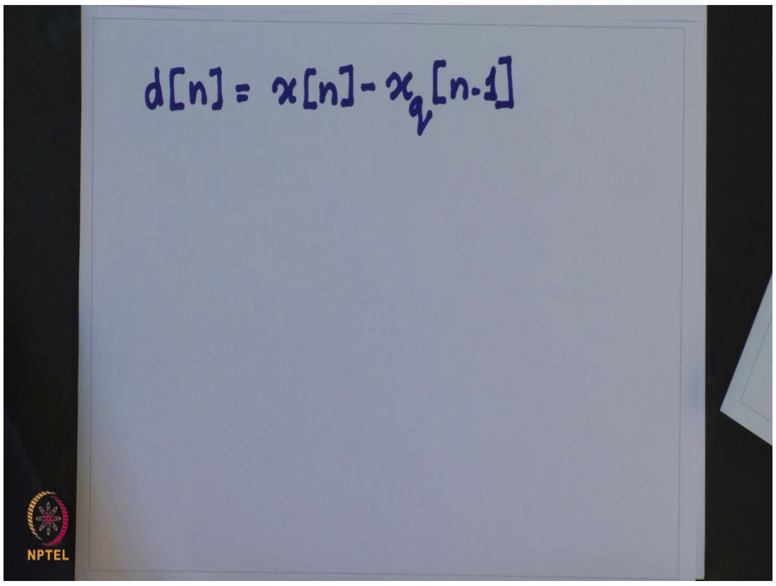

$$d[n] = x[n] - x[n-1]$$
$$d_q[n] = d[n] + q[n]$$
$$x_q[n] = x[n] + \underbrace{\sum_{k=1}^{n} q[k]}$$

So, there is a interesting thing we have got we see that the this is the quantization error which occurs in the reconstruction of the nth sample of the sequence the original sequence x n.

So, this quantization error accumulates as the process continues. Now, theoretically if the quantization error process is a 0 mean process then the errors will cancel each other in the long run. But in practice often long before that can happen the finite precision of the machine causes the reconstruction value to overflow. So, why is this happening? It is important to note that the encoder at the transmitter end and the decoder that is the reconstruction at the receiver end are operating with different pieces of information the encoder generates the different sequence based on the original sampled values, while the decoder adds back the quantized difference onto a distorted version of the original signal correct.

So, now, this problem can be solved by forcing both the encoder and decoder to use the same information during the differencing and reconstruction operation. Now, remember the only operation available at the receiver about the sequence x n is the reconstructed sequence x cube n. Now, at this information is also available at that transmitter end, we can modify this differencing operation to use the reconstructed value of the previous sample instead of using the previous sample itself that is what we will do is that we can obtain d n as x n minus x cube n minus 1.

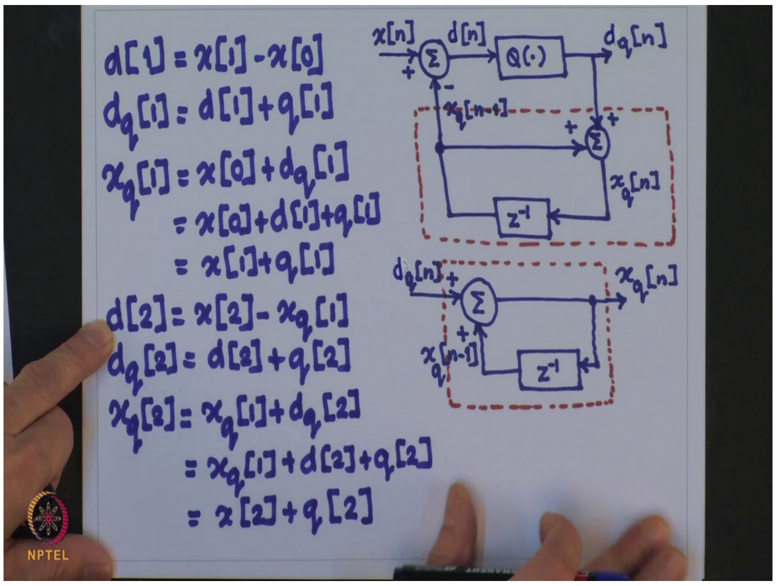
(Refer Slide Time: 19:30)



$$d[n] = x[n] - x_q[n-1]$$

So, let us see if you follow this strategy what would happen.

(Refer Slide Time: 19:49)



$$d[1] = x[1] - x[0]$$
$$d_q[1] = d[1] + q[1]$$
$$x_q[1] = x[0] + d_q[1]$$
$$= x[0] + d[1] + q[1]$$
$$= x[1] + q[1]$$
$$d[2] = x[2] - x_q[1]$$
$$d_q[2] = d[2] + q[2]$$
$$x_q[2] = x_q[1] + d_q[2]$$
$$= x_q[1] + d[2] + q[2]$$
$$= x[2] + q[2]$$

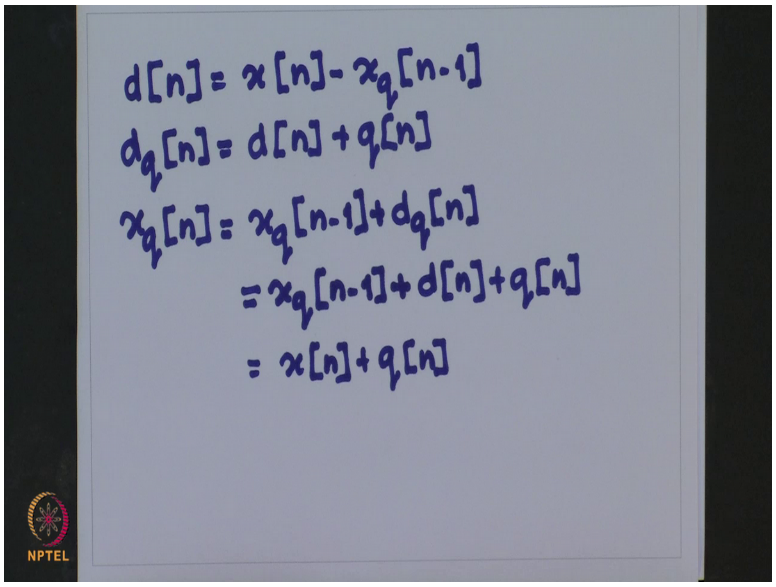
So, I have at the receiver, this block. Now, what I am saying is that since this reconstruction can also be done at the transmitting end because it is using for reconstruction at the receiver end only the different sequence and which is also available at the transmitting end. So, this block this red block which I have marked in dotted can be put at the transmitting at itself correct. Now, if we do this let us see what are we going to get?

So, now, my d 1 is going to be x 1 minus x 0 again I assume that x 0 is known to the receiver it could be assumed to be 0. Now, this is going to be quantized, so I will get as a quantization noise. So, your reconstructed value at the receiver would be this is known at the receiver. So, the quantization error which I get for the first sample is the same as I got in the earlier case. Now, let us look for the second different sample.

Now, here I use the current sample and I use the previously reconstructed sample and then I pass it through a quantizer. Now, I reconstruct at the receiver as follows. My previously reconstructed value plus my received difference signal this I can rewrite it as. And from here you can see that this is nothing, but x 2 plus q 2 interesting, I get only one term out here which is going to bring about the distortion in the reconstruction of the second sample unlike the previous case where I had q 1 also.

Now, if I proceed in this manner what I will get is as follows.

(Refer Slide Time: 23:02)

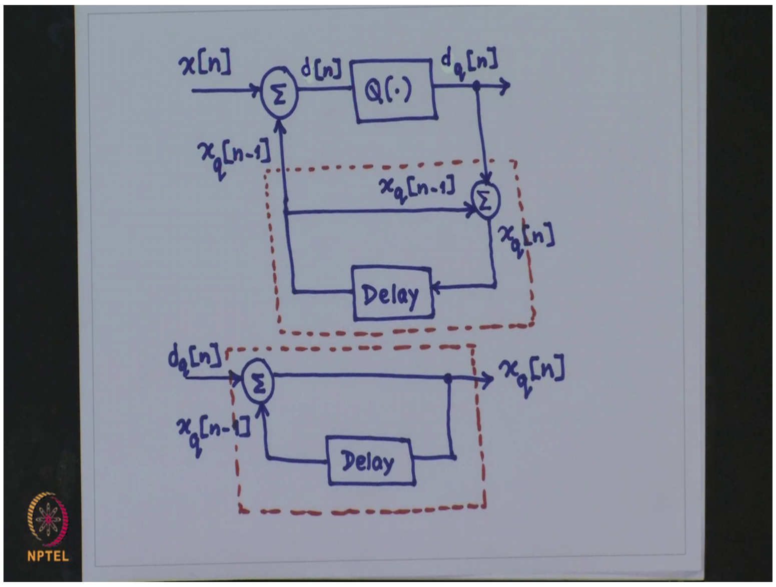


$$d[n] = x[n] - x_q[n-1]$$
$$d_q[n] = d[n] + q[n]$$
$$x_q[n] = x_q[n-1] + d_q[n]$$
$$= x_q[n-1] + d[n] + q[n]$$
$$= x[n] + q[n]$$

My d n will be given by this expression this will get quantized. And my reconstruction at the receiver would be as follows. I take my previously reconstructed value to that I add the current different sample which I can rewrite it as. Now, this plus this is nothing, but equal to x n plus q n. So, at the nth iteration there is no accumulation of the quantization noise, in fact, the quantization noise in the reconstructed sequence is the quantization noise incurred by the quantization of the nth difference.

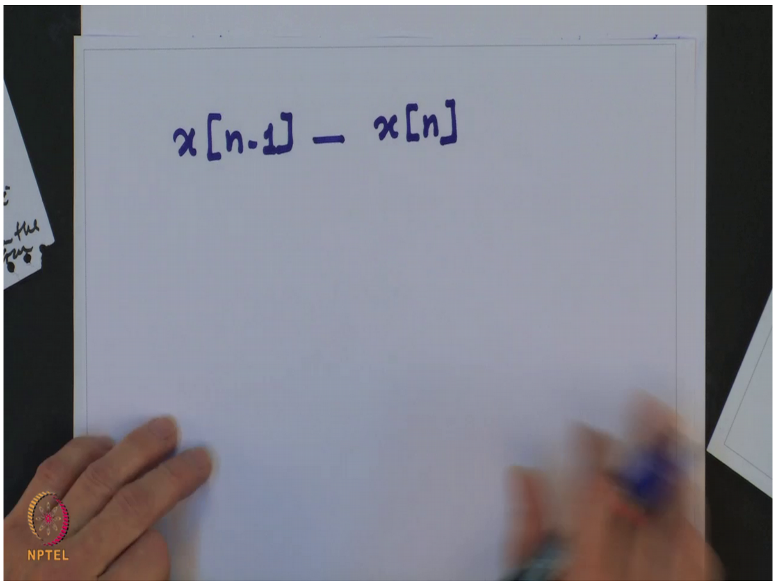
So, the quantization error for the different sequence is substantially less than the quantization error for the original sequence therefore, this procedure will lead to the overall reduction of the quantization error. So, what we could say that if you are satisfied with the quantization error for given number of bits per sample then we can use fewer bits with a differential encoding procedure to obtain the same distortion.

(Refer Slide Time: 25:16)



Now, in our discussion so far what we have said is that we are using this kind of differential encoding scheme. So, in this scheme we would like this d n to be as small as possible, correct. What this implies that this x cube should be as close as possible to this x. However, this x cube is the reconstructed value of x n minus 1, correct.

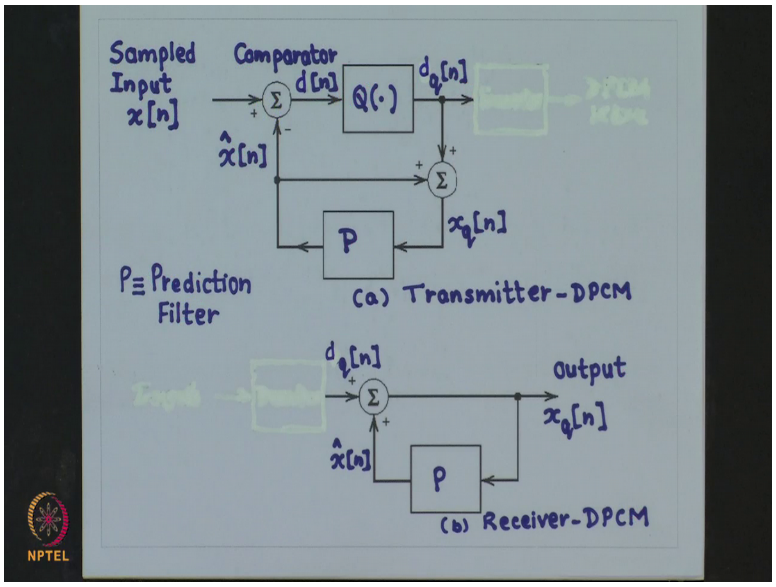
(Refer Slide Time: 25:46)



$x[n-1] - x[n]$

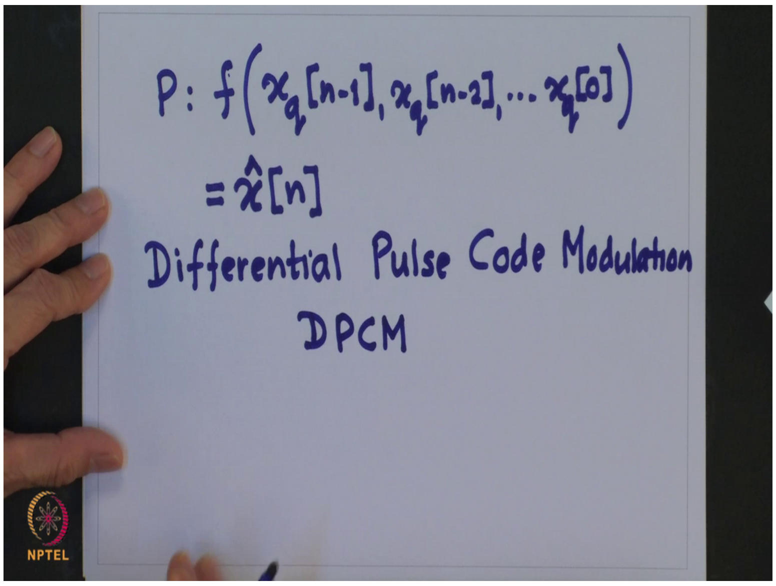
So, what we would like that this x cube should be very close to x n minus 1. Now, unless this x n minus 1 is very close to x n some past values of the reconstructed sequence can often provide a better prediction of this x n. So, let us modify this figure to replace the delay with a block called a predictor block and we will get this figure as shown here.

(Refer Slide Time: 26:35)



Now, this p out here predictor block is nothing, but a function of the reconstructed past values and this let us call as x hat n. So, this is the predictor which is obtained as a function of this reconstructed past value.

(Refer Slide Time: 26:44)



$$P: f(x_q[n-1], x_q[n-2], \dots x_q[0])$$
$$= \hat{x}[n]$$

Differential Pulse Code Modulation

DPCM

So, this basic differential encoding system is known as differential pulse code modulation in short, DPCM. So, this DPCM consists of two blocks, one is the predictor block and the other is the quantizer block.

We will study the predictor block and the quantizer block design and see how they function in a differential encodings system, and that we will do next time.

Thank you.