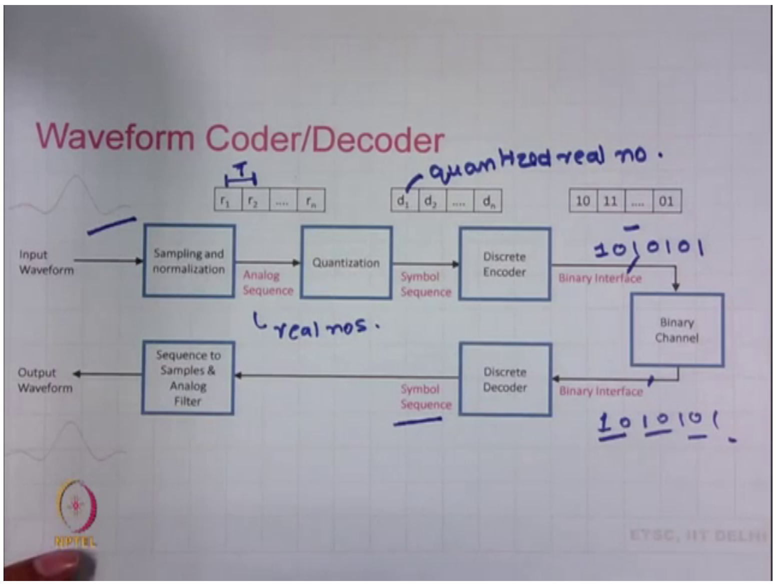
Principles of Digital Communication
Prof. Abhishek Dixit
Department of Electrical Engineering
Indian Institute of Technology, Delhi

Lecture – 18
Waveform Coding

Good morning, welcome to the new lecture and in this lecture we will be talking about Waveform Coding. So, what is waveform coding?

(Refer Slide Time: 00:30)



Waveform coding deals with the subject of converting an input waveform; so, we have this input waveform into a binary strip ok. So, this is the objective of waveform coder you have an analog waveform and you convert this analog waveform into a binary stream. So, binary stream is a sequence of 1's and 0's.

So, as you would know that many natural sources are analog sources they produce analog waveform and the job of waveform coder is to convert those analog waveforms into a binary stream. So, let us look at the blocks of a waveform coder. So, first we have the sampling and normalization process. The sampling and normalization process converts this analog waveform into an analog sequence ok. So, sequence of real numbers right, what is the sequence? So, we can understand sequence as a bunch of numbers right.

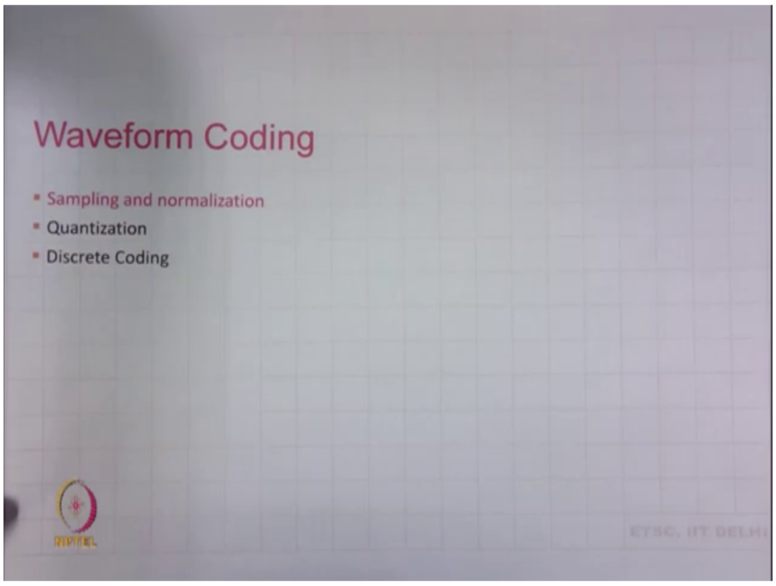
And these numbers are real numbers right, what is the time difference between these real numbers? The time difference between these real numbers, let us say is T and this T depends upon this sampler, we will see about this in more details later on ok. Then we have a quantizer; the quantizer converts these real numbers into symbols. What is a symbol? Symbol is nothing, but it is a quantized real number so, that is the job of a quantizer.

It just rounds off these real numbers to whatever precision it is required ok. Then you have a discrete encoder, discrete encoder converts these symbols into binary digits ok. We will see in this waveform coder about these 3 processes sampling and normalization quantization and discrete encoding then had the output of a waveform coder we have a binary sequence available. So, sequence of 1's and 0's and this sequence in 1's and 0's goes through a binary channel.

So, at the output of that binary channel, we have already seen what that binary channel is in the introductory lecture you can look at it again at the output of a binary channel again, we have a binary sequence if we are lucky we will be having the same binary sequence as was fed to the binary channel. Then discrete decoder converts this binary sequence again into symbol sequence and then this symbol sequence passes through an analog filter which converts the symbol sequence into a waveform.

So, most of this processes are almost covered in the courses like signals and systems. So, in this course we will not be looking at each block in that much detail, I will assume that you have already seen this and we would just try to present the flavors of these blocks from the point of view of digital communication system. So, let us look at what we are going to study now in more detail.

(Refer Slide Time: 03:55)

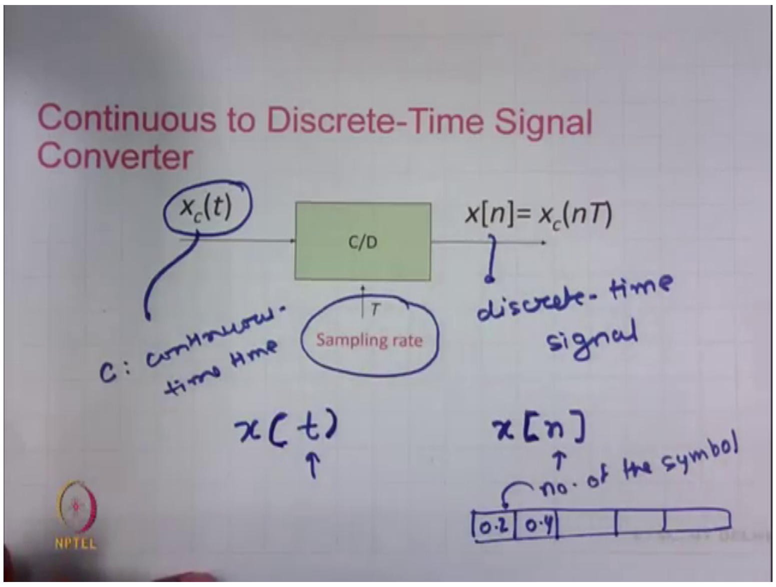


Waveform Coding

- Sampling and normalization
- Quantization
- Discrete Coding

So, today we will have a basic recap of this sampling and normalization procedure we will see, what are the key steps there we look at quantization in much more detail then is covered in the basic courses and then we will look at discrete coding. So, let us start with sampling and normalization this is what we are going to see.

(Refer Slide Time: 04:18)

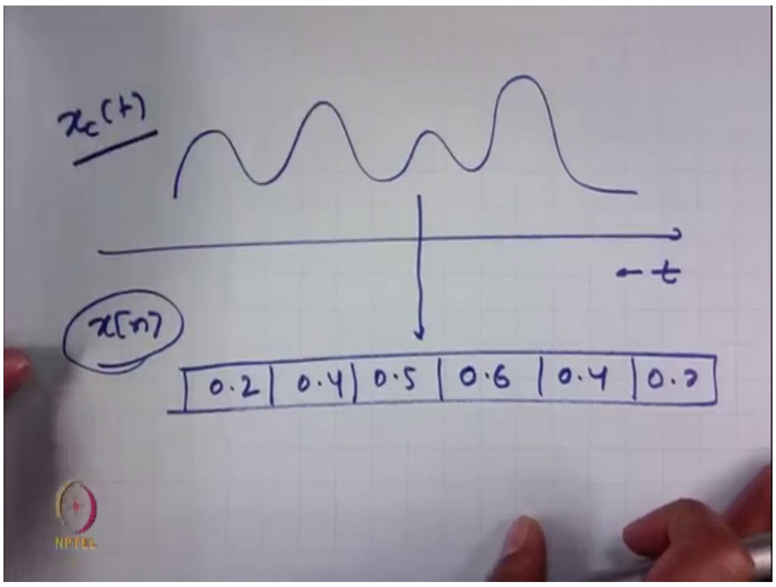


Continuous to Discrete-Time Signal Converter

So, what is the job of a sampler and normalization unit it can be understood as a block which takes in a continuous time signal like x c t. So, c here represents that this is a continuous time signal. So, it takes in a continuous time signal and, it converts this continuous time signal into a discrete time signal. So, this is x n is a discrete time signal and you must have known by convention this continuous time signal chooses an independent variable t.

Whereas, a discrete time signal by convention chooses an independent variable n because n represents not the time, but it represents the number of the symbol ok. So, n is the number of the symbol. For example, if you have an array and you have some values here. So, n represents the index of this array whereas, t is time. The second difference in the notation is when we are referring to continuous time signal, we use this parenthesis, but when we refer to discrete time signal we use these square brackets ok. So, this is the difference between continuous time signal and discrete time signal by notation.

(Refer Slide Time: 06:10)

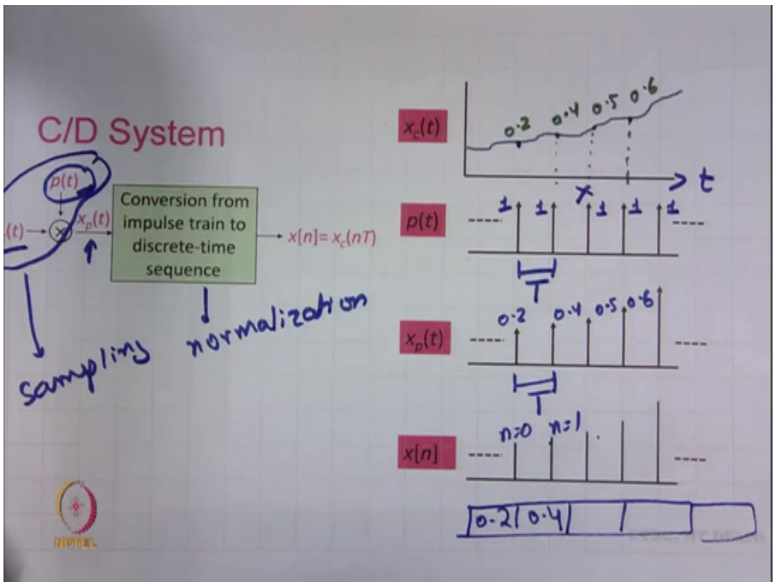


Also, if we have to look at a continuous time signal; so, we know that time is the independent variable so that means, this independent variable takes values in uncountable set; that means, this belongs to the set of real numbers and we have a signal like this ok. So, this is a continuous time signal. When we are talking about a discrete time signal it is nothing, but you can think of this as an array.

And you have some numbers in this array, and we want to go from this signal to this signal. So, this continuous time signal to discrete time signal and we know why is this important? This is important because this is a physical signal this can travel over physical mediums. So, if you have an electrical circuit you would produce the current and voltage like this right current and voltage signals are continuous time signals and this is a discrete time signal. So, if you want to store a signal in computer you want to store this as an array, right because this would take infinite memory because independent variable is defined for all real values ok.

This time here belongs to an uncountable set. So, this signal cannot be stored right. So, for a storage we like to use discrete time signal whereas, continuous time signals are important for transmission over physical channels and we want to make use of both kind of signals. So, it is an important idea to be able to convert this continuous time signal into discrete time signal and this is what happens in this block which, we have seen before so, this continuous to discrete time converter. So, here you can see that this continuous and discrete time converter is fed by clock, and this clock is working at a rate of T this will be more clear in a while.

(Refer Slide Time: 08:19)


C/D System

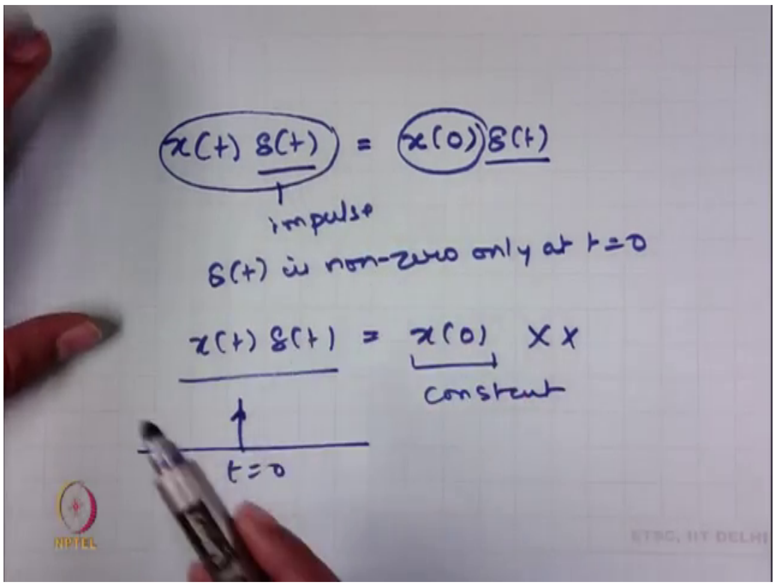
Conversion from impulse train to discrete-time sequence

So, let us now see what happens in the C D system continuous time to discrete time converter system. So, you take in a signal x c t so this is a continuous time signal right. So, here we have t, t is defined for all real numbers then p t. So, this multiplies x e t with an impulse train. So, p t is an impulse train we already know what is an impulse train impulse train contains impulses. So, these are impulses and these impulses let us say are separated by duration of T; so, this is what an impulse train is.

So, first step is multiplying this continuous time signal with an impulse train what happens on the multiplication. So, if I want to multiply this signal with this signal what happens. So, first of all when we are talking about an impulse train let us say we have an unit impulse train. So, all these impulses have the weights as 1 right. So, when I am saying 1 it means that the area of this impulse is 1. So now, when you multiply this impulse train with this continuous time signal the weights of this impulse is modified by this continuous time signal.

So, let us say if I have some value let us say 0.2 and at this timing instance, we have some value 0.4 at this timing instance we have some value let us say 0.5, at this timing instance we have some value 0.6 and so on so forth what happens is the weights of this impulse is modified by these numbers. So, we would have here the weight as 0.2, 0.4, 0.5, 0.6. So, why is this so, let us write a basic equation. So, if I is let us say multiply x t where delta t.

(Refer Slide Time: 10:29)



$$x(t)\,\underbrace{\delta(t)}_{\text{impulse}} = x(0)\,\delta(t)$$

$\delta(t)$ is non-zero only at $t=0$

$$x(t)\,\delta(t) = \underbrace{x(0)}_{\text{constant}} \quad \times\times$$

$t=0$

What is the result? First of all this delta t is an impulse and this delta t is non-zero only at t equals to 0, delta t is non-zero only at t equals to 0 right. So, when you multiply x t with delta t you only have x of 0, which is the value of this signal at t equals to 0 and then you have delta t right. Sometimes a students try to think that x t delta t is just x of 0 which is incorrect, why is this incorrect because this is a constant number and this is a function of time is not it.

So, if I want to plot x t times delta t this will be a signal only define a t equals to 0, where x of 0 is a constant, it is a number like 2, 3, 4 and so on so forth. So, x t delta t cannot be x of 0, but it would be x of 0 times delta t. This delta t is in here to say that this product only has a non-zero value at t equals to 0 and this effect is created by this delta t. Now, we know what happens when you have a continuous time signal and you multiply it with an impulse; what happens the weight of this impulse, is modified by x of 0 and what is this x of 0 it is the time at which this delta t is non-zero.

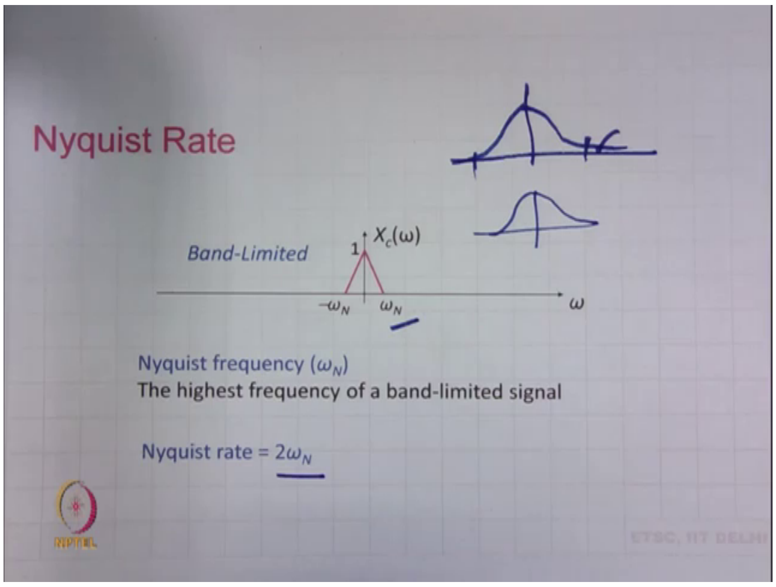
Similarly, now if we go back to that picture this impulse, has its effect only at this time instants and the weight of this impulse would be modified by the value of the signal at this time instants ok. In short, what happens when you multiply a continuous time signal with an impulse train you get a weighted impulse train, where the weights of this impulses is decided by the signal with which you are multiplying this impulse train. So, this is what we get, we get x p t. And, then the next step is to go from this x p t, which is still a continuous time signal you can understand that this is a continuous time signal because of this t and because of this parenthesis.

We want to go to x n which is a discrete time signal; that means, this is a signal which you can store. So, what happens is instead of impulses by notation, we represent them using lines. So, these are lines these are impulses and say this simply means that this is a signal, which is in the form of an array and where elements of this array are nothing, but it is the weights of these impulses. So, this is your discrete time signal. In this course we are not very much interested in how to do this right.

So, for that you have to see and look at the courses like digital electronics and other courses, where they basically talk about how to modify something. In this course we are just interested in what are the processes which happens, and what is the impact of this processes on the transmission of a signal. So, this is the viewpoint that we are taking in this course. So, in short a C D system, or continuous to discrete time system takes in a continuous time signal multiplies this with an impulse train gets a signal, which is like a weighted impulse train where the weights are modified by this x t and then from this x p t you go into a discrete time signal.

So, this happens in these sampling and normalization. So, C D system continues to discrete time system carries out this effect of sampling and normalization. So, this process is known as sampling and conversion from impulse train to discrete time sequence, we refer to as normalization why we want to say this as normalization because these impulses, were separated by a time duration of t whereas, these numbers are just separated by 1 unit in the memory or of an array. So, this is for example, n equals to 0, n equals to 1 and so on so forth right.

(Refer Slide Time: 15:26)



Nyquist Rate

Nyquist frequency (ω_N)
The highest frequency of a band-limited signal

Nyquist rate = $2\omega_N$

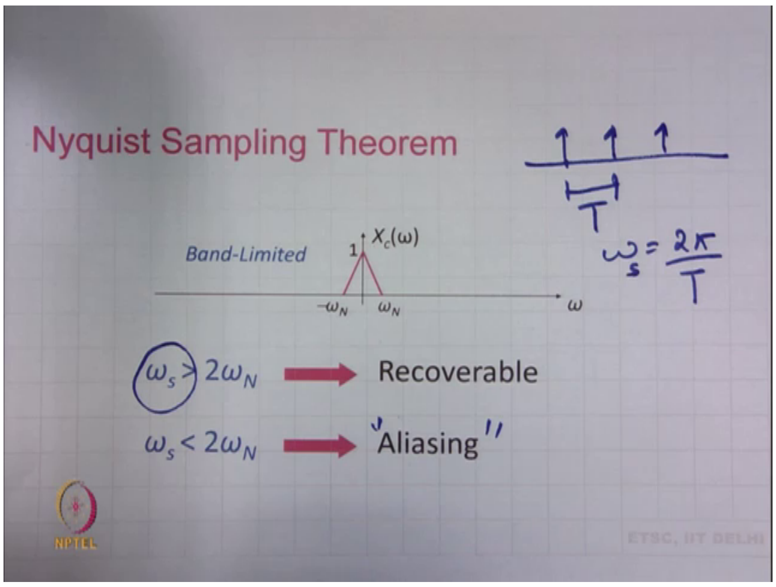
Band-Limited
$X_c(\omega)$
1
$-\omega_N$
ω_N
ω

Let us, now see certain things that are important if you want to do this conversion faithfully and the first thing that you have to understand is most of this theory works only for band limited signal. So, you can only convert a continuous time signal faithfully to a discrete time signal if the underlying continuous time signal is a band limited signal. And, if it is a band limited signal we have talked about this often, then it would have a maximum frequency for which most of this energy is confined. For example, in this case if I have this spectrum you see here, that most of the energy or all its energy of this signal is confined between 0 and omega N frequencies.

Of course, we have already seen that ideal band limited signals do not exist in practice and thus, for practical signals what works is some approximation and we normally consider the bandwidth of those signals in terms of the region in which most of their energy is confined ok. For example, if I have a spectrum like this you know ideally it will run from 0 to infinity because it is a practically realizable signal, but we would like to approximate this as a signal probably like this, because this part of the energy would not be impacting the signal too much.

So, when we say band limited signal you always mean approximately band limited signal ok. Then we have some terminology running in here first is Nyquist frequency which is the highest frequency of a band limited signal for in this case the highest frequency is omega N. So, Nyquist frequency is omega n and then we have Nyquist rate which is twice of the Nyquist frequency. So, Nyquist rate in this case is 2 times omega N and then we have a very celebrated sampling theorem, which goes by the name of Nyquist sampling theorem.

(Refer Slide Time: 17:35)



Nyquist Sampling Theorem

Band-Limited
$X_c(\omega)$
1
$-\omega_N$
ω_N
ω
T
$\omega_s = \frac{2\pi}{T}$

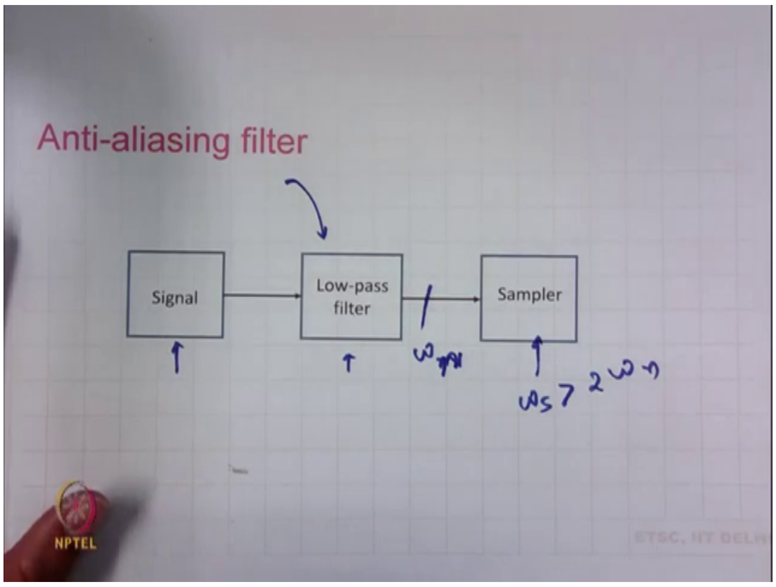
$\omega_s > 2\omega_N \rightarrow$ Recoverable

$\omega_s < 2\omega_N \rightarrow$ "Aliasing"

It says that the sampling frequency should be greater than twice omega N, if you want to have a faithful conversion from continuous time domain to discrete time domain. The sampling frequency should be greater than twice omega, what is the sampling frequency. So, we have already seen that to sample we have to multiply with an impulse train and this impulse train let us say, has a period of t then the frequency is 2 pi by t. So, Nyquist sampling theorem says that this frequency if I like to call this as omega s should be greater than 2 times omega N where omega is the maximum frequency of this band limited signal.

Then you can have a faithful reconstruction ok, if you do not follow this if omega S is less than twice of omega, then you have an effect called as aliasing in which high frequencies pose themselves as low frequencies. So, high frequency alias as low frequencies; I would not be discussing too much about this aliasing, but I will just like to point out that aliasing happens and sometimes it deteriorates the perception of the signal.

(Refer Slide Time: 19:10)

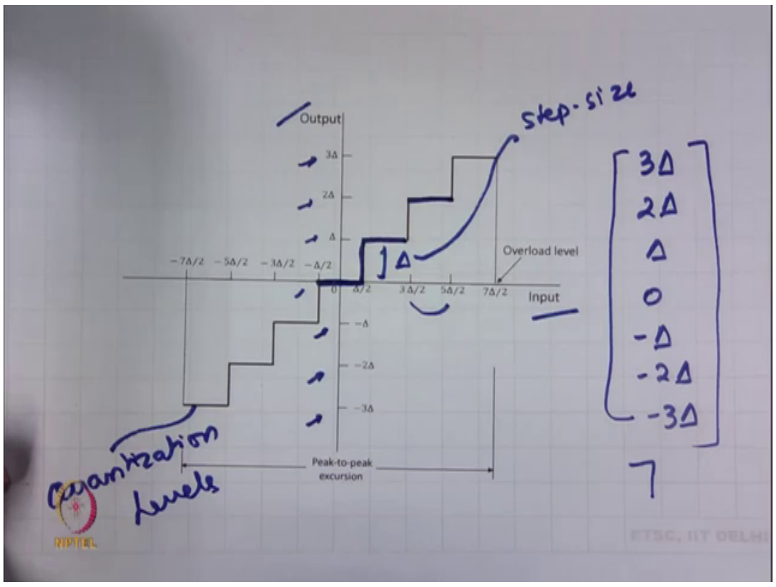


So, last point about this sampling is, because these signals are not band limited or may not be band limited, it would be a good idea to first band limit them by having a low pass filter. So, that after this low pass filter the maximum frequency becomes omega M or omega N and then you use a sampler, whose frequency is greater than twice omega n. So, we use a low pass filter just to truncate this bandwidth of the signal to frequencies around omega N and this filter goes by the name of anti aliasing filter. So, sampling is easy there is nothing much in here only few things you have to know the first thing is, about this C D block this is only important you do this continuous time to discrete time conversion.

Following these steps taking a continuous time signal multiplied with an impulse train you carry weighted impulse train. And then, you convert these impulses into lines and you just store the weights of these impulses and you get a discrete time signal. And for faithful conversion you have to follow the sampling theorem the sampling theorem states that the sampling frequency has to be greater than twice omega N for the faithful conversion ok. I think we are done with this sampling and let us now move to second stuff, and that is about quantization because after sampling what we are getting is so sequence right, of real numbers and you can also not store real numbers because real numbers also require infinite memory right real numbers have infinite decimal representation.

So, if you have to store these real numbers they will come up with infinite memory. So, I have to take the decision to round off those real numbers to whatever precision you are with and this process of rounding off in engineering is referred to as quantization.

(Refer Slide Time: 21:22)



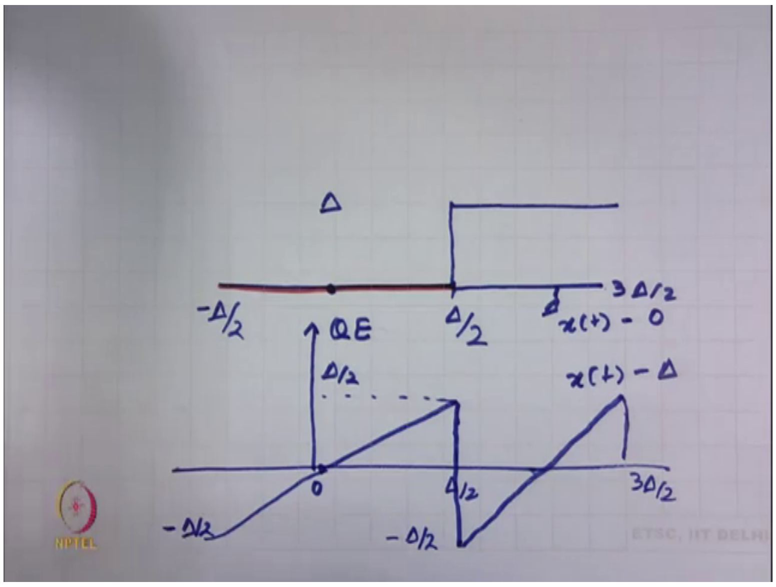
So, let us look at the quantizer transfer characteristic. So, quantizer is a rather simple block. So, let us say we have input here, and this is the output of a quantizer. So, in this case in this quantizer the quantizes can be of various kinds. So, the kind of quantizer that we are looking into, what happens is if the amplitude of input lies between minus delta by 2 to plus delta by 2 the output of the quantizer is 0.

If the amplitude of input lies between delta by 2 to 3 delta by 2 the output of the quantizer is delta of course, there is a discontinuity at a single point and we rather do not worry about discontinuity at a single point or we do not even care about the signals, which are discontinuous at a single point because we have seen that they belong to class of indistinguishable functions right. So, two functions which are different only at countable number of points are indistinguishable we have seen all of that stuff the only point that is important is, if the amplitude lies between this to this range the quantizer output is delta.

If the amplitude of input lies in this range the quantizer output is 2 delta and so on and so forth. Now, you can see that input amplitude could have taken any value right because it has a continuously varying amplitudes. So, this amplitude belongs to an uncountable set, but the amplitudes at the output of a quantizer belongs to a discrete set, because now my amplitude can be either 3 delta, 2 delta, delta, or 0 right or if you want to look at this x s it can be minus delta, minus 2 delta, minus 3 delta.

So, at the output of the quantizer we have only finite number of amplitudes available so, how many in this case in this case we have 7 amplitudes available. So, the output of a quantizer we have a discrete set, is not it. So, here only 7 amplitudes are possible whereas, in the input the amplitude can take any real number between minus 7 delta by 2 to 7 delta by 2. So, the output of a quantizer is a discrete set. Of course, there must be some error because of this, quantization and this error is known as quantization error. Let us look at the quantization error let me do it before showing you the picture.

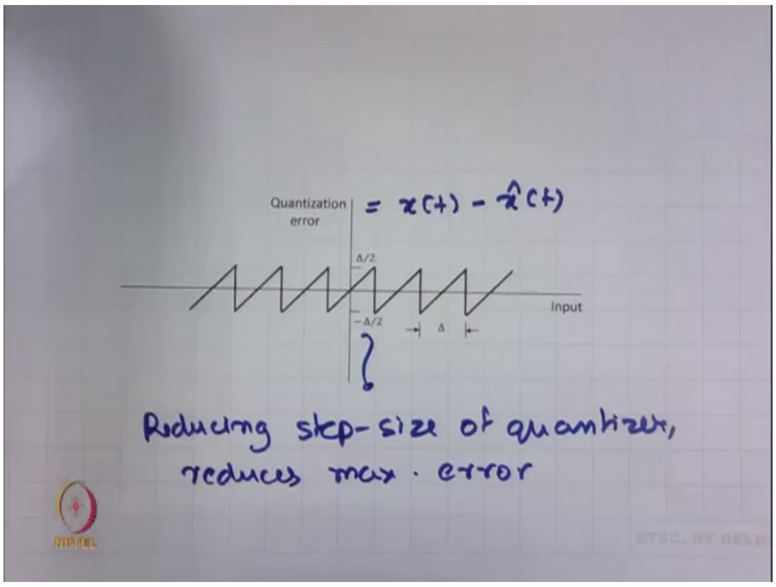
(Refer Slide Time: 24:25)



Let us consider the case, where input amplitude varies between minus delta by 2 to delta by 2 and the output of the quantizer is 0, let me plot the output of the quantizer. So now, what is the error? Error is let us say input amplitude minus, the quantization output which is 0. So, in this region the error is just same as the inputs amplitude ok. So, its 0 here and it goes up to delta by 2. So, input is also delta by 2 and on this axis what we have is the quantization error, similarly it will go up to minus delta by 2, I will show you neat a picture in a while, this is just for understanding.

Now let us look at this region, when the input amplitude varies between delta by 2 to three delta by 2 and the quantizer output is delta. So now, we have x t minus delta as the error in this region. So, at this point the error is minus delta by 2, and now the input increases, and the error reduces, and a delta let us say delta is here the error becomes o and then it becomes positive. At this point at 3 delta by 2 let us say three delta by 2 is here the error magnitude is 3 delta by 2 minus delta which is delta by 2.

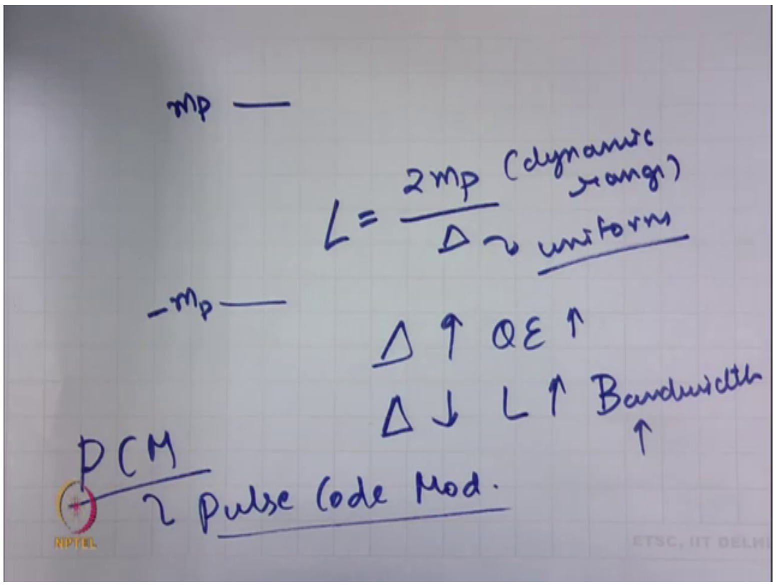
(Refer Slide Time: 26:35)



So, if we look at the quantization error which is nothing, but x t minus quantized value of x t, you get a saw tooth wave in this case. If you look at the magnitude of the quantization error the maximum magnitude of the quantization error is delta by 2, the minimum is minus delta by two and what is this delta this delta is important. So if I look at that quantizer again this delta is, the step size of the quantizer. So, this is known as the step size it is an important tau.

So, if you reduce the step size or the quantizer, what happens the maximum quantization error reduces because maximum quantization error is delta by 2. So, let me write this reducing step size of quantizer, reduces maximum error a maximum quantization error. So, why not we have a very small step size; what is its disadvantage? If you have a very small step size, then what happens is you end up with many quantization levels. So, these are known as also quantization levels or simply levels. In this case how many levels we have 7, how to find out the levels that you would have in a quantizer that is also simple exercise.

(Refer Slide Time: 28:38)


mp
$L = \frac{2mp}{\Delta}$ (dynamic range)
~ uniform
-mp
$\Delta \uparrow$ QE $\uparrow$
$\Delta \downarrow$ L $\uparrow$ Bandwidth $\uparrow$
PCM
~ Pulse Code Mod.

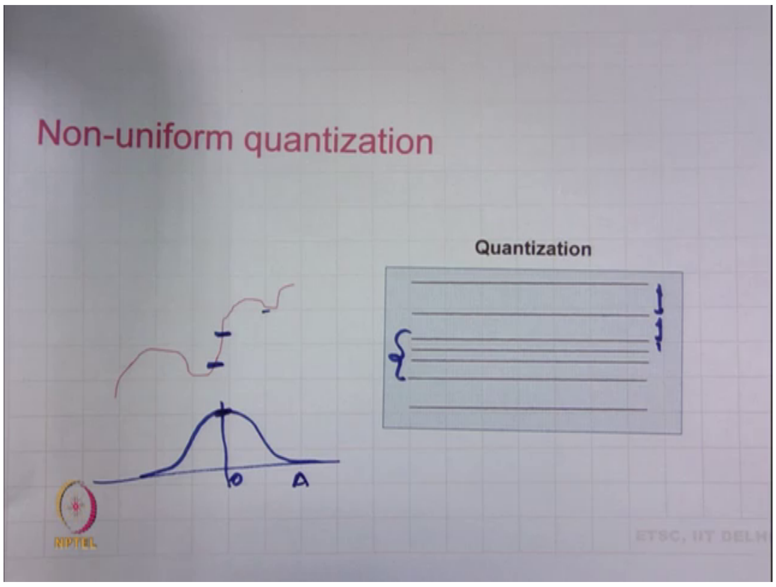
So, let us say if the signal amplitude varies between mp so maximum signal amplitude is m p minimum is minus mp. So, the total dynamic range of the signal is 2 m p understand these words clearly. So, this is known as a dynamic range. So, it is the maximum minus minimum of a signal, there you want to measure or transmit this the dynamic range of the signal in this case is 2 m p and if you use, quantizer with the step size of delta. And in this case we are assuming a simple situation where the step size is uniform that means, difference between any two quantization level is delta that is fixed then the number of levels is 2 m p by delta easy.

So, if I have a reduced delta then the number of levels increases and as we will see shortly, that increases the bandwidth requirement for the transmission of the signal and that is one important trade of that if we want to see how to choose this delta, delta increases, quantization error increases. We have seen this delta reduces L increases number of levels increases and we will see that also increases the bandwidth requirement, which is also an important resource in communication system. So, initial studies in communication systems which are also referred to as pulse code modulation based systems.

Let me use another word introduce pulse code modulation, we will see and understand the name later on, but whenever you are talking about waveform coding an important way to do this waveform coding is by using this pulse code modulation techniques ok. So, whatever we have studied falls into the regime of pulse code modulation and so, initial studies in pulse code modulation systems dealt with how to choose this delta and so on so forth ok. So, what we have seen is an important attribute in quantization is the step size step size influences the quantization error, and step size also influences the number of levels that you have at the output of a quantizer.

So, we have also seen that if the step size is constant; that means, the difference between two quantization levels is constant, then we call that quantizer as uniform quantizer.

(Refer Slide Time: 31:42)

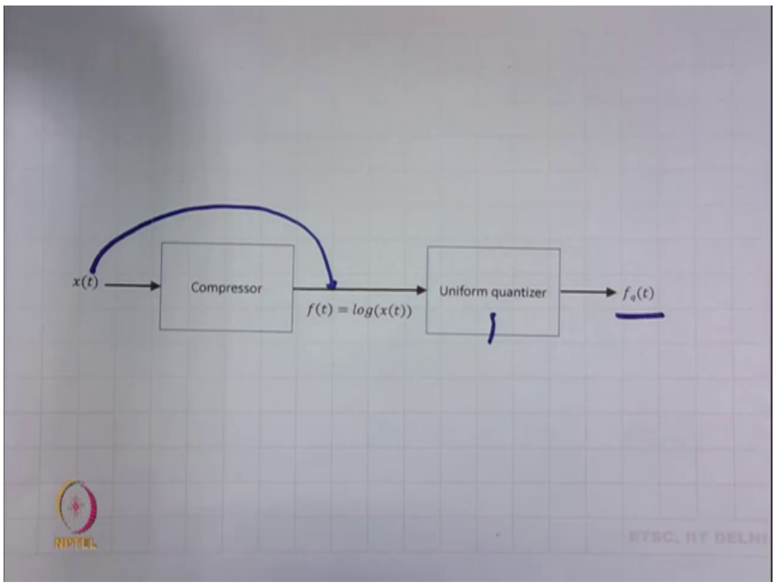


And there is another interesting quantizer which is non uniform quantizer and as you would have guessed in non uniform quantizer, the difference between the quantization levels is not constant right. So, for example, in this case the difference between two levels is not constant and the idea is because, the probabilities of low amplitudes is much more than the probabilities of occurrence of higher amplitudes, it is better to use more levels to code lower amplitudes whereas, it is good to use fewer levels to code bigger amplitudes.

For example, even in the case of noise and even in the case of Gaussian PDF we have seen, that the Gaussian PDF is like a bell curve and what does it say it says that the probabilities of amplitudes in a case of noise at least is higher at the amplitude levels around here so around 0. And as the amplitude level increases so, to state it carefully the probabilities of amplitudes, around a decreases as a increases this is what happens in the Gaussian case and for natural occurring waveforms like a speech waveforms or music waveforms it is also seen that the lower amplitudes happen more frequently, than the higher amplitudes and thus its wiser to use more levels to code lower amplitudes than large amplitudes and this what happens in non uniform quantization.

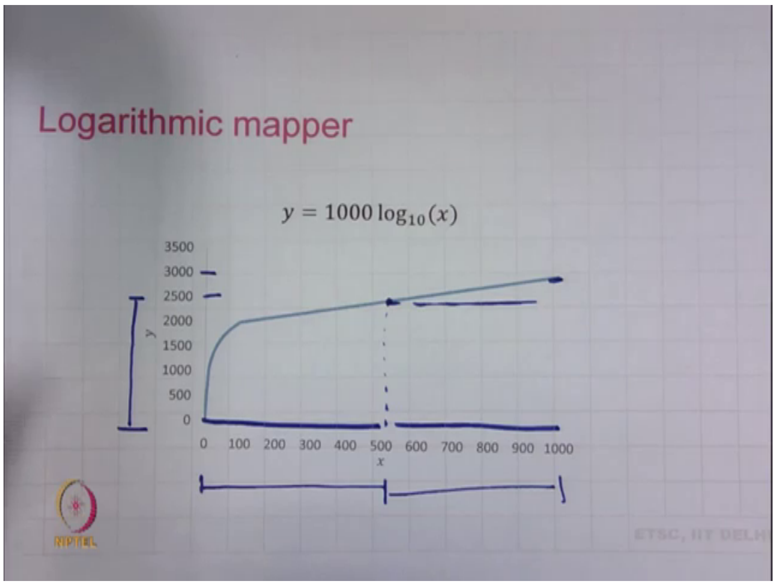
This is exactly like you are texting rich more than poor right. So, what is how to do this non uniform quantization is, to first pass the signal through a compressor.

(Refer Slide Time: 33:58)



So, compressor changes the signal the normal signal to another signal in this case ft and then you use in uniform quantizer for this ft to get the quantized version of f t which I call as f q t. And, so instead of using a non-uniform quantizer what we are doing is we first pass the signal through a compressor and mostly the transfer characteristic of this compressor a logarithmic. For example, in this simple case what we have assumed as f t is nothing, but log of x t and then you pass this f t through a uniform quantizer, how does it solve our problem of coding lower amplitudes more than the higher amplitudes will be clear if you look at this picture of logarithmic mapper.

(Refer Slide Time: 34:49)



Logarithmic mapper

$y = 1000\log_{10}(x)$

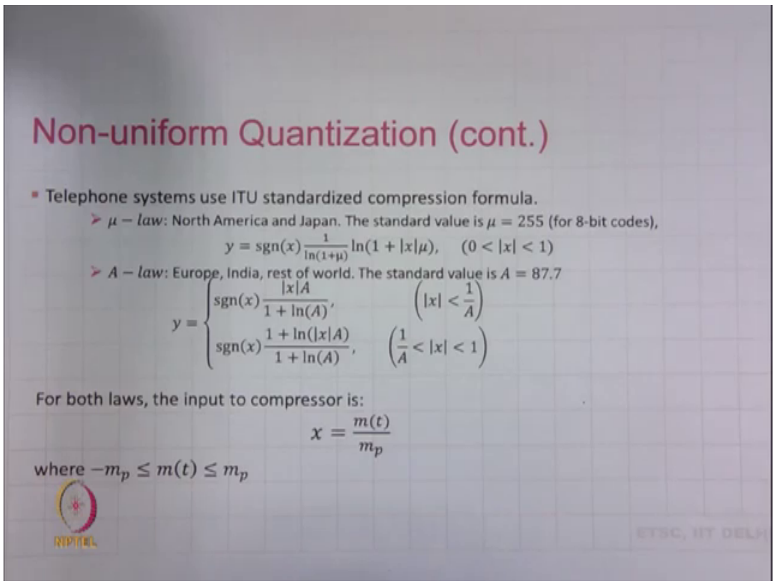
Here you can see clearly that let us say, we have two regions in here. So, I have two regions for amplitude levels one region goes from 0 to 500 let me put 500 here and then there is a region between 500 to 1000. So, these regions are uniform, but now what you see is this region is mapped to values between 0 to 2500.Whereas, for this region we only have y going in between 2500 to 3000 does this lower region has an output dynamic range much, larger than this region which has a very small output dynamic range.

So, this happens when you pass a signal through a logarithmic mapper and now if you use a uniform quantizer at ft what would happen is, uniform quantizer we will have the quantization level at constant interval. And because this range is much larger than this range you would have more quantization levels in this range than in this range and thus to code this lower range, we are using many more quantization levels. Than what we are using to code the higher range and this is what logarithmic mapper doing for us.

So, let me summarize what we have said so far is, because the lower amplitudes happen with a higher probabilities in naturally occurring signals like a speech and music signals, it has been understood as a good idea to have more levels to code lower amplitudes than for higher amplitudes. And this is the concept behind non-uniform quantizer and to do non uniform quantizer you do not change anything in quantizer the quantizer remains uniform.

But what you do is you pass this signal through a mapper, which is also known as compressor and usually in practice we use logarithmic mapper's which takes in a signal, takes this log and produces another signal and when it does automatically what follows is a lower range of amplitudes has a much more output dynamic range than the higher amplitudes. So, what we end up with is, you use many more quantization levels to code lower amplitudes than what you use to code higher amplitudes of course, this transfer function.

(Refer Slide Time: 37:51)



Non-uniform Quantization (cont.)

- Telephone systems use ITU standardized compression formula.
  - $\mu - law$: North America and Japan. The standard value is $\mu = 255$ (for 8-bit codes),

$$y = \text{sgn}(x)\frac{1}{\ln(1+\mu)}\ln(1+|x|\mu), \quad (0 < |x| < 1)$$

  - $A - law$: Europe, India, rest of world. The standard value is $A = 87.7$

$$y = \begin{cases} \text{sgn}(x)\dfrac{|x|A}{1+\ln(A)}, & \left(|x| < \dfrac{1}{A}\right) \\ \text{sgn}(x)\dfrac{1+\ln(|x|A)}{1+\ln(A)}, & \left(\dfrac{1}{A} < |x| < 1\right) \end{cases}$$

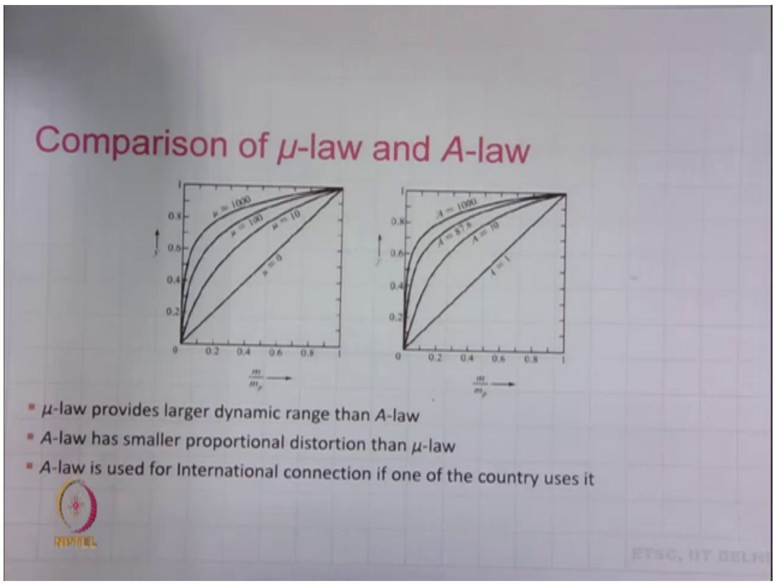
For both laws, the input to compressor is:

$$x = \frac{m(t)}{m_p}$$

where $-m_p \leq m(t) \leq m_p$

I have introduced for simplicity in practice you have little bit more complicated transfer functions. In fact, you have laws for this mapping the two important laws mu law, and a law mu law is used in North America and Japan. The standard value of mu is 255 for 8 bit codes, and this is a transfer characteristic I will not read out this, but important point to appreciate is that it is a log function. A law is used in Europe, India and rest of the word and the standard value of A is 87.7, you do not have to learn these laws by heart the only thing that is of interest here is to appreciate that these laws also have logarithmic mapping which, we know intuitively will work because any log mapping whatsoever is that would produce a larger dynamic range for a lower range of amplitudes. So, as I have seen here is x what is x? x is nothing, but m t divided by m p m p is the maximum value of the signal.

(Refer Slide Time: 39:20)



Comparison of μ-law and A-law

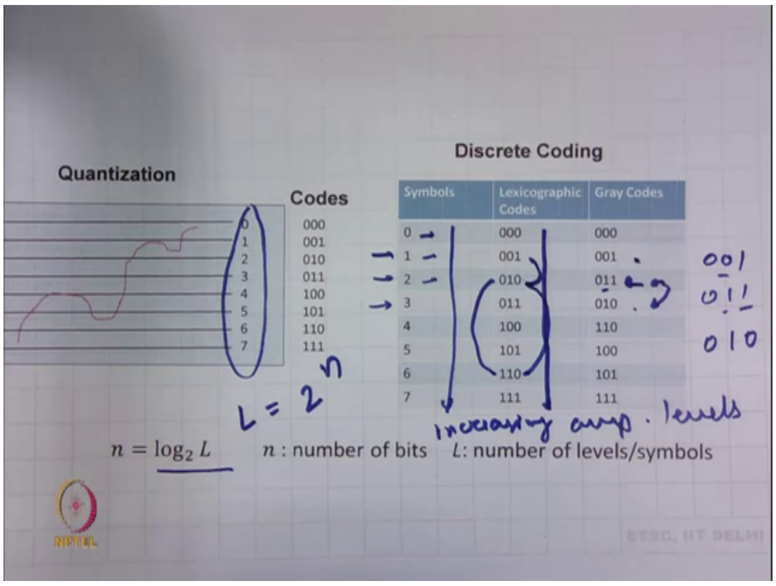
- μ-law provides larger dynamic range than A-law
- A-law has smaller proportional distortion than μ-law
- A-law is used for International connection if one of the country uses it

So, just to get a some practical flavor on, the differences between mu and A law is pretty similar as the value of mu increases you see that you have more and more compression and we also know what values of mu and A are used in practice, mu law provides larger dynamic range than A law. A law on the other hand has a smaller proportional distortion and because many countries are using a laws other than North American Japan it is used for international connection if one of the country uses it, just to get some flavor on the differences between mu and A law.

Let us now get to the third business and there is about discrete coding. So, far what we have done is we had this continuous time signal we converted this into discrete time signal and then what we have done is in this conversion we have also got a sequence. But, the sequence consisted of real numbers, bunch of real numbers, sequence of real numbers, and then you have this quantizer which converts the sequence of real numbers into sequence of quantized real numbers, and we call those as symbols. Now once you have got the symbols or the levels you want to find the code binary representation for those symbols and this is easy in waveform coding was pretty hard in the source coding and that is the main difference.

So, when we see waveform coding, we normally assume that all these levels are of equal probability and we also want to use the same number of bits for each representation.

(Refer Slide Time: 41:21)



Discrete Coding

Symbols	Lexicographic Codes	Gray Codes
0	000	000
1	001	001
2	010	011
3	011	010
4	100	110
5	101	100
6	110	101
7	111	111

$n = \underline{\log_2 L}$ n : number of bits L: number of levels/symbols

Whereas, in source coding you want to optimize that, so we will not be discussing a source coding in this course. We have simplified it to waveform coding where we assume that the levels that we have caught, 8 levels in this case all this levels happen with equal probability and we want to find good binary representation for these levels and this is really a simple issue. The one thing that you have to decide in here is, how many bits would you need to represent a level and the number of bits that you would require is simply given by this log 2 L, where L is the number of levels. So, for example, if you have 8 levels as in this case you would need 3 bits.

If you have 9 levels you would need 4 bits and then you would see that, it is a wastage because if anywhere you are using 4 bits it would have been a better idea to use 16 levels is not it. So, what we want to do is we want to make the number of levels L as some 2 to the power n so, that there is no wastage is not it. So, once you know the number of levels then finding the number of bits that you would require to represent that level is simple, it is just log to L and these bit assignments follows either lexographic coding or gray coding. In lexographic codes you take in a numerical value and you choose the binary representation in accordance with the numerical value that you have in here.

For example, if this is the order of the numerical values of these symbols for example, 0 represents the lowest amplitude, one represents a little higher amplitude, two represents higher amplitude than 1 and so on so forth. So, this is in the let us say increasing amplitude levels and what you can do is? You can simply assign the codes the binary codes, in the increasing order of binary value ok. So, you use 000 to represent 0, 001 to represent 1, 010 to represent 2 and so, on so forth and this is known as lexographic coding. So, you arrange the symbols in the order of their values and then you generate binary codes and assign the symbols their binary codes, based on the values of these binary codes.

In gray coding we are little bit more careful because we want to make sure that, the difference between the codes of two adjacent symbols does not differ more than 1 bit. For example, the adjacent symbol to symbol 2 is 1 and 3 and let us say the code that we want to use for 2 is 011, then we want to make sure that the code of 1 and 3 does not differ by more than 1 bit. So, 001 and 011 differs by 1 bit in this location. So, there is a difference in this location and 010 and 011 differs in this location, but there is only difference on 1 bit. Whereas, you see in lexicographic coding that is not the case for example, these two quotes differ by 2 bits and what do we want to make sure that in grey codes the difference between the codes assigned to adjacent symbols is not more than 1 bit is because the probability of bit error, for 1 bit is much larger than probability of bit error for 2 bits.

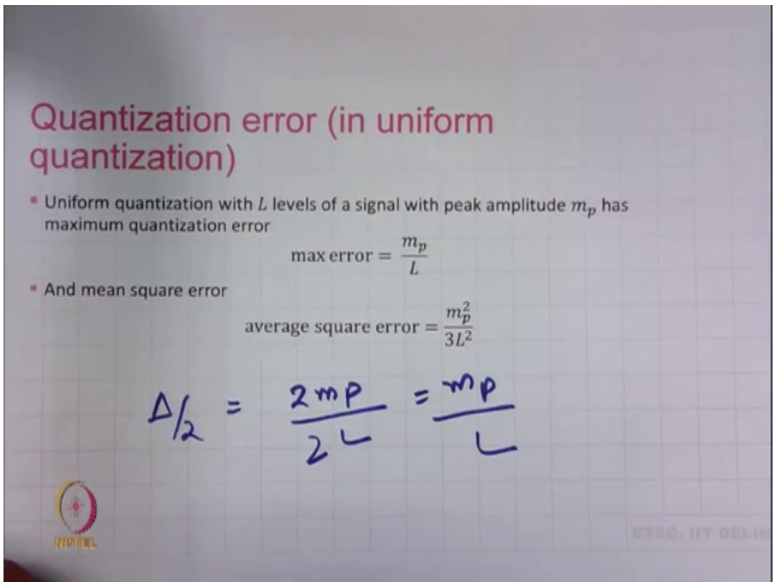
Thus for example, if I assume that there is 1 bit error, this was transmitted and because of some error they receiver thought that this is transmitted you only slip two adjacent symbols and does the deterioration level is not to significant. Whereas, in this case, let us say here and let me choose this and this for explanation. So, these two codes also differ by 1 bit and the errors 1 bit errors will be much more dominant than 2 bit errors and as soon as there is a 1 bit error you slip from 2 to 6 and there is a serious deterioration, because there is a much more difference between 2 and 6 then would have been between adjacent symbols.

And that is because 1 bit errors are more probable than 2 bits errors in a code, then the gray code serves you better because whenever there is a 1 bit error in the code do you only slip two adjacent symbols. And thus the detoriation is not significant on the other hand in lexicographic coding, even if there is a 1 bit error you might move to a symbol which is too far from the symbol which is actually transmitted. And, thus you face lot of detoriation and thus it is a better idea to use gray coding whenever you want to map symbols 2 bits.

This idea will be reinforced when we talk about modulation and this is exactly what happens in coding. So, discrete coding is simple, what we have seen is as soon as you identify the number of levels you can find the number of bits that you want to use for the binary representation of those symbols, we have a formula log to L to find that and you want to make sure that the number of levels is 2 to the power n. So, that you do not waste the bits right you make the optimum use of the resources and thirdly we have seen that the symbols should be assigned the bit spaced on gray coding, that is most optimal because then one bit error does not slip you too far.

Now, it is the time that we do this performance characterization of these PCM systems. This is exactly what happens in a PCM systems, sampler quantizer, discrete coding and there is little bit more that we will see later on.

(Refer Slide Time: 48:50)



Quantization error (in uniform quantization)

- Uniform quantization with L levels of a signal with peak amplitude m_p has maximum quantization error

$$\text{max error} = \frac{m_p}{L}$$

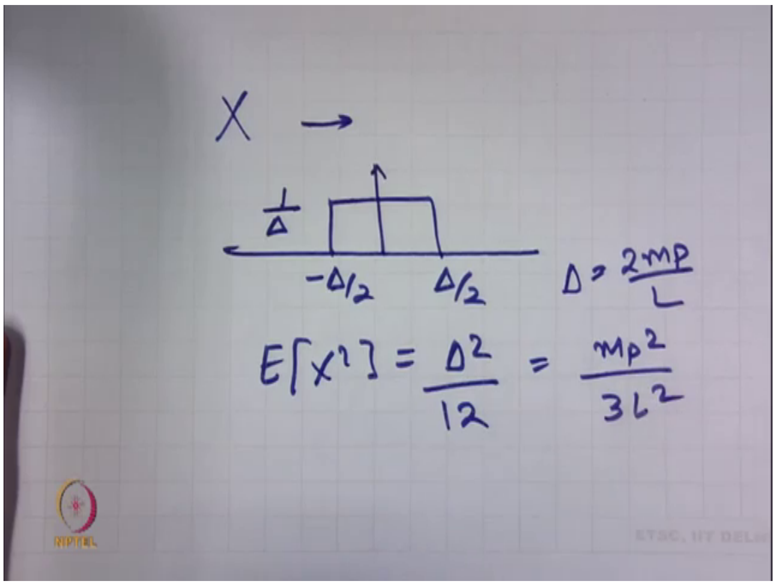
- And mean square error

$$\text{average square error} = \frac{m_p^2}{3L^2}$$

$$\Delta/2 = \frac{2mp}{2L} = \frac{mp}{L}$$

But all these three steps happens in PCM systems. So, maximum error in the case of uniform quantizer, we have seen what is what is the maximum error. So, the maximum error was delta by 2 and delta we have seen is nothing, but 2 mp by L. So, delta by 2 is 2 mp by 2 L which is nothing, but mp by L. So, that is the maximum error in uniform quantization and if I assume that this error can be modeled by assuming uniformly distributed random variables. So, you must have seen if x is a uniformly distributed random variable.

(Refer Slide Time: 49:32)



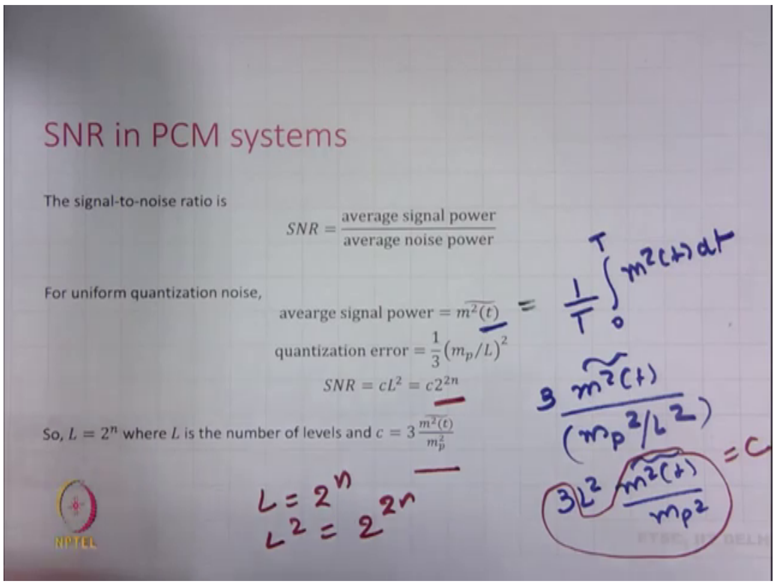
$X \rightarrow$
$\frac{1}{\Delta}$
$-\Delta/2$
$\Delta/2$
$\Delta = \frac{2m_p}{L}$
$E[x^2] = \frac{\Delta^2}{12} = \frac{m_p^2}{3L^2}$

And, this we have seen in one of the lectures on random variables its probably lecture number 9 and there we have seen that if you have uniformly distributed random variable, which spans from minus delta by 2 to plus delta by 2. Then the probability density function takes in a value of 1 by delta and the power in that random variable, is delta square by 12.

This we have seen I will not derive it, this is the reason that we derived in there when we discussed uniformly distributed random variable. The point is if we assume that the quantization error is also uniformly distributed random variable, quantization error has a maximum value of delta by 2 and minimum value of minus delta by 2 where delta is the step size of the quantizer, then the power in that noise or in that error process is delta square by 12 and delta as we have seen is nothing, but 2 mp by L.

So, substituting that in here we get mp square by 3 L square. So, average is square error is m p square by 3 L square mp is the maximum amplitude or the peak amplitude of the signal L is the number of quantization levels. Signal to noise ratio is the average signal power divided by average noise power, the signal to noise ratio is an important characteristic for digital communication systems.

(Refer Slide Time: 51:14)



SNR in PCM systems

The signal-to-noise ratio is

$$SNR = \frac{\text{average signal power}}{\text{average noise power}}$$

For uniform quantization noise,

$$\text{avearge signal power} = \widetilde{m^2(t)} = \frac{1}{T}\int_0^T m^2(t)\,dt$$

$$\text{quantization error} = \frac{1}{3}(m_p/L)^2$$

$$SNR = cL^2 = c2^{2n}$$

So, $L = 2^n$ where L is the number of levels and $c = 3\frac{\widetilde{m^2(t)}}{m_p^2}$

$$\frac{3\widetilde{m^2(t)}}{(m_p^2/L^2)}$$

$$3L^2\frac{\widetilde{m^2(t)}}{m_p^2} = c$$

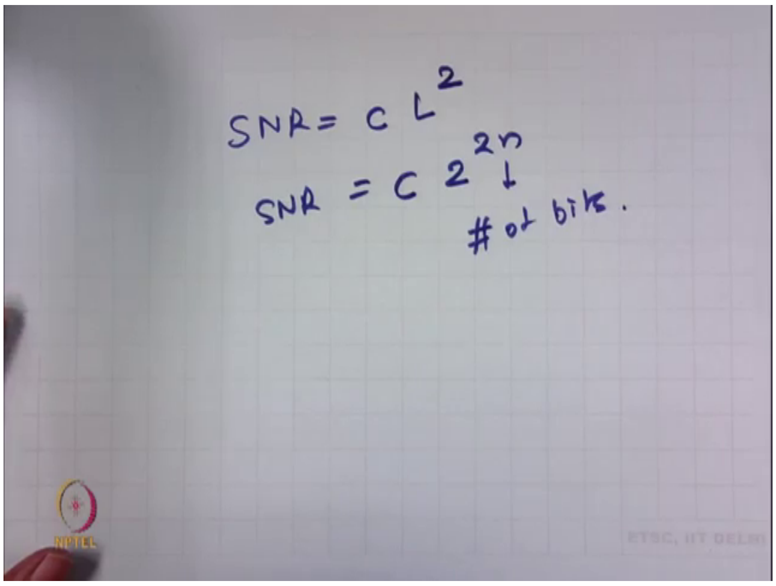
$$L = 2^n$$

$$L^2 = 2^{2n}$$

We will see a lot about it when we do detection, at this point I just like to stay that this is an important metric and the more is the signal to noise ratio better is the receiver performance or the communication system performance. So, we want to have a communication system with as high signal to noise ratio as possible, because you want to have more signal power and less noise power is not it. So, a higher signal to noise ratio is better. So, and what is signal to noise ratio it is average signal power divided by average noise power, average signal power is given by this m square t tilde and this is nothing, but it is simply 1 by t, 0 to t, m square t dt which is the mean square value of the signal, where t is the duration for which the signal empty exist.

Quantization error as we have seen before, is given by this and signal to noise ratio can be obtained by dividing this by this. So, if we divide m square t by m p square L square and so, we get three L square m square t by m p square, let me put tilde here and we combine all of this in a constant c because this is really constant. So, we get signal to noise ratio is constant times L square and L we know, what is L? L is 2 to the power n. So, L square is 2 to the power 2 n which is this. So, we get signal to noise ratio is c times 2 to the power 2 n where the value of c as we have said is this thing and this is an important result, in the case of the performance of PCM systems.

(Refer Slide Time: 54:00)



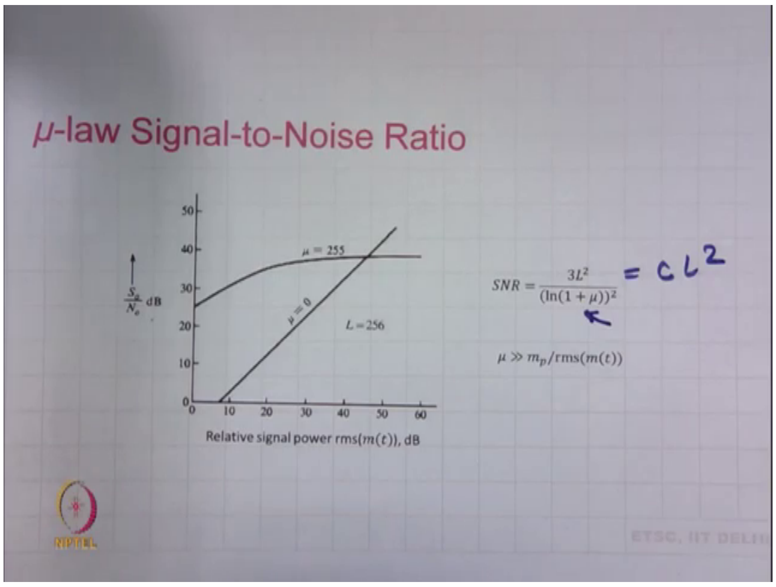
$$SNR = c\,L^2$$

$$SNR = c\,2^{\underset{\#\text{ of bits.}}{\underset{\downarrow}{2n}}}$$

So, in short we can say that the signal to noise ratio in the case of PCM systems is constant times, L square which is constant times 2 to the power 2 n. What is n? n is the number of bits that your coder use. So, this is for the uniform quantizer because this quantization error that we have obtained is for uniform quantization.

(Refer Slide Time: 54:32)

μ-law Signal-to-Noise Ratio

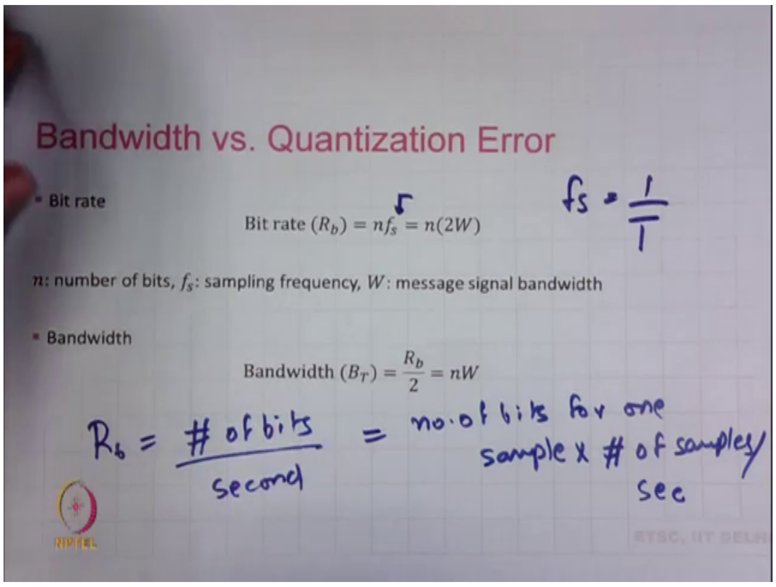


$$SNR = \frac{3L^2}{(\ln(1+\mu))^2} = cL^2$$

$\mu \gg m_p/\text{rms}(m(t))$

Let us see what happens in the case of the quantizer where you have the compressor based on mu law, we will not derive it because it is really complicated, but it is good to see that the signal to noise ratio follows the same form it is constant times L square. But, here the constant depends just on 3 and the mu that you use and it does not depend upon the signal power for large values of mu so, it is pretty constant the signal to noise ratio.

(Refer Slide Time: 55:14)



Bandwidth vs. Quantization Error

- Bit rate

$$\text{Bit rate } (R_b) = nf_s = n(2W) \qquad f_s = \frac{1}{T}$$

n: number of bits, f_s: sampling frequency, W: message signal bandwidth

- Bandwidth

$$\text{Bandwidth } (B_T) = \frac{R_b}{2} = nW$$

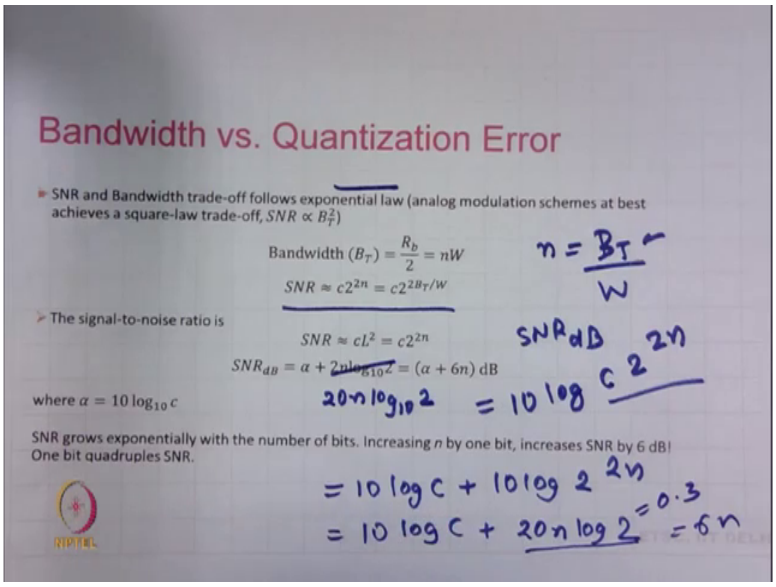
$$R_b = \frac{\#\text{ of bits}}{\text{second}} = \text{no. of bits for one sample} \times \#\text{ of samples/sec}$$

Other important point is the bandwidth performance of the PCM systems. So, first we see what is the bit rate? In the case of PCM systems, so, bit rate would depend upon the number of bits that you are transmitting per second R b is number of bits per second, first of all you can see that how many samples are happening per second. So, this would be nothing, but it is the number of bits for one sample times the number of samples per second. And number of bits for one sample is n we have said that we are using n bits to code one sample, sample is same as symbol and how many samples are we having per second that is given by the sampling frequency which we are seeing f s.

So, f s is the sampling frequency number samples per second, and sampling frequency is nothing, but it is 1 by t where t was the time duration between two impulses. And we have seen in one of the previous lectures that the sampling frequency and we have seen and revised today as well should be at least twice of W, where W is the signal bandwidth. So, we have to take the sampling frequency to follow Nyquist sampling theorem as greater than twice of W, we have chosen it to be twice of w in this case and so, the bit rate would be n times 2 W and the bandwidth as we will see in one of the lectures is nothing, but R b by 2.

So, if you know bit rate the bandwidth requirement or rather let us say the minimum bandwidth that you would require would be R b by 2. So, we are assuming that our communication system is optimal and its requiring the minimum bandwidth for a given bit rate. So, minimum bandwidth that you would require for a given bit rate is R b by 2. So, it will be n times w.

(Refer Slide Time: 57:52)



Bandwidth vs. Quantization Error

- SNR and Bandwidth trade-off follows exponential law (analog modulation schemes at best achieves a square-law trade-off, $SNR \propto B_T^2$)

$$\text{Bandwidth } (B_T) = \frac{R_b}{2} = nW$$

$$SNR \approx c2^{2n} = c2^{2B_T/W}$$

$$n = \frac{B_T}{W}$$

- The signal-to-noise ratio is

$$SNR \approx cL^2 = c2^{2n}$$

$$SNR_{dB} = \alpha + \cancel{2n\log_{10}2} = (\alpha + 6n)\text{ dB}$$

where $\alpha = 10\log_{10}c$

$$20n\log_{10}2$$

$$SNR_{dB} = 10\log \underline{c\,2^{2n}}$$

SNR grows exponentially with the number of bits. Increasing n by one bit, increases SNR by 6 dB! One bit quadruples SNR.

$$= 10\log c + 10\log 2^{2n}$$

$$= 10\log c + \underline{20n\log 2}\ \overset{=0.3}{} = 6n$$

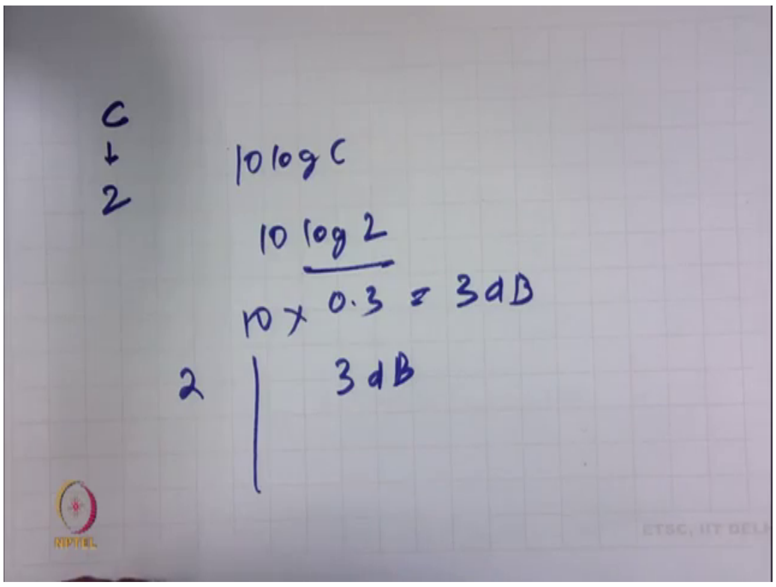
So, let us see the tradeoff between bandwidth and quantization error. So, one thing that you see is bandwidth increases with n so, as you increase end the bandwidth increases and signal to noise ratio also increases with n and signal to noise ratio is good. So, if we increase n you get a better signal to noise ratio, but you also require more bandwidth. So, signal to noise ratio as we have seen already is c times, this and what is the n? So, if we can write n is B T by W, where B T is the required bandwidth, W is the signal bandwidth.

So, we can replace n by BT by W and then you can see is, there is a tradeoff between signal to noise ratio and transmission bandwidth. So, if you can allow for a larger transmission bandwidth you get more signal to noise ratio and the other way around as well. So, normally you must have seen that there is a tradeoff between this signal to noise ratio and the required transmission bandwidth and you also see this trade off in here. The important point is that this signal and bandwidth trade off follows exponential law, you see the transmission bandwidth is an exponent. So, there is an exponential tradeoff between signal to noise ratio and required bandwidth whereas, if you have seen analog modulation a schemes like amplitude modulation or phase modulation of frequency modulation these modulation schemes particularly phase, and frequency modulation at best achieves the square law tradeoff that means, signal to noise ratio is proportional to transmission bandwidth the square.

But here the tradeoff follows an exponential law that means, by spending a little bit more on transmission bandwidth you can get a lot of signal to noise ratio at vantage in case of these modulation techniques PCM. So, signal to noise ratio we have seen is proportional to c times L square, we have already seen this and if I want to write it in dB scale, I can take log of this thing I will come to dB in a while. So, I take 10 log c 2 2 to the power n and you know that n log you can write this as 10 log c plus 10 log 2 into 2 to the power n which you can write 10 log c plus 20 n log 2 and 20 into log 2 log 2 is 0.3. So, this becomes 6 n.

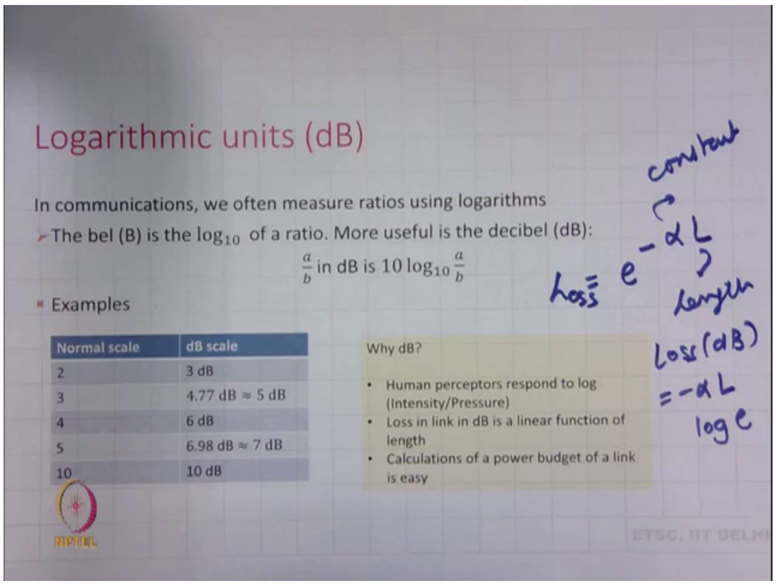
So, more or less we get alpha plus 6 n. So, this has to be 20 n log 10 2. So, we get 10 log c plus 20 n log 2, log 2 is 0.3 20 log 2 times n become 6 n. So, we get signal to noise ratio is alpha plus 6 n in dB. So, what you see is in this case if I look at this signal to noise ratio grows exponentially with the number of bits, if I want to increase the bit by 1 signal to noise ratio grows by 6 dB look at this if I change n by 1, I get 6 dB advantage in signal to noise ratio. Every bit would, quadruples my signal to noise ratio. I will come to this dB in a while at this point you can simply understand dB as 10 log c.

(Refer Slide Time: 62:12)


C
↓
2
$10 \log C$
$\underline{10 \log 2}$
$10 \times 0.3 = 3dB$
2 | 3 dB

So, if you want to talk about the dB you just if you want to change c, c can be some number like 2. So, if you want to change this in dB scale you just find 10 log of 2, then you know log 2 is 0.3 multiplied by 10 and you get 3 dB. So, 2 in dB scale is 3 dB. Let me talk about this now, and then we will talk about other things.

(Refer Slide Time: 62:36)



Logarithmic units (dB)

In communications, we often measure ratios using logarithms

- The bel (B) is the $\log_{10}$ of a ratio. More useful is the decibel (dB):

$$\frac{a}{b} \text{ in dB is } 10\log_{10}\frac{a}{b}$$

- Examples

Normal scale	dB scale
2	3 dB
3	4.77 dB ≈ 5 dB
4	6 dB
5	6.98 dB ≈ 7 dB
10	10 dB

Why dB?

- Human perceptors respond to log (Intensity/Pressure)
- Loss in link in dB is a linear function of length
- Calculations of a power budget of a link is easy

constant

$$\text{Loss} = e^{-\alpha L}$$

length

$$\text{Loss (dB)} = -\alpha L \log e$$

ETSC, IIT DELHI

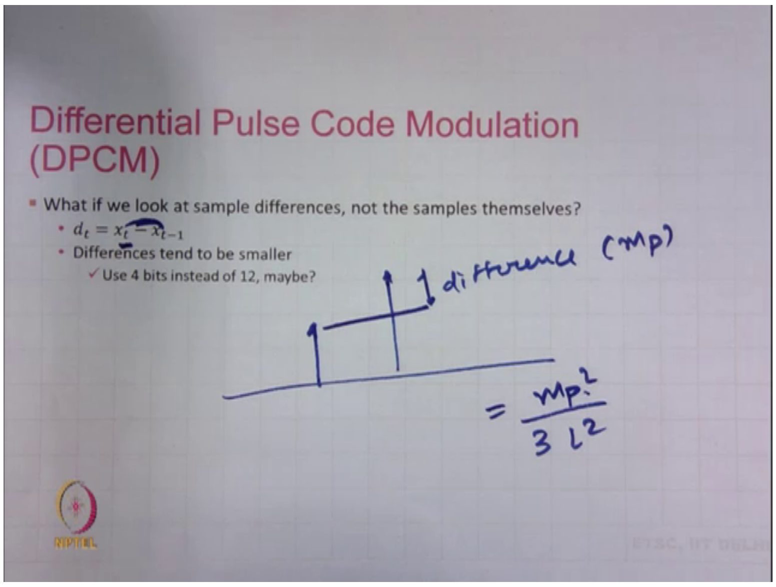
So, dB as I have said, is used in communication systems because we often measure ratios using logs. So, when you say bel is the log 10 of a ratio and we want to use dB which is decibel, which is 10 times log 10 of a ratio. And, this is an important scale because of various reasons first reason is that the human perceptors like ears and eyes respond to log of intensity and pressure, it is very natural this log scale.

Secondly, when you are talking about the communication link you may see that the loss in a communication link follows typically this relationship. So, this is let us say loss and it follows typically this relationship where this is some constant and this L is the length of the link. And so, if you want to express loss in dB it becomes so, do not worry about a factor of 10 and so on and so, forth. Just see that if loss follows this relationship, loss in dB becomes a linear function of the length of the link and thus it is easier to estimate the loss right.

So, there are other reasons as well when we do power budget and things like that, we will see that log scale is more convenient. It is also helpful to learn some numbers this conversions we have already seen that, 2 in dB scale is 3, 3 in dB scale is approximately

5, 4 in dB scale is approximately 6, 5 in dB scale is approximately 7, 10 in dB scale is 10. So, we have finished with the pulse code modulation techniques and it would be a good idea to quickly see other modulation techniques as well.

(Refer Slide Time: 64:53)



Differential Pulse Code Modulation (DPCM)

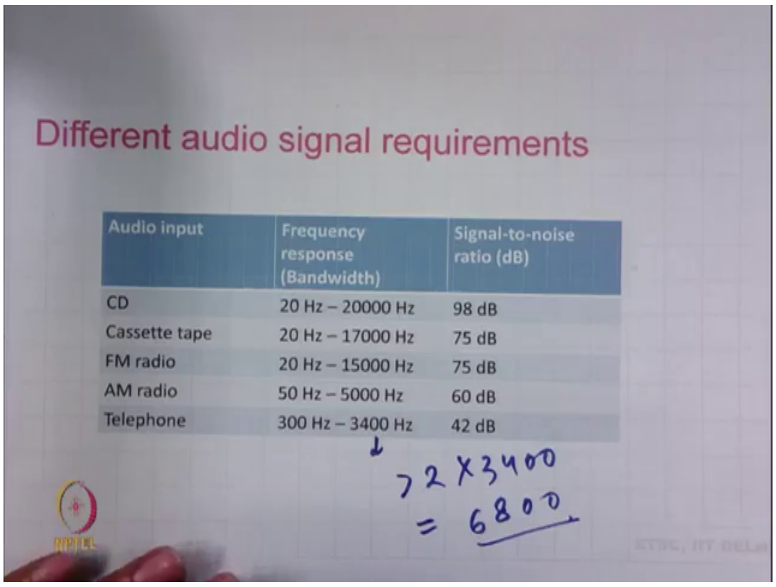
- What if we look at sample differences, not the samples themselves?
  - $d_t = x_t - x_{t-1}$
  - Differences tend to be smaller
    - ✓ Use 4 bits instead of 12, maybe?

difference (mp)
$= \frac{mp^2}{3L^2}$

For example, we have this differential pulse code modulation technique in differential pulse code modulation technique what happens is instead of sending the signal and coding the signal, we code the difference between the two samples of a signal ok. For example, if you have taken one sample here one sample here instead of coding each sample individually, we calculate the difference between the samples and we rather code that. What advantage we will get is that this difference would be much smaller than the signal itself and thus what would happen is the m p or peak value of this signal will reduce right and so, to get the same quantization performance you can use fewer number of levels.

Because, we have seen that the quantization error power depends upon m p square by a quantization error power is m p square divided by 3 L square. So, if m p reduces, m p reduces because instead of coding each sample individually we code the difference between the samples and thus the peak value of these different signal reduces and there is the power of the quantization noise reduces ok, if we are using the same number of levels as before if you want to have the same quantization noise power you can also reduce the number of levels because m p reduces. So, it cut both ways. So, that is the idea behind differential pulse code modulation technique. So, finally, we come to the slide where I capture in the different audio signal requirements.

(Refer Slide Time: 66:33)



Different audio signal requirements

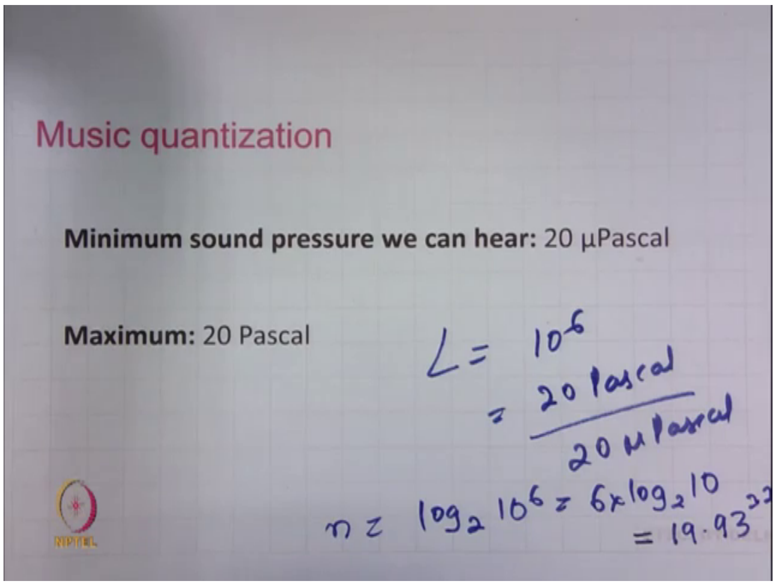
Audio input	Frequency response (Bandwidth)	Signal-to-noise ratio (dB)
CD	20 Hz – 20000 Hz	98 dB
Cassette tape	20 Hz – 17000 Hz	75 dB
FM radio	20 Hz – 15000 Hz	75 dB
AM radio	50 Hz – 5000 Hz	60 dB
Telephone	300 Hz – 3400 Hz	42 dB

$$> 2 \times 3400 = \underline{6800}$$

So, we say that depending upon the kind of audio input different audio inputs require different bandwidth or has different bandwidth and thus appropriately you have to choose the sampling frequency, sampling frequency is normally twice this maximum frequency that we have in here for example, what telephone we would need something like 8 kilohertz right. So, sampling frequency that we normally choose is typically more than twice of maximum frequency component.

So, here you get around, but normally we take it as 8 kilohertz. So, the sampling frequency is decided by the maximum frequency of these different audio inputs and this is the signal to noise ratio requirement for these audio signals. So, the messages that these requirements are too much specific on the kind of source signal that we are dealing with, let us just take the example of the music signal.

(Refer Slide Time: 67:44)



Music quantization

Minimum sound pressure we can hear: 20 μPascal

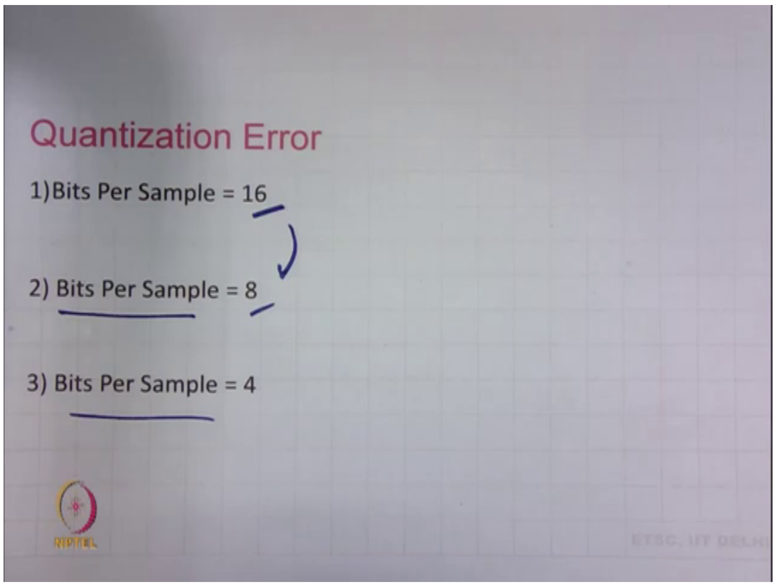
Maximum: 20 Pascal

$L = 10^6 = \frac{20 \text{ Pascal}}{20\, \mu\text{Pascal}}$

$n = \log_2 10^6 = 6 \times \log_2 10 = 19.93$

So, we know from psycho acoustic studies, that the minimum sound pressure that we can here is 20 micro Pascal and the maximum that you can here is 20 Pascal; that means, the number of levels that we would have is 1 million approximately right. So, 20 Pascal divided by 20 micro Pascal. And, thus if you want to code these many levels typically the number of bits that we will require is, log 2 of 10 to the power 6 which is 6 into log 2 of 10 ok. So, you can easily understand the number of bits that you would require and this turns out to be approximately 19.93 or approximately 20.

(Refer Slide Time: 68:46)



Quantization Error

1)Bits Per Sample = 16

2) Bits Per Sample = 8

3) Bits Per Sample = 4

So, let us see what we have got is if you have to have 1 million levels which is optimal because the ears can hear minimum 20 micro Pascal and maximum 20 Pascal. So, we have seen that we need around 20 bits and in C D what we do is we use 16 bits per sample right, and there are certain C Ds where you have 8 bits per sample ok. So, if you have fewer bits per sample what happens is probably there is little iteration in the quality, but you get more memory right. So, you can stuff more music if you are using fewer number of bits per sample.

Idea could be to have even fewer bits per sample like 4, if the quality of music does not iterate seriously. So, let us do a demo and see how the music quality is affected by the number of bits per sample. So, we will first see what happens when we use 16 bits per sample and now, we will see the quality of music which is produced using 8 bits per sample. And now, we will see the quality of music which is produced using 4 bits per sample and we can see clearly that up to 8 bits per sample the quality of music did not degrade seriously, but as soon as we use 4 bits per sample we could hear lot of hissing in the music and this is because we were really using very few number of levels than what is required for good quality.

So, this with this we conclude today's lecture and the main thing that we have seen in this lecture is how to go from an analog source to binary sequence we have seen the steps like sampling, quantization and discrete coding. And we have seen that an interesting class of system that do this is pulse code modulation systems, we have seen their performance metric we have seen that there is an exponential tradeoff between signal to noise ratio and offer transmission bandwidth in PCM systems, it is much better trade off then what analog modulation schemes offer which typically offers square law tradeoff.

And then we have seen that different kinds of signals and sources would have different kinds of requirements of sampling frequency and the signal to noise ratio that they would require from next lecture, we will talk about modulation.