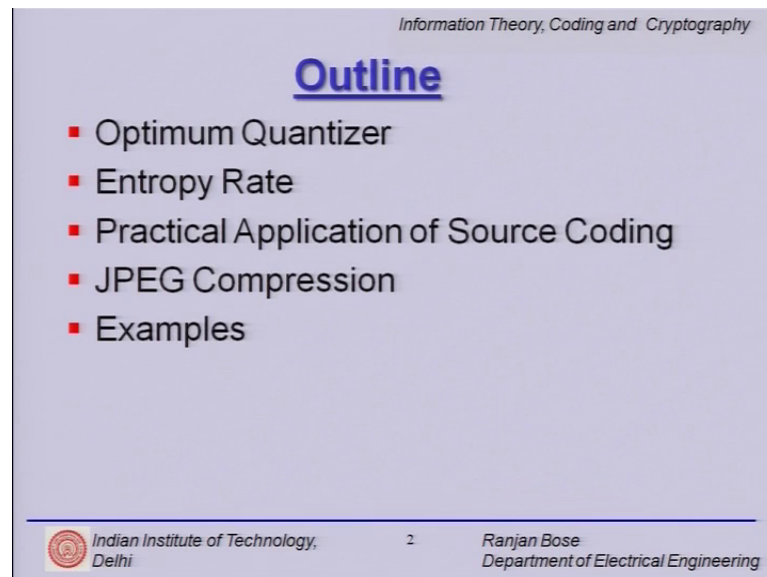


Information Theory, Coding and Cryptography
Dr. Ranjan Bose
Department of Electrical Engineering
Indian Institute of Technology, Delhi

Module - 07
Source Coding
Lecture – 07

Hello and welcome to module 7 on Source Coding. Let us start with a brief outline of today's talk.

(Refer Slide Time: 00:23)



Information Theory, Coding and Cryptography

Outline

- Optimum Quantizer
- Entropy Rate
- Practical Application of Source Coding
- JPEG Compression
- Examples

Indian Institute of Technology, Delhi 2 Ranjan Bose
Department of Electrical Engineering

We will discuss the practical applications of source coding and we will start with an Optimal Quantizer. We will discuss about entropy rate and then finally, some practical applications of source coding, leading to JPEG compression and of course, we will look at some examples so that is the road map for today.

(Refer Slide Time: 00:51)

Information Theory, Coding and Cryptography

Recap

- Huffman Coding
- Arithmetic Coding
- Lempel Ziv Coding
- Run Length Coding
- Examples

Indian Institute of Technology, Delhi

3

Ranjan Bose
Department of Electrical Engineering

We begin by a quick recap we have already done Huffman coding, which gives us a prefix code. We saw arithmetic coding how it is done, how it is better than Huffman coding because we go on the real line between 0 and 1. But both Huffman coding and arithmetic coding required the input probabilities a priori.

If we do not have the input probabilities, both at the transmission side and the receiving side we are in trouble. Lempel Ziv coding on the other hand does not required you to have the measurement or estimate of the input probabilities, before you start encoding, so that is the good part. We also looked at this notion of run length coding.

What is a run? It is a long sequence of 1 or a long sequence of 0's. These are typically found in facts data or in images or sometimes in peculiar encoding, that is required to be done. So, run length coding takes care of this long runs of 1's and 0's in which you pretty much encode. How many of what that is the basic idea behind run length coding, and we looked at some examples in the last class.

(Refer Slide Time: 02:18)


Information Theory, Coding and Cryptography

Optimum Quantizer

- Consider a continuous amplitude signal whose amplitude is not uniformly distributed, but varies according to a certain probability density function, $p(x)$.
- We wish to design the optimum scalar quantizer that minimizes some function of the quantization error $q = \tilde{x} - x$, where \tilde{x} is the quantized value of x .
- The distortion resulting due to the quantization can be expressed as

$$D = \int_{-\infty}^{\infty} f(\tilde{x} - x)p(x)dx$$

where $f(\tilde{x} - x)$ is the desired function of the error.

 Indian Institute of Technology, Delhi 4 Ranjan Bose
Department of Electrical Engineering

Now let us look at an optimum quantizer design, this is a practical problem and we would see how the concept of source coding can be applied here. So, let us consider a continuous amplitude signal whose amplitude is to be quantized, because amplitude is not uniformly distributed is this a good assumption well. In many real life cases this happens to be true even if I take my speech or I measure the vibrations of a bridge most slightly the amplitude is distributed closer to the mean value and the very high, or very low values on either side of 0 happens rarely.

So, let us have this continuous amplitude signal whose amplitude is not uniformly distributed. We also do something else the probability density function of this amplitude varies and it is given by p of x . Now we wish to design an optimum scalar quantizer that minimizes some function of the quantization error ok. So, quantization error q is x tilde minus x where x tilde is the quantized value of x .

Now, we define the distortion as D , it is integrated from minus infinity to infinity some function of x tilde minus x . So, this is the kind of quantization error f is a function of that weighted with a probability distribution of the amplitude p of x t of x fine. So, I can have any function of properly defined function f of x tilde minus x .

(Refer Slide Time: 04:19)


Information Theory, Coding and Cryptography

Optimum Quantizer

- An optimum quantizer is one that minimizes D by optimally selecting the output levels and the corresponding input range of each output level.
- The resulting optimum quantizer is called the *Lloyd-Max quantizer*.
- For an L -level quantizer the distortion is given by

$$D = \sum_{k=1}^L \int_{x_{k-1}}^{x_k} f(\tilde{x}_k - x) p(x) dx$$

▶

 Indian Institute of Technology, Delhi

5

Ranjan Bose
Department of Electrical Engineering

Now, an optimum quantizer is one that, minimizes D by optimally selecting the output levels and the corresponding input range of each output level basically that is the problem of a quantizer design. So, if you do not want to do any of this hard work, you have a uniform quantizer but that is clearly not going to solve our problem correctly, because the amplitude itself is non-uniformly distributed.

So, this kind of a problem was first formulated by Lloyd Max and it is called the Lloyd Max quantizer. Now suppose we have one levels of this quantizer the distortion is simply given by D , equal to summation over all the levels k is equal to 1 through l and then I have this integration of f of this quantization error x_k tilde minus x . So, this time it is x_k and weighted by $p(x) dx$.

(Refer Slide Time: 05:27)


Information Theory, Coding and Cryptography

Optimum Quantizer

- The necessary conditions for minimum distortion are obtained by differentiating D with respect to $\{x_k\}$ and $\{\tilde{x}_k\}$.
- As a result of the differentiation process we end up with the following system of equations

$$f(\tilde{x}_k - x_k) = f(\tilde{x}_{k+1} - x_k), \quad k = 1, 2, \dots, L-1$$

$$\int_{x_{k-1}}^{x_k} f'(\tilde{x}_{k+1} - x) p(x) dx, \quad k = 1, 2, \dots, L$$

 Indian Institute of Technology, Delhi
6
Ranjan Bose
Department of Electrical Engineering

So, the necessary conditions for minimum distortion are obtained by differentiating D with respect to x_k and x_k tilde. And if you do that basic math you will come up with a condition that $f(x_k \text{ tilde} - x_k)$ is equal to $f(x_{k+1} \text{ tilde} - x_k)$ for k is equal to 1 to up to $L-1$ right.

(Refer Slide Time: 05:56)


Information Theory, Coding and Cryptography

Optimum Quantizer

- For $f(x) = x^2$, i.e., the mean square value of the distortion, the equations simplify to

$$x_k = \frac{1}{2}(\tilde{x}_k + \tilde{x}_{k+1}), \quad k = 1, 2, \dots, L-1,$$

$$\int_{x_{k-1}}^{x_k} (\tilde{x}_k - x) p(x) dx = 0, \quad k = 1, 2, \dots, L.$$

 Indian Institute of Technology, Delhi
7
Ranjan Bose
Department of Electrical Engineering

And then if you take a special case of mean square error. So, I want to design an optimum quantizer that minimizes the mean square error it is a good function to take. You come up with these levels the quantization levels x_k given as $x_k \text{ tilde} + x_{k+1} \text{ tilde}$


tilde by 2. And we have this condition that $\int x_k \tilde{p}_k dx$ integration over 1 interval is equal to 0. So, this is just an example of how you solve it. But the question is now how do we fit in source coding here?

(Refer Slide Time: 06:41)

Information Theory, Coding and Cryptography

Optimum Quantizer

- The non uniform quantizers are optimized with respect to the distortion.
- However, each quantized sample is represented by *equal* number of bits (say, R bits/sample).
- It is possible to have a more efficient variable length coding. The discrete source outputs that result from quantization can be characterized by a set of probabilities p_k .
- These probabilities can then be used to design efficient variable length codes (source coding).
- In order to compare the performance of different nonuniform quantizers, we first fix the distortion, D , and then compare the average number of bits required per sample.

 Indian Institute of Technology, Delhi

8

Ranjan Bose
Department of Electrical Engineering

So, these non uniform quantizers are optimized with respect to distortion that is what we have done. So, we have defined a distortion measure and we optimized according to that, but the question is each quantized sample is represented by equal number of bits.

So, so far what we have done is broken up the total range of the amplitude into quantization levels, but then suppose there are 8 quantization levels we would put 3 bits per sample, but that is not clearly the best way to do things. Because of the p_x the amplitude distribution all quantization levels are not equally likely.

So, I will treat the quantization levels as symbols they are not equally likely just like my example earlier. Suppose I am talking and my voice has been recorded I will have probably more samples closer to the mean value and I scream only once in a while. So, very high values are taken very rarely. So, what we do is we would like to have a more efficient variable length coding for this optimum quantizer, which was optimized for the distortion measure, and these probabilities are then used to design the efficient variable length code therefore, the source coding.

So, quantization itself is a compression to begin with why I have this continuous signal and it has infinite information content. Take any sample and you should require infinite number of bits to quantize and represent it, but the moment we quantize it we have a finite number of bits.

The number of levels of the quantizer actually decides how many bits, but sample, but we are not content with that what we would like to further do is understand that different samples occur with different probabilities and we will do Huffman coding on that alright.

(Refer Slide Time: 09:03)

Example Information Theory, Coding and Cryptography

- Consider an eight level quantizer for a Gaussian random variable.
- The random variable has zero mean and variance equal to unity.
- For a mean square error minimization:

Level, k	x_k	\tilde{x}_k	$P(x_k)$	Huffman Code
1	-1.748	-2.152	0.040	0010
2	-1.050	-1.344	0.107	011
3	-0.500	-0.756	0.162	010
4	0	-0.245	0.191	10
5	0.500	0.245	0.191	11
6	1.050	0.756	0.162	001
7	1.748	1.344	0.107	0000
8	∞	2.152	0.040	0011

Indian Institute of Technology,
Delhi

9

Ranjan Bose
Department of Electrical Engineering

So, let us look at an example. So, we have 8 level quantizers, but my amplitude is not uniformly distributed and therefore, I have a simple x_k which tells me how I have distributed my levels. So, if you look at this graphically.

more frequently occurring levels quantization values as with fewer number of bits as opposed to they are rarely occurring very high and very low values.

So, level 1 if you would like to call it compression happened when I quantized it. The infinite number of bits which was supposed to be used to represent a continuous random variable got represented with finite number of bits. But here in this case ideally I should have had 3 bits for every level, but looking at the different probabilities I have put a more efficient Huffman code ok.

(Refer Slide Time: 13:16)

Information Theory, Coding and Cryptography

Example

- For these values, $D = 0.0345$ which equals -14.62 dB.
- The number of bits/sample for this optimum 8-level quantizer is $R = 3$.
- On performing Huffman coding, the average number of bits per sample required is $R_H = 2.88$ bits/sample.
- The theoretical limit is $H(X) = 2.82$ bits/sample.

Level, k	x_k	\hat{x}_k	$P(x_k)$	Huffman Code
1	-1.748	-2.152	0.040	0010
2	-1.050	-1.344	0.107	011
3	-0.500	-0.756	0.162	010
4	0	-0.245	0.191	10
5	0.500	0.245	0.191	11
6	1.050	0.756	0.162	001
7	1.748	1.344	0.107	0000
8	∞	2.152	0.040	0011

Indian Institute of Technology,
Delhi
10
Ranjan Bose
Department of Electrical Engineering

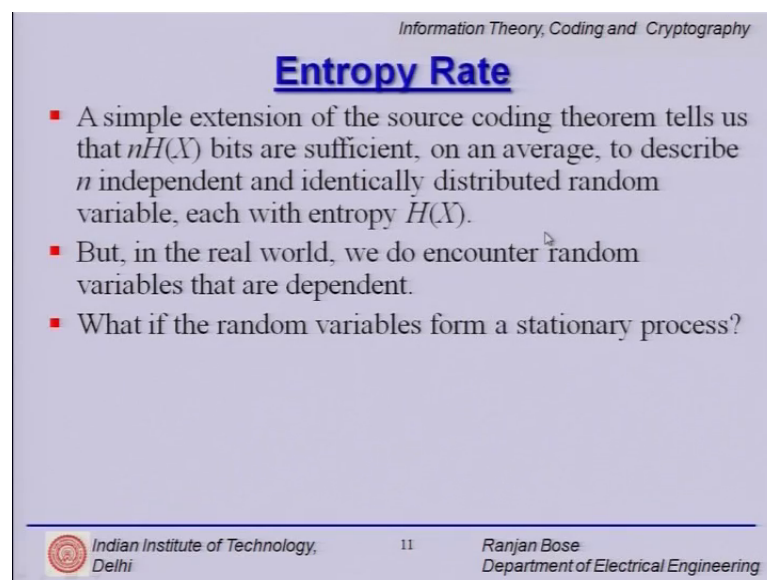
Just to complete this example we have now a distortion value D is equal to 0.0345 which is can be represented in terms of dB is minus 14.62 dB. So, you can compare 2 different quantizers with respect to the distortions. If I have defined mean square error and 2 people are selling you 2 quantizers you can calculate the distortions with the respective mean square error and then choose to buy any one.

Now clearly we did not go for R is equal to 3 bits per symbol, we chose not to have equal number of bits per level. And since we did Huffman coding we ended up having 2.88 bits per sample as opposed to 3. And of course, if you look at this P of X_x and calculate the theoretical limit it tells you that it is 2.82 bits per sample that I could have achieved the best it is not in my control.

In this practical scenario I can at best give you 2.88 bits per sample on an average. So, that is the 3 most critical words on an average, it is not just if you just pick up it is my bad day and a very very high value of the sample comes in of course I am spending 4 bits per symbol.

Another high value sample comes in again I end up spending 4 bits, but then the law of averages will take over soon or later I will start getting values closer to the mean and then I will have 2 bits per symbol and so on and so forth, and the average out will be roughly 2.88. So, this is an example where we have been able to use source coding to reduce the bits per sample to be sent.

(Refer Slide Time: 15:19)



Information Theory, Coding and Cryptography

Entropy Rate

- A simple extension of the source coding theorem tells us that $nH(X)$ bits are sufficient, on an average, to describe n independent and identically distributed random variable, each with entropy $H(X)$.
- But, in the real world, we do encounter random variables that are dependent.
- What if the random variables form a stationary process?

Indian Institute of Technology, Delhi 11 Ranjan Bose
Department of Electrical Engineering

Now, we change gears and we talk about something called as Entropy Rate. Now we observed in the last class when we talked about using blocks for coding, and trying to achieve the source code theoretical limit. We observe that a simple extension of the source coding theorem is that $nH(X)$ bits are sufficient on an average to describe n independent and identically distributed random variables, each with entropy $H(X)$ it is just very intuitive.

But in real world we do encounter random variables that are dependent and the question is what if the random variables form a stationary process. We do encounter that quite a bit in electrical engineering. So, how do we define the corresponding entropy for that do we have a different measure?

(Refer Slide Time: 16:18)


Information Theory, Coding and Cryptography

Stationary stochastic process

- A stochastic process is said to be **stationary** if the joint distribution of any subset of the sequence of random variables is invariant with respect to shifts in time index, i.e.,

$$P(X_1 = x_1, X_2 = x_2, \Lambda X_n = x_n) = P(X_{1+m} = x_1, X_{2+m} = x_2, \Lambda X_{n+m} = x_n)$$

for every shift m and for all $x_1, x_2, \Lambda x_n \in X$

 Indian Institute of Technology, Delhi 12 Ranjan Bose
Department of Electrical Engineering

So, let us define a stochastic process is said to be stationary if the joint distribution of any subset of the sequence of random variables is invariant with respect to shifts in the time index. So, if see in this definition earlier, but we put it again together and here this lambda is actually dot dot dot.

So, we have X_1 is equal to x_1 capital X_2 is equal to x_2 dot dot dot up to X_n equal to x_n right is given therefore, every shift m . So, I have shifted the indices 1 plus m 2 plus m n plus m and therefore, this kind of a definition tells us that the stochastic process is indeed stationary ok.

(Refer Slide Time: 17:13)

Information Theory, Coding and Cryptography

Markov Process


- A discrete stochastic process X_1, X_2, \dots is said to be a **Markov Chain** or a **Markov Process** if, for $n = 1, 2, \dots$

$$P(X_{n+1} = x_{n+1} | X_n = x_n, X_{n-1} = x_{n-1}, \Lambda, X_1 = x_1) = P(X_{n+1} = x_{n+1} | X_n = x_n)$$

for all $x_1, x_2, \Lambda, x_n, x_{n+1} \in X$

- The probability density function of a Markov process can be written as

$$p(x_1, x_2, \Lambda, x_n) = p(x_1) p(x_2 | x_1) p(x_3 | x_2) \Lambda p(x_n | x_{n-1})$$

 Indian Institute of Technology,
Delhi13Ranjan Bose
Department of Electrical Engineering

Now we go a little bit further and we define what is a Markov Process or a Markov chain. So, a discrete stochastic process X_1, X_2, \dots . So, it can go up to infinity is said to be a Markov Chain or a Markov Process. If for n is equal to $1, 2, \dots$, so it can go up to infinity is defined as the conditional probability $p(x_{n+1})$ should be equal to x_{n+1} given all the previous one is nothing, but the conditional probability $p(x_{n+1})$ given x_n that is it just depends on the previous sample for all x_1, x_2, \dots, x_n .

Therefore the probability density function of a Markov process can be written as $p(x_1, x_2, \dots, x_n)$ this is nothing, but $p(x_1)$ times $p(x_2 | x_1)$, but rest do not matter times $p(x_3 | x_2)$, the rest do not matter \dots up to x_n given x_{n-1} . So, that is the beauty of a Markov process.

(Refer Slide Time: 18:32)

Information Theory, Coding and Cryptography

Entropy Rate


- If we have a sequence of n random variables, it is interesting to explore how the entropy of the sequence grows with n .
- Entropy rate is used to define this rate of growth.
- The **Entropy Rate** of a stochastic process X is given by

$$H(X) = \lim_{n \rightarrow \infty} \frac{1}{n} H(X_1, X_2, \dots, X_n)$$

provided the limit exists.

For **Stationary Markov Chain**, the entropy rate is given by

$$H(X) = \lim_{n \rightarrow \infty} H(X_n | X_{n-1}, \dots, X_1) = \lim_{n \rightarrow \infty} H(X_n | X_{n-1}) = H(X_2 | X_1)$$

 Indian Institute of Technology, Delhi 14 Ranjan Bose
Department of Electrical Engineering

So, why are we talking about Markov process all of a sudden? We will connect it quickly if we have a sequence of n random variables please note a sequence of n random variables it is interesting to explore how the entropy of the sequence grows within. So, it is a question we have posed to ourselves. So, entropy rate is used to define the rate of growth.

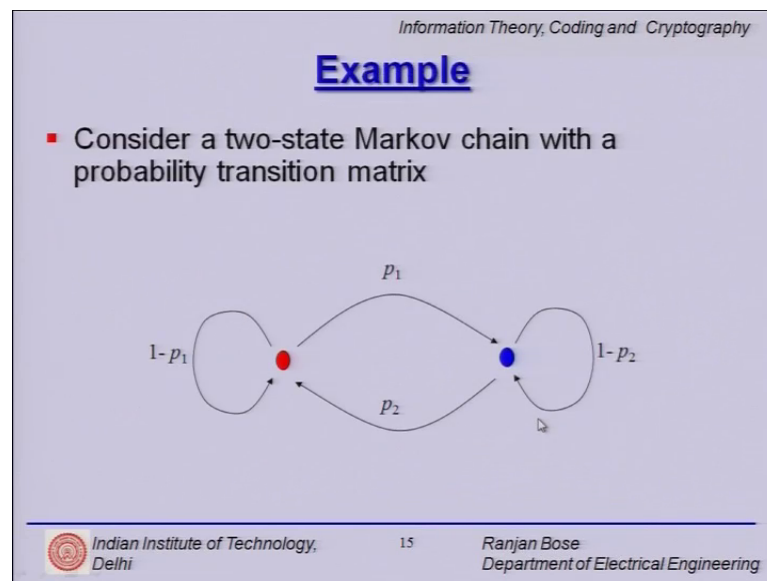
So, let us understand the physical significance we have a sequence of random variables, and they are not independent that dependent ok. So, the occurrence of each one probably depends on the previous one, and possibly the previous to previous one and so and so forth.

Now there random variables there dependent, so there is an entropy associated with it. But I led this sequence play along and as n increases how do you define this rate of growth? So, we define it the entropy rate of a stochastic process x is given by H of X , where limit n tends to infinity normalized by 1 over n $H(X_1, X_2, \dots, X_n)$ provided the limit exists, so it is possible that the limit may not exist.

So, what do we already know we know H joined entropy H of X_1 comma X_2 dot dot dot up to X_n right. What are these n ? These are these n random variables. So, we are talking about the joint entropy and have normalized it with them and I am letting this n tend to infinity.

If this quantity exists that is defined as the entropy rate ok. So, this thing has an equivalent definition for stationary Markov chain. So, we know stationarity, we know Markov chain. So, for stationary Markov chain the entropy rate is given by H of X again limit n tends to infinity X_n given, X_{n-1} dot dot dot dot up to X_1 and that can be simplified as n tends to infinity limit X_n given X_{n-1} . It is simply equal to $H(X_2)$ given X_1 using the properties of the Markov chain.

(Refer Slide Time: 21:14)




So, let us look at a simple example. So, let us consider a two state Markov chain with the probability transition matrix, such that I have got state one by the red dot, state two with the blue dot. So, this state transitions into the blue dot with probability p_1 , and it goes back to red dot with probability p_2 , and sometimes it stays as it is with probability $1 - p_2$. Similarly the red dot stays back with probability $1 - p_1$, so this is a simple state diagram of a two state Markov chain, they are fairly easy to understand this.

(Refer Slide Time: 22:10)

Information Theory, Coding and Cryptography

Example

- For stationary distribution, the net probability distribution across any cut set in the state transition graph should be zero.
- Let α and β be the stationary probabilities of the two states.
- The stationary distribution is given by
$$\alpha = \frac{p_2}{p_1 + p_2} \quad \beta = \frac{p_1}{p_1 + p_2}$$
- The entropy of the state X_n at time n will be
$$H(X_n) = H\left(\frac{p_2}{p_1 + p_2}, \frac{p_1}{p_1 + p_2}\right) = H(\alpha, \beta)$$

 Indian Institute of Technology, Delhi 16 Ranjan Bose
Department of Electrical Engineering

So, for stationary distribution, the net probability distribution across any cut set in the state transition graph should be 0 that is a standard definition. So, let alpha and beta be the stationary probabilities of the two states. What are we talking about? Well what is the probability of being in red, or the probability of being in the blue state ok that can be written as alpha is equal to p_2 over $p_1 + p_2$ is a distribution stationary distribution for alpha being the probability of state red.

And similarly beta is p_1 over $p_1 + p_2$ it does not require too much thinking to note that alpha plus beta should be 1, I should be either in state red or in state blue. But there is a probability of being in red and the probability of being in blue and they keep flipping with certain probabilities it switches back and forth sometimes it does not switch.

And therefore, I can determine what is the probability of being in state 1 and state 2? And looking at the definition you have the H of X_n , the entropy for state X_n at any time n can be written as well there only two possibilities. So, it is like tossing of the coin sometimes head comes, sometimes tail comes, sometimes in alpha sometimes I am in state blue, sometimes in state red. So, alpha and beta are their probabilities and I get H of alpha comma beta right.

(Refer Slide Time: 23:51)

Information Theory, Coding and Cryptography


Example

- The entropy of the state X_n at time n will be

$$H(X_n) = H\left(\frac{p_2}{p_1+p_2}, \frac{p_1}{p_1+p_2}\right) = H(\alpha, \beta)$$

- The entropy rate of this two-state Markov chain is given by

$$H(X) = H(X_2 | X_1) = \frac{p_2}{p_1+p_2} H(p_1) + \frac{p_1}{p_1+p_2} H(p_2)$$

 Indian Institute of Technology,
Delhi17Ranjan Bose
Department of Electrical Engineering


So, if you want to find out the entropy rate of these two state Markov chain, it is given by $H(X)$ is $H(X_2 | X_1)$ right is the weighted probability, so $\frac{p_2}{p_1+p_2}$ which is nothing, but α . So I have got $\alpha H(p_1) + \beta H(p_2)$.

(Refer Slide Time: 24:20)

Information Theory, Coding and Cryptography

Application of Source Coding

- Lets consider the lossless compression option of the Joint Photographic Experts Group (JPEG) image compression standard.
- The JPEG image compression standard is actually a description of 29 distinct coding systems for compression of images.
- Why are there so many approaches?
- It is because the needs of users vary so much with respect to quality versus compression and compression computation time.
- We shall briefly discuss here two methods that use entropy coding.

 Indian Institute of Technology,
Delhi18Ranjan Bose
Department of Electrical Engineering

So, this is a simple example of how you can apply your concepts to entropy rate and you can define it. So, you could have 3 state; 4 state, up to n state Markov process alright.

So, now we come to a practical application of source coding. We have already looked at the optimum quantizer, but now let us look at a more widespread application which

touches our lives almost on a daily basis. Whenever we download an image on a mobile phone, or send a picture to our friend we most likely are using some kind of an image compression technique. And one of the most commonly used one is the JPEG which stands for joint photographic experts group.

So, even though it is almost a defacto standard, it is commonly used other JPEG standards available today. But JPEG is an image compression standard defacto which is actually description of several 25 distinct coding systems. Why there are so many approaches, because the users the needs of the users vary a lot with respect to the quality, compression, and computation time. So, JPEG permits the user to probably have a poor quality image which is compressed, but requires lesser time and uses fewer number of bits. But let us describe 2 methods that we have studied which are used in JPEG compression standard.

(Refer Slide Time: 26:01)

Information Theory, Coding and Cryptography

JPEG Compression

- The two lossless JPEG compression options differ only in the form of the entropy code that is applied to the innovations data.
- The user can choose to use either a **Huffman code** or an **Arithmetic code**.
- Some compression can be achieved if we can predict the next pixel using the previous pixels.
- In this way we just have to transmit the prediction coefficients (or difference in the values) instead of the entire pixel.

Indian Institute of Technology, Delhi

19

Ranjan Bose
Department of Electrical Engineering

So, the two techniques that we have looked at are Huffman coding, and Arithmetic coding ok. So, some compression can be achieved if we can predict the next pixel using the previous pixel and this kind of a differential coding is also used.

Now please note JPEG has two kinds of compression built into it one is lossy, where we lose information, and one is lossless where we can completely reconstruct the image the original image from the compressed image without any loss of data. So, we will talk briefly about both of them. So, what are we going to explore in the next few slides. We

are looking at how lossy compression works? How lossless compression works and how differential coding is used to our advantage in JPEG compression.

(Refer Slide Time: 26:58)


Information Theory, Coding and Cryptography

JPEG Compression: DCT

- Discrete Cosine Transform (DCT):

$$Y(k, l) = \sum_{i=0}^{N-1} \sum_{j=0}^{M-1} 4y(i, j) \cos\left(\frac{\pi k}{2N}(2i+1)\right) \cos\left(\frac{\pi l}{2M}(2j+1)\right)$$

where the input image is N pixels by M pixels, $y(i, j)$ is the intensity of the pixel in row i and column j , $Y(k, l)$ is the DCT coefficient in row k and column l of the DCT matrix.

 Indian Institute of Technology, Delhi 20 Ranjan Bose
Department of Electrical Engineering

So, let us look at the nuts and bolts the basic building box the JPEG compression uses something called discrete cosine transform. Just as Fourier transform takes you from say time domain to frequency domain, or spatial domain to spatial frequency domain..

Similarly DCT takes you from the spatial domain to the spatial frequency domain. So, it is defined as follows, so there is a two double double summation and then $y_{i j}$ is the spatial domain, and capital Y_{kl} represents the spatial frequency domain.

Now, please note this is a real transform it also does the job similar to Fourier it is a real transform. And in images we would rather work with real transforms because we would not like to put an additional effort to handle complex numbers.


(Refer Slide Time: 28:08)

Information Theory, Coding and Cryptography

JPEG Compression: DCT

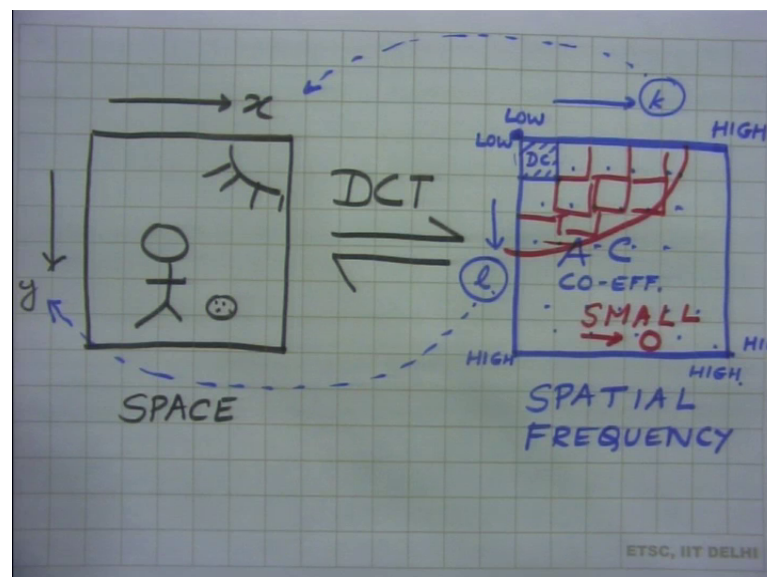
$$Y(k,l) = \sum_{i=0}^{N-1} \sum_{j=0}^{M-1} 4y(i,j) \cos\left(\frac{\pi k}{2N}(2i+1)\right) \cos\left(\frac{\pi l}{2M}(2j+1)\right)$$

- All DCT multiplications are real.
- This lowers the number of required multiplications, as compared to the discrete Fourier transform.
- For most images, much of the signal energy lies at low frequencies, which appear in the upper left corner of the DCT.
- The lower right values represent higher frequencies, and are often small (usually small enough to be neglected with little visible distortion).

 Indian Institute of Technology, Delhi 21 Ranjan Bose
Department of Electrical Engineering

So, all DCT multiplications are real right. So, this lowers the number of required multiplications as compared to discrete Fourier transform. But why do we use this DCT, what is so great about it. Well for most real world images most of the signal energy lies at the low frequency, which appears at the upper left corner of the DCT, the lower right values represent the higher frequency and often small values.

(Refer Slide Time: 28:58)



So, let us understand this a little better. Suppose we have a standard image ok. So, you can have anything that you would like any standard image ok. Now if we take the DCT,

it will also give you another matrix, but this is in spatial frequency domain. We are comfortable with time and frequency here we have space and frequency ok.

So, just as we have x and y we can have here k and l . But in some sense this k may correspond to the spatial frequency along the x and this l may correspond to the spatial frequency along y .

Now if you look at this, this is the low frequency and this is the high frequency. Again this is the low and going down this is the high and of course, this is high and high. this guy is the DC part. So, the first value is nothing, but the average of all the pixels taken. So, add them up and divide by the number of total number of pixels and you get this DC value, rest all of the coefficients are the AC coefficients.

Now, why DCT? If there is fair amount of correlation in this original image, then maximum energy is compartmentalized in the lower terms. So, here, here, here, here, here what does it mean? Please note this side represents low frequency as we go along this axis it represents high frequency, but of what dealing with x .

If we go along this side the same is along y if this high frequency spatial frequency, then you have high values here, but we live in a real world inertia dominates things do not change very frequently. And so there is a lot of correlation within the image and if the input image is highly correlated, then DCT does a great job of concentrating that energy into the lower values the low frequency components.

So, ideally I would probably use a kl transform and show that I can squeeze in the maximum information in terms of the frequency domain representation in minimum number of samples, but DCT does a great job if the input image is correlated. On the other hand if I have a computer generated image which is totally uncorrelated the pixels are not correlated then numbers will be all over the place here in DCT.


But since we are dealing with real world images we are able to focus and lead to maximum coefficients here and here all the coefficients become small, so small they tend to 0. Now, this critical observation will go a long way in compressing your image. So, let us see how DCT is applied.

(Refer Slide Time: 33:44)

Information Theory, Coding and Cryptography

JPEG Compression

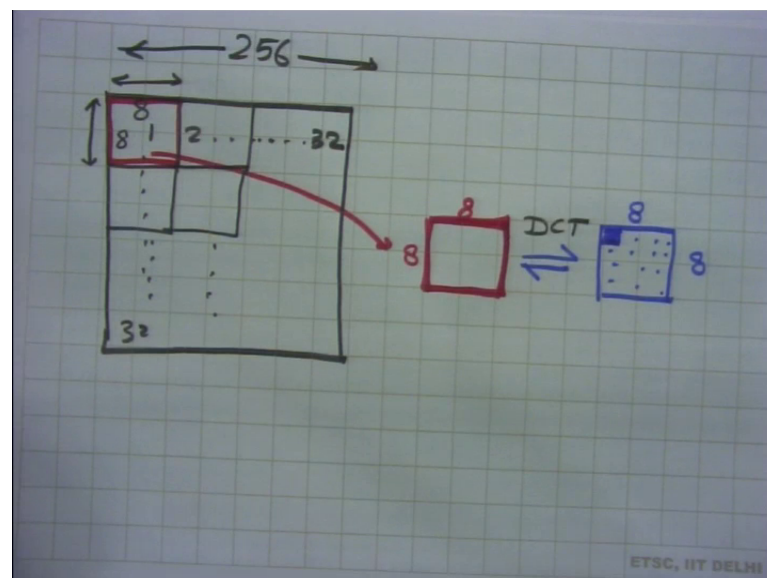
- DCT is applied to 8 by 8 pixel blocks of the image.
- Hence, if the image is 256 by 256 pixels in size, we break it into 32 by 32 square blocks of 8 by 8 pixels and treat each one independently.
- The 64 pixel values in each block are transformed by the DCT into a new set of 64 values.
- These new 64 values, known also as the DCT coefficients, form a whole new way of representing an image.
- The DCT coefficients represent the spatial frequency of the image sub-block.

 Indian Institute of Technology, Delhi

22

Ranjan Bose
Department of Electrical Engineering

(Refer Slide Time: 33:50)



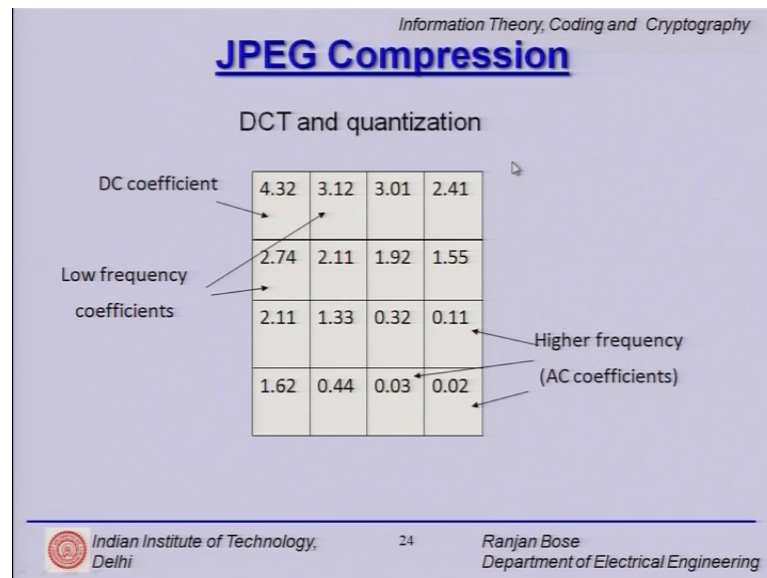
So, the first step is that you take an image. And you do not work with the whole image at a time you first divide the image into blocks. So, this could be 8 cross 8, or 16 cross 16, or 32 cross 32, but let us just focus on 8 cross 8 blocks. Why do we divide? Well if you take small blocks of 8 cross 8 you have much higher correlation, and this correlation will play a critical role in compression. It goes without saying higher the correlation, the fewer number of bits are required to represent that block right.

So, if we go back if the image is 256 by 256 pixels in size, we break it up into 32 by 32 square blocks of 8 by 8 pixels and treat each one of them independently. So, if this was 256 then I would have 1, 2, 3 up to 32, 1, 2, 3, up to 32. So, this will basically decide my sub block size.

Now, we only deal with 1 8 by 8 block at a time. So, this is where we start and the first step of course, is to take a DCT. So, this will also result in an 8 cross 8 DCT samples with the first one is a DC coefficient. So, we have taken a DCT here.

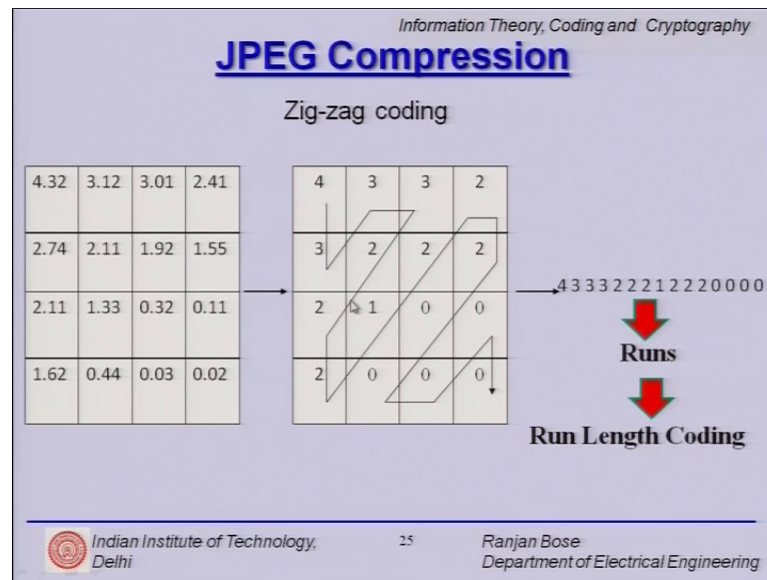
Now, what is what we observe is that first few coefficients have some larger values, but as we go along they are extremely close to 0. Let us look at a an example here.

(Refer Slide Time: 36:23)



This is a toy example I have not taken a block 8 by 8, but I have taken a 4 by 4 block. And for a real world image if you do a DCT these are the typical values you may encounter. So, let us go through this example the first one is a DC coefficient rest all our AC coefficients this one is just the average. So, 4.32 is average of the 4 point 4 cross 4 sized spatial image that we took rest all as you can see keep decreasing. And as we go to the really high frequency part there is very little value here ok.

(Refer Slide Time: 37:08)



So, now what do we do? What JPEG compression standard suggests is take that and first step is to quantize, I am going to do lossy compression, I am going to throw away things. What am I going to throw away from this 4.32 I throw away 0.32, for this 3.12 I throw away 0.1, for this 3.01 I throw away 0.01. It is just a toy example I can choose how much to throw away, but I have quantized it.

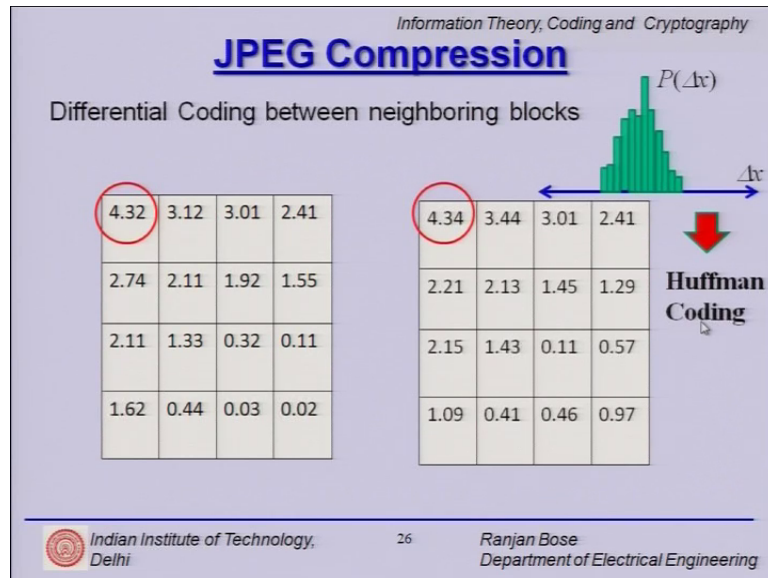
So, I get from this really 4.32 to 3.12 to 3.01 kind of a table I get 4 3 3 2 3 2 2 2 and so on so forth. Then I make another observation that this end is low low, this end is low high, high low, high high in terms of frequency. So, the frequency gradually increases as we move away from the center. So, we perform something called a zigzag coding, where I go zigzag and this is the path we follow to encode.

So, we do not do this raster line by line and line by line because this gradually decreases 4 3 3 2 jumps, 3 2 2 2 jumps, 0 0 2, so it does not make sense. So, we do this zigzag encoding and what does it lead well if I read out the numbers 4 3 3 2 2 2. So, I get this 4 3 3 suddenly I start seeing runs ok. The moment I see a run I am likely to do my run length coding, and this 3 3's or 3 2's will not really give me the compression..

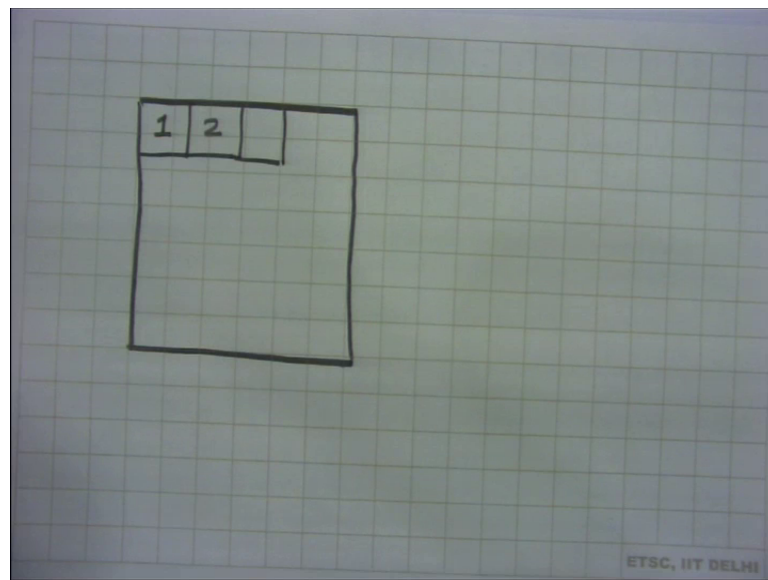
But what is amazing is this, lot of these AC coefficients have become 0's thanks to DCT which put packed all the energy into the first few coefficients rest all as 0's. So, of huge runs of 0's here 0 0 0 0 0 0 and efficiently I can do my run length coding here. Please

remember we are not talking about the whole image we are only talking about the sub block.

(Refer Slide Time: 39:44)



(Refer Slide Time: 40:02)



Now, we do this same thing with all of them; now please note it is not just one block. So, if you look at your image which you broke it up into blocks. Now let us take block 1, and block 2, so let us consider these 2 blocks. Because these 2 blocks are close to each other and their average values are likely to be similar.

So, let us just write down the DCT coefficients of block 1 and block 2 which are adjacent. Now the DC coefficient the first box please remember this is in the spatial transform domain represents the average value, this represents the average value of the next one, they are quite close to each other.

In fact, if you take the delta it is a very small number. So, we do the differential coding between these two. What is differential coding? Well what you do is you find the difference and I do this for blocks 2, and 3; blocks 3 and 4; 4 and 5 and all the neighboring blocks. If we now just do an analysis if we plot on the x axis the delta x which is the difference between these two values and the histogram the probability measure of sorts and then we see a distribution that is very close to 0 we see lot of samples which is not surprising because they do not change drastically.

But as we move away from 0 the probability is changing, the moment we have a distribution we should be able to do Huffman coding. You give me symbols or samples with different probabilities associated with that and of course, I will do Huffman coding. So, differential coding plus Huffman coding, or differential encoding or arithmetic coding. Because we have learned both of them and we should be able to deliver a much higher level of compression alright.

(Refer Slide Time: 42:26)

Information Theory, Coding and Cryptography

Summary

- Optimum Quantizer
- Entropy Rate
- Practical Application of Source Coding
- JPEG Compression
- Examples

Indian Institute of Technology, Delhi 27 Ranjan Bose
Department of Electrical Engineering

So, we would like to now summarize what we have done in today's class. We started our discussion with the design of optimum quantizer, which minimizes a predefined

distortion function. We took an example of mean square error distortion, once you get that distortion measure and you minimize it you end up with the quantization levels.

Now if your input amplitude which is a continuous random variable it has a probability distribution. Based on that we associate probabilities with the different levels of the quantizer, and then we encode them using Huffman coding. So, no 2 levels of the quantizer would yield equal number of bits per sample. So, different samples require and will be represented by different number of bits depending upon what kind of Huffman coding we are doing.

We then looked at entropy rate ok, and then finally we discussed this application of source coding which is JPEG compression. We looked at how Huffman coding or arithmetic coding or even run length coding together can be used to compress the image. So, that we also looked at some examples that brings us to the end of today's module. And we will look at other aspects of source coding in the subsequent module.