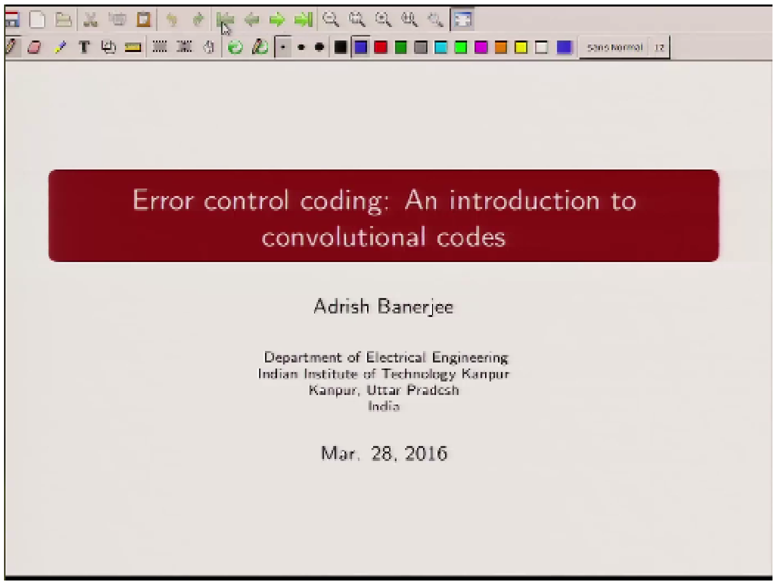
# Indian Institute of Technology Kanpur
**National Programme on Technology Enhanced Learning (NPTEL)**
**Course Title**
**Error Control Coding: An Introduction to Convolutional Codes**

**Lecture-7**
**Turbo Codes**

**by**
**Prof. Adrish Banerjee**
**Dept. Electrical Engineering, IIT Kanpur**

Welcome to the course on error control coding, an introduction to convolutional code.

(Refer Slide Time: 00:21)



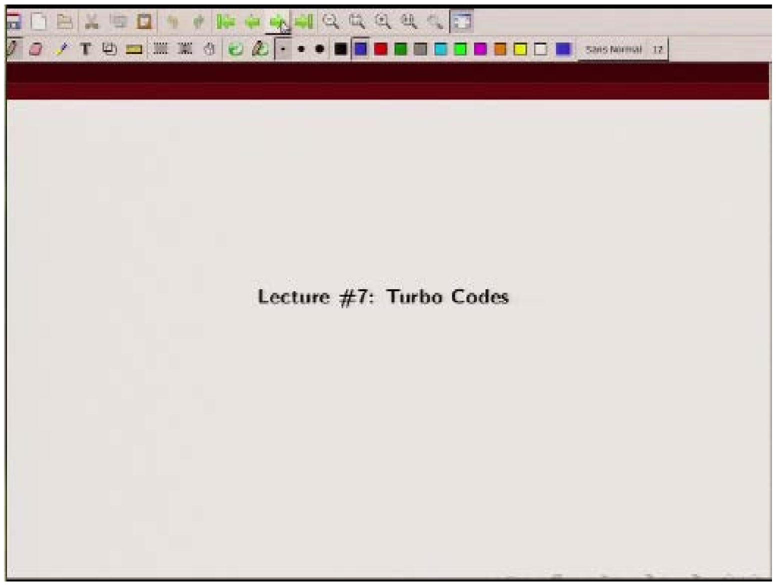
Error control coding: An introduction to convolutional codes

Adrish Banerjee

Department of Electrical Engineering
Indian Institute of Technology Kanpur
Kanpur, Uttar Pradesh
India

Mar. 28, 2016

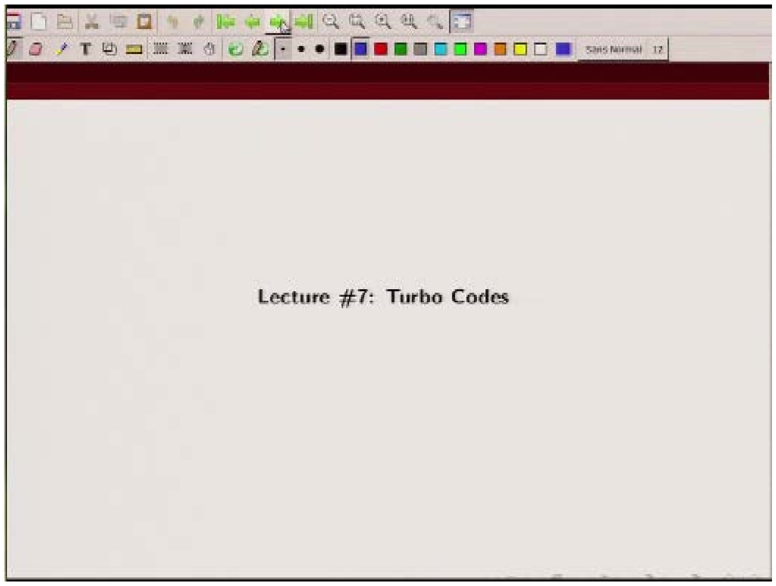
(Refer Slide Time: 00:20)



Lecture #7: Turbo Codes

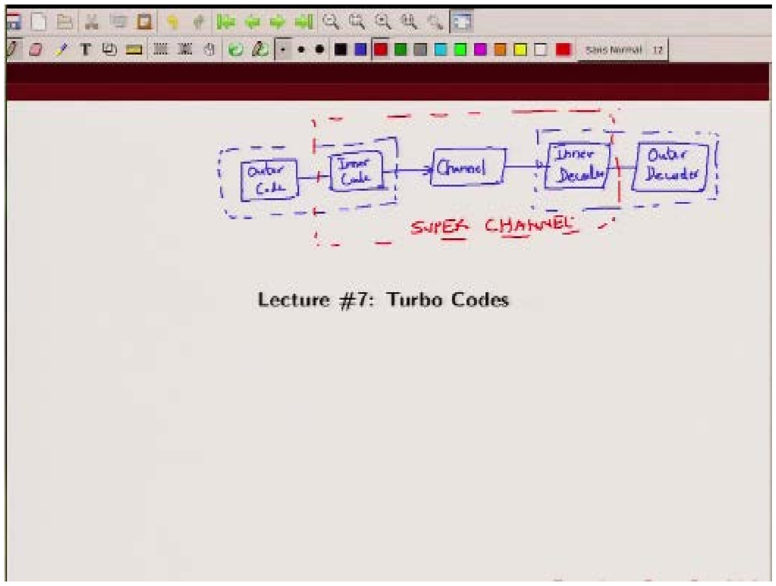
So today we are going to talk about a class of parallel concatenated convolutional code called turbo codes.

(Refer Slide Time: 00:31)


Lecture #7: Turbo Codes

Now what is concatenation? In concatenation we take small codes and we combine them to create a more powerful code.

(Refer Slide Time: 00:48)

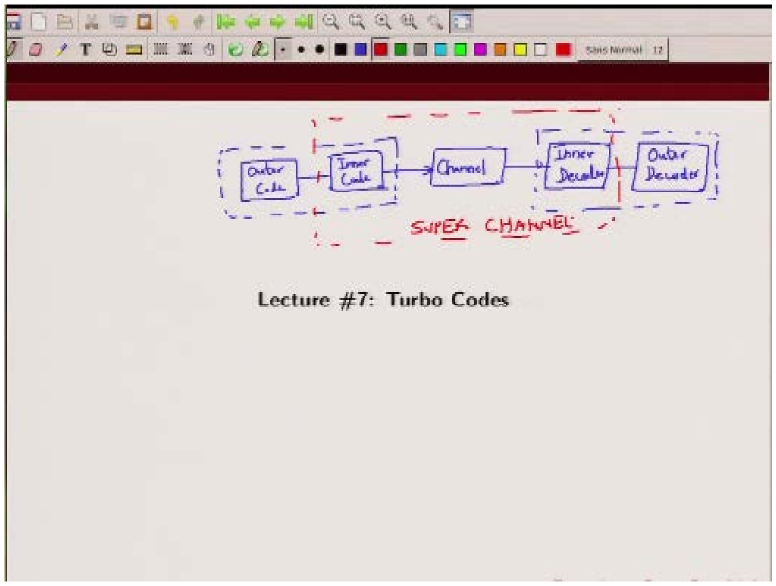


Lecture #7: Turbo Codes

Now this idea of concatenation was proposed by David Forney and what he did was he took two codes so there was one code called outer code and then here another code called inner code. And so this is your – you are creating up more powerful code using two codes and you send your coded bits through a channel and then you have the decoder for inner code.

So you have inner decoder and then you have a outer decoder. So this is your – I could also view this as, this as your super channel, this as your super channel, these are codes on that. Now this is a very simple idea of combining two relatively simpler code to create a more powerful code.

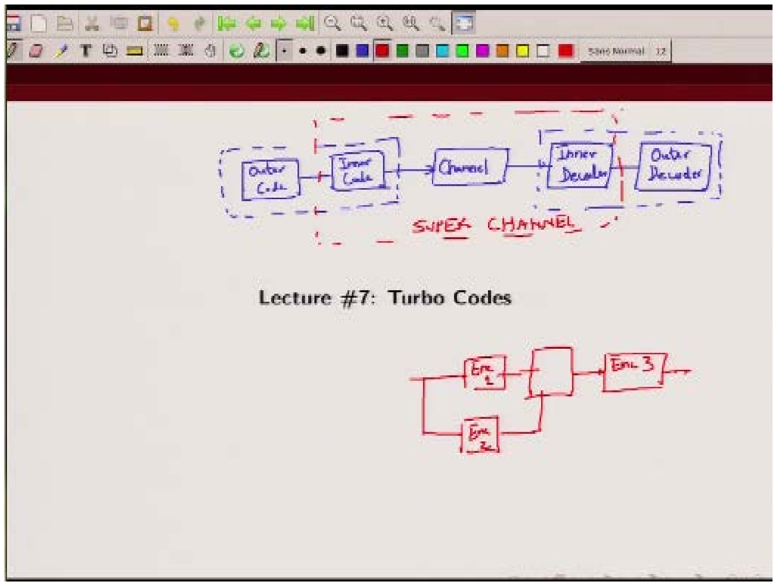
(Refer Slide Time: 02:08)



Lecture #7: Turbo Codes

And these codes could be block codes, convolutional codes, we could use anyone of them. Now in – this is one class of concatenated codes. Now there are various ways in which you could concatenate code. This is an example where two codes are serially connected, if you see the output of the first encoder is given as input to second encoder.

(Refer Slide Time: 02:41)

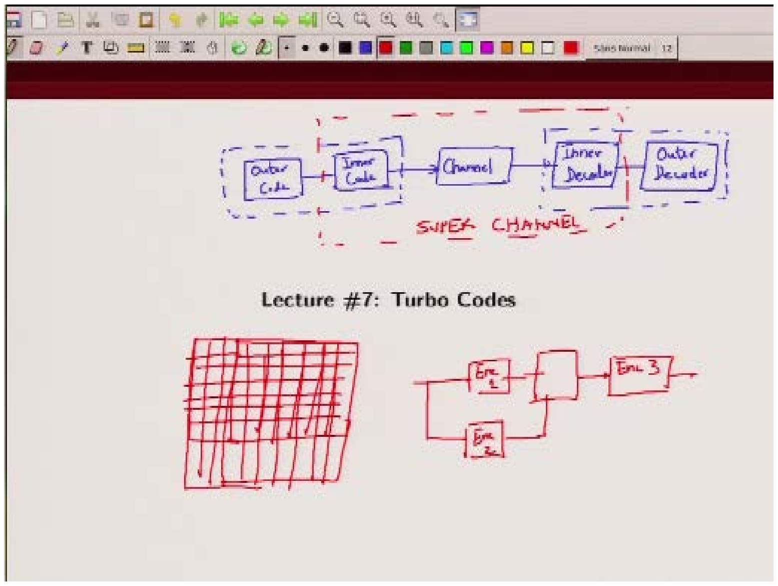


Lecture #7: Turbo Codes

Now there could be a configuration where two encoders are connected in parallel. So we have two encoders they are connected in parallels, this is your encoder one, encoder two, you could have like this or you could have a combination of both. So do you have some parallel concatenation the set is the combiner and then you have another code encoder three which is serial.

So this is a combination of parallel concatenation and serial concatenation. This is like hybrid concatenation, so there are various ways in which you can concatenated two codes.

(Refer Slide Time: 03:13)

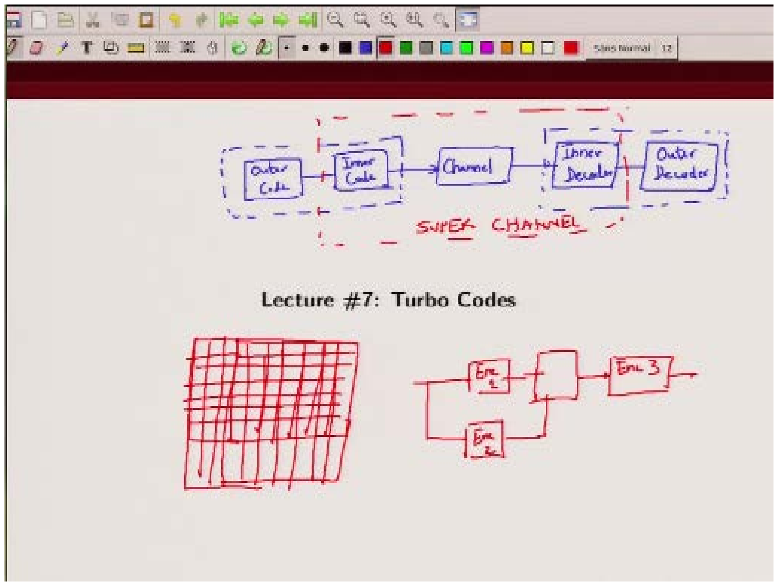


Lecture #7: Turbo Codes

There is a configuration which is called product code, so you have a – let us say k x k information bits you add some parity bits here corresponding to each of these K bits you add some parity bits and then you can read these information sequence column wise also and you can add parity bits corresponding to these column.

So you can have codes like this, this is called incomplete product code and then you could also have checks on checks. So that is the product code. So there are various ways in which you could concatenate it or combine to smaller codes to create a more powerful code.

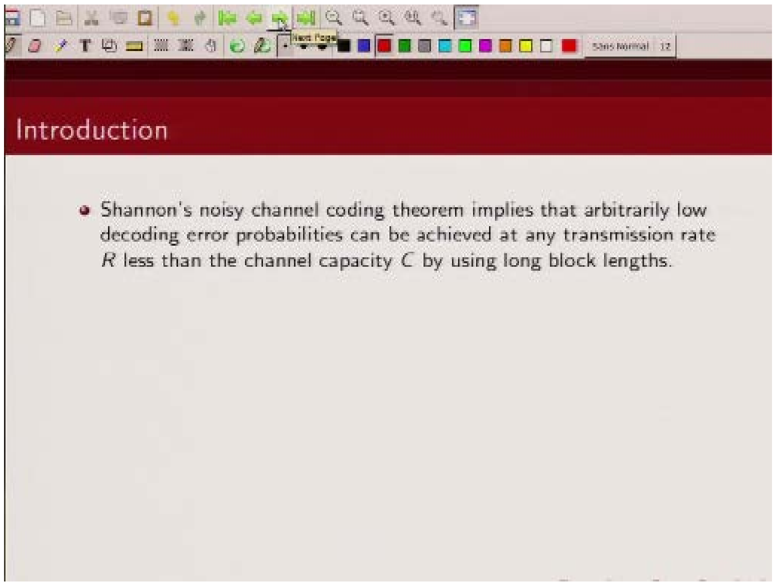
(Refer Slide Time: 04:07)



Lecture #7: Turbo Codes

In this lecture we are going to talk about one such class of concatenated codes which is called turbo codes. It is essentially a class of parallel concatenated codes.

(Refer Slide Time: 04:13)

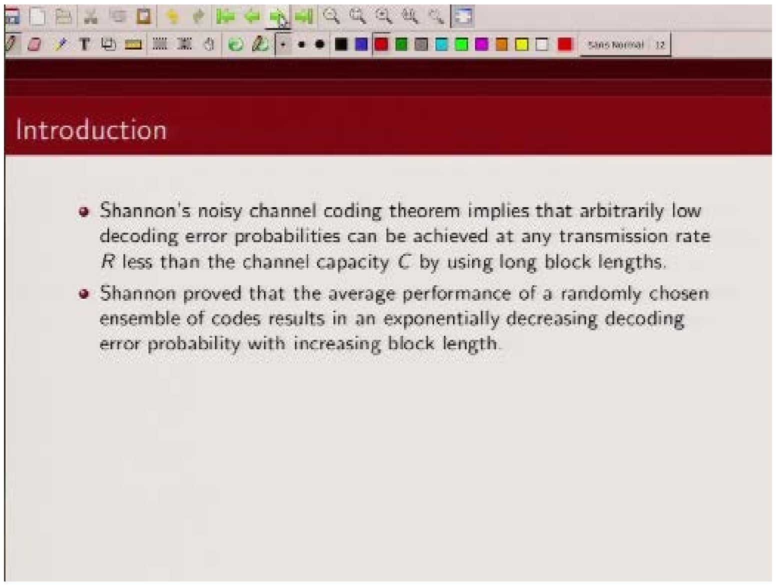


Introduction

- Shannon's noisy channel coding theorem implies that arbitrarily low decoding error probabilities can be achieved at any transmission rate $R$ less than the channel capacity $C$ by using long block lengths.

Now if you recall Shannon in his celebrated noisy channel coding theorem has mentioned that as long as we choose our transmission rate to below channel capacity we can reliably communicate order communication link. And – and his prove was basically based on fact that if you randomly select a code word of very large length then you can show that probability of error will go to zero as long as your transmission rate is below channel capacity and you transmit with long code words.

(Refer Slide Time: 04:55)

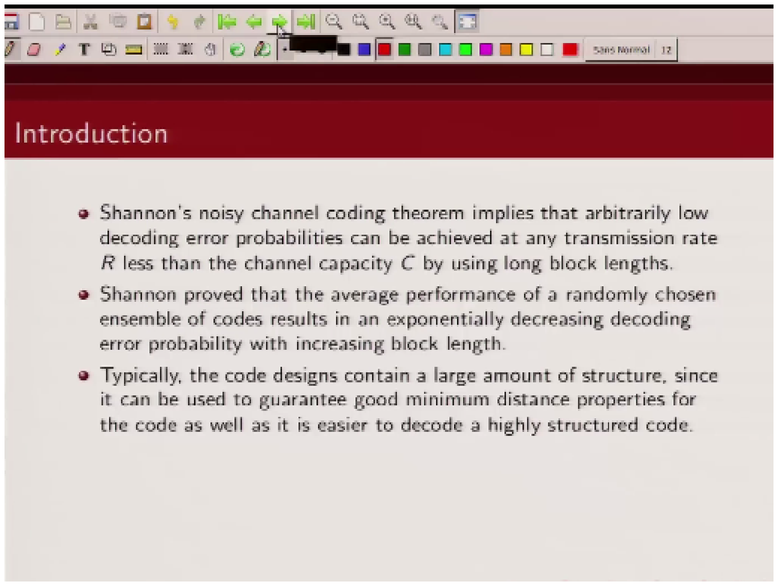


## Introduction

- Shannon's noisy channel coding theorem implies that arbitrarily low decoding error probabilities can be achieved at any transmission rate $R$ less than the channel capacity $C$ by using long block lengths.
- Shannon proved that the average performance of a randomly chosen ensemble of codes results in an exponentially decreasing decoding error probability with increasing block length.

Now people from 1948 onwards were trying to design codes close to channel capacity, and the problem was if you design a very random like code which does not have any decoding structure then the decoding complexity is very, very large. However if you put lot of structure into the code then it is not really random like and its performance is not very good.

(Refer Slide Time: 05:23)

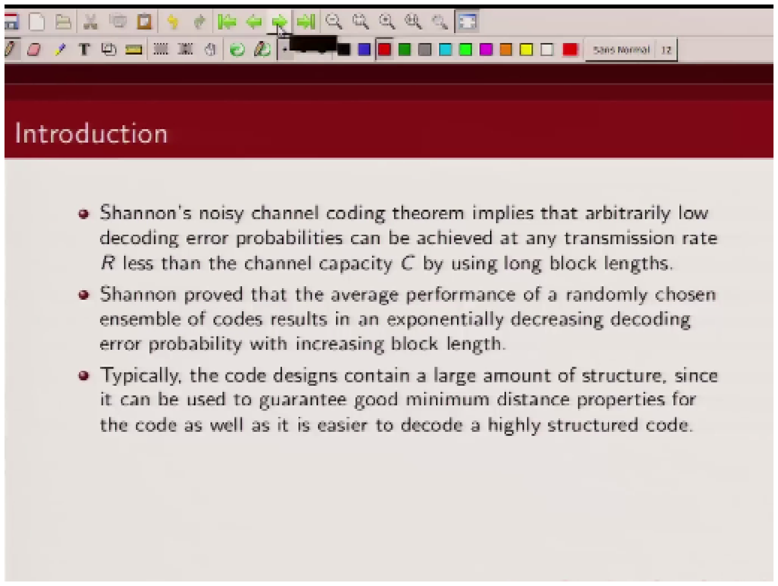


## Introduction

- Shannon's noisy channel coding theorem implies that arbitrarily low decoding error probabilities can be achieved at any transmission rate $R$ less than the channel capacity $C$ by using long block lengths.
- Shannon proved that the average performance of a randomly chosen ensemble of codes results in an exponentially decreasing decoding error probability with increasing block length.
- Typically, the code designs contain a large amount of structure, since it can be used to guarantee good minimum distance properties for the code as well as it is easier to decode a highly structured code.

So how do you design a code which has enough randomness into it, but then enough structure also which can be exploited for decoding the code?

(Refer Slide Time: 05:37)

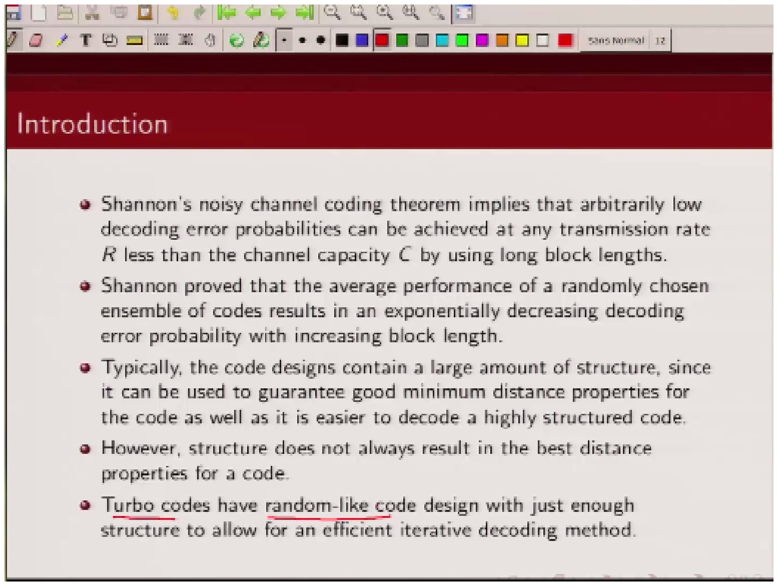


## Introduction

- Shannon's noisy channel coding theorem implies that arbitrarily low decoding error probabilities can be achieved at any transmission rate $R$ less than the channel capacity $C$ by using long block lengths.
- Shannon proved that the average performance of a randomly chosen ensemble of codes results in an exponentially decreasing decoding error probability with increasing block length.
- Typically, the code designs contain a large amount of structure, since it can be used to guarantee good minimum distance properties for the code as well as it is easier to decode a highly structured code.

So that is basically the challenge to design a code which has good minimum distance but then should be – we should be able to decode it also.

(Refer Slide Time: 05:46)

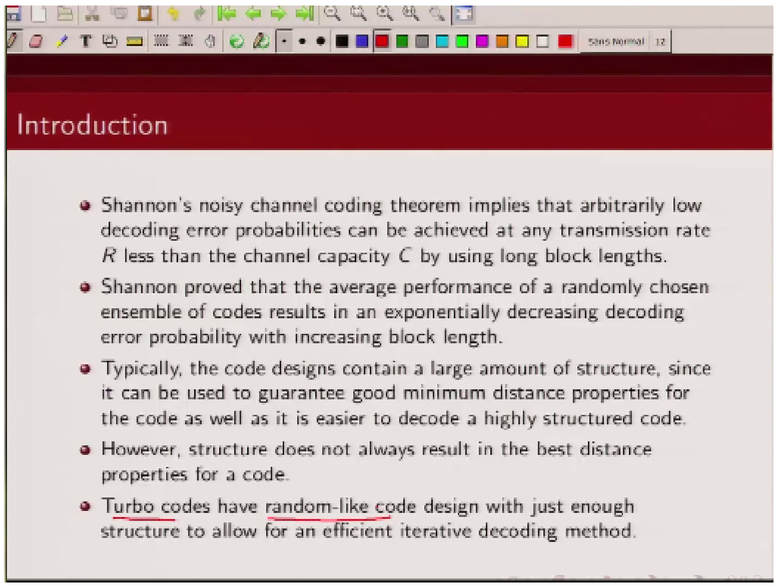


## Introduction

- Shannon's noisy channel coding theorem implies that arbitrarily low decoding error probabilities can be achieved at any transmission rate $R$ less than the channel capacity $C$ by using long block lengths.
- Shannon proved that the average performance of a randomly chosen ensemble of codes results in an exponentially decreasing decoding error probability with increasing block length.
- Typically, the code designs contain a large amount of structure, since it can be used to guarantee good minimum distance properties for the code as well as it is easier to decode a highly structured code.
- However, structure does not always result in the best distance properties for a code.
- Turbo codes have random-like code design with just enough structure to allow for an efficient iterative decoding method.

Now these turbo codes are a class of codes which are random like and we will show you how this random look like because of the inherent interleaver structure fitted in this parallel concatenation structure. And there is enough structure in the code which allows us to do efficient decoding of these error correcting codes.

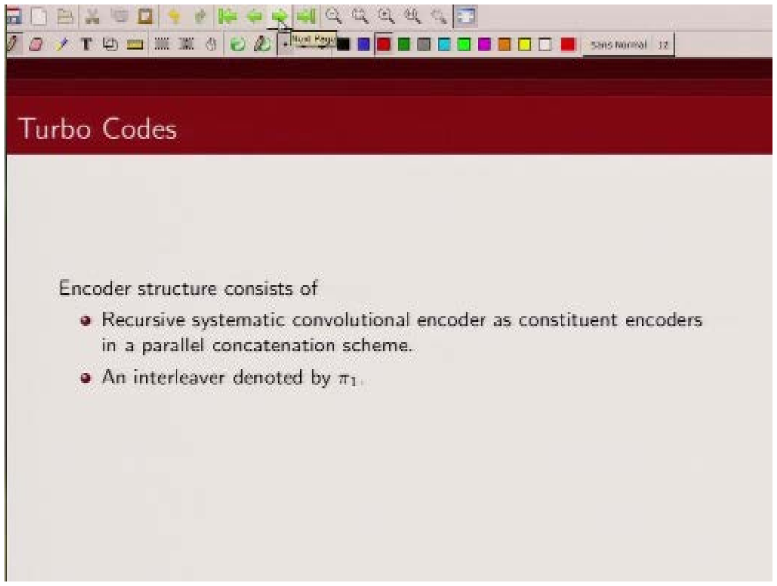
(Refer Slide Time: 06:14)



## Introduction

- Shannon's noisy channel coding theorem implies that arbitrarily low decoding error probabilities can be achieved at any transmission rate $R$ less than the channel capacity $C$ by using long block lengths.
- Shannon proved that the average performance of a randomly chosen ensemble of codes results in an exponentially decreasing decoding error probability with increasing block length.
- Typically, the code designs contain a large amount of structure, since it can be used to guarantee good minimum distance properties for the code as well as it is easier to decode a highly structured code.
- However, structure does not always result in the best distance properties for a code.
- Turbo codes have random-like code design with just enough structure to allow for an efficient iterative decoding method.

(Refer Slide Time: 06:14)



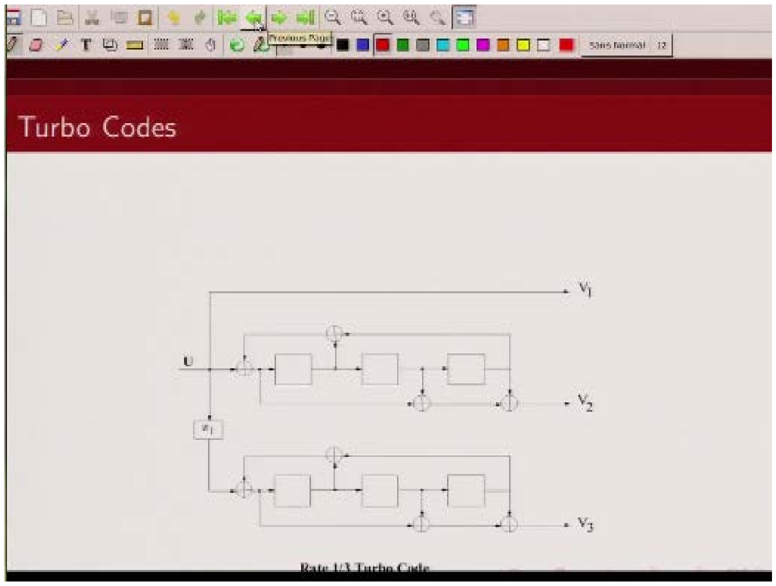
## Turbo Codes

Encoder structure consists of

- Recursive systematic convolutional encoder as constituent encoders in a parallel concatenation scheme.
- An interleaver denoted by $\pi_1$.

So what does an encoder of a turbo code looks like, it consist of parallel concatenation.

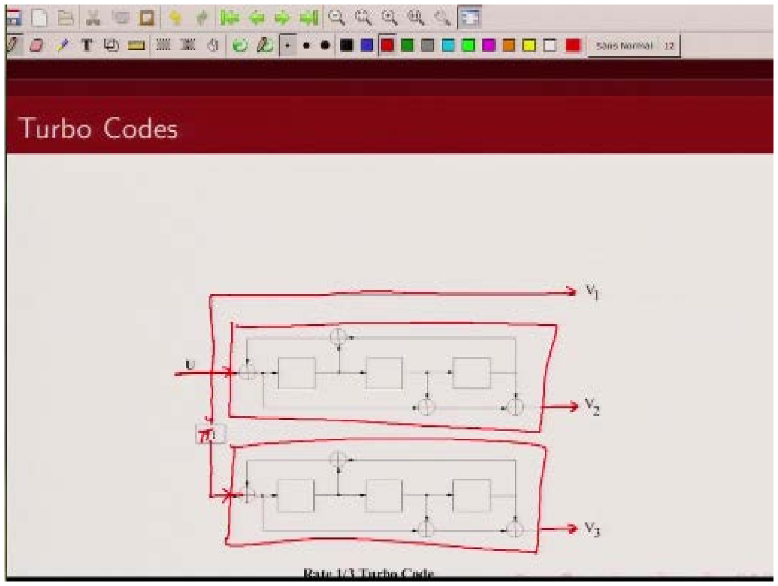
(Refer Slide Time: 06:26)


## Turbo Codes

Rate 1/3 Turbo Code

So maybe I will first show you the diagram. It consists of parallel concatenation of two encoders.

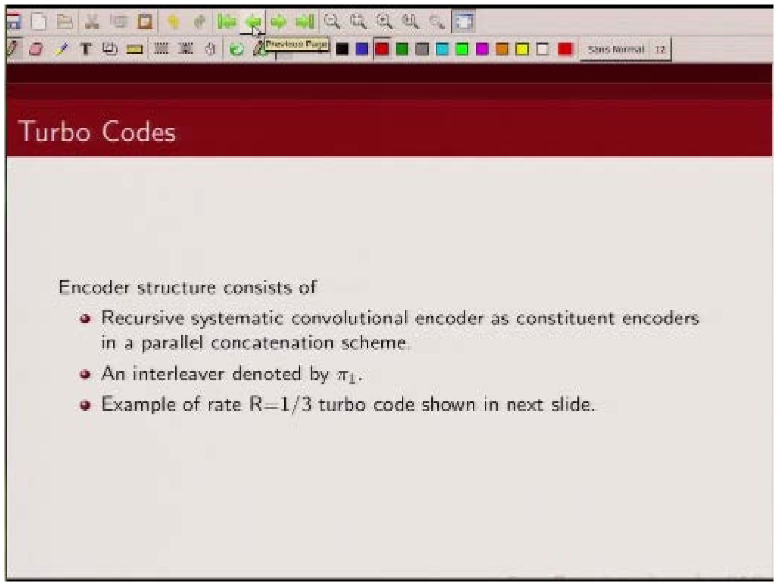
(Refer Slide Time: 06:31)


## Turbo Codes

Rate 1/3 Turbo Code

This is one encoder and this is second encoder. And note the same information bit is going to two encoders, so this information bit is coming here and the same information bit after getting interleaved, interleave is nothing but reordering of the message bits, it is entering this particular encoder and this is your systematic code, so you have your information bits here. And then this encoder is generating this parity bit, this encoder is generating this parity bit.

(Refer Slide Time: 07:10)



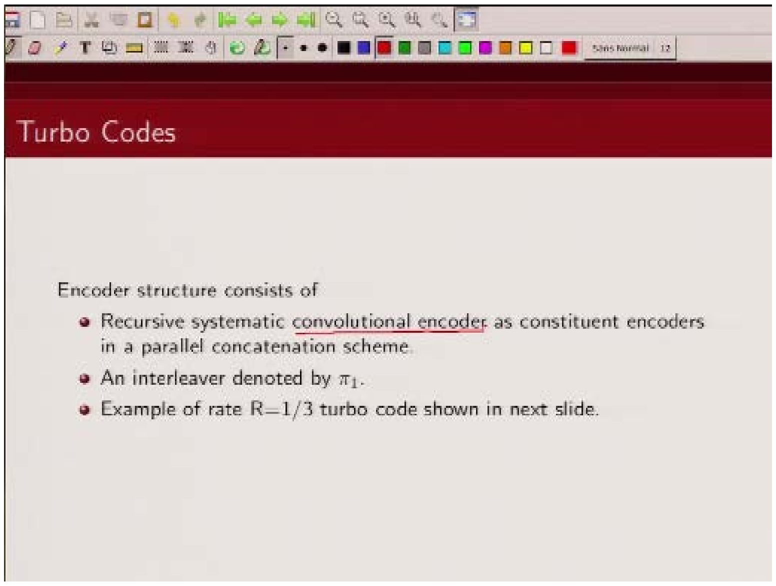
## Turbo Codes

Encoder structure consists of

- Recursive systematic convolutional encoder as constituent encoders in a parallel concatenation scheme.
- An interleaver denoted by $\pi_1$.
- Example of rate R=1/3 turbo code shown in next slide.

So if you notice this encoder consist of convolutional encoder

(Refer Slide Time: 07:17)



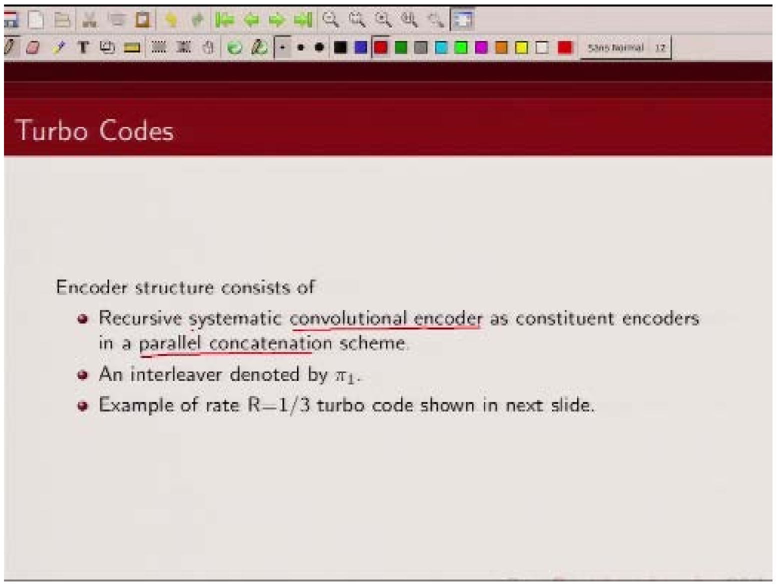
## Turbo Codes

Encoder structure consists of

- Recursive systematic convolutional encoder as constituent encoders in a parallel concatenation scheme.
- An interleaver denoted by $\pi_1$.
- Example of rate R=1/3 turbo code shown in next slide.

As a constituent encoder in a parallel concatenation scheme.

(Refer Slide Time: 07:23)



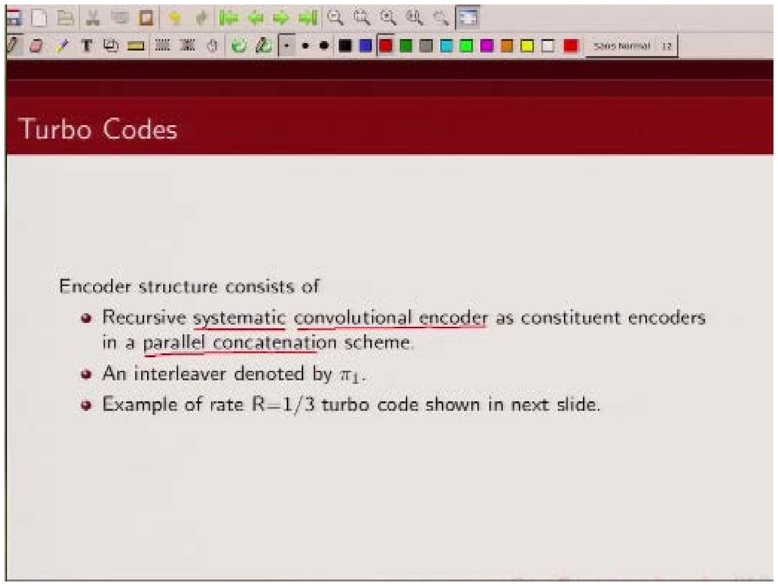
## Turbo Codes

Encoder structure consists of

- Recursive systematic convolutional encoder as constituent encoders in a parallel concatenation scheme.
- An interleaver denoted by $\pi_1$.
- Example of rate R=1/3 turbo code shown in next slide.

So the key is parallel concatenation. So the same input is going to both these convolutional encoder.

(Refer Slide Time: 07:32)



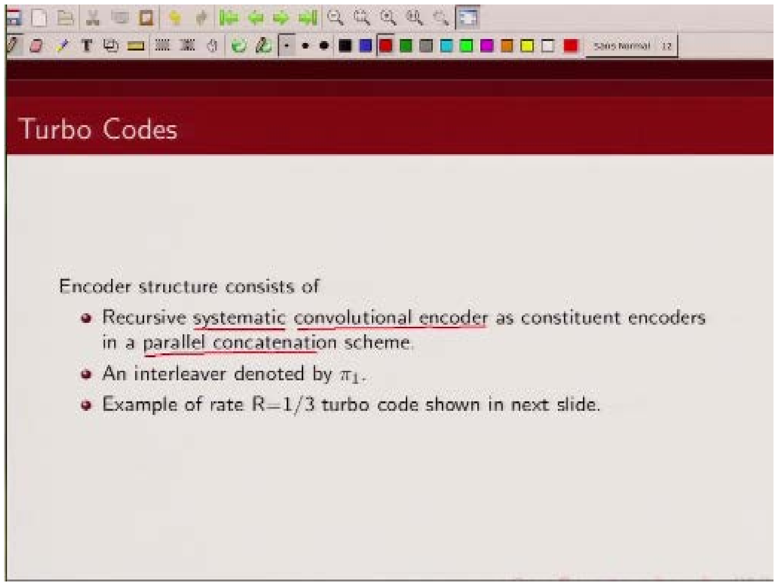
## Turbo Codes

Encoder structure consists of

- Recursive systematic convolutional encoder as constituent encoders in a parallel concatenation scheme.
- An interleaver denoted by $\pi_1$.
- Example of rate R=1/3 turbo code shown in next slide.

Next thing to remember is we are using systematic convolutional encoders.

(Refer Slide Time: 07:40)



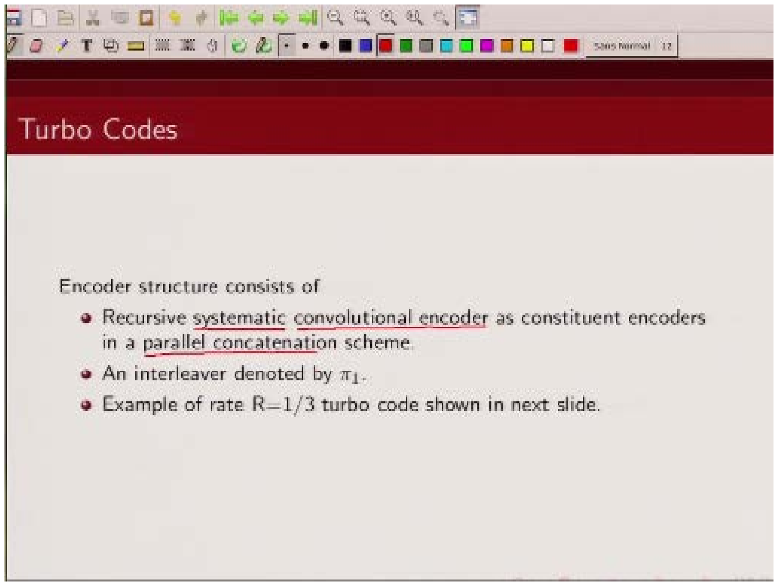
## Turbo Codes

Encoder structure consists of

- Recursive systematic convolutional encoder as constituent encoders in a parallel concatenation scheme.
- An interleaver denoted by $\pi_1$.
- Example of rate R=1/3 turbo code shown in next slide.

Why are we using systematic convolutional encoder, when we talk about the convergence properties of turbo code then we will mention basically these systematic encoders have better convergence property under iterative decoding algorithm.

(Refer Slide Time: 07:58)



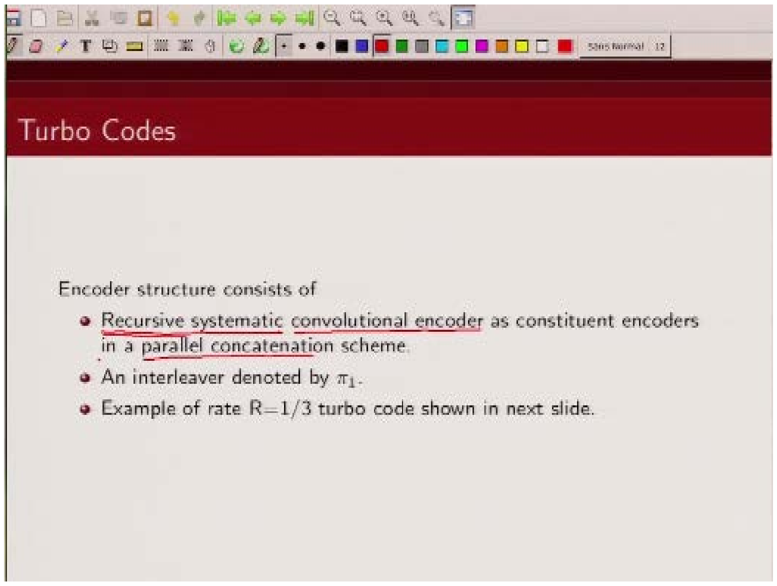
## Turbo Codes

Encoder structure consists of

- Recursive systematic convolutional encoder as constituent encoders in a parallel concatenation scheme.
- An interleaver denoted by $\pi_1$.
- Example of rate R=1/3 turbo code shown in next slide.

And that is why initially proposed turbo codes use systematic convolutional encoder, subsequently people have also worked on designing design of nonsystematic turbo codes, but the ones which were initially proposed in 1993 by Berrou [indiscernible][00:08:15] used systematic convolutional encoder.

(Refer Slide Time: 08:23)



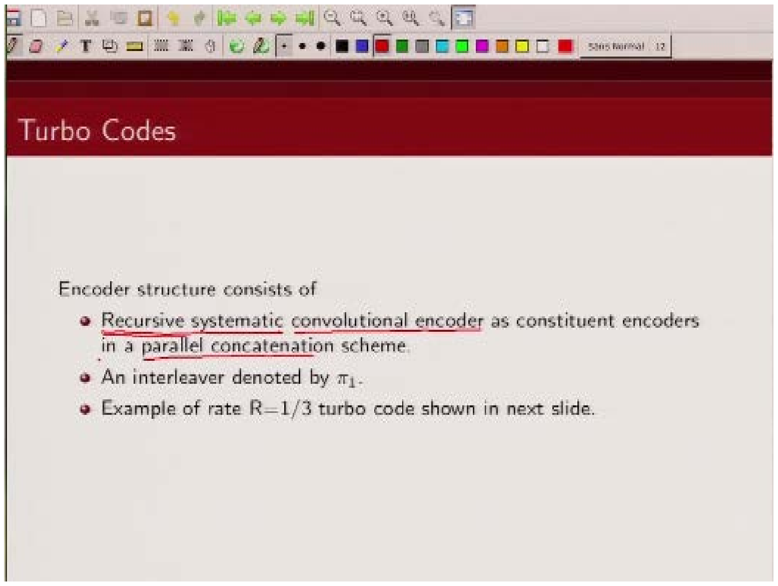
## Turbo Codes

Encoder structure consists of

- Recursive systematic convolutional encoder as constituent encoders in a parallel concatenation scheme.
- An interleaver denoted by $\pi_1$.
- Example of rate R=1/3 turbo code shown in next slide.

The third thing to note very crucial is the use of recursive encoders. What is recursive encoders, these are feedback encoders.

(Refer Slide Time: 08:34)

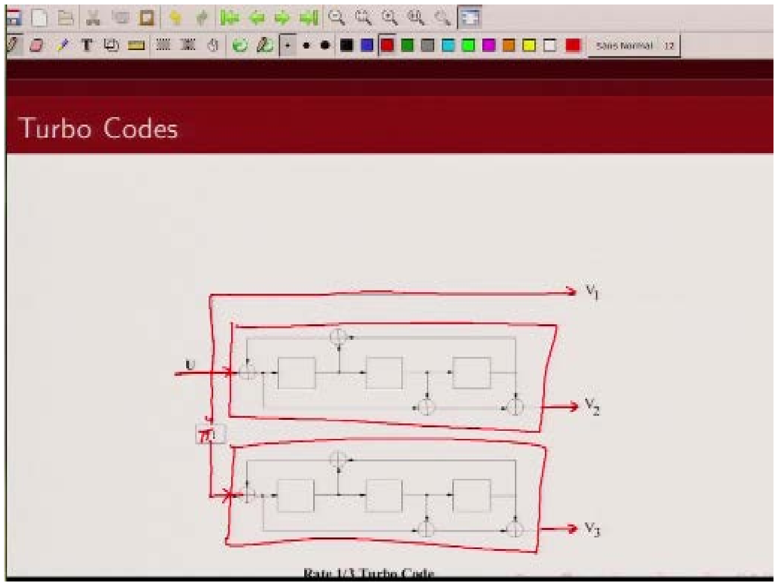


## Turbo Codes

Encoder structure consists of

- Recursive systematic convolutional encoder as constituent encoders in a parallel concatenation scheme.
- An interleaver denoted by $\pi_1$.
- Example of rate R=1/3 turbo code shown in next slide.

(Refer Slide Time: 08:35)



## Turbo Codes

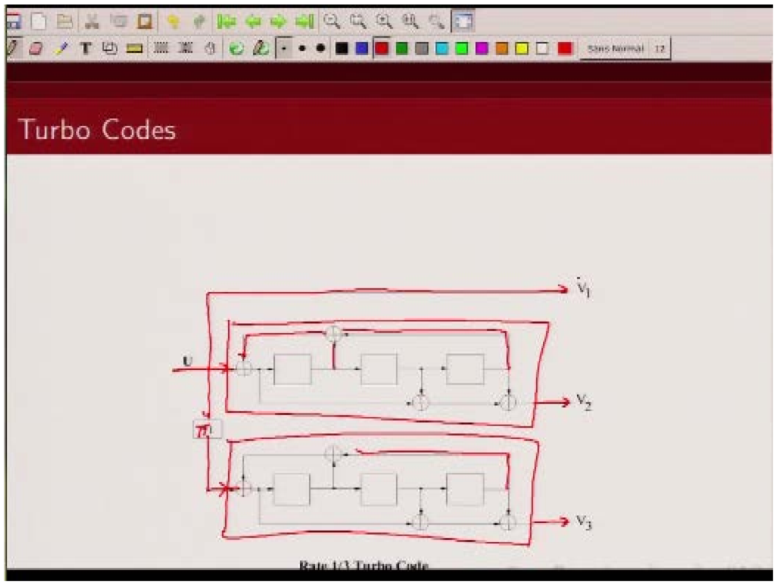
Rate 1/3 Turbo Code

So there is a feedback from the output to the input.

(Refer Slide Time: 08:37)

Note here there is a feedback from the output going to the input, there is a feedback from the output going to the input. So these are all feedback encoders, now why do we need feedback encoders? I will just give a very simple example to illustrate.

(Refer Slide Time: 08:58)



## Turbo Codes

Rate 1/3 Turbo Code

So we know that we would like to have a large minimum distance of these codes, because larger the minimum distance, better is the error correcting capability of the code. So let us just considered a very simple case.

(Refer Slide Time: 09:09)

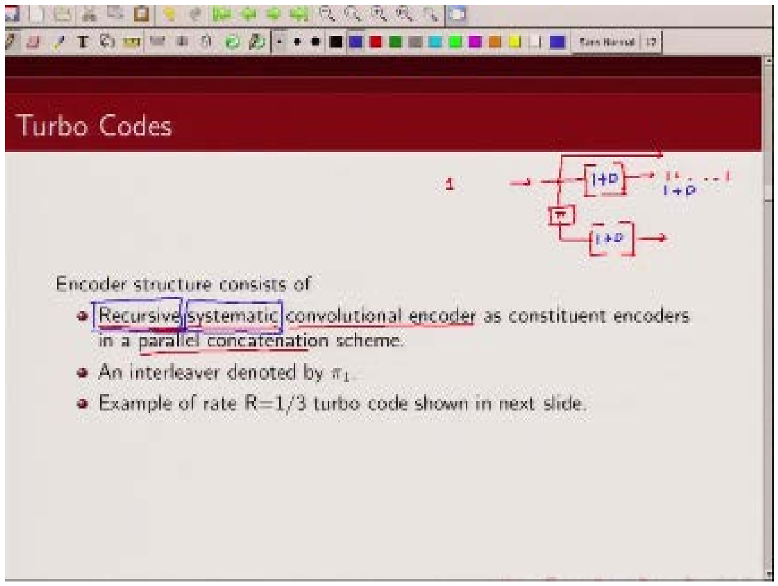
Let us just consider that you have a memory one code, so let us just say memory one code could be G(D) so this could be, let us say 1/1+D, 1/1+D. Now usually the input weights and this is systematic right, so this comes here, so usually the information sequence that have low weight can create low output weight sequence also.

So let us look at weight one sequence, let us say one and all zeros, so that is just one. Now if this input comes in here to this particular encoder, note this will produced an infinite length of one. And same this one when it comes here after interleaving this will also create a large number of ones whereas.

If instead of recursive encoder if we would have used a non recursive encoder, let us say if we had used a feedforward encoder 1, 1+D then what would have happened, this would, would have been just 1+D and of course this would have been also 1+D some or some shifted version of that, so you can see that

We can get better distance by using feedback convolutional encoder, so that is why this is very important that and this was a key innovation for this code that when they use two parallel concatenated codes the convolutional encoder which was used was recursive convolutional code.

(Refer Slide Time 11:08)



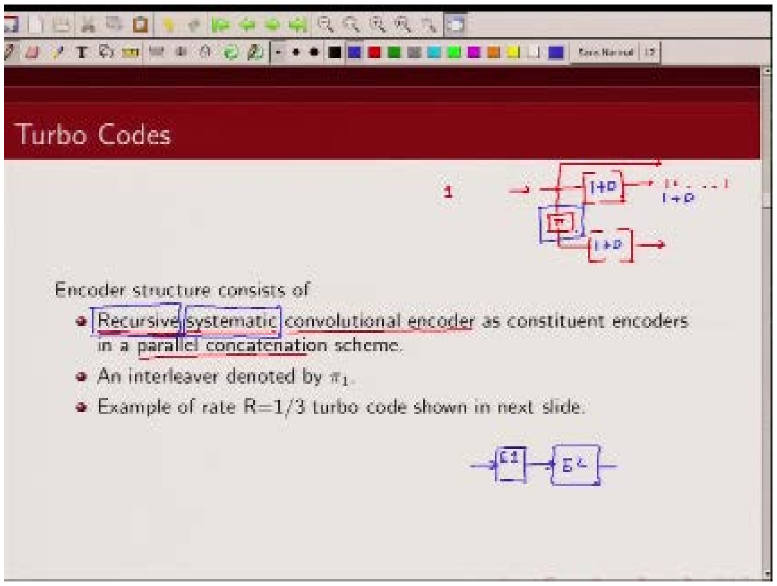
## Turbo Codes

Encoder structure consists of

- Recursive systematic convolutional encoder as constituent encoders in a parallel concatenation scheme.
- An interleaver denoted by $\pi_1$.
- Example of rate R=1/3 turbo code shown in next slide.

It was systematic because systematic codes had better convergence property so remember these two things, so what are the components of the turbo code. So first thing I said it is a concatenation of 2 or more encoders in a parallel fashion as opposed to serial concatenation. In serial concatenation

(Refer Slide Time 11:36)


## Turbo Codes

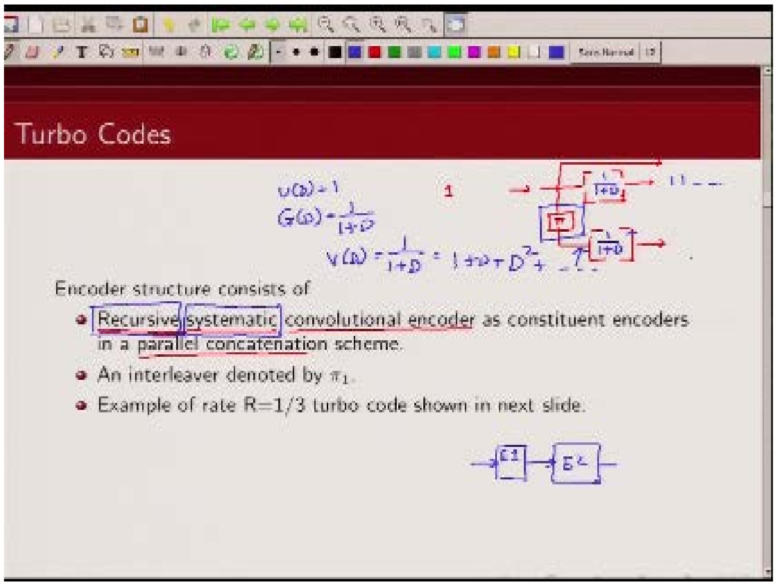
Encoder structure consists of

- Recursive systematic convolutional encoder as constituent encoders in a parallel concatenation scheme.
- An interleaver denoted by $\pi_1$.
- Example of rate R=1/3 turbo code shown in next slide.

What happens is so this there are bits coming in here, this output for first encoder that is fed as input to the second encoder so this is your encoder one, encoder two, so the output of encoder one is fed as input to the second encoder but in parallel concatenation you are sending the same information parallely to both the encoders. Now what is the role of interleaver and what is interleaver? As I said interleaver just permutes the bit, so it just reorders the bits, so some, some bits

Which were there in the say bit location one maybe it will be put in bit version 56, and it just shuffles bits here.

(Refer Slide Time 12:27)



## Turbo Codes

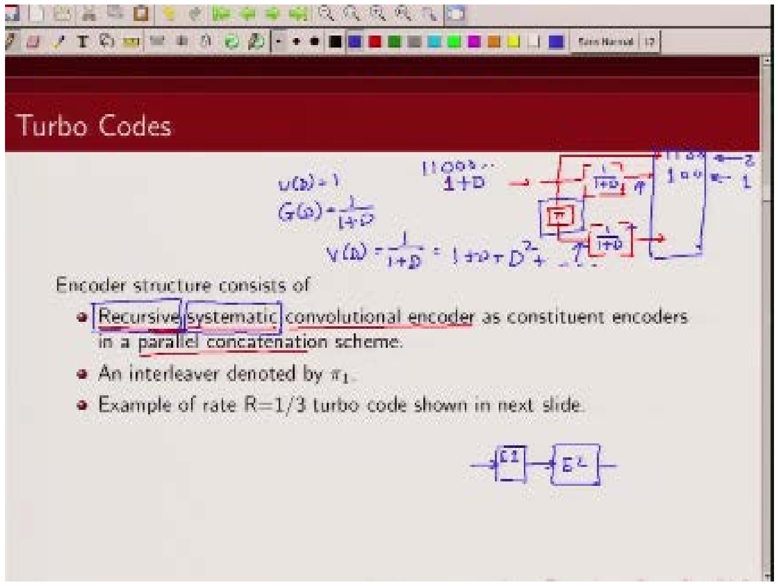
Encoder structure consists of

- Recursive systematic convolutional encoder as constituent encoders in a parallel concatenation scheme.
- An interleaver denoted by $\pi_1$.
- Example of rate R=1/3 turbo code shown in next slide.

Now the design of interleaver and the role of interleaver is very, very crucial for turbo codes, so let us illustrate that again with a simple example. So what we are considering is this recursive or a feedback encoders and we are conceding a very simple feedback encoders, which is memory once two state convolutional encoder. Now as I said, now input 1 sequence cannot terminate this encoder if my U of D is one and G of D is 1/1+D then what is my V(D) this U times V which is 1, 1+D.

And this is 1+D +D$^2$ so this I am getting a string of ones, a same thing when you permit it does not change the way distribution it just reorders the bit. So again the weight one sequence that you will get here cannot terminate this encoder, so I will get a large output weight, now let us look at weight 2 sequences let us say

(Refer Slide Time 13:53)



## Turbo Codes

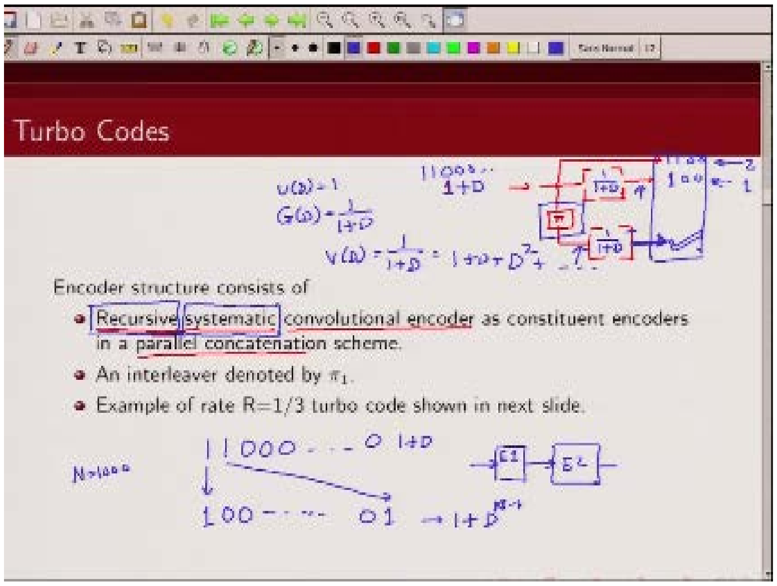
Encoder structure consists of

- Recursive systematic convolutional encoder as constituent encoders in a parallel concatenation scheme.
- An interleaver denoted by $\pi_1$.
- Example of rate R=1/3 turbo code shown in next slide.

If I have one, one and all zeros so let us say one, one and all zero that is 1+D so if my input is 1+D then what happens to the output at the first decoder? Now this sequence can terminate this encoder and what I will get here is I will just get a one. I will just get a one because my U(D) is 1+D and G(D) is 1/1+D so what I will get here is one so I am getting parity weight out of this parity check bit, I am just getting one and all zeros here. Now to have a code with overall weight large, remember my input is only to weight it is it my input is one, one and all zeros.

So here also I will get one, one and all zeros so here I am getting only weight 2, here I am getting only weight 1, now to have an overall weight large what would you like? You would like to have

(Refer Slide Time 15:01)



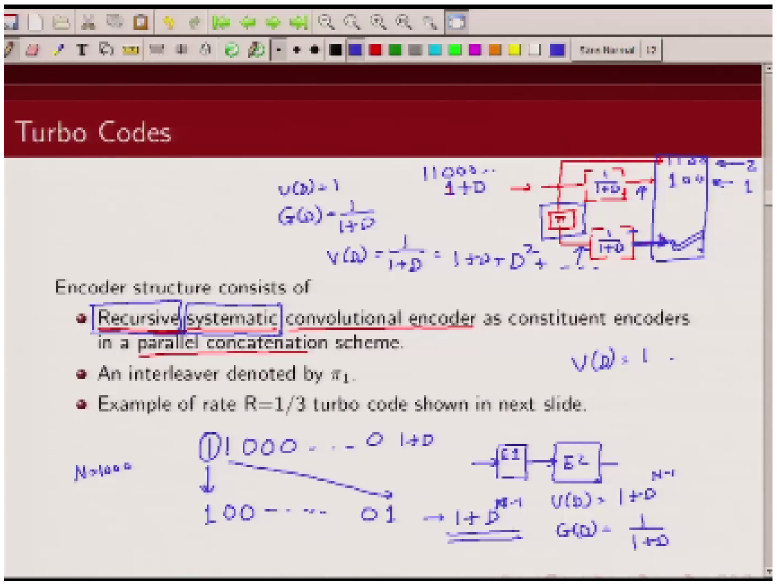
## Turbo Codes

Encoder structure consists of

- Recursive systematic convolutional encoder as constituent encoders in a parallel concatenation scheme.
- An interleaver denoted by $\pi_1$.
- Example of rate R=1/3 turbo code shown in next slide.

A large number of one’s coming out from this parity check bit. Now how can you do that, now remember what is interleaver, interleaver is just a per muter it is just shuffling bits so if you had bits which are like this let us one, one and all zeros and typically the block sizes are very large I mean 1000, 10,000 listen 10,000 or something like that and let us say your interleaver what it did, did was it kept this one here but moved this one to last location and in between you had all zeros, so this was your 1+D, and this becomes $1+D^{2,}$ let us say N-1 because the way you design this interleaver

(Refer Slide Time 16:05)



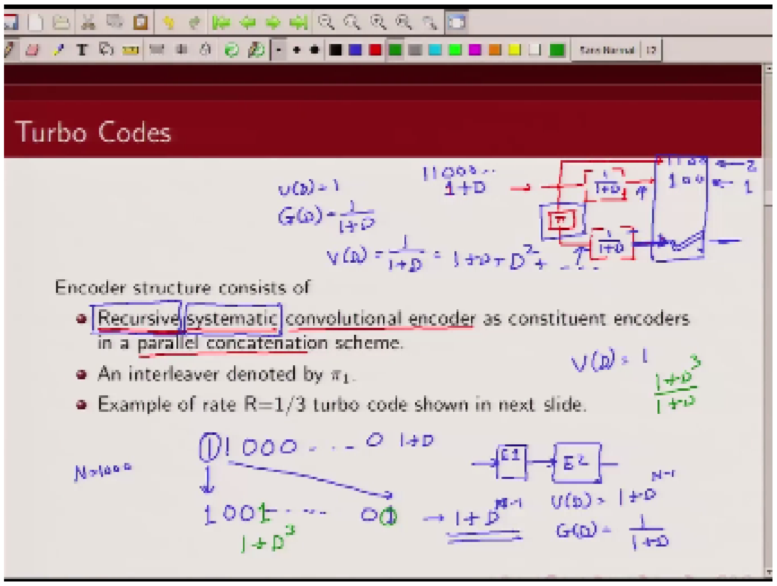
## Turbo Codes

Encoder structure consists of

- Recursive systematic convolutional encoder as constituent encoders in a parallel concatenation scheme.
- An interleaver denoted by $\pi_1$.
- Example of rate R=1/3 turbo code shown in next slide.

It rearranged the bits is such a way that let say this bit was put here, this bit was put here and other zeros were put in between. Now if you feed in this sequence to this encoder so your U(D) is 1+D$^2$ N – 1, G (D) is 1+D what will you get, you will get a output sequence which will have large number of ones, correct? So what I did was I just designed my interleaver in such a way that 2 adjacent ones were spread out and as a result you noticed that

(Refer Slide Time 16:51)


## Turbo Codes

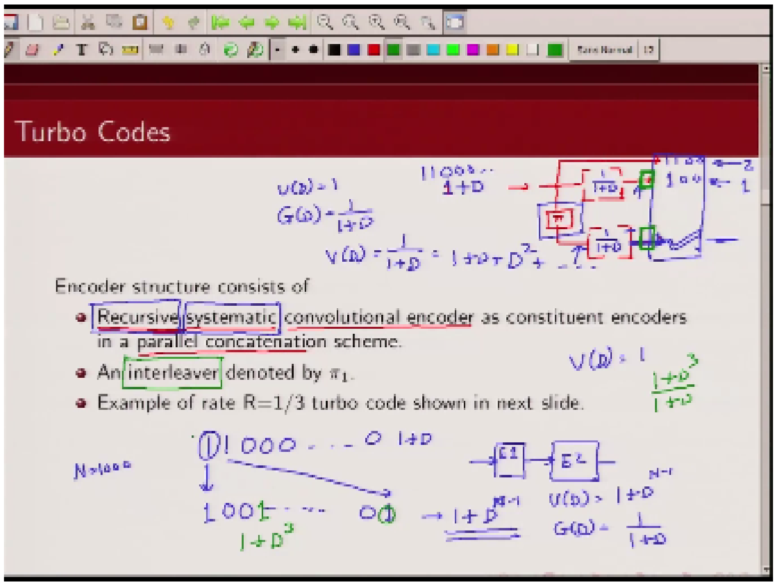
Encoder structure consists of

- Recursive systematic convolutional encoder as constituent encoders in a parallel concatenation scheme.
- An interleaver denoted by $\pi_1$.
- Example of rate R=1/3 turbo code shown in next slide.

I am getting a large number of one’s coming out from this parity bit, now contrast it with a situation where let say instead of shuffling this bit here if I would have kept this bit let us say here and this would have been zero so this would have been $1+D^3$ then what would I have get? I would have got $1+D^3$ by 1+D so this would be $1+D+D^2$ so I would have only got 3ones here, but now I am getting a large number of one’s so I hope I am able to convey the role of interleaver, the design of interleaver is very, very crucial.

You should design the interleaver in such a way such that the adjacent ones are separated out far apart so that after the interleaving when the same information sequences passing through the encoder it generates a large parity, a parity sequence with large weight.

(Refer Slide Time 18:01)

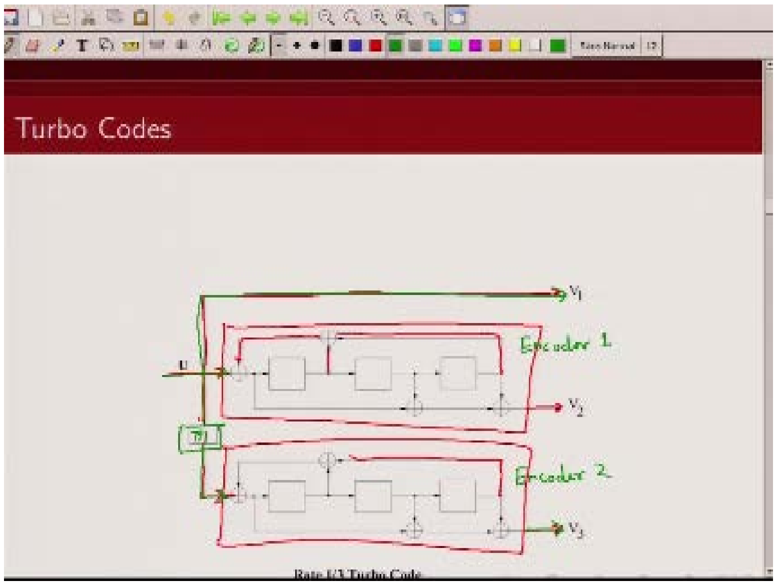

## Turbo Codes

Encoder structure consists of

- Recursive systematic convolutional encoder as constituent encoders in a parallel concatenation scheme.
- An interleaver denoted by $\pi_1$.
- Example of rate R=1/3 turbo code shown in next slide.

Ideally you would what you would like is if the parities sequence coming out here has low weight then this should generate large weight parity sequence, and if this guy is generating parity sequence with low weight then this sequence this should generate a parity sequence with large weight, so again I cannot emphasize enough the role of interleaver because this a very, very crucial component of turbo encoder and it also helps in some sense randomizing the goat leg, make the code look random like if you, if you want to use that word so let us look at

(Refer Slide Time 18:47)



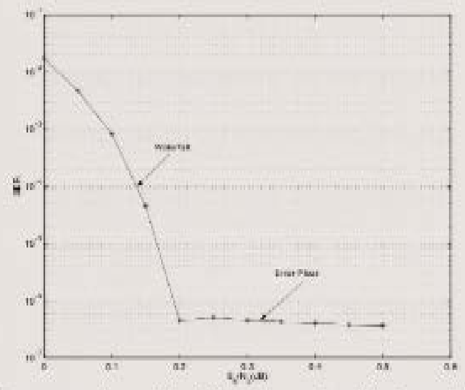
## Turbo Codes

The structure of a turbo code again so you note here I have 2 encoders, this is my encoder one, and this my encoder two, both of them are recursive or feedback encoders. You can see that each one of them has eight states in this particular example, now the information sequences coming here it is a systematic code so I, the same information is coming out directly as one of the output and then the input to the second encoder this is interleaved, now interleaving is nothing but as I said reordering of the bits.

The reordered bits are coming to the second encoder which will then generate a set of parity bits, so this is the structure of a parallel concatenated code. In this particular example we are using convolutional encoders as constituent encoders.

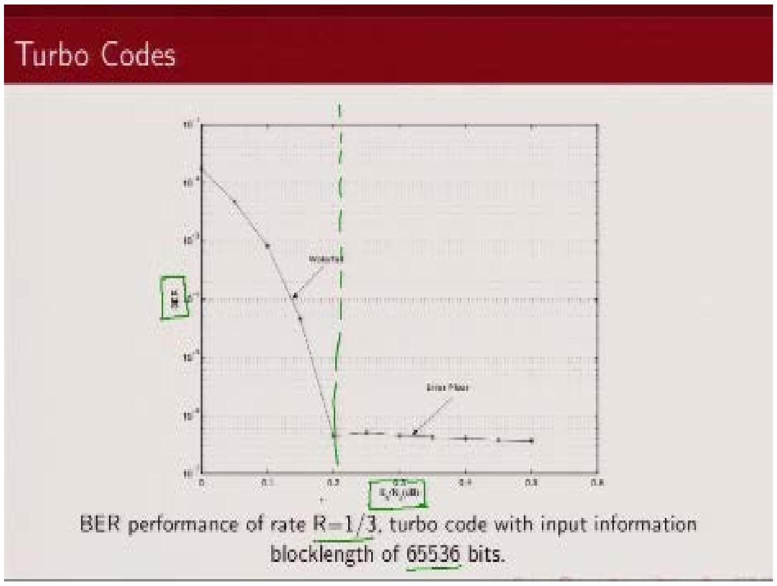
(Refer Slide Time: 19:59)

## Turbo Codes



BER performance of rate R=1/3, turbo code with input information blocklength of 65536 bits.

(Refer Slide Time: 20:05)

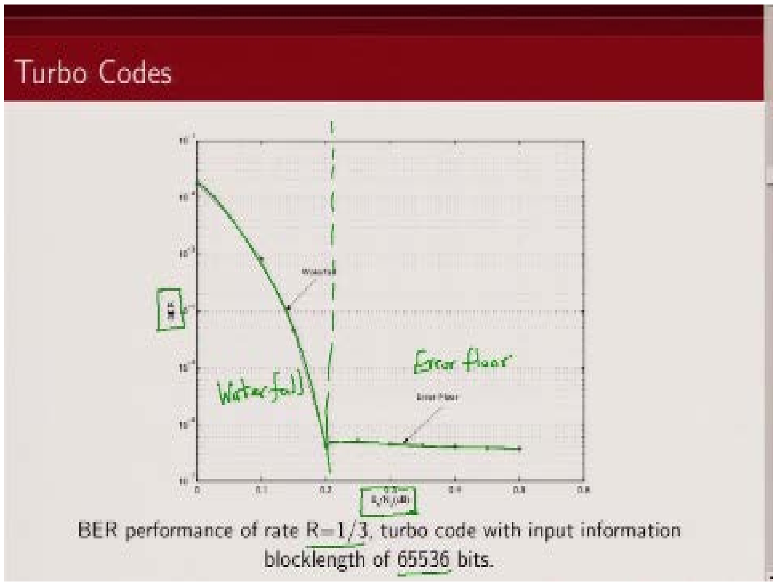


## Turbo Codes

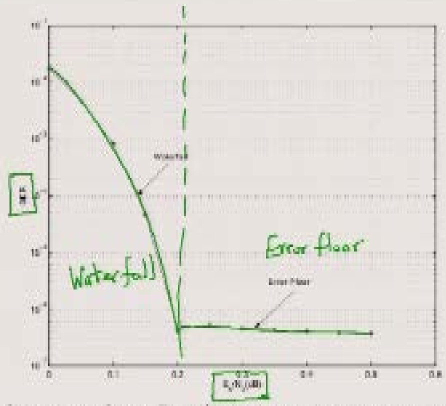
BER performance of rate R=1/3, turbo code with input information blocklength of 65536 bits.

Now in this graph I have plotted bit error rate performance versus signal to noise ratio for a rate 1/3 turbo code and for block size more than sixty five thousand. Now note the typical performance so there is a region, where there is a steep fall in bit error rate performance.

(Refer Slide Time: 20:33)



## Turbo Codes

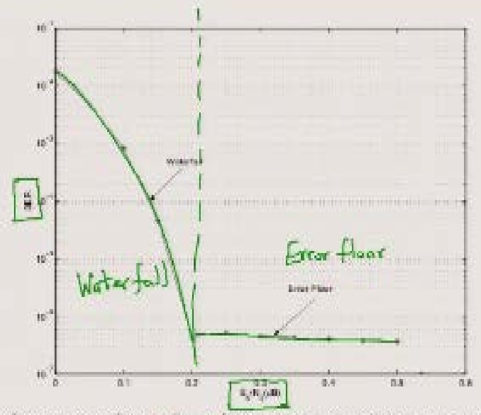


BER performance of rate R=1/3, turbo code with input information blocklength of 65536 bits.

And this region we call it waterfall region, so this is called waterfall region, just like a typical waterfall that performance is coming like this, and then there is a region where there is hardly any improvement in bit error rate in spite of increasing the signal to noise ratio and this is known as error floor region. Now performance in the waterfall region is governed by the convergence behavior of the constituent encoders under iterative decoding algorithm and we will discuss this in subsequent lectures.

(Refer Slide Time: 21:21)

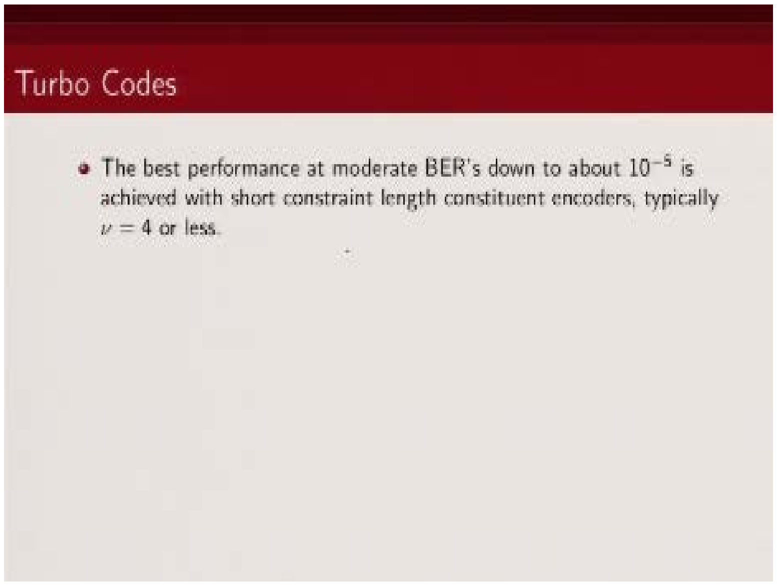
## Turbo Codes



BER performance of rate R=1/3, turbo code with input information blocklength of 65536 bits.

And performance in the error floor region is governed by the distance spectrum of the turbo codes.

(Refer Slide Time: 21:33)

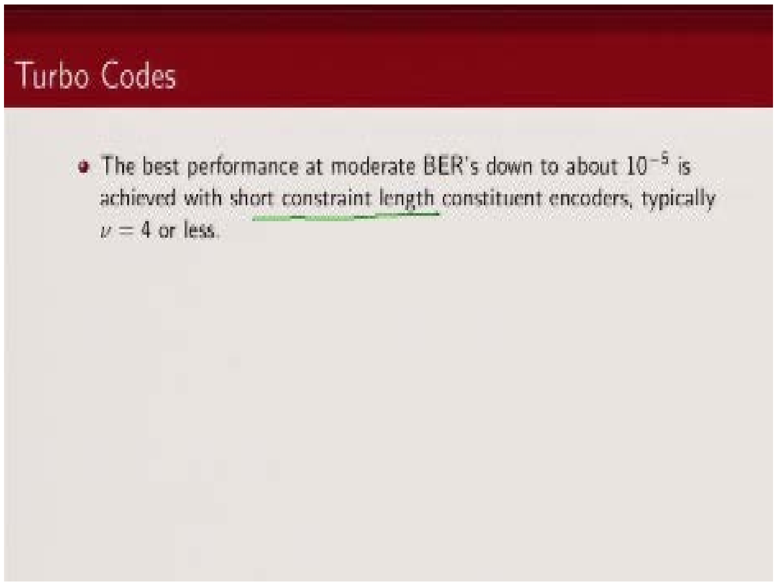


## Turbo Codes

- The best performance at moderate BER's down to about $10^{-5}$ is achieved with short constraint length constituent encoders, typically $\nu = 4$ or less.

So typically the constituent encoders, convolutional constituent encoders that gives good performance are the ones

(Refer Slide Time: 21:45)

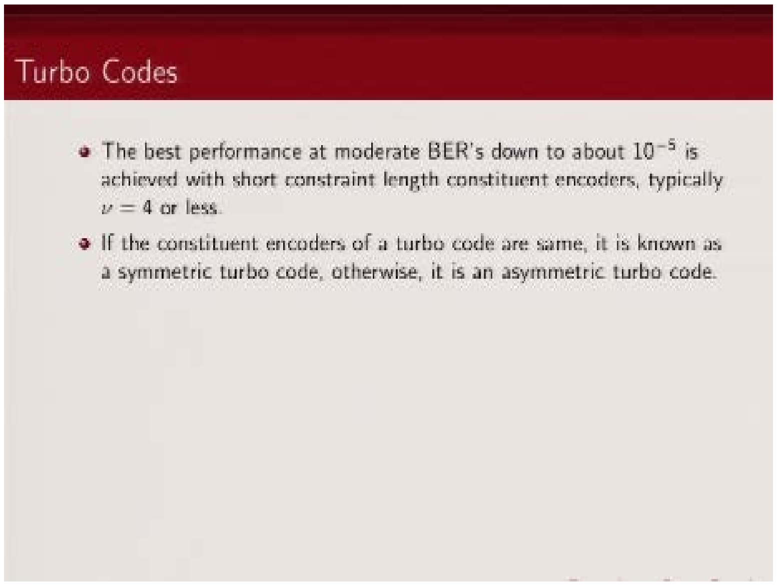


## Turbo Codes

- The best performance at moderate BER's down to about $10^{-5}$ is achieved with short constraint length constituent encoders, typically $\nu = 4$ or less.

Which have short constraint length typically three, four like memory three, memory four, either the ones that give good performance, if we use a more stronger code with memory five, six then typically the performance in a waterfall region is not good.

(Refer Slide Time: 22:04)

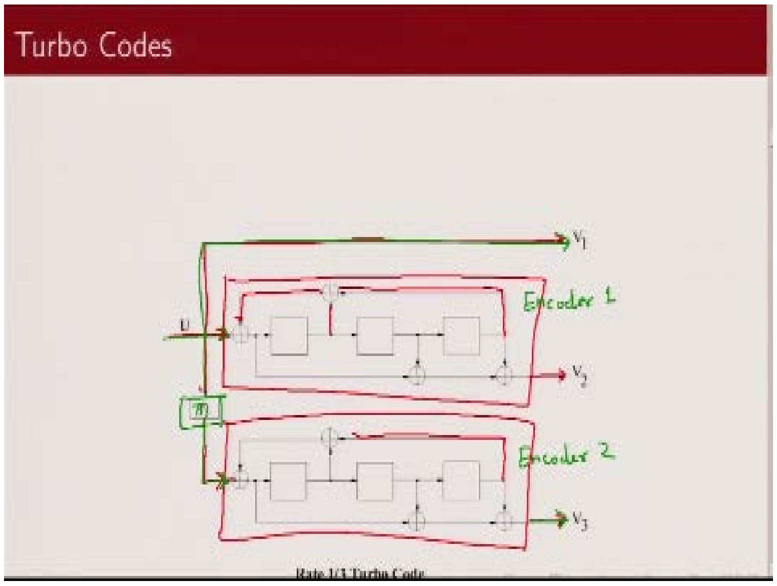


## Turbo Codes

- The best performance at moderate BER's down to about $10^{-5}$ is achieved with short constraint length constituent encoders, typically $\nu = 4$ or less.
- If the constituent encoders of a turbo code are same, it is known as a symmetric turbo code, otherwise, it is an asymmetric turbo code.

Now what I have shown you is a parallel concatenation of two encoders, now these two encoders can be the same encoder like what I have shown here.

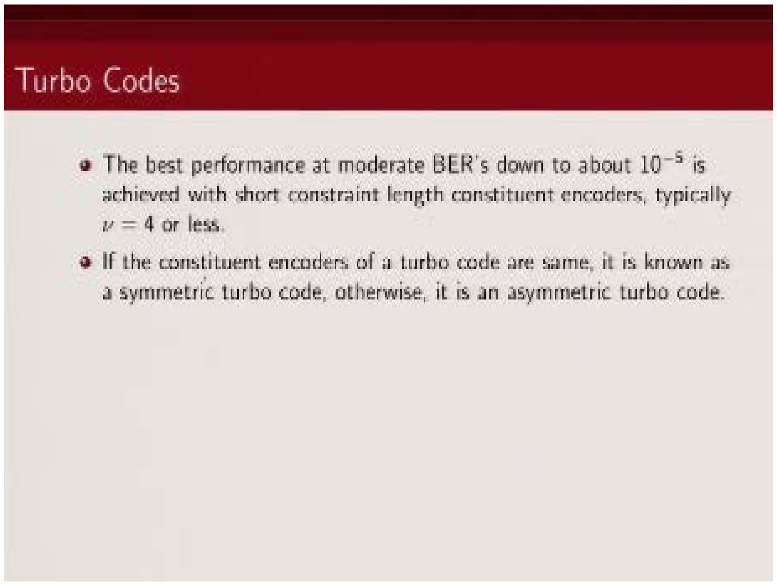
(Refer Slide Time: 22:16)


## Turbo Codes

Rate 1/3 Turbo Code

If you see here both of encoder 1 and encoder 2 are same, now they could be same or they could be different.

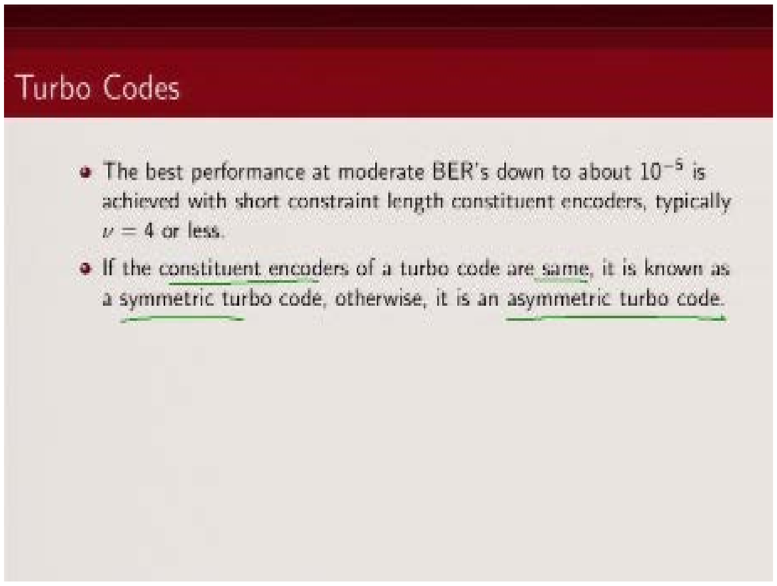
(Refer Slide Time: 22:26)



## Turbo Codes

- The best performance at moderate BER's down to about $10^{-5}$ is achieved with short constraint length constituent encoders, typically $\nu = 4$ or less.
- If the constituent encoders of a turbo code are same, it is known as a symmetric turbo code, otherwise, it is an asymmetric turbo code.

(Refer Slide Time: 22:29)

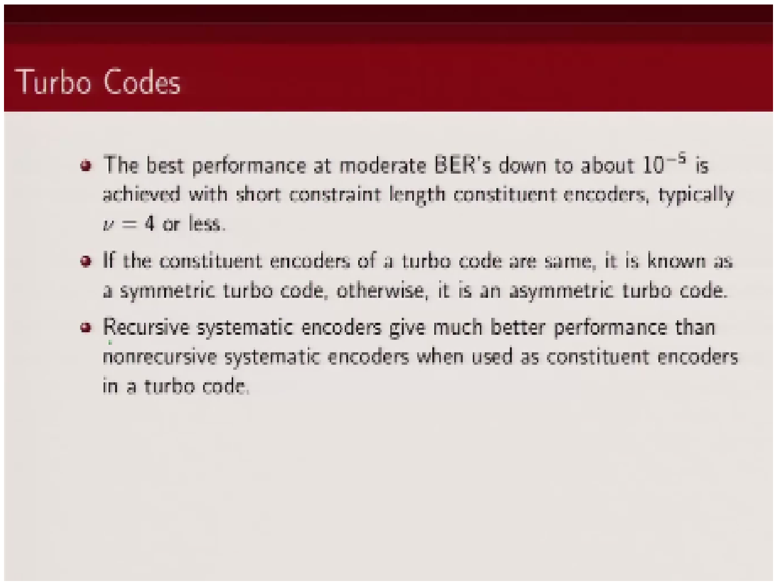


## Turbo Codes

- The best performance at moderate BER's down to about $10^{-5}$ is achieved with short constraint length constituent encoders, typically $\nu = 4$ or less.
- If the constituent encoders of a turbo code are same, it is known as a symmetric turbo code, otherwise, it is an asymmetric turbo code.

So if the constituent encoders are same we call it a systematic symmetric turbo code and if these constituent encoders are different we call it asymmetric turbo code.

(Refer Slide Time: 22:44)

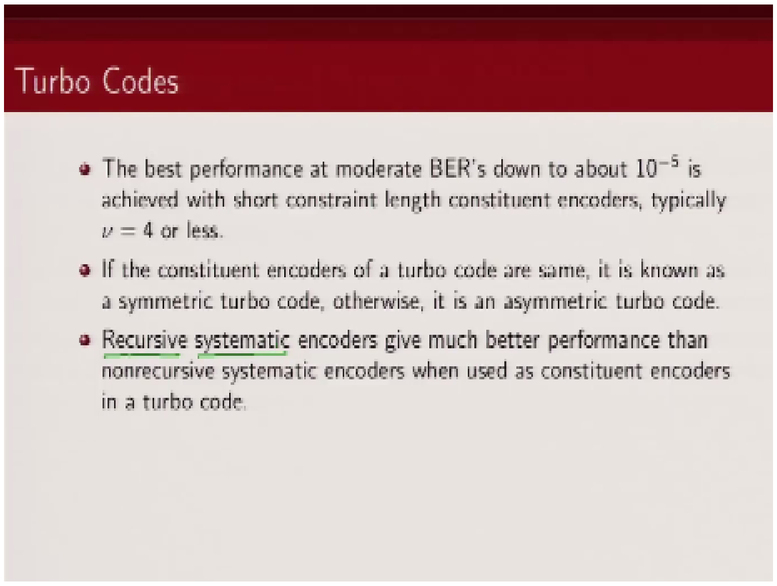


## Turbo Codes

- The best performance at moderate BER's down to about $10^{-5}$ is achieved with short constraint length constituent encoders, typically $\nu = 4$ or less.
- If the constituent encoders of a turbo code are same, it is known as a symmetric turbo code, otherwise, it is an asymmetric turbo code.
- Recursive systematic encoders give much better performance than nonrecursive systematic encoders when used as constituent encoders in a turbo code.

I already mentioned this point.

(Refer Slide Time: 22:48)

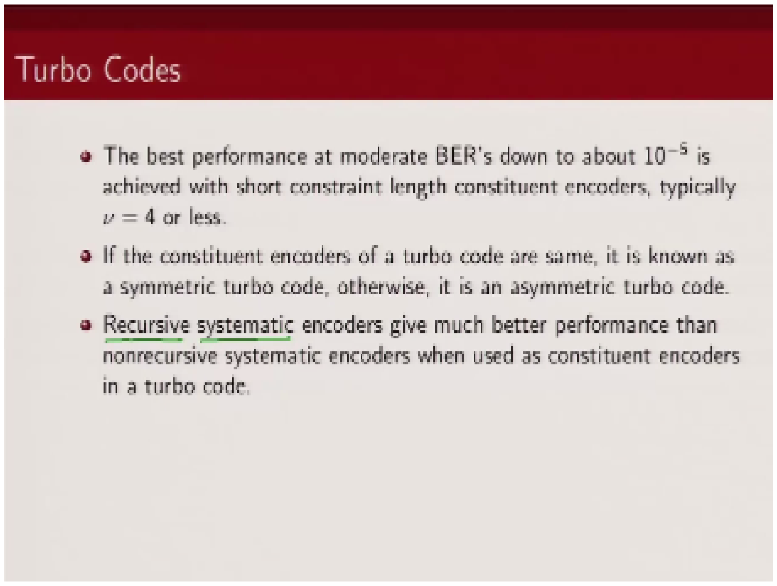


## Turbo Codes

- The best performance at moderate BER's down to about $10^{-5}$ is achieved with short constraint length constituent encoders, typically $\nu = 4$ or less.
- If the constituent encoders of a turbo code are same, it is known as a symmetric turbo code, otherwise, it is an asymmetric turbo code.
- Recursive systematic encoders give much better performance than nonrecursive systematic encoders when used as constituent encoders in a turbo code.

That choice of recursive and systematic encoders is very, very crucial because for a recursive encoder a weight one pattern cannot terminate the encoder whereas for a feedforward encoder a weight one pattern can terminate the encoder.

(Refer Slide Time: 23:10)

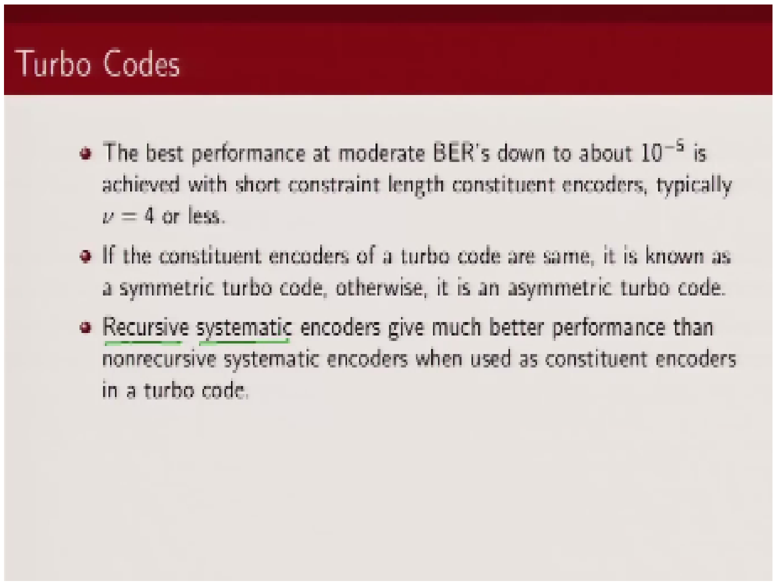


## Turbo Codes

- The best performance at moderate BER's down to about $10^{-5}$ is achieved with short constraint length constituent encoders, typically $\nu = 4$ or less.
- If the constituent encoders of a turbo code are same, it is known as a symmetric turbo code, otherwise, it is an asymmetric turbo code.
- Recursive systematic encoders give much better performance than nonrecursive systematic encoders when used as constituent encoders in a turbo code.

That is why we should use a recursive encoder and systematic encoders help in initial performance in waterfall region and that is why they are preferred over non systematic encoders.

(Refer Slide Time: 23:24)



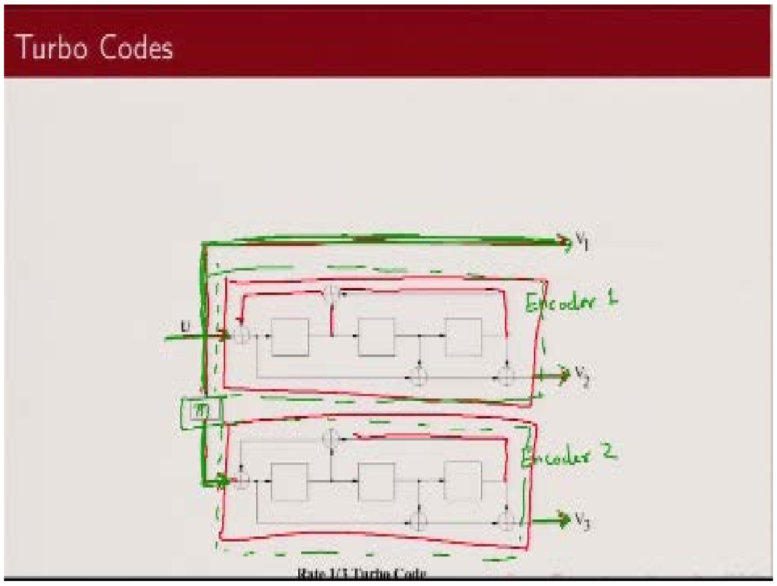
## Turbo Codes

- The best performance at moderate BER's down to about $10^{-5}$ is achieved with short constraint length constituent encoders, typically $\nu = 4$ or less.
- If the constituent encoders of a turbo code are same, it is known as a symmetric turbo code, otherwise, it is an asymmetric turbo code.
- Recursive systematic encoders give much better performance than nonrecursive systematic encoders when used as constituent encoders in a turbo code.

Especially if you are interested in performance in the waterfall region.

(Refer Slide Time: 23:34)

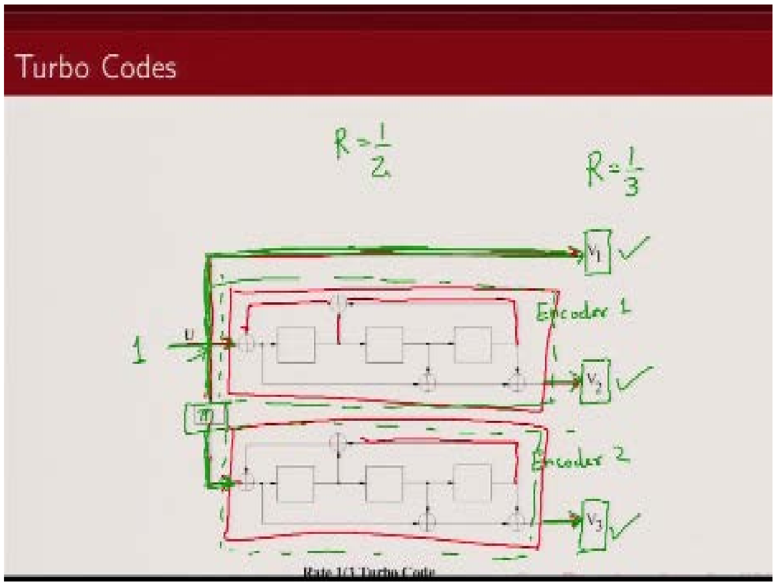
## Turbo Codes



Rate 1/3 Turbo Code

So what we are doing here is so this is information bits right? This is the information bit that is coming out of an output and then you have this encoder, which is generating this parity bit, right? Now this interleaved bit is, this interleaved information sequence is coming here and what this encoder 2 generates, this encoder 2 generates this parity bit, please note that we are not sending the information bit corresponding to second encoder, why?

(Refer Slide Time: 24:24)

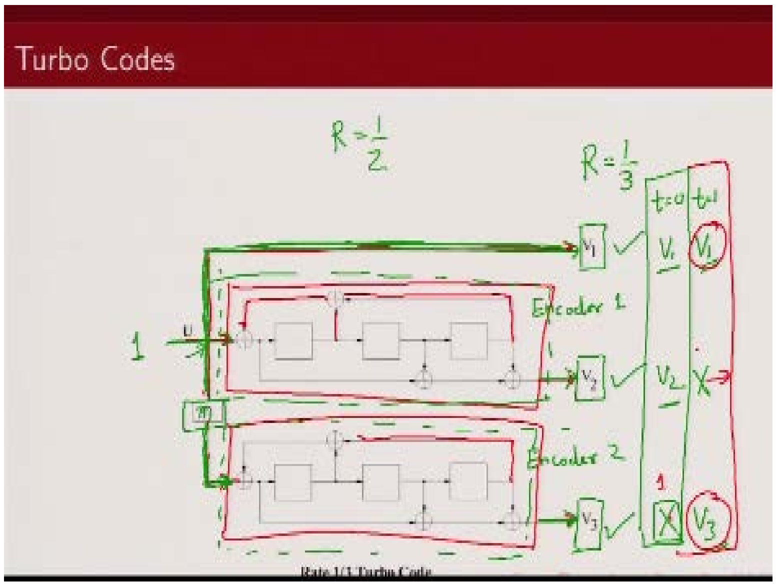


## Turbo Codes

Because the information bits correspond to second encoder is nothing but interleaved version of the information sequence that has been fed to encoder 1 and that is why we are not sending, we are not sending the information bits here, we are not sending the information bits of the second encoder we are not sending it across to the channel. That is because from this information bit if we just interleave we can get back the information bits for the second encoder.

So you can see this is a rate 1/3 code because there is one input and there are three output, this is one output, this is one output, and this is one output, right? Now how do you get different rates code so this is a rate 1/3 code? Now how do you get other rates let us say I want to design a rate 1 ½ turbo code, how do I do that? So I will do what I can do let us say what is called puncturing, so what you do in puncturing? In puncturing you remove some of the bits do not send some of the bits.

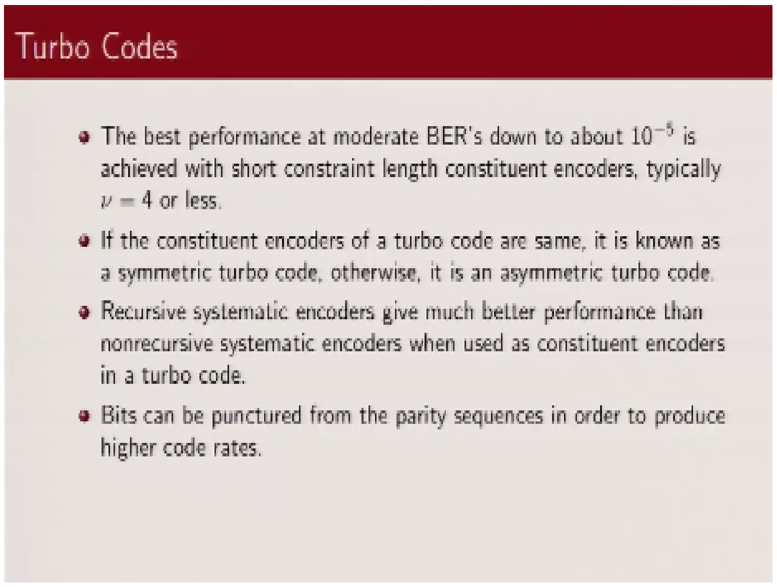
(Refer Slide Time: 25:39)



So let us say at time t=0, I transmit $v_1$ and I transmit $v_2$ so I am transmitting two bits I do not transmit $v_3$. Then at time t=1, I transmit $v_1$ and $v_3$, I do not transmit $v_2$ and this I repeat so every odd time I transmit $v_1$ and $v_2$ and every even at t=0, 2, 4 I transmit $v_1$ and $v_2$ and for other time I transmit $v_1$ and $v_3$, and at the receiver what I will do is so at let us say I have received these bit now I am since I am not transmitting all the three bits, so what I will do is for the bit which was not transmitted I will assume that this bit is equally likely to be zero or one.

So I will put is as likely would ratio of this bit being one or zero to be same so that is what I will do, and similarly here since I am getting receive bit corresponding to only this and this what I will assume here is I will feed to a decoder that it is equally likely that this particular bit is zero or one because I do not have any other information about this bit, so this is known as puncturing, so I am removing some of the bits I am not sending some of the bits so if you are interested in getting higher rates you can do puncturing to get higher rate codes.

(Refer Slide Time: 27:22)

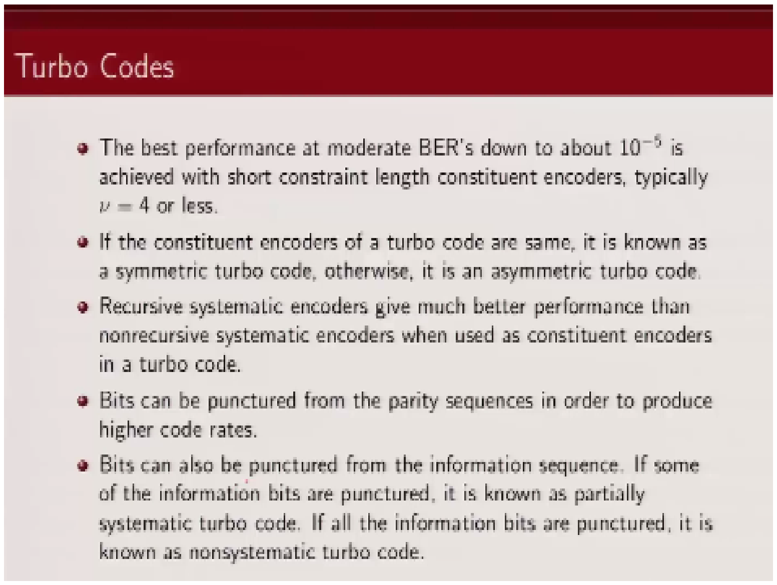


## Turbo Codes

- The best performance at moderate BER's down to about $10^{-5}$ is achieved with short constraint length constituent encoders, typically $\nu = 4$ or less.
- If the constituent encoders of a turbo code are same, it is known as a symmetric turbo code, otherwise, it is an asymmetric turbo code.
- Recursive systematic encoders give much better performance than nonrecursive systematic encoders when used as constituent encoders in a turbo code.
- Bits can be punctured from the parity sequences in order to produce higher code rates.

And that is what I mean when I said we could, bits can be punctured from the parity sequence to produce higher code rates.

(Refer Slide Time: 27:32)

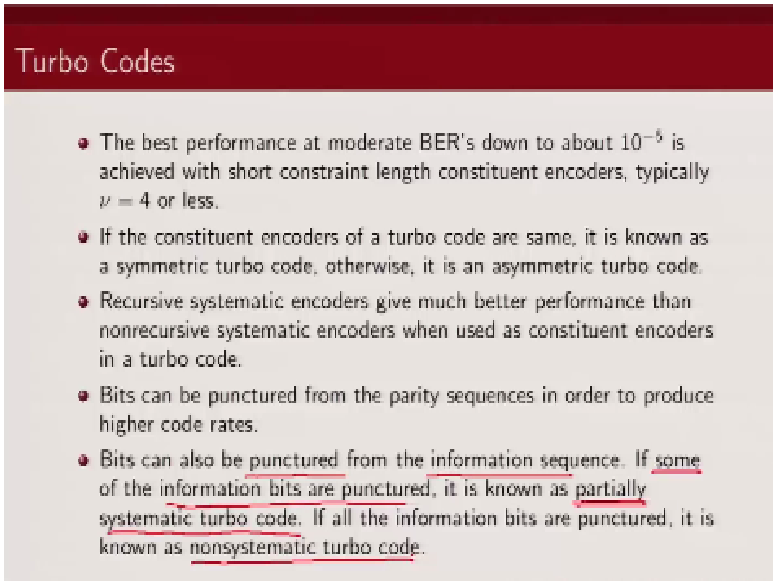


## Turbo Codes

- The best performance at moderate BER's down to about $10^{-5}$ is achieved with short constraint length constituent encoders, typically $\nu = 4$ or less.
- If the constituent encoders of a turbo code are same, it is known as a symmetric turbo code, otherwise, it is an asymmetric turbo code.
- Recursive systematic encoders give much better performance than nonrecursive systematic encoders when used as constituent encoders in a turbo code.
- Bits can be punctured from the parity sequences in order to produce higher code rates.
- Bits can also be punctured from the information sequence. If some of the information bits are punctured, it is known as partially systematic turbo code. If all the information bits are punctured, it is known as nonsystematic turbo code.

We could also do puncturing of the information bits.

(Refer Slide Time: 27:36)

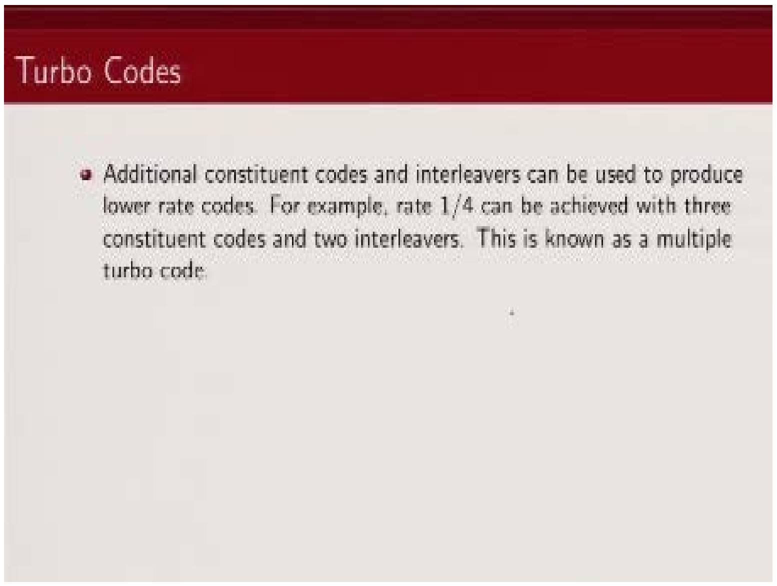


## Turbo Codes

- The best performance at moderate BER's down to about $10^{-5}$ is achieved with short constraint length constituent encoders, typically $\nu = 4$ or less.
- If the constituent encoders of a turbo code are same, it is known as a symmetric turbo code, otherwise, it is an asymmetric turbo code.
- Recursive systematic encoders give much better performance than nonrecursive systematic encoders when used as constituent encoders in a turbo code.
- Bits can be punctured from the parity sequences in order to produce higher code rates.
- Bits can also be punctured from the information sequence. If some of the information bits are punctured, it is known as partially systematic turbo code. If all the information bits are punctured, it is known as nonsystematic turbo code.

So I just showed you an example of puncturing of the parity bits, I could also decide not to send the information bits. Now if some of the information bits are not sent that means they are punctured, this particular code is known as partially systematic turbo code and if I do not send any information bit then of course you already know this is a non systematic code.

(Refer Slide Time: 28:07)

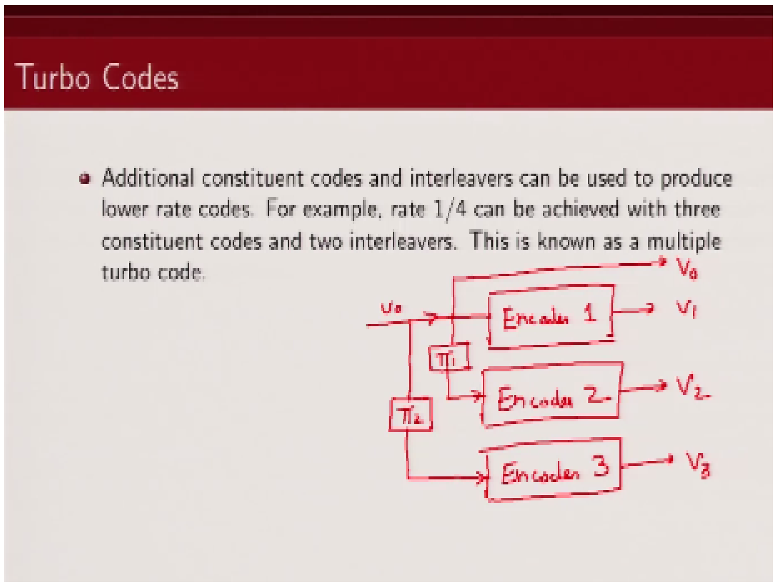


## Turbo Codes

- Additional constituent codes and interleavers can be used to produce lower rate codes. For example, rate 1/4 can be achieved with three constituent codes and two interleavers. This is known as a multiple turbo code.

Now the encoder diagram that I showed you was using two encoders and one interleaver, now this structure can be extended for multiple encoders and multiple interleavers, so I could have a situation

(Refer Slide Time: 28:27)

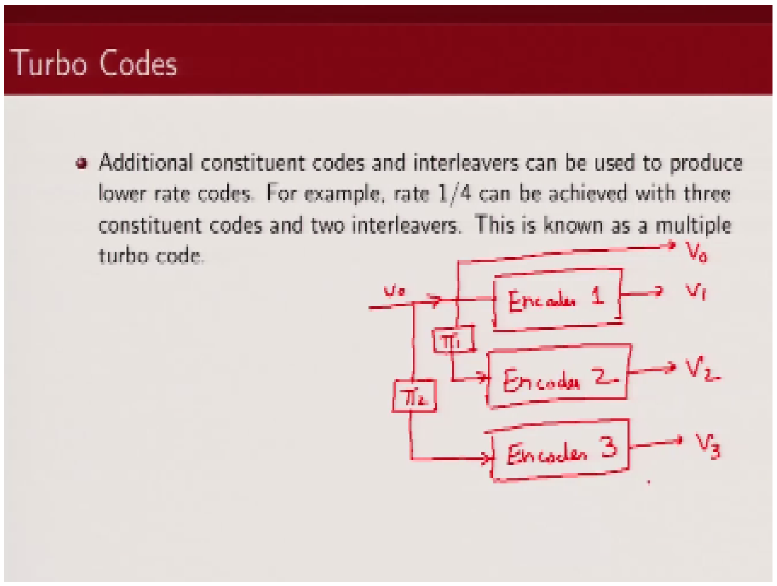


## Turbo Codes

- Additional constituent codes and interleavers can be used to produce lower rate codes. For example, rate 1/4 can be achieved with three constituent codes and two interleavers. This is known as a multiple turbo code.

Let us say I have encoder 1, I have encoder 2, I can have encoder 3 and what I can do is I can have information sequence coming in let us say u so this is directly going out this is $v_0$ this is $v_1$ then I have interleaver which I am denoting by this by so interleaved bits are coming to encoder 2, it is generating a parity bits $v_2$ then I have another set of interleaver, I am just denoting it by $\Pi_2$ and it is being fed to encoder 3 and I get another separate, so the structure that I have shown you for two encoders and one interleaver can be extended

(Refer Slide Time: 29:29)

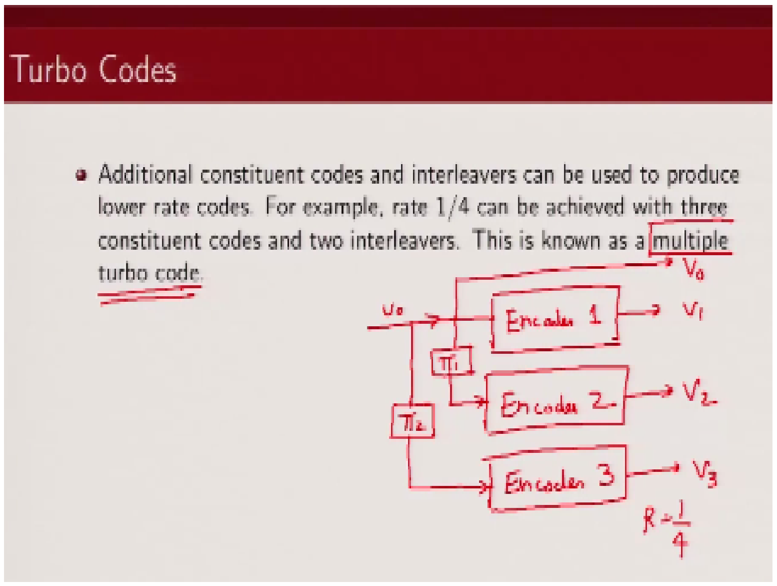


## Turbo Codes

- Additional constituent codes and interleavers can be used to produce lower rate codes. For example, rate 1/4 can be achieved with three constituent codes and two interleavers. This is known as a multiple turbo code.

For multiple encoders and multiple interleavers, so this will result in a rate ¼

(Refer Slide Time: 29:35)

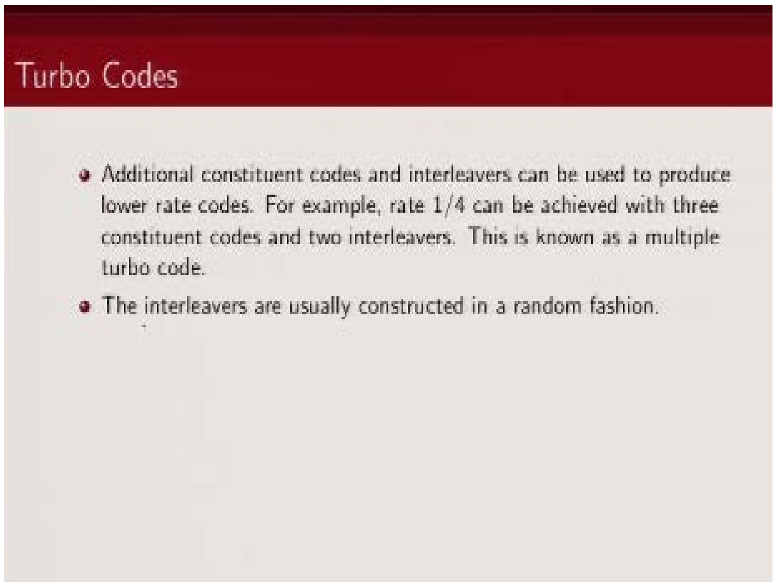


## Turbo Codes

- Additional constituent codes and interleavers can be used to produce lower rate codes. For example, rate 1/4 can be achieved with three constituent codes and two interleavers. This is known as a multiple turbo code.

What we call multiple turbo code so we use the term multiple turbo code, so this is an extension of the convolutional turbo code which uses two encoders and one interleaver.

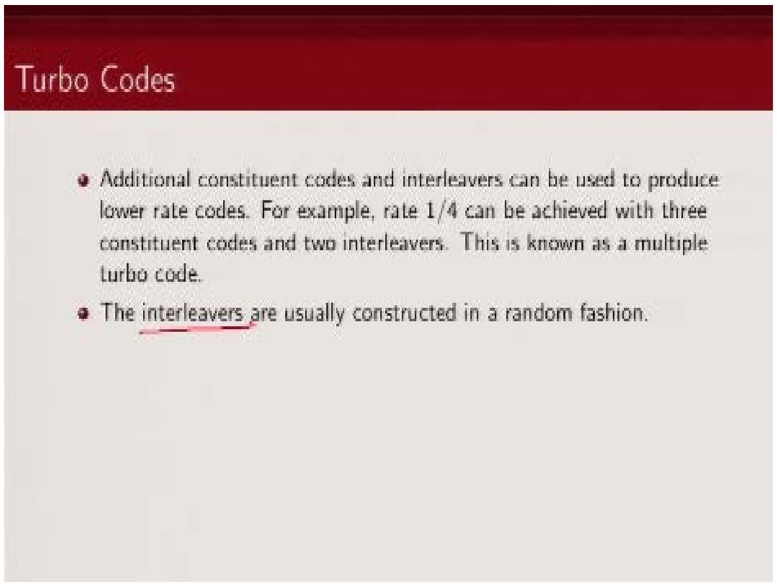
(Refer Slide Time: 29:56)



## Turbo Codes

- Additional constituent codes and interleavers can be used to produce lower rate codes. For example, rate 1/4 can be achieved with three constituent codes and two interleavers. This is known as a multiple turbo code.
- The interleavers are usually constructed in a random fashion.

(Refer Slide Time: 30:00)

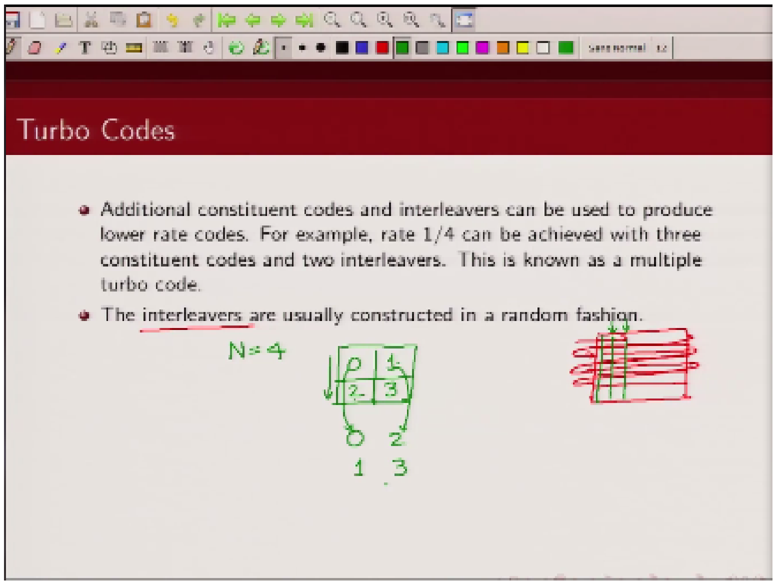


## Turbo Codes

- Additional constituent codes and interleavers can be used to produce lower rate codes. For example, rate 1/4 can be achieved with three constituent codes and two interleavers. This is known as a multiple turbo code.
- The interleavers are usually constructed in a random fashion.

Now I have shown you that design of interleaver is very, very crucial and there are multiple techniques you can use for designing interleaver. A very simple way is what is call block interleavers, so you

(Refer Slide Time: 30:15)



## Turbo Codes

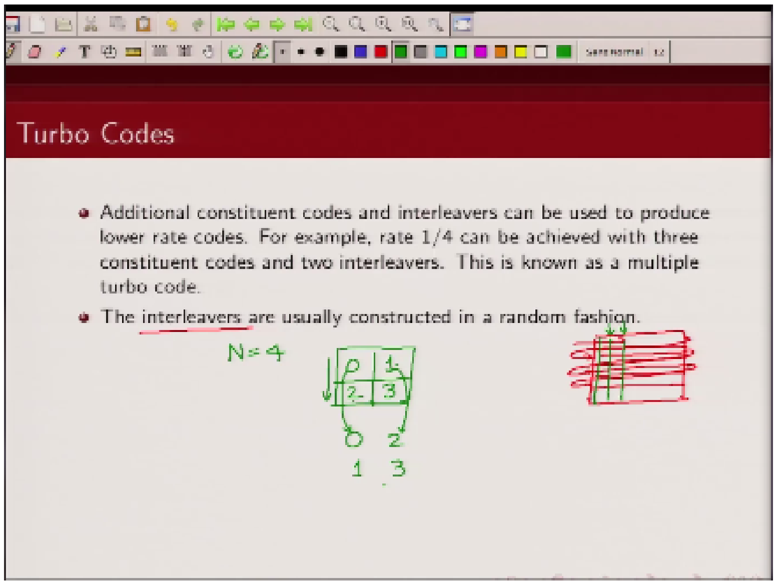
- Additional constituent codes and interleavers can be used to produce lower rate codes. For example, rate 1/4 can be achieved with three constituent codes and two interleavers. This is known as a multiple turbo code.
- The interleavers are usually constructed in a random fashion.

The information that you want to interleave or you want to send to the second encoder, what you can do is you can store the data block wise row wise, so you fill that data row wise so once you come here you then fill up. So this is how you fill up the data okay. So you have, you are filling up the data row wise. But when reading out you could read the data column wise so I can read data like this and then I can come here read this data come here this, so this is known as block interleaver.

A very simple example, let us say my block size is 4, so what I can do is I can put my data like this first, data I can put it here, second data I can put it here, third data I can put here and the fourth data I put here. But while reading instead of reading row wise I can read it column wise. So I first read 0, then I read 2nd, I read 1, and I read 3rd right. So what did I do so this was my original data but interleave version is now what was there in, so in location this same bit is there, but in this location now I am having what was earlier there in this location. And in this location I am having what was earlier I had in this location. So interleaving is nothing but just reordering of data.

You could also design it randomly, I will decide okay first bit I will put in this location, second bit I will put in this location. I just have to ensure that every bits are put in distinct location. No two bits are put in the same location right. And ideally what you would like is if there are like bit streams coming in which have consecutive ones they should be spread out after interleaving that is,

(Refer Slide Time: 32:17)

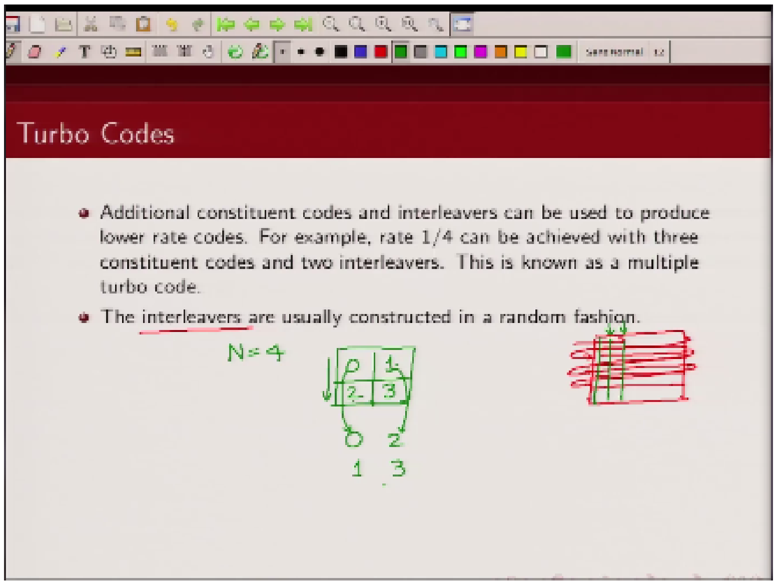


Turbo Codes

- Additional constituent codes and interleavers can be used to produce lower rate codes. For example, rate 1/4 can be achieved with three constituent codes and two interleavers. This is known as a multiple turbo code.
- The interleavers are usually constructed in a random fashion.

You would like to do, and there are class of interleavers like S random interleaver which imposes some additional constraints which will allow that two adjacent bits are at least spread by some amount okay?

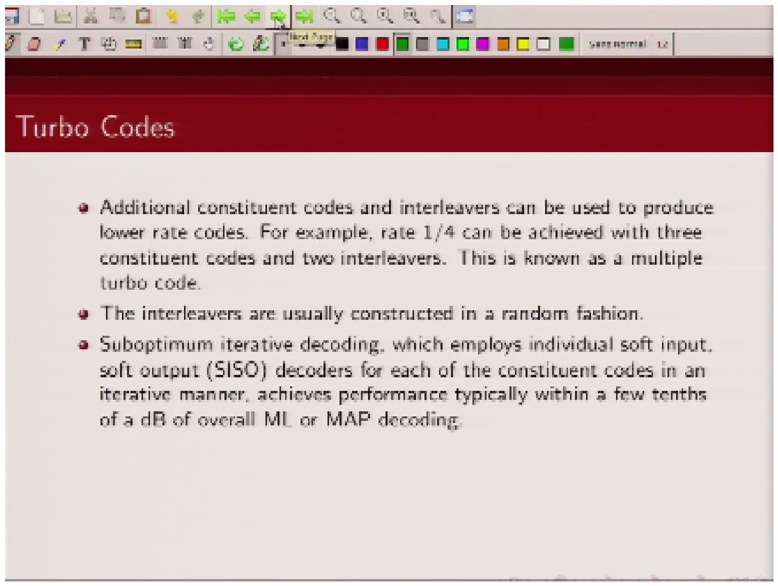
(Refer Slide Time:  32:30)



## Turbo Codes

- Additional constituent codes and interleavers can be used to produce lower rate codes. For example, rate 1/4 can be achieved with three constituent codes and two interleavers. This is known as a multiple turbo code.
- The interleavers are usually constructed in a random fashion.

(Refer Slide Time: 32:31)



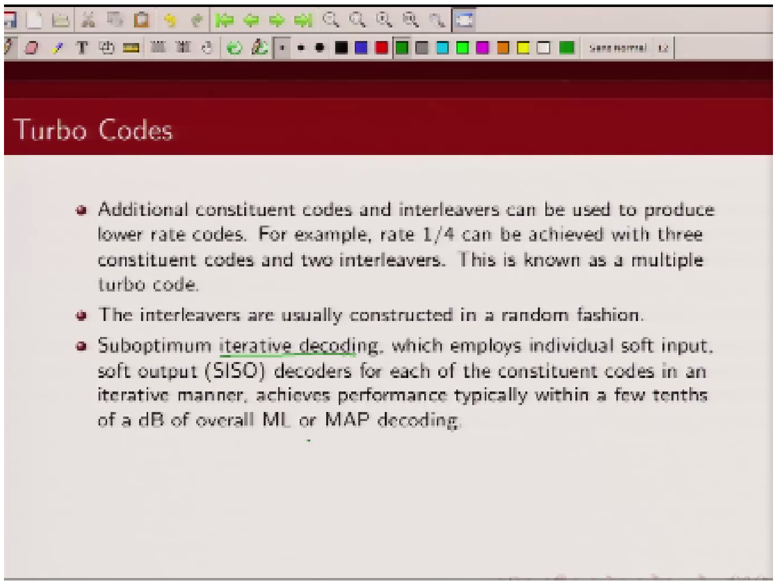
## Turbo Codes

- Additional constituent codes and interleavers can be used to produce lower rate codes. For example, rate 1/4 can be achieved with three constituent codes and two interleavers. This is known as a multiple turbo code.
- The interleavers are usually constructed in a random fashion.
- Suboptimum iterative decoding, which employs individual soft input, soft output (SISO) decoders for each of the constituent codes in an iterative manner, achieves performance typically within a few tenths of a dB of overall ML or MAP decoding.

So that is interleaver, so you have seen how we are constructing the encoder. So parallel concatenation of two encoders right and you have a interleaver, now how do we decode these codes? We are going to use an iterative algorithm so we are first going to decode one encoder then we are going to decode second encoder and these decoders will pass on information to among each other. So after the first encoder decodes it will pass on some information to second decoders saying okay I think these are the information bits.

Now the second encoder will take that input and will process and then it will give a new set of estimates to the first encoders saying, oh no I think this are the values of information sequence. And this process happens in a iterative fashion. In fact the name turbo codes come from the decoding structure, the turbo engine has this feedback mechanism and the decoding structure of this turbo code is also the feedback so from one encoder to other encoder and this is basically going around.

(Refer Slide Time: 33:45)

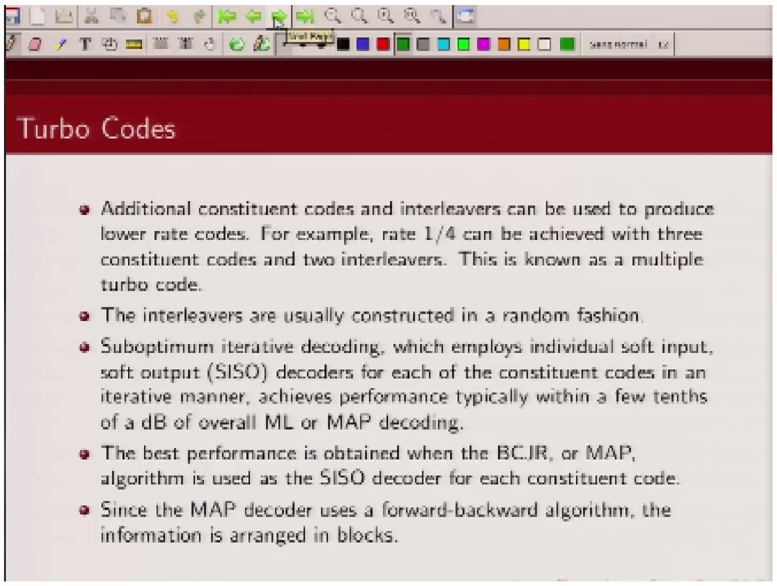


## Turbo Codes

- Additional constituent codes and interleavers can be used to produce lower rate codes. For example, rate 1/4 can be achieved with three constituent codes and two interleavers. This is known as a multiple turbo code.
- The interleavers are usually constructed in a random fashion.
- Suboptimum iterative decoding, which employs individual soft input, soft output (SISO) decoders for each of the constituent codes in an iterative manner, achieves performance typically within a few tenths of a dB of overall ML or MAP decoding.

So from there this name turbo codes have come. So we are going to use iterative decoding algorithm and each of these component encoders are going to use this BCJR algorithm and we already, when we talked about decoding of convolutional code we already talked about how BCJR algorithm works. In the next lecture we will talk about how the turbo decoding algorithm works.

(Refer Slide Time: 34:14)

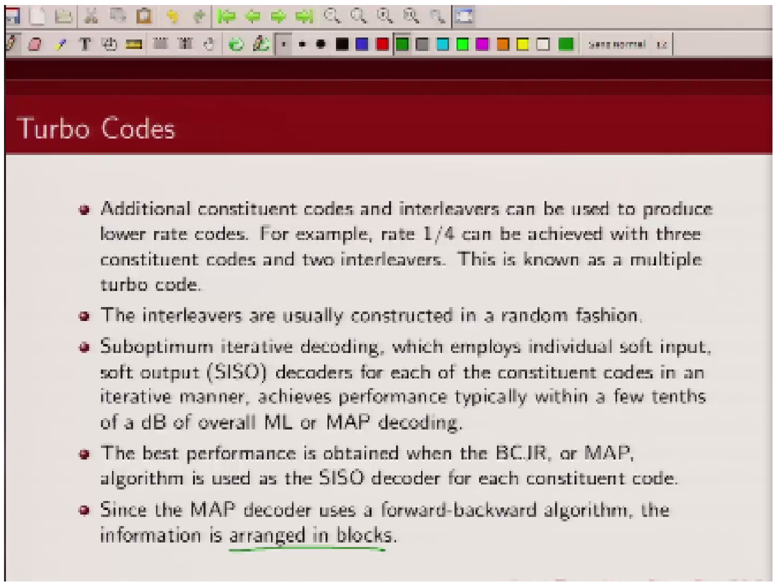


## Turbo Codes

- Additional constituent codes and interleavers can be used to produce lower rate codes. For example, rate 1/4 can be achieved with three constituent codes and two interleavers. This is known as a multiple turbo code.
- The interleavers are usually constructed in a random fashion.
- Suboptimum iterative decoding, which employs individual soft input, soft output (SISO) decoders for each of the constituent codes in an iterative manner, achieves performance typically within a few tenths of a dB of overall ML or MAP decoding.
- The best performance is obtained when the BCJR, or MAP, algorithm is used as the SISO decoder for each constituent code.
- Since the MAP decoder uses a forward-backward algorithm, the information is arranged in blocks.

And this decoding as I said happens by block fashion, so you wait until you receive n bits of data because remember this interleaving operation though it can be parallelized typically done in blocks of data. So you process, you decode the bits also in block by block fashion.

(Refer Slide Time: 34:37)

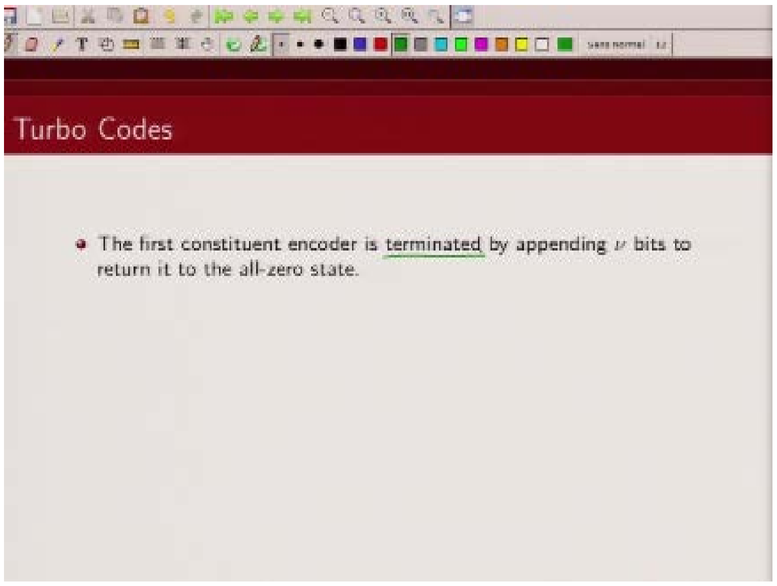


## Turbo Codes

- Additional constituent codes and interleavers can be used to produce lower rate codes. For example, rate 1/4 can be achieved with three constituent codes and two interleavers. This is known as a multiple turbo code.
- The interleavers are usually constructed in a random fashion.
- Suboptimum iterative decoding, which employs individual soft input, soft output (SISO) decoders for each of the constituent codes in an iterative manner, achieves performance typically within a few tenths of a dB of overall ML or MAP decoding.
- The best performance is obtained when the BCJR, or MAP, algorithm is used as the SISO decoder for each constituent code.
- Since the MAP decoder uses a forward-backward algorithm, the information is $\underline{\text{arranged in blocks}}$.

So the information can be arranged in blocks of data.

(Refer Slide Time: 34:41)

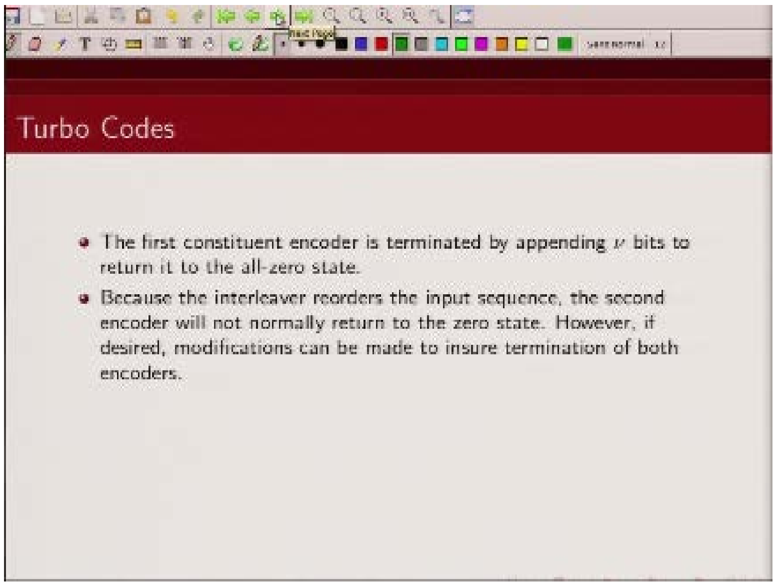


## Turbo Codes

- The first constituent encoder is terminated by appending $\nu$ bits to return it to the all-zero state.

Now typically for better performance as I mentioned we typically tried initially the encoder is in all-zero state and we like to bring back the encoder back to all-zero state after we have transmitted our information sequence. So that is known as termination and we terminate this turbo encoder by appending some tail bits to our information bits.

(Refer Slide Time: 35:14)

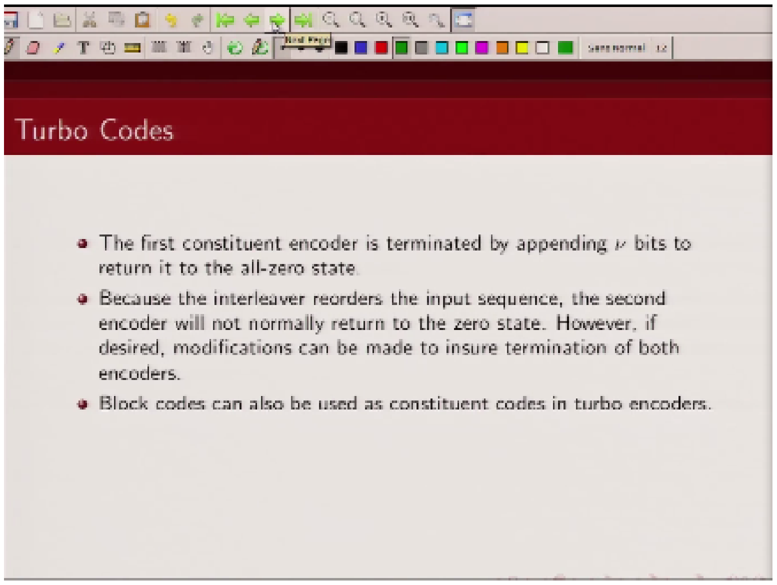


## Turbo Codes

- The first constituent encoder is terminated by appending $\nu$ bits to return it to the all-zero state.
- Because the interleaver reorders the input sequence, the second encoder will not normally return to the zero state. However, if desired, modifications can be made to insure termination of both encoders.

Now because of the interleaver it is not always possible to have the same tail bits terminate both the encoders. You could design a interleaver which will allow you to terminate both the encoder, that is possible but in general it is not possible to terminate both the encoders with same termination bits. So you could either terminate first encoder, leave the second encoder un-terminated, or you could send additional bits to terminate the second encoder.

(Refer Slide Time:  35:48)

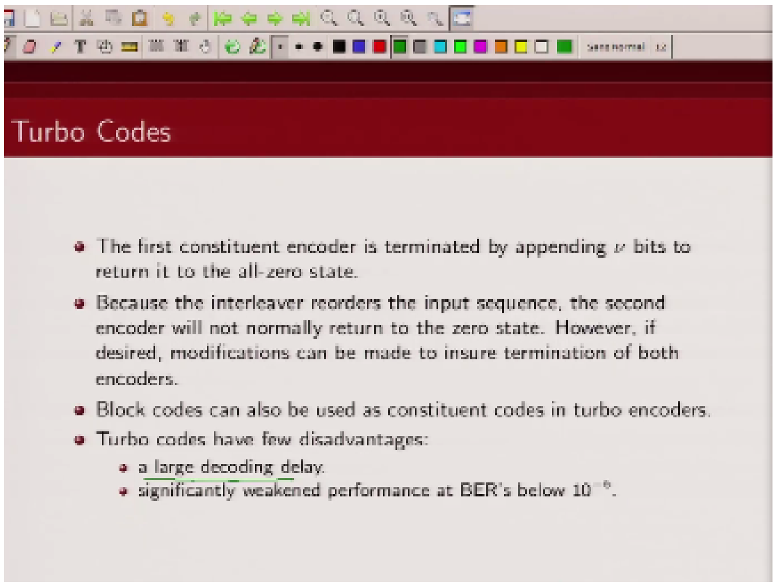


## Turbo Codes

- The first constituent encoder is terminated by appending $\nu$ bits to return it to the all-zero state.
- Because the interleaver reorders the input sequence, the second encoder will not normally return to the zero state. However, if desired, modifications can be made to insure termination of both encoders.
- Block codes can also be used as constituent codes in turbo encoders.

So far I have shown you that we are using recursive systematic convolutional encoder in a parallel configuration. We could also design same thing using block codes as well.

(Refer Slide Time: 36:04)

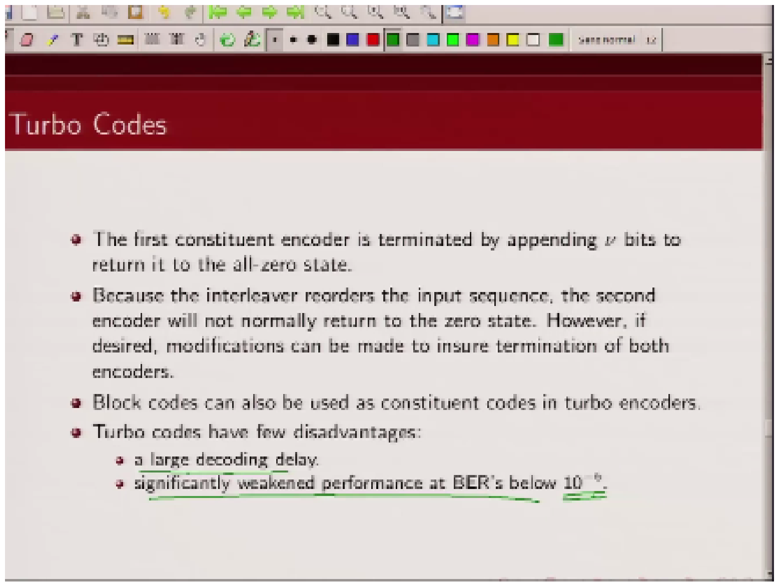


## Turbo Codes

- The first constituent encoder is terminated by appending $\nu$ bits to return it to the all-zero state.
- Because the interleaver reorders the input sequence, the second encoder will not normally return to the zero state. However, if desired, modifications can be made to insure termination of both encoders.
- Block codes can also be used as constituent codes in turbo encoders.
- Turbo codes have few disadvantages:
  - a large decoding delay.
  - significantly weakened performance at BER's below $10^{-5}$.

And we are not going to discuss those in this lecture. Now there are some disadvantages of turbo code, though their performance is extremely good there are some inherent problems with turbo code, one is they have a large decoding delay typically their performance is good for large block size. So when you are dealing with large sizes and there is a lot delay involved because you are doing block by block processing.

(Refer Slide Time:  36:34)

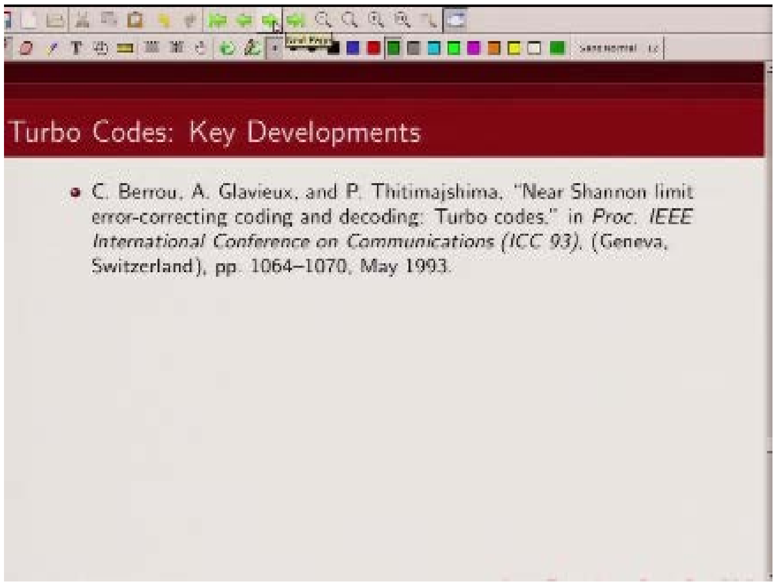


## Turbo Codes

- The first constituent encoder is terminated by appending $\nu$ bits to return it to the all-zero state.
- Because the interleaver reorders the input sequence, the second encoder will not normally return to the zero state. However, if desired, modifications can be made to insure termination of both encoders.
- Block codes can also be used as constituent codes in turbo encoders.
- Turbo codes have few disadvantages:
  - a large decoding delay.
  - significantly weakened performance at BER's below $10^{-5}$.

And second thing is typically if you are looking at very, very low bit error rate performance there you, you just saw that because of error flow their performance is not very good at very, very low bit error rate.

(Refer Slide Time:  36:55)

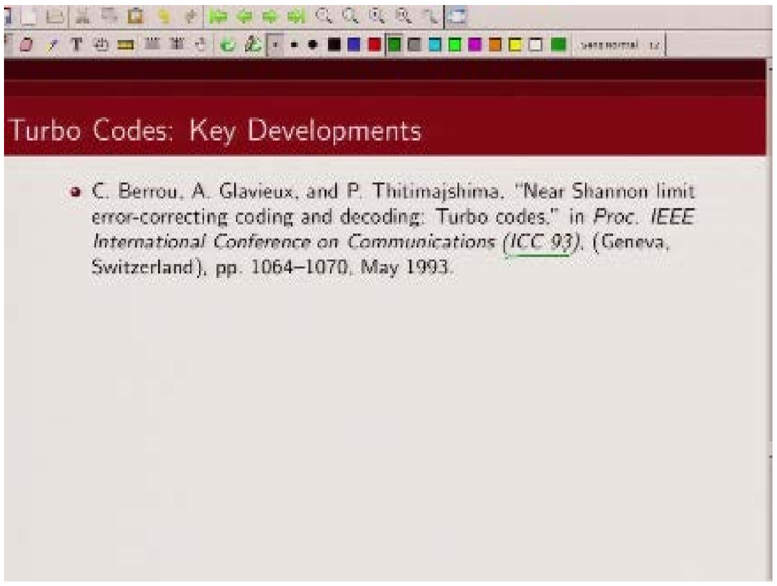


## Turbo Codes: Key Developments

- C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo codes," in *Proc. IEEE International Conference on Communications (ICC 93)*, (Geneva, Switzerland), pp. 1064–1070, May 1993.

Now I will just conclude this lecture by giving you some references of some key early development in turbo codes and you can read those references.

(Refer Slide Time: 37:06)

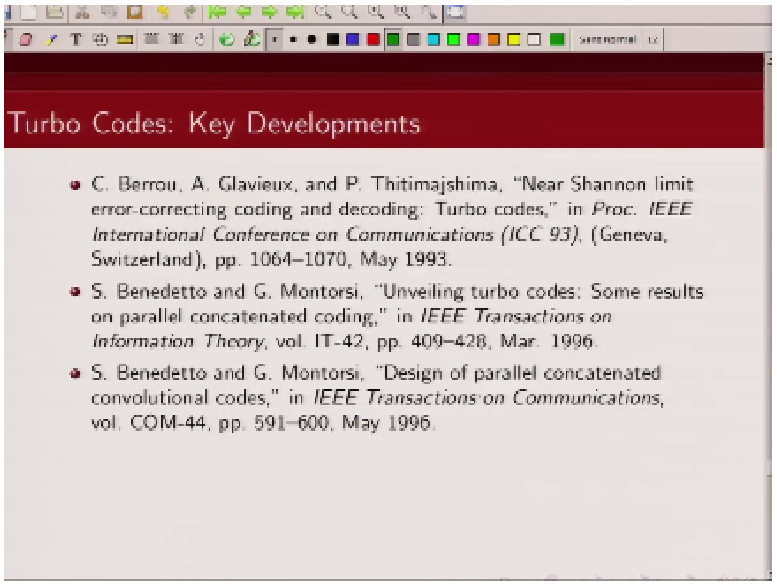


## Turbo Codes: Key Developments

- C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo codes," in *Proc. IEEE International Conference on Communications (ICC 93)*, (Geneva, Switzerland), pp. 1064–1070, May 1993.

So this is where turbo codes were initially introduced by paper by Berrou Atal in ICC 1993. When this paper came people did not actually believe the results because they were very good performing codes but subsequently multiple research groups were able to reproduce their results.

(Refer Slide Time: 37:27)

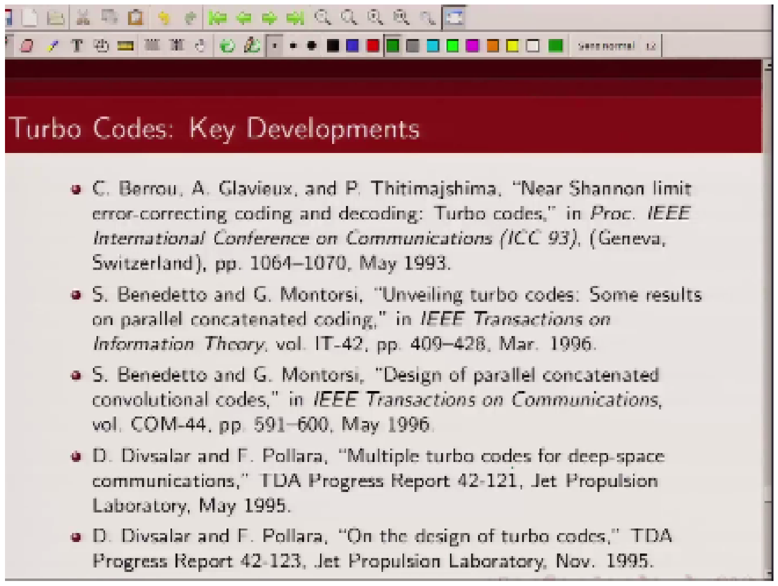


## Turbo Codes: Key Developments

- C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo codes," in *Proc. IEEE International Conference on Communications (ICC 93)*, (Geneva, Switzerland), pp. 1064–1070, May 1993.
- S. Benedetto and G. Montorsi, "Unveiling turbo codes: Some results on parallel concatenated coding," in *IEEE Transactions on Information Theory*, vol. IT-42, pp. 409–428, Mar. 1996.
- S. Benedetto and G. Montorsi, "Design of parallel concatenated convolutional codes," in *IEEE Transactions on Communications*, vol. COM-44, pp. 591–600, May 1996.

Now the next set of papers, these two papers by Benedetto Atal "Unveiling turbo codes. Some results on parallel concatenated coded" and "Design of parallel concatenated convolutional codes". These were one of the first paper to explain why the performance of turbo codes is not very good at high signal to noise ratio and why do they suffer from error flow?

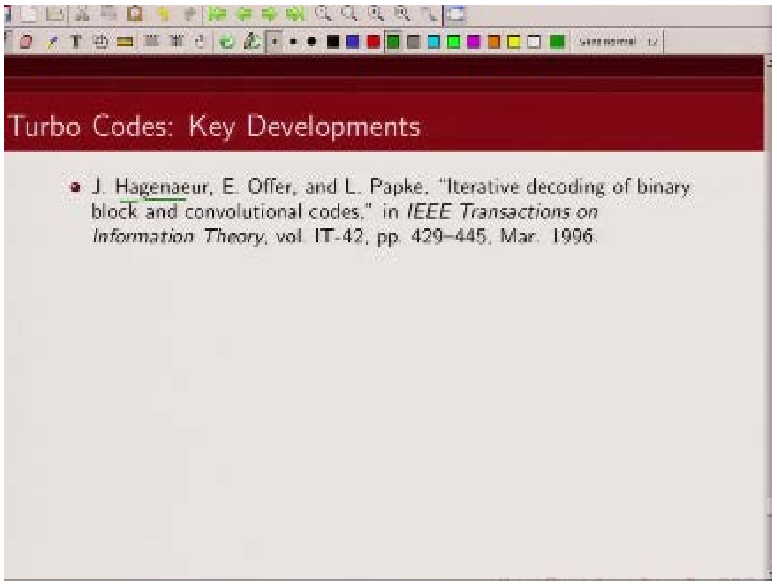
(Refer Slide Time: 37:54)


## Turbo Codes: Key Developments

- C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo codes," in *Proc. IEEE International Conference on Communications (ICC 93)*, (Geneva, Switzerland), pp. 1064–1070, May 1993.
- S. Benedetto and G. Montorsi, "Unveiling turbo codes: Some results on parallel concatenated coding," in *IEEE Transactions on Information Theory*, vol. IT-42, pp. 409–428, Mar. 1996.
- S. Benedetto and G. Montorsi, "Design of parallel concatenated convolutional codes," in *IEEE Transactions on Communications*, vol. COM-44, pp. 591–600, May 1996.
- D. Divsalar and F. Pollara, "Multiple turbo codes for deep-space communications," TDA Progress Report 42-121, Jet Propulsion Laboratory, May 1995.
- D. Divsalar and F. Pollara, "On the design of turbo codes," TDA Progress Report 42-123, Jet Propulsion Laboratory, Nov. 1995.

These were the first paper when they were, where they generalized by Divsalar at JPL lab where they generalized it to multiple turbo codes and this paper also gives some design rules of well how to design turbo codes.

(Refer Slide Time: 38:11)

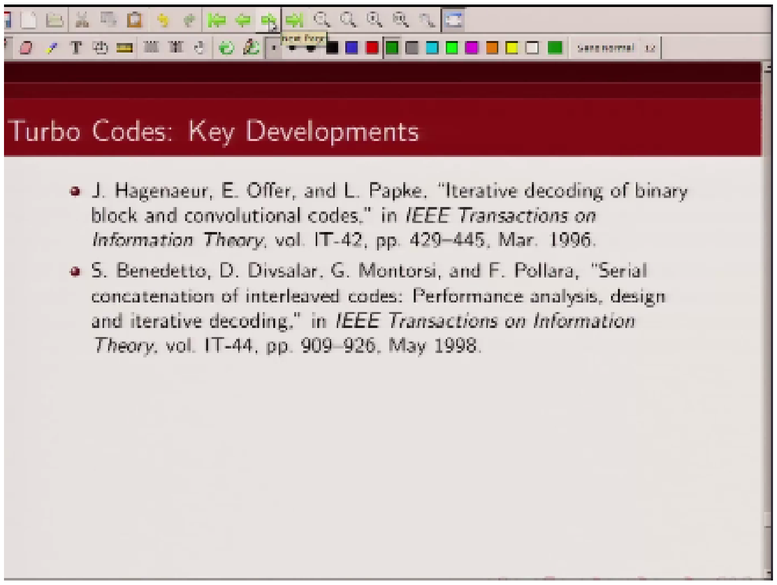


## Turbo Codes: Key Developments

- J. Hagenaeur, E. Offer, and L. Papke, "Iterative decoding of binary block and convolutional codes," in *IEEE Transactions on Information Theory*, vol. IT-42, pp. 429–445, Mar. 1996.

This is a nice by Hagenaeur, this is a nice paper on "Iterative decoding of binary block and convolutional code", this explains the BCJR algorithm and turbo decoding.

(Refer Slide Time: 38:28)

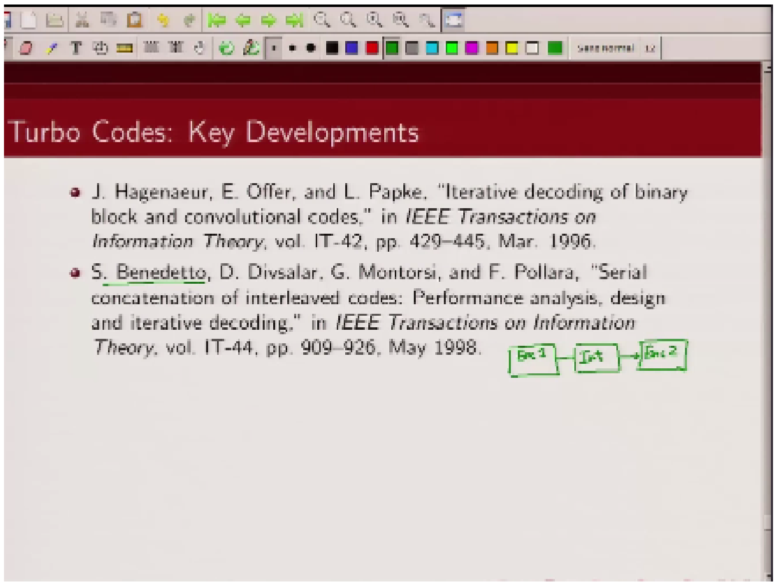


## Turbo Codes: Key Developments

- J. Hagenaeur, E. Offer, and L. Papke, "Iterative decoding of binary block and convolutional codes," in *IEEE Transactions on Information Theory*, vol. IT-42, pp. 429–445, Mar. 1996.
- S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "Serial concatenation of interleaved codes: Performance analysis, design and iterative decoding," in *IEEE Transactions on Information Theory*, vol. IT-44, pp. 909–926, May 1998.

And BCJR algorithm in a very nice way, now in this lecture we talked about parallel concatenation, now there is a different variant where you could have a serial concatenations.

(Refer Slide Time: 38:39)



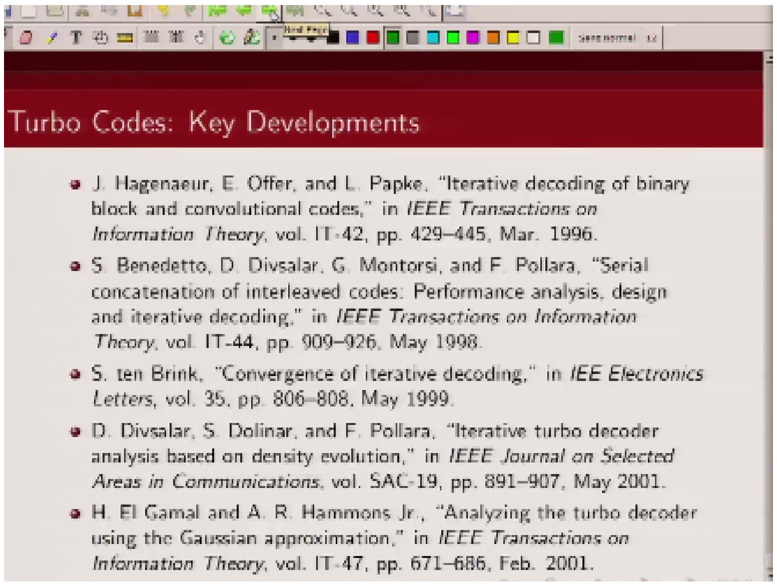
## Turbo Codes: Key Developments

- J. Hagenaeur, E. Offer, and L. Papke, "Iterative decoding of binary block and convolutional codes," in *IEEE Transactions on Information Theory*, vol. IT-42, pp. 429–445, Mar. 1996.
- S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "Serial concatenation of interleaved codes: Performance analysis, design and iterative decoding," in *IEEE Transactions on Information Theory*, vol. IT-44, pp. 909–926, May 1998.

$$\begin{tikzcd} \boxed{\text{Enc 1}} \arrow[r] & \boxed{\text{Int}} \arrow[r] & \boxed{\text{Enc 2}} \end{tikzcd}$$

So you could have a encoder 1 here then you could have one interleaver here, interleaver and then you could have another encoder. So this variation was proposed by Benedetto and others and this serial concatenation though has little inferior performance in the waterfall region it has better performance in the error flow region.

(Refer Slide Time: 39:12)

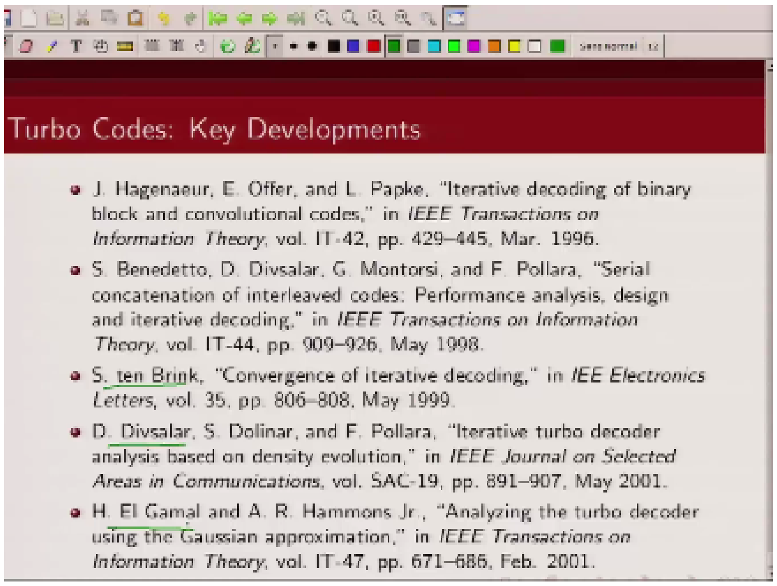


## Turbo Codes: Key Developments

- J. Hagenaeur, E. Offer, and L. Papke, "Iterative decoding of binary block and convolutional codes," in *IEEE Transactions on Information Theory*, vol. IT-42, pp. 429–445, Mar. 1996.
- S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "Serial concatenation of interleaved codes: Performance analysis, design and iterative decoding," in *IEEE Transactions on Information Theory*, vol. IT-44, pp. 909–926, May 1998.
- S. ten Brink, "Convergence of iterative decoding," in *IEE Electronics Letters*, vol. 35, pp. 806–808, May 1999.
- D. Divsalar, S. Dolinar, and F. Pollara, "Iterative turbo decoder analysis based on density evolution," in *IEEE Journal on Selected Areas in Communications*, vol. SAC-19, pp. 891–907, May 2001.
- H. El Gamal and A. R. Hammons Jr., "Analyzing the turbo decoder using the Gaussian approximation," in *IEEE Transactions on Information Theory*, vol. IT-47, pp. 671–686, Feb. 2001.

And finally these three papers that you have discusses the behavior of the iterative decoding algorithm. Why choice of some encoders is good, how should you choose the encoders so that you have better performance in the waterfall region?

(Refer Slide Time: 39:31)



## Turbo Codes: Key Developments

- J. Hagenaeur, E. Offer, and L. Papke, "Iterative decoding of binary block and convolutional codes," in *IEEE Transactions on Information Theory*, vol. IT-42, pp. 429–445, Mar. 1996.
- S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "Serial concatenation of interleaved codes: Performance analysis, design and iterative decoding," in *IEEE Transactions on Information Theory*, vol. IT-44, pp. 909–926, May 1998.
- S. ten Brink, "Convergence of iterative decoding," in *IEE Electronics Letters*, vol. 35, pp. 806–808, May 1999.
- D. Divsalar, S. Dolinar, and F. Pollara, "Iterative turbo decoder analysis based on density evolution," in *IEEE Journal on Selected Areas in Communications*, vol. SAC-19, pp. 891–907, May 2001.
- H. El Gamal and A. R. Hammons Jr., "Analyzing the turbo decoder using the Gaussian approximation," in *IEEE Transactions on Information Theory*, vol. IT-47, pp. 671–686, Feb. 2001.

This was nicely explained in these three papers and each one of them use different technique. Ten Brink used mutual information, Divsalar used the evolution of the densities of the extrinsic information and El Gamal used mean and variance of the extrinsic information, and they tracked it to get information about the convergence of the turbo code. So with this I am going to conclude my discussion on turbo codes. Thank you.

**Acknowledgement**

**Ministry of Human Resource & Development**

**Prof. Satyaki Roy**

**Co-ordinator, NPTEL IIT Kanpur**

**NPTEL Team**

**Sanjay Pal**

**Ashish Singh**

**Badal Pradhan**

**Tapobrata Das**

**Ram Chandra**

**Dilip Tripathi**

**Manoj Shrivastava**

**Padam Shukla**

**Sanjay Mishra**

**Shubham Rawat**

**Shikha Gupta**

**K. K.  Mishra**

**Aradhana Singh**

**Sweta**

**Ashutosh Gairola**

**Dilip Katiyar**

**Sharwan**

**Hari Ram**

**Bhadra Rao**

**Puneet Kumar Bajpai**

**Lalty Dutta**

**Ajay Kanaujia**

**Shivendra Kumar Tiwari**

**an IIT Kanpur Production**