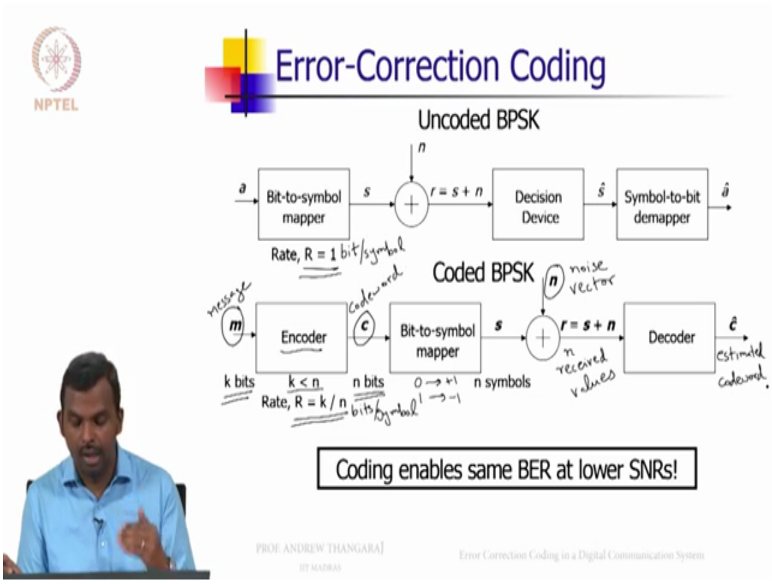
LDPC and Polar codes in 5G Standard
Error Correction Coding in a Digital Communication System
Professor Andrew Thangaraj
Department of Electrical Engineering
Indian Institute of Technology, Madras
Error-Correction Coding

(Refer Slide Time: 0:18)



Now we are ready now to say what error control coding does to uncoded BPSK System. So here you have an uncoded BPSK System, you have a bit you converted into a symbol, noise gets added, there is a decision device, you do a symbol to bit mapping a hat, okay. So the rate R is said to be 1 here, 1 in the sense 1 bit per symbol. Every bit carries 1 bit every symbol carries 1 bit of information, every bit is converted into only 1 symbol, 1 bit 1 symbol so your rate is also 1 and that is the uncoded BPSK.

What happens in coded BPSK? You do not transmit 1 bit at a time, you collect k bits into a message m, so this terminology of message we will use again and again, you collect k bits into a message here. Now this k can be greater than 1 is usually yes and it can be quite high also into entropical system today people use k of thousand, ten thousand, five hundred, hundred, two hundred numbers like that keep those kind of numbers in mind.

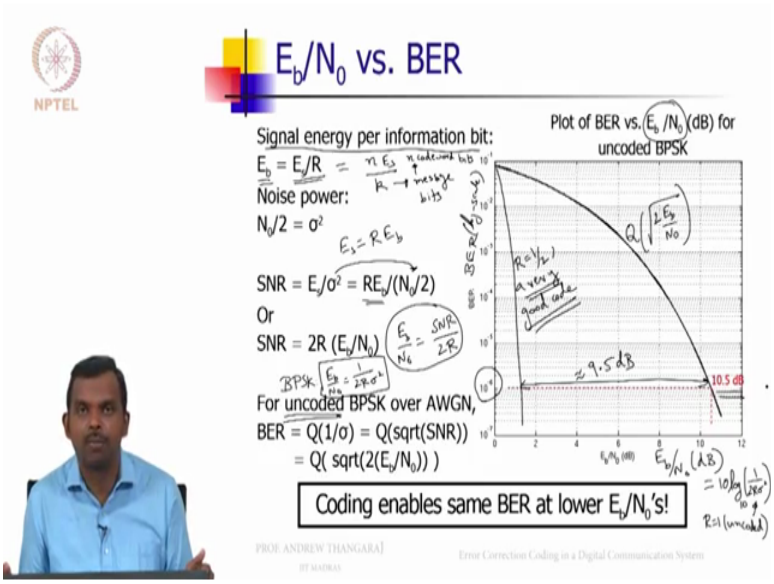
Now you will take this message m and you will encode it using some encoder into a code word c and this code word will be n bits in length and this n will be greater than k. So you will take say hundred bits and produce a two hundred bit code word. So now when you do that your rate goes down, your hundred got converted into two hundred so the rate becomes k by n bits per symbol, so if k is hundred, n is two hundred you have a rate 1 by 2.

So once you converted into a code word what happens after that is similar to the encoded BPSK case, you do a bit to symbol mapper and you get n symbols, how does the bit to symbol mapper work? 0 goes to plus 1, 1 goes to minus 1, there is nothing different about the bit to symbol mapper than the uncoded BPSK case, exactly same thing happens, you get a symbol vector now s and this goes through the AWGN channel and you have a noise vector here and it will have n values of noise, they are all independent Gaussian but still it will have n values of noise and you will have n received values.

Now you have the task of building a decoder which will take this n received values and put out estimated code word which is actually n bits. Now why will you go to all this trouble? Why should you do all these things? Why should you do coding? Why cannot you just do uncoded case? The reason is written here, coding enables the same BER the same bit error rate at a lower SNR, so this is an important story to remember about coding, why would you do coding? You have a certain BER versus SNR trade-off, right BER is q of square root of SNR, if you do coding you can get the same BER at a lower signal to noise ratio.

Now why would you want that because in your system the signal power costs you something, if you think of a mobile phone the signal you are transmitting costs you energy and that energy comes from the battery, battery is getting drained, if you use lesser energy then you can use the battery energy for other things like may be playing games on your phone, I do not know may be other things you can do with that or your phone's battery will last much longer, so your signal power has to be reduced as much as possible, but you also want to achieve the same bit error rate, so if you do coding you get this benefit without going for a higher signal to noise ratio you can get the same bit error rate, get bit error rate lower bit error rate at the same SNR or the same BER can be achieved at a lower SNR, so this is what is important.

(Refer Slide Time: 4:36)



So that is nice you might say it is interesting, so let me begin by describing this plot here, this plot here actually shows BER versus E b over N not, now most of you are familiar with digital communication will know that SNR is not the correct metric in the presence of coding, you have to use something called E b over N not, now what is this E b over N not? Where does it come from?

So let me describe that briefly here, so if you remember from the previous picture the uncoded case had a rate of 1, every symbol carried 1 bit of information. Now once you do coding every symbol is not carrying 1 bit of information, it is carrying less than 1 bit of information. So uncoded and coded the comparison is not fair because if you spend so much energy in the uncoded case for a symbol you are transmitting 1 bit, okay but if you spend the same amount of energy in a coded case you are not sending 1 bit, you are sending less than 1 bit, if you want to send k bits you have to send n symbols so you have to spend n times the symbol energy and you will be sending only k bits on the other hand in the uncoded case if you send n bits with n times s you will be sending n information bits, so you have to equalize these two things and that is what E b over N not does, so let me describe that.

So signal E b is the signal energy per information bit and it is defined as E s divided by R, how does that come? This is n times E s divided by k, so k information bits k message bits gets converted into n information bits n code word bits, you have k information bits or message bits, you convert them into n code word bits and you actually transmit them on the channel, so the total energy you take up is n times E s, okay E b is actually the energy per information bit, so n times E s is the total energy consumed for k information bits, for k information bits you consume an energy of n times E s, so for 1 information bit you consume an energy of n times E s divided by k and that is E b.

So the noise power remains the same n not by 2 is sigma squared, so SNR is E by sigma squared one can write it in terms of E b and N not so instead of E s so here what did I do E s is R times E b, so I have put that here R times E b and sigma squared is N not by 2 and if you simplify you see that SNR is two times R times E b over N not or another way to write this is E b over N not is SNR divided by 2 times R, this is the main result. So you have to divide by 2 times R and plot it versus (when) convert SNR into E b over N not, you have to convert SNR into E b over N not dividing by 2 times rate.

So in the BPSK case we know SNR is 1 by sigma squared, so for BPSK (maybe I should write that here) for BPSK E b over N not is 1 by 2 R sigma squared, so you remember this formula it is quite important, BPSK E b over N not is 1 by 2 R sigma squared, so if your rate is 1 E b over N not is 1 by 2 sigma squared, if your rate is half rate is one half if you are taking N to be two times k then your E b over N not is 1 by sigma squared, so that is the formula for E b over N not.

Now you can convert your BER versus SNR formula that you had into E b over N not, you will get an additional factor of 2 because for uncoded rate is 1, so your SNR is two times E b over N not and that is what you have here and you can make a plot, you can make a plot of bit error rate, this is BER on the Y axis versus E b over N not in dB, how do you convert E b over N not to dB? You do the same thing you take 1 by 2 R sigma squared and then you do ten times log base 10 (so maybe I should write that down here) this 10 times log base 10 of 1 by 2 R sigma squared whatever rate you have.

So if you put rate equals 1, then it is uncoded. So for the uncoded case we know what BER is with E b over N not, so it is Q of 1 by sigma and that will convert as square root of 2 E b over N not and you can make this plot, okay. So I urge all of you to write a small piece of MATLAB or Python or any code you like and generate this graph, this graph is very very important for any error control coding class, you should be able to generate this plot of BER versus E b over N not, BER in the log scale, so you can see this is log scale and E b over N not in dB scale and for uncoded BPSK remember uncoded BPSK this plot is simply Q of square root of 2 E b over N not, so it is an easy enough formula and you plot this.

And one particular data point is 10.5 dB which corresponds to a bit error rate of 10 power minus 6, so just remember this keep this in mind, it is a good metric to remember. So if you want a bit error rate of 10 power minus 6 with uncoded BPSK you need an E b over N not of 10 dB or so, 10.5 dB. So that means roughly signal energy is 10 times the noise energy, 10 dB is the same as factor of 10, so that is the conversion that you have here.

Now what will coding do? If you do coding, if you do a very good code in fact you can even achieve something like this, so let us say R equals half a very good code. What is this very good code? We will see later on in this class, the codes in the 5G standard are very good codes and they can give you a picture like this. So what have you gained? Look at this point, so you have gained roughly about 9.5 dB or so, maybe even more look at the calculation.

So (instead of having a noise power which is) instead of having a signal power which is 10 times as much as the noise power, it is enough if you have a signal power which is just a little bit more than the noise power, okay image the savings, your battery will last that much longer, things do not heat up that much, everything is wonderful in this scenario. So this is essentially what we will see in this class, we will see how these very good codes in the 5G standard can be build encoder can be built for them, decoder can be built for them and how you can achieve these kind of coding games.

So our presentation will mostly be from an implementation point of view, how to implement these algorithms? But we will spend some time discussing about where these algorithms come from the or where these codes come from and all that but most of our focus will be on having an implementation for the codes in the 5G standard and maybe even show these kind of graphs, maybe generate a graph like this and show how this works, thank you very much.