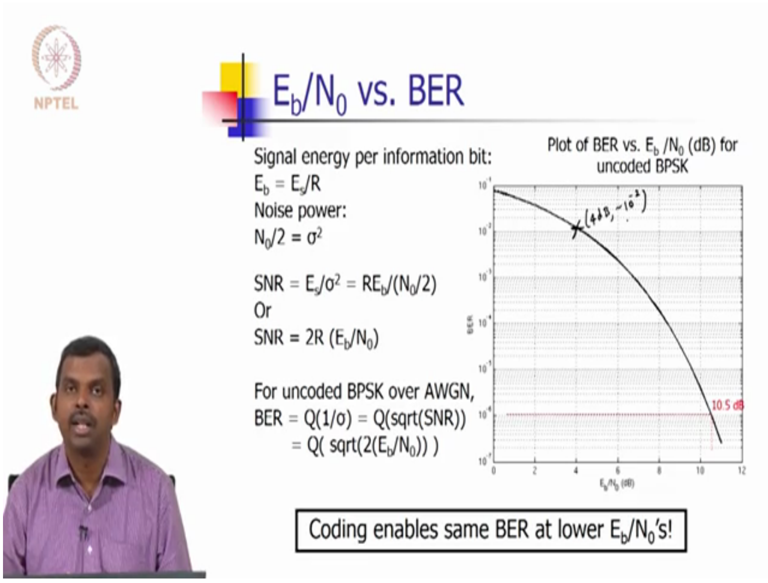
LDPC and Polar Codes in 5G Standard
Complementary Error Function
Professor Andrew Thangaraj
Department of Electrical Engineering
Indian Institute of Technology, Madras

(Refer Slide Time: 00:17)



Okay, so in the previous lecture we stopped at this plot, a plot of bit error rate versus Eb over N not for uncoded BPSK and I was claiming this is the curve for instance at 10.5 dB of Eb over N not, we have a bit error rate of 10 power minus 6 and coding is going to help you move this curve to the left, maybe at a lower rate than 1 but it can do that okay.

So what people do in this lecture is multiple things, first is this course will involve some programming language okay, so I am not very particular on which one you should use, you can use the one which you like but what we will demonstrate in this class will be MATLAB or its equivalent or clone or limited clone which is the GNU Octave program okay so now MATLAB is licensed, you may have to get a license for it, your institution may have it or your organization may have it, if it does you can use MATLAB.

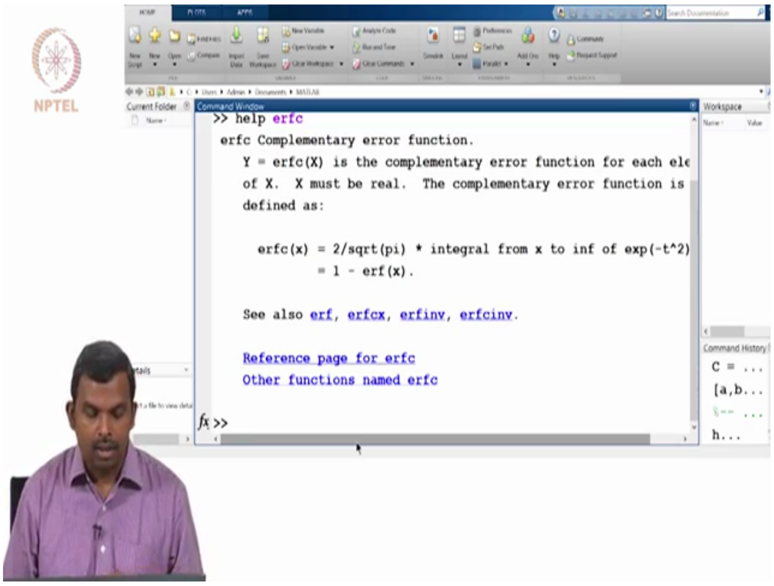
If you do not have such a license you can download Octave for free, it is a free and open source software and you can use it, the reason why I am sticking to MATLAB Octave is it has a common syntax and it is easy and many people know it so that is why I am doing it but none of the assignments will actually require you to stick to MATLAB but you should do some programming, you might use Python if you like or C if you like, you will have to write some programs but I will show you what I will demonstrate to you in class will be MATLAB and Octave also. I will use both just to show you how they are both quite equivalent okay.

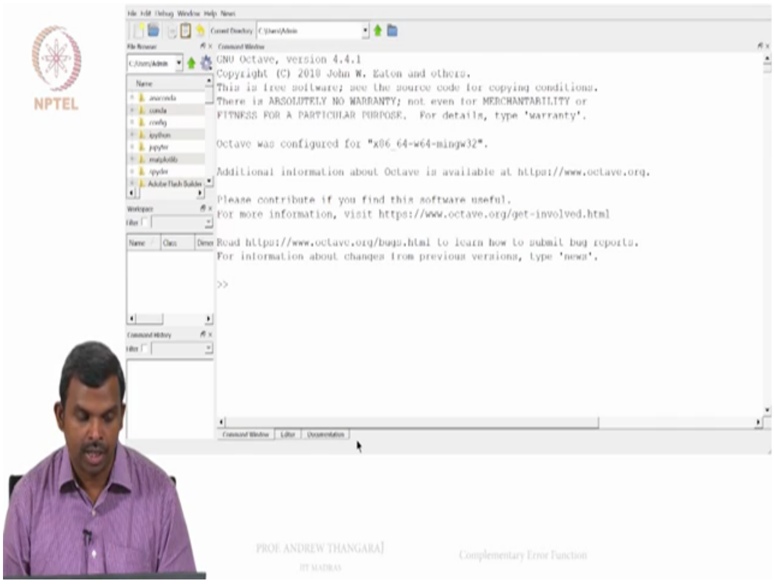
So to begin with that in this lecture we are going to see couple of things one mainly is how to generate this graph okay, we will see that using MATLAB and Octave as well and then how to write a quick simulation to validate this graph okay so this graph for instance predicts 4 dB, for 4 dB over N not it predicts bit error rate of around 10 power minus 4 dB, roughly around 10 power minus 2 right.

So how to validate this point, okay so how do you validate it by simulation so we will write a quick small simulation to check what will be the error rate at 4 dB over N not, we will do it for the uncoded BPSK case, it is good to keep this program with you cause soon enough we will add coding into this program and at that point you can keep the rest of the framework the same, you just have to change some encoding and you can keep using this framework.

So the little piece of code we write today which is the simulator for uncoded BPSK is very very useful for the entire course and we will build on it and eventually have our encoders and decoders for LDPC codes and polar codes written as part of that program, okay so let us get started, I am going to show you first which has MATLAB.

(Refer Slide Time: 03:36)

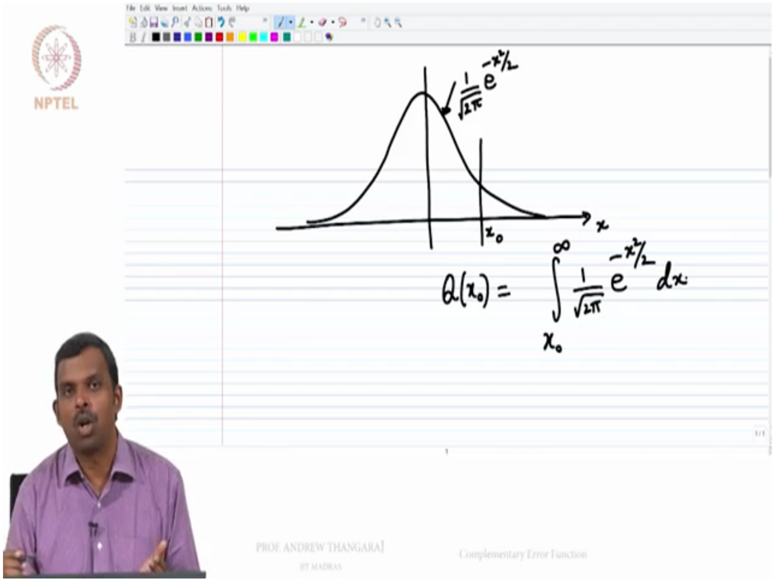




So this is how MATLAB looks, it is of course like I said it is licensed software so you need to get a license for it as well and I will show you how Octave looks okay and this is credits for octave, it is free and open source distributer and GNU licensed, there is no warranty and things like that that most software comes with no warranty but it is free right so you can just do that and most of the programs and code that we will write in this class will require only elementary use of computing platform MATLAB, we would not use any of the inbuilt packages, we will write everything from scratch.

So most of the things I write you will be able to run it on Octave just like that as well okay so just for completeness today I will show both Octave and MATLAB, going forward maybe we will stick to one of the two, just to keep things simple okay. So let us get started, the first thing I want to say is about this queue function so let me say a few words about what this queue function is.

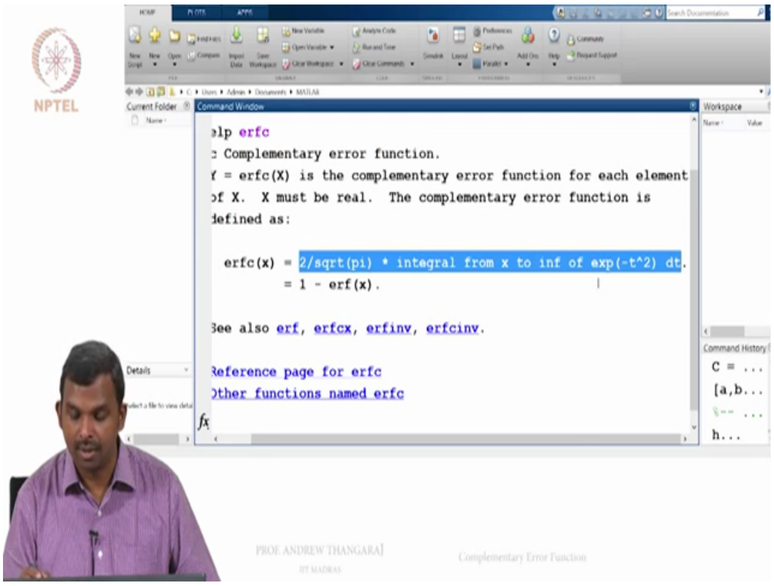
(Refer Slide Time: 04:50)



Like I mentioned you have this Gaussian integral right and then this the Gaussian PDF which is 1 by root 2 pi e power minus x squared by 2 okay so this is the 0 mean variance 1 Gaussian PDF. So maybe I should write this 2 pi a bit more clearly okay and if you fix a point like say x not okay Q of x not is defined as integral from x not to infinity of this function 1 by root 2 pi e power minus x squared by 2 dx okay.

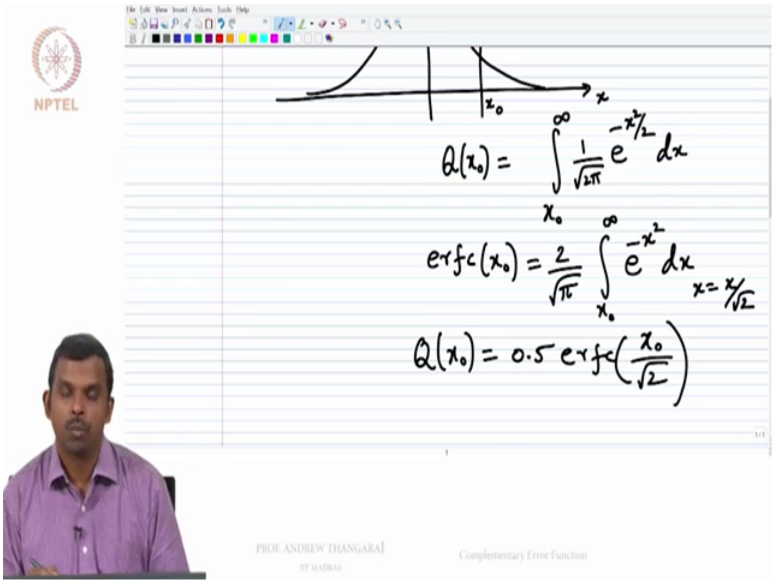
So this is the function that we have to implement to be able to make a plot and that plot was with Q functions, now it turns out many for instance MATLAB as a package which will evaluate the queue function for you but we don't have to use that package because there is another function called ERFC, complementary error function of x not which will look very similar to this, in fact you can go to MATLAB and type help.

(Refer Slide Time: 06:08)



ERFC which is what I have done here and you see ERFC is very similar to Q, you can see what it is here, it is 2 by square root of pi integral from x to infinity of this okay so this you can see sort of similar so let me write it on the other page and we will see how to manipulate one to get the other it is quite straight forward I will show how to do that okay.

(Refer Slide Time: 06:38)



So this function is 2 by square root of pi integral x not to infinity e power minus x squared dx. So if you stare at these two very closely it turns out q of x not is 0.5 times erfc of x not by square root of 2 okay, so you can check that so how to prove this, so you will have to do a substitution here of x equals x by root 2 okay if you do that this will become integral x not by root 2 to infinity, 2 by root pi e power minus you will get the x squared by 2 that you want, this will be 1 by root 2 okay.

So this 1 by root 2 will cancel here and you will get a root 2 but you need a 1 by 2 root 2 so you need a half so this is a formula that I am not going to give you more details of, please check this if you this substitution here you will get this equality okay so 0.5 erfc x not by root 2 is Q of x not okay. So this is good thing for you to remember I think I had put this up in the previous slide as well, the nice thing is ERFC is natively supported in most platforms, you will have a function for it and you can write a quick conversion to Q okay. So you can keep this in mind, so once we have this we are ready to write our little price of program to generate the curve.