**ESE-3014 Lab 4 - The receiver of a PCM system**

**Theory**

GNU Octave is a high-level language, primarily intended for numerical com-putations. It provides a convenient command line interface for solving linearand nonlinear problems numerically, and for performing other numerical exper-iments using a language that is mostly compatible with Matlab. It may also beused as a batch-oriented language.Octave has extensive tools for solving common numerical linear algebra prob-lems, finding the roots of nonlinear equations, integrating ordinary functions,manipulating polynomials, and integrating ordinary differential and differential-algebraic equations. It is easily extensible and customizable via user-defined functions written in Octave’s own language, or using dynamically loaded mod-ules written in C++, C, Fortran, or other languages.

**Task1.**

Simulate all operations performed in the regeneration circuit in of the PCMsystem receiver include amplifier equalizer, timing circuit, and decision making device. You should add appropriate noise in distorted signal, and calculate the bit error rate at the end of receiver.

**SUBMITTED BY : DATE: 23/10/2019**

**FRANCIS VIJITH - C0742980**

**VIJAYAPRAKASH V I - C0744450**

>> clear all

--Plotting the PCM distorted signal--

>> N = 1000; %setting the sampling range

>> noise = 0.5; %setting the noise factor

>> noise1 = noise\*randn(1,N); %noise signal

>> t = 5\*pi\*linspace(0,1,1000);

>> y = sin(pi\*t) + sin(2\*pi\*t) + cos(4\*pi\*t) + sin(3\*pi\*t) + cos(2\*pi\*t) + cos(5\*pi\*t) + sin(4\*pi\*t); >> %input signal without noise

>> abssig = abs(y); %absolute value of the signal to generate a PCM signal

>> sig = abssig + noise1; %adding the noise

>> plot(sig); %plotting the signal with noise

>> title(‘input signal’);

--Amplifying the signal--

>> ampvalue = 2; %aamplifying factor

>> ampsig = sig \* ampvalue; %amplifying the signal

>> plot(ampsig); %plotting the amplified signal

>> line ([2.5 1000], [2.5 2.5], "linestyle", "-", "color", "r"); %plotting the threshold value

>> title('amplified signal & threshold = 2.5');

--Timing circuit--

>> n = 1:1000;

>> s = [ones(1,1000)];

>> stem(n,s); %plotting the pulse train

>> title(‘pulse train’);

--Decision making device which uses **Return-to-Zero** PCM encoding method--

>> i = 1;

>> while(i <= 1000) %while loop to regenerate the signal using a decision making device

if(ampsig(1,i) > 2.5)

regsig(1,i) = 1; %returns bit 1 if the value is above the threshold value

else

regsig(1,i) = 0; %returns bit 0 if the value is below the threshold

endif;

i++;

endwhile;

>> plot(regsig);

>>title(‘output signal of decision making device’);

--to find bit error rate with threshold value = 2.5--

>> b1 = 0;

>>i =1;

>> while(i <=1000); %using while loop to determine number of bits transmitted out of 1000

if(sig(1,i) > 0)

b1++;

endif;

i++;

endwhile;

>> b1

b1 = 885 %number of transmitted bits

b2 = 0

>> i = 1;

>> while(i <=1000); %using while loop to determine number of bits received out of 1000

if(regsig(1,i) > 0)

b2++;

endif;

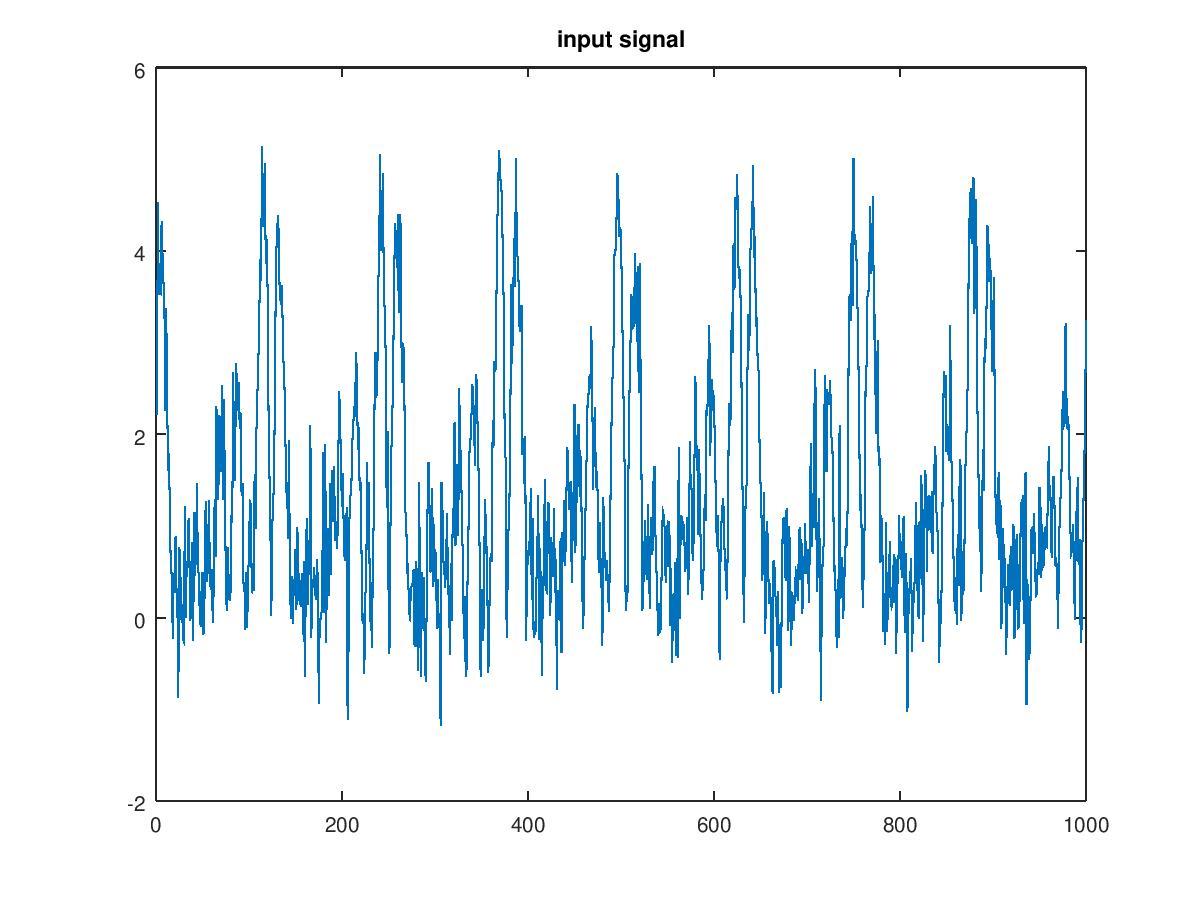
i++;

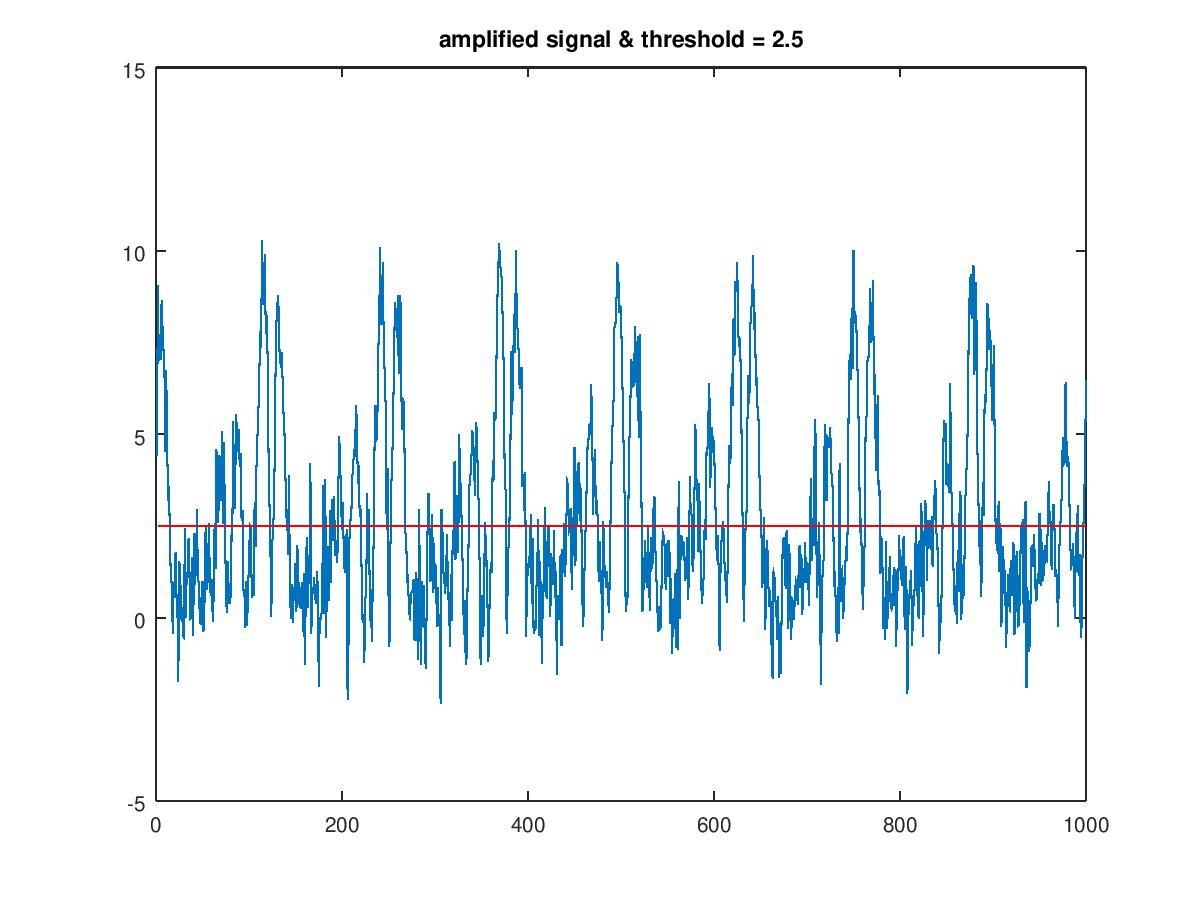
endwhile;

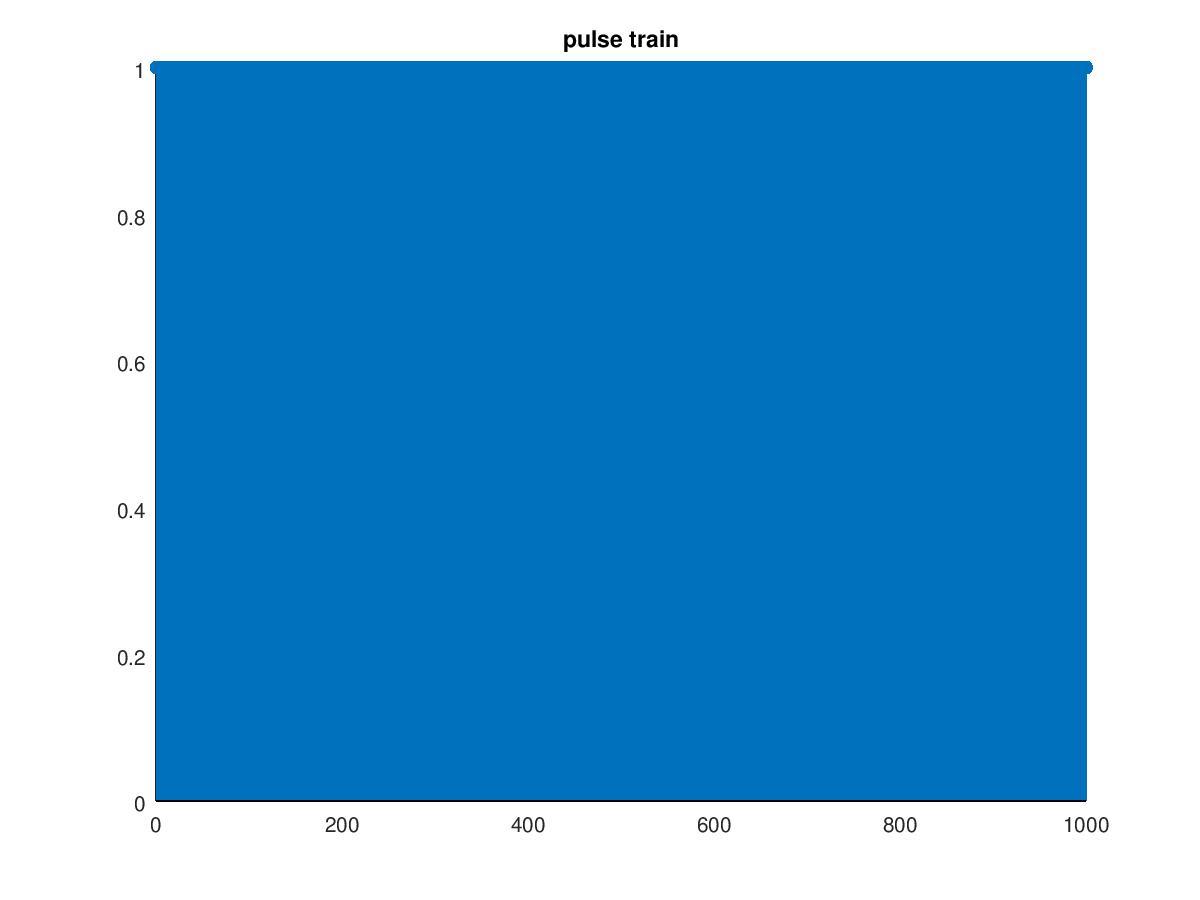
>> b2

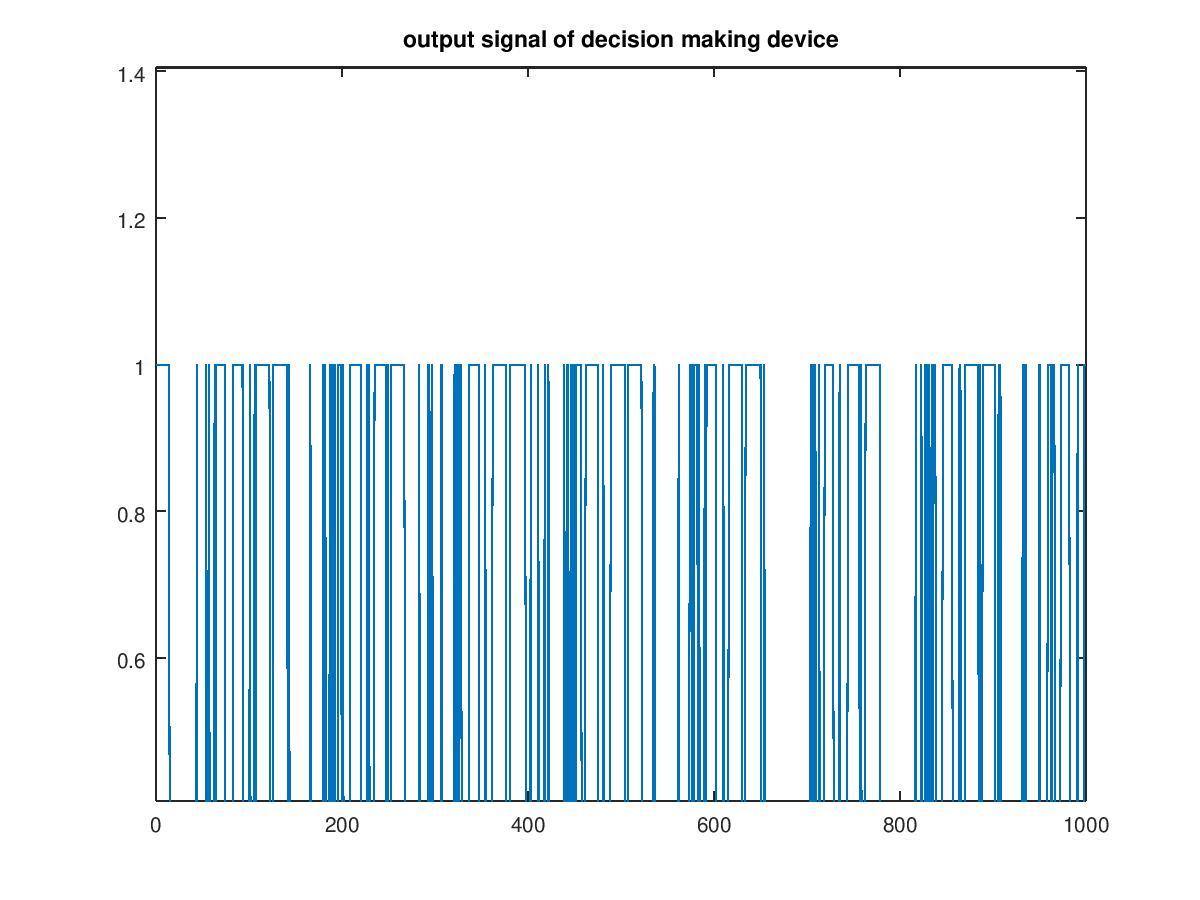
b2 = 428 %number of received bits

Bit error ratio = (b1-b2)/b1 =(885-428)/885 = 0.51638









Output signal in text form:

