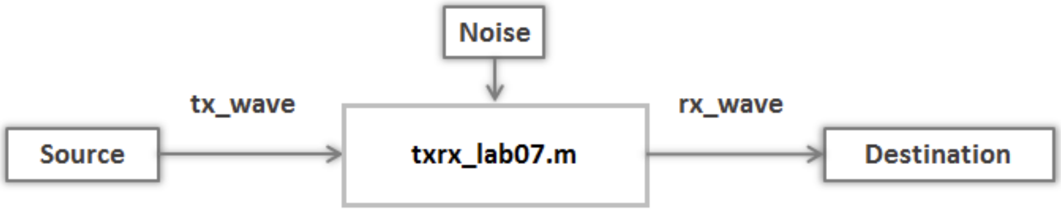
# LAB 7 TASK 1 - STATISTICAL DISTRIBUTION OF THE RECEIVED SIGNAL (1/1 point)

In this task, you will write code to visualize the histogram of the received signal when a "0" bit is transmitted (IN=0), and compare it with the histogram predicted by assuming the the noise is Gaussian distributed.



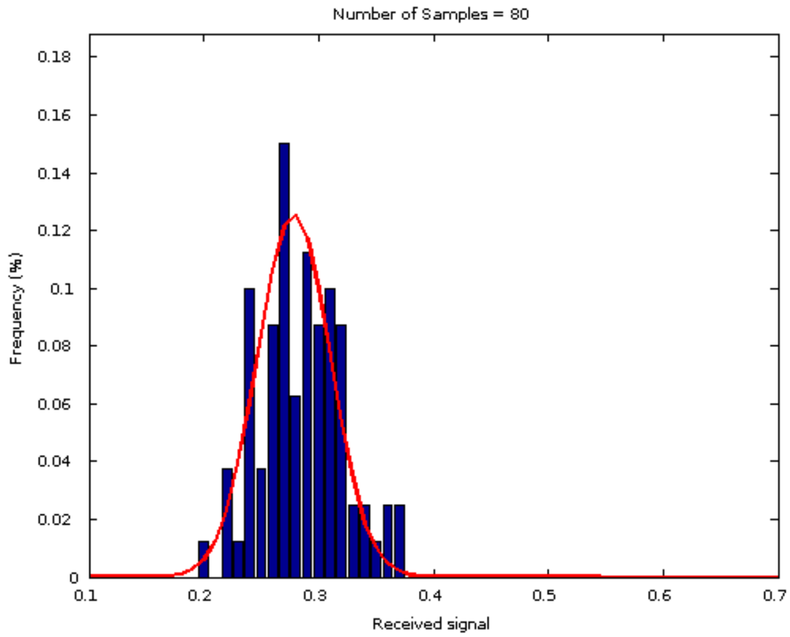

```
SPB = 50;                              % bit time in samples
tx_bs = zeros(1,1280);                 % generate the all-zero bit sequence
tx_wave = format_bitseq(tx_bs,SPB);    % create waveform following protocol

% transmit and receive over noisy channel
[rx_wave,start_ind,rx_min,rx_max,sigma] = txrx_lab07(tx_wave);
sample_ind = start_ind+2*SPB-1+SPB*[0:1279]; % set subsampling points
signal_samples = rx_wave(sample_ind);        % get the received samples

xhist = 0.1:0.01:0.7;    % centers of histogram bins

% Do not modify code above this line
nsamp = 80;   % number of samples to use for histogram
signal_samples_first80bits = signal_samples(1:80);

```

Correct

```
h = hist(signal_samples(1:nsamp),xhist)/nsamp;   % generate histogram
bar(xhist,h);
```

Figure 1

Help




Check | Reset | Save | Hide Answer

*You have used 3 of 10 submissions*

## INSTRUCTIONS

Step 1: Run the code as presented

Your job in this task is to compare the empirical histogram of the received signal when IN=0 with the theoretical distribution predicted by assuming that the noise follows a Gaussian distribution. Let's first look at how the code works.

Assuming that the received signal is the summation between the response of the channel to the transmitted signal and the additive noise, the given code transmits an all-zero sequence (1280 zeros) so that the variation in the received signal is purely due to the additive noise. The function **txrx_lab07** simulates the transmission of **tx_wave** over a noisy channel, and returns the channel output in **rx_wave**, the index of the start bit in **start_ind**. It also uses the portion of the received signal corresponding to the training sequence to compute estimates of the values of the response of the noise free channel to IN=0 and IN=1 and the standard deviation of the noise, and returns these as **rx_min**, **rx_max** and **sigma**. The receiver samples the received signal at 1280 sampling positions, stored in **sample_ind,** to obtain 1280 samples of the received signal stored in **signal_samples**.

After you hit the **Run Code** button, a plot will be generated by Matlab showing the predicted distribution of the samples. This predicted distribution is calculated from the values of **rx_min**, **rx_max** and **sigma** returned by **txrx_lab07**. Since the input is always zero, we expect the received signal to have a Gaussian distribution with mean **rx_min** and standard deviation **sigma** (variance **sigma^2**). The function **PLOT_GHIST(X,MEAN,SIGMA)** plots the theoretically predicted percentage distribution of Gaussian data with mean **MEAN** and standard deviation **SIGMA** among bins with centers specified by **X**.

Step 2: Plot the distribution of the received signal

Your code should compute the *percentage* histogram from the first **nbits**=80 signal samples. The value of the histogram in each bin should be the ratio between the number of samples in that bin and the total number of samples. This percentage should be expressed as a number between 0 and 1 (100%). We use the percentage here so that histograms computed from different numbers of samples give comparable values.

For more information on how to generate histogram in MATLAB, please review the video Making Histograms. The MATLAB function, **N = HIST(Y,X)**, where **X** is a vector, returns the distribution of **Y** among bins with centers specified by **X**. The first bin includes data between -infinity and the first center and the last bin includes data between the last center and infinity. Note that the **hist** function returns the total number of samples in each bin, not the percentage.

Help

Plot the histogram as a bar chart on the same plot as the theoretically predicted histogram to enable easy comparison. The MATLAB command, **BAR(X,Y)**, plots the values of **Y** versus **X** as a bar chart.

Put your code under the comment starting with % Place your code below this line... Pay attention to the four specifications listed in the code window.

Step 3: Submit your work

Once you have completed your work, click on the **Check** button to submit your answer.