Department of Electrical and Electronic Engineering
Summer 21

EEE 344/ECE344

Digital Signal Processing Laboratory

Section: 01

Open Ended Lab Project

Group-8

Group members:

1	19121141	Lazib Sharar Shaiok
2	19121133	Fiaz Sadid
3	19121101	Sahha Munzer
4	19121126	Quazi Rian Hasnaine

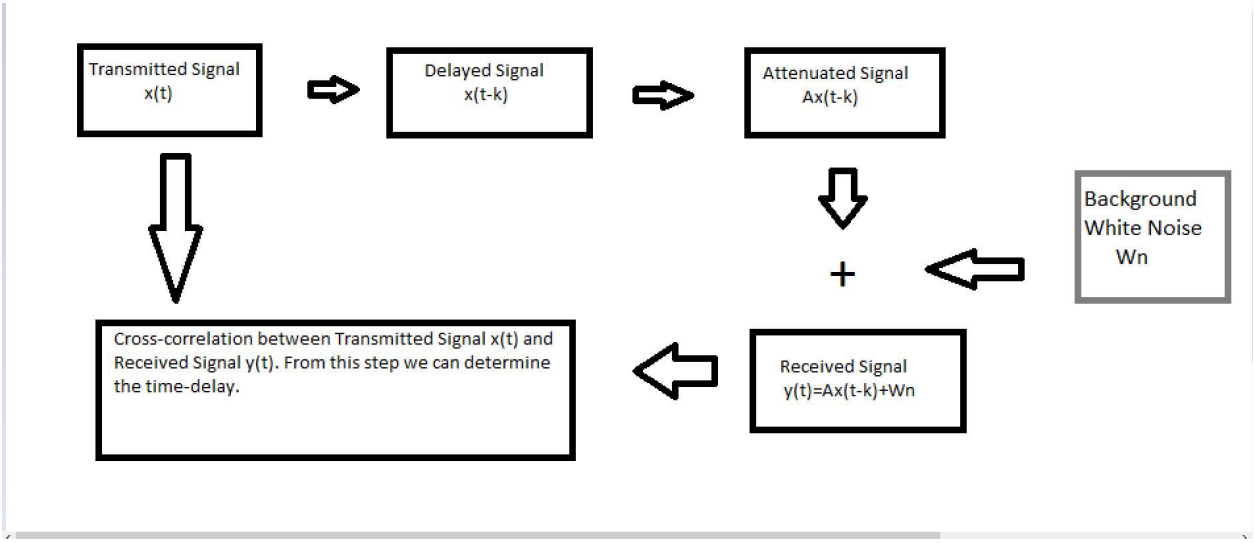
Date of Submission: 16th September 2021

Objective:

In this open ended lab, we were asked to model the behaviour of a radar system which sends out a signal and then receives a attenuated,delayed and noisy signal on MATLAB.Despite the signal being so altered upon receiving the delay was asked to be found out for different attenuation and SNR Levels.

Methodology:

Block Diagram :



Theoretical Calculation:

Velocity of Sound Wave in Air= 330m/s

Distance between radar and target=165km

We have to realize that the actual distance travelled by the signal as it travels to and fro between the radar and target is $2 \times 165 = 330$km.

we know that $t = \frac{d}{v}$

Where

t=time (s)

d= distance (m)

v=velocity (m/s)

So

$t = \frac{330 \times 1000}{330} = 1000$ s

Thus the time taken for the signal to travel to and fro between the radar and the target is 1000s

Work Flow:

Please note that the code has been explained line by line using comments in the appendix section. This Work Flow section mainly aims to highlight some of the key areas of the code and its surrounding theory.

In the challenges and troubleshooting section more information about the code can be found.

- First we had to set up the transmitted signal (x). The unique characteristic of the transmitted signal (x) is that it was a sinusoid with a period of T=150s and ran for only two pulses (from 0 to 300 s). Yet, the entire graph ran from 0 to 1000 secs.
  To plot this graph we utilized the fact that two 1D vectors(a,b) can be appended in MATLAB
  to make another 1D vector (c) using the syntax c=[a b].
  So, a sinusoid (x1) which had period of 150s was generated for 0<t<300
  Another sinusoid (x2) with 0 amplitude was generated for 300<t<1000
  Then the two were appended to one another to generate the transmitted signal.
  i.e x=[x1 x2]

- Then the received signal(y) needed to be generated.

  The formula for the received signal is
  received signal (y)= attenuated and delayed signal (Ax(t+k)) + noise
  where,
  k=time delay
  A= Attenuation factor (the gain of the received signal)
  Noise is set based on SNR

  Now let us see how each of the parameters were determined and eventually the received signal was generated.

  k was simply calculated using MATLAB arithmetic operations.

A was simply a MATLAB variable.

Noise signal determination is as follows:

SNR formula= $10\,log10^{\frac{SIGNAL\ POWER}{NOISE\ POWER}}$

The Signal Power was determined by squaring each element of the signal(attenuated and delayed signal) then dividing it number of elements in the signal.

This is in sync with what we learned in theory class:

Power of periodic signal = $\frac{1}{T}\int_0^T x(t)^2\,dt$

Now for a particular SNR we can determine the Noise Power.

Then the noise signal is generated using: noise=sqrt(NP)*randn(1,length(ax))

Now the randn function generates a random sequence of numbers with 0 variance.

rand(1,length(ax)) thus generates a 1D array of random numbers with 0 variance.

Then each of these elements in this array is multiplied with a the square root of our noise power to generate the white background noise signal.

Finally the received signal (y)= attenuated and delayed signal (Ax(t+k)) + noise

- The correlation between the transmitted signal (x) and received signal (y) can be used to determine the time delay between the two.

  Cross correlation is a powerful mathematical tool that is used to track the movements of two or more time dependent data relative to one another.Thus it can also be used to compare how similar the data sets are to each other and at what point the data sets are the most similar.

  It just so happens that for an original signal and slightly delayed version of the same signal, that point,the point where they are the most similar to each other, is the delay time.

  MATLAB has its own cross correlation function xcorr(). So we used this function instead of the convolution method. In xcorr(), two arguments were inputted received signal(y) and transmitted signal(x). The indices for cross correlation rxy(t) is 1/150 increments

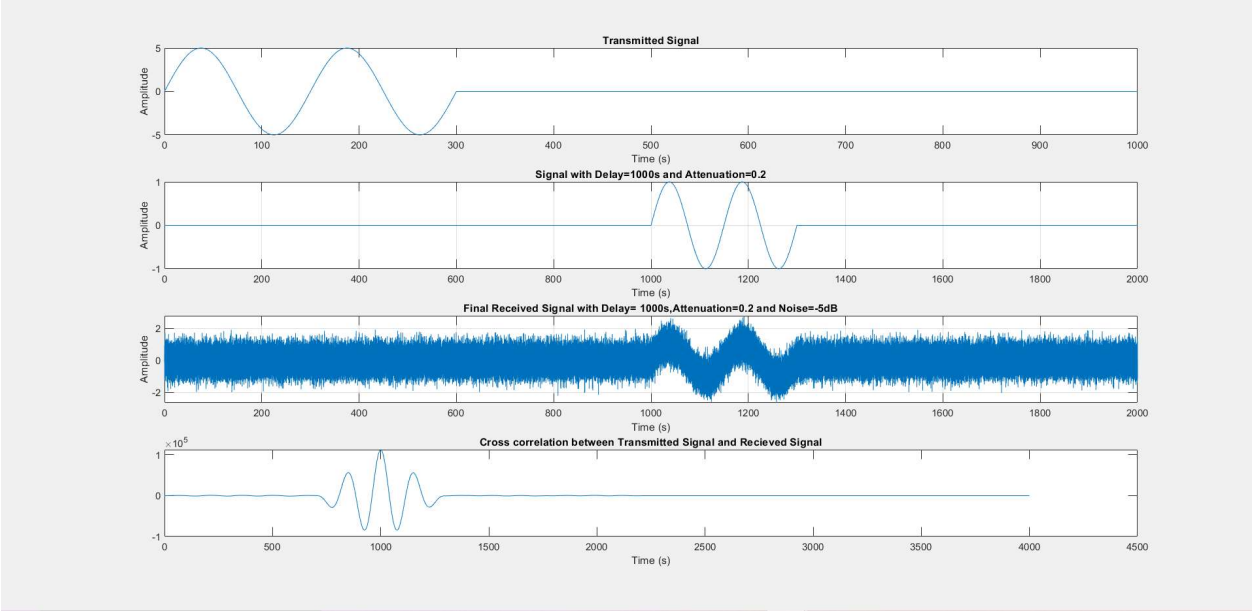
from 1 to the number of elements in rxy(t). This method for determining the indices was given in the lab sheet of Experiment-5 and also shown in the corresponding bux video.

- To show all the graphs, the entire grid was divided into 4 rows and 1 column using the subplot() function. Each row held one graph. plot() function was utilized as all graphs were continuous time graphs.Extra caution was taken to ensure that the y-axis and x-axis had the same number of elements or else plot() function wouldn't work.

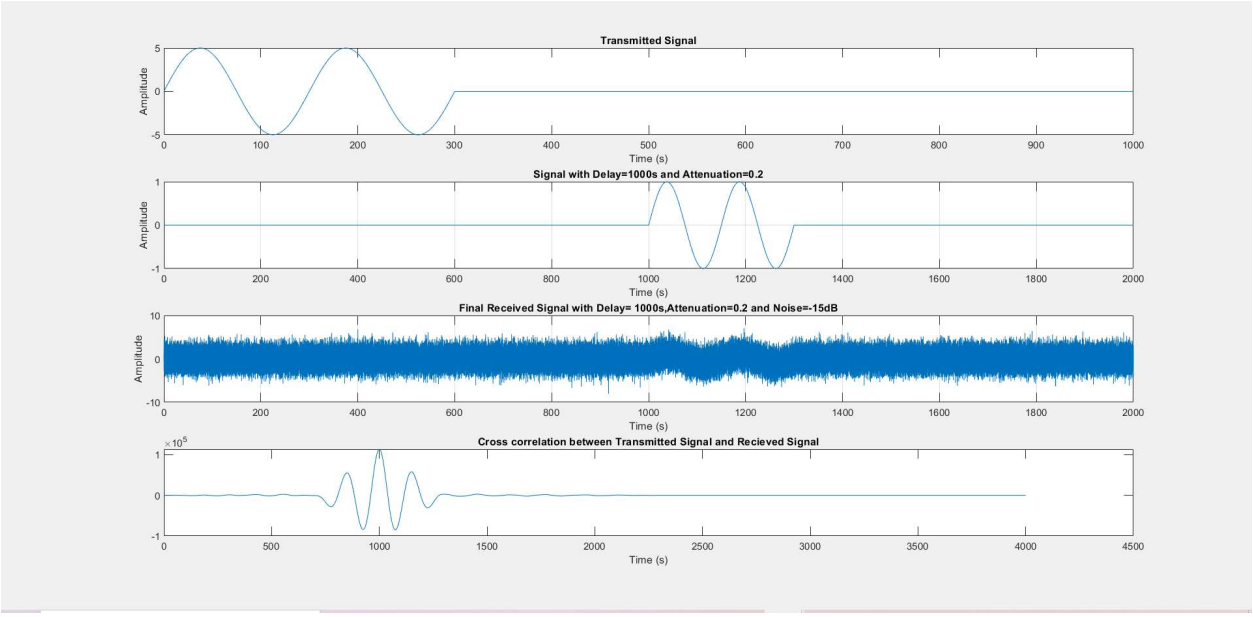
Results:

Attenuation factor (A=0.2):

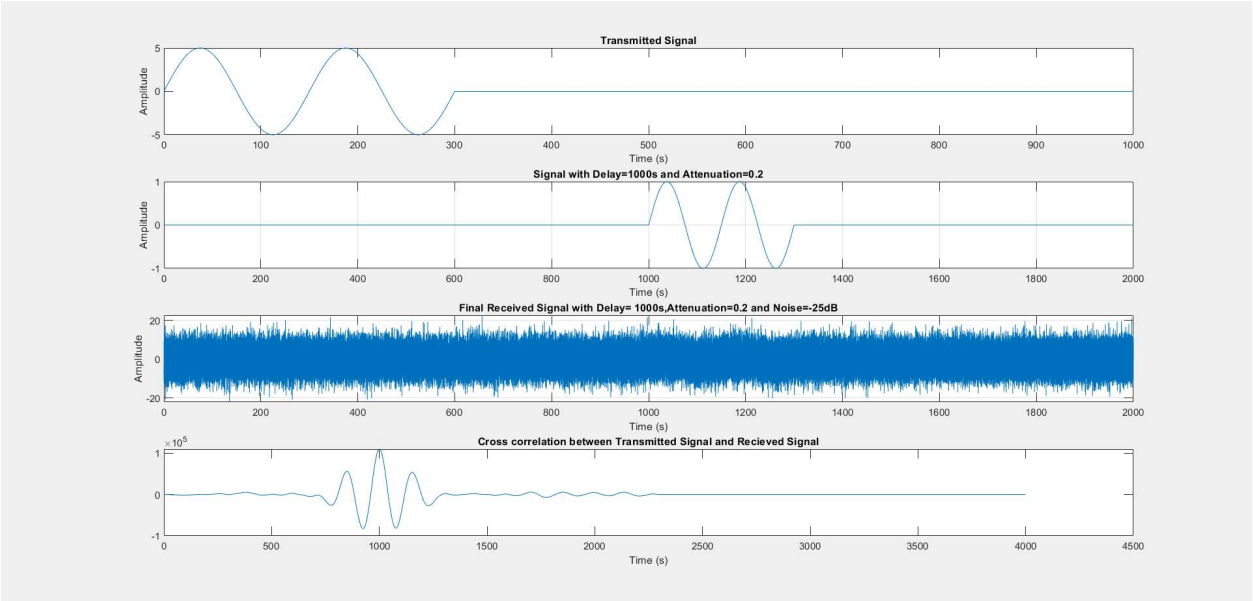
SNR=-5dB:



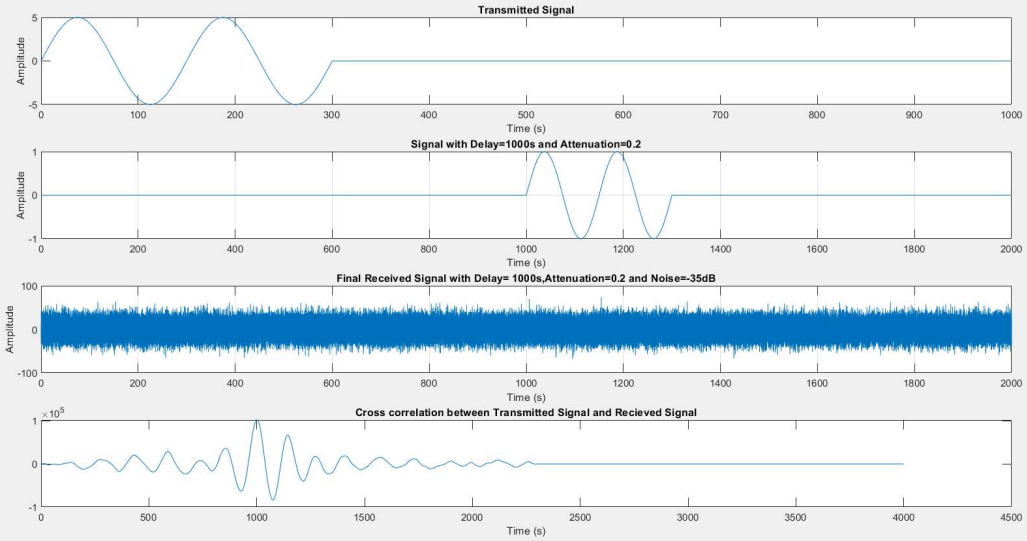
SNR=-15dB:



SNR=-25dB:

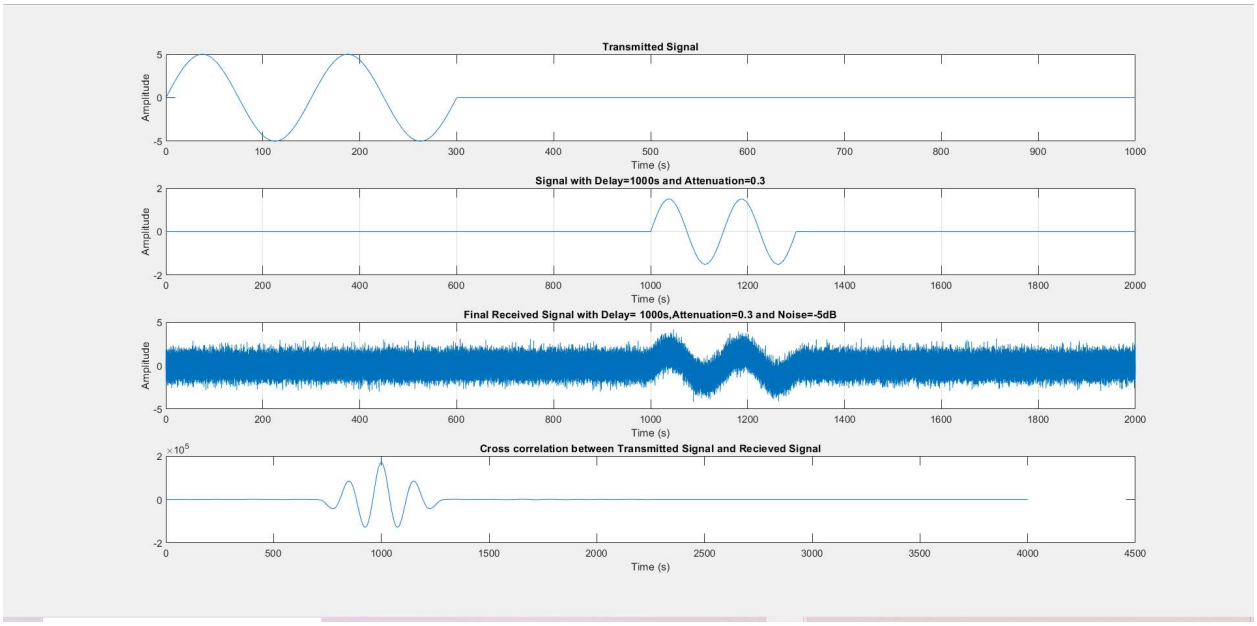


SNR=-35dB:

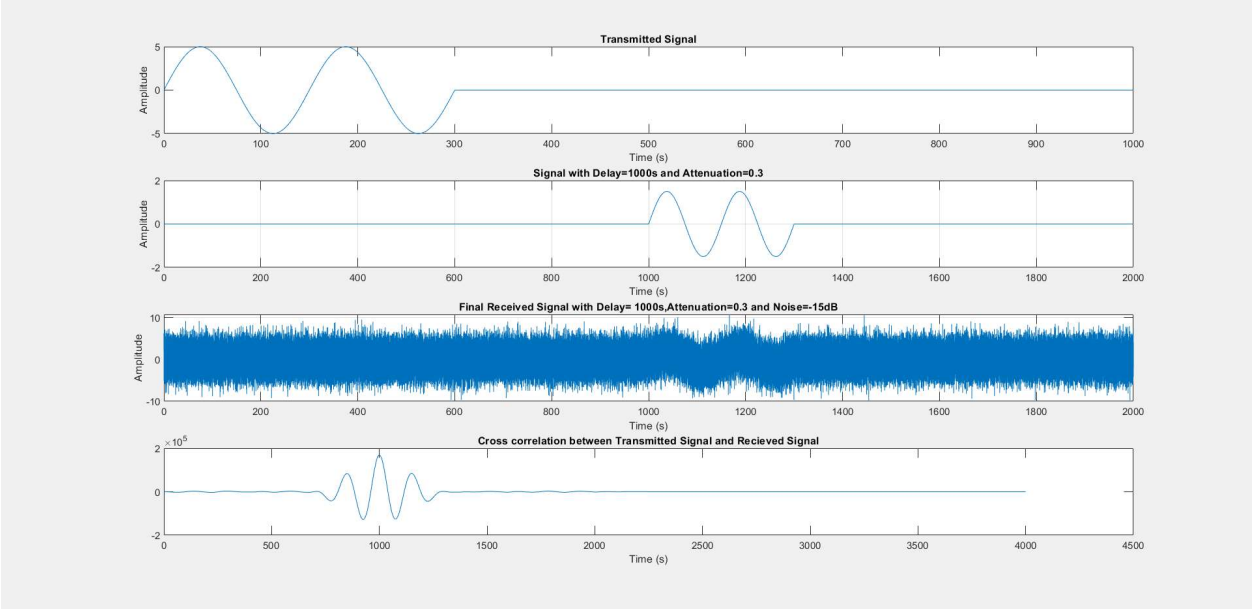

Transmitted Signal
Amplitude
Time (s)
Signal with Delay=1000s and Attenuation=0.2
Amplitude
Time (s)
Final Received Signal with Delay= 1000s,Attenuation=0.2 and Noise=-35dB
Amplitude
Time (s)
Cross correlation between Transmitted Signal and Recieved Signal
×10^5
Time (s)

Attenuation factor (A=0.3):

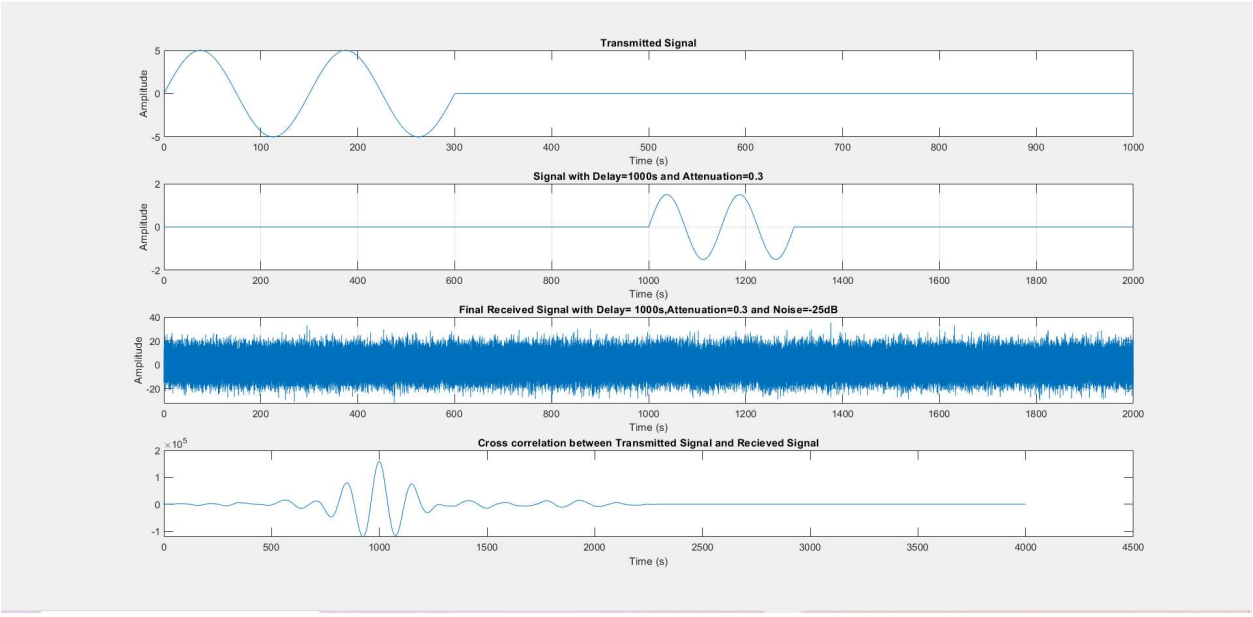
SNR=-5dB:



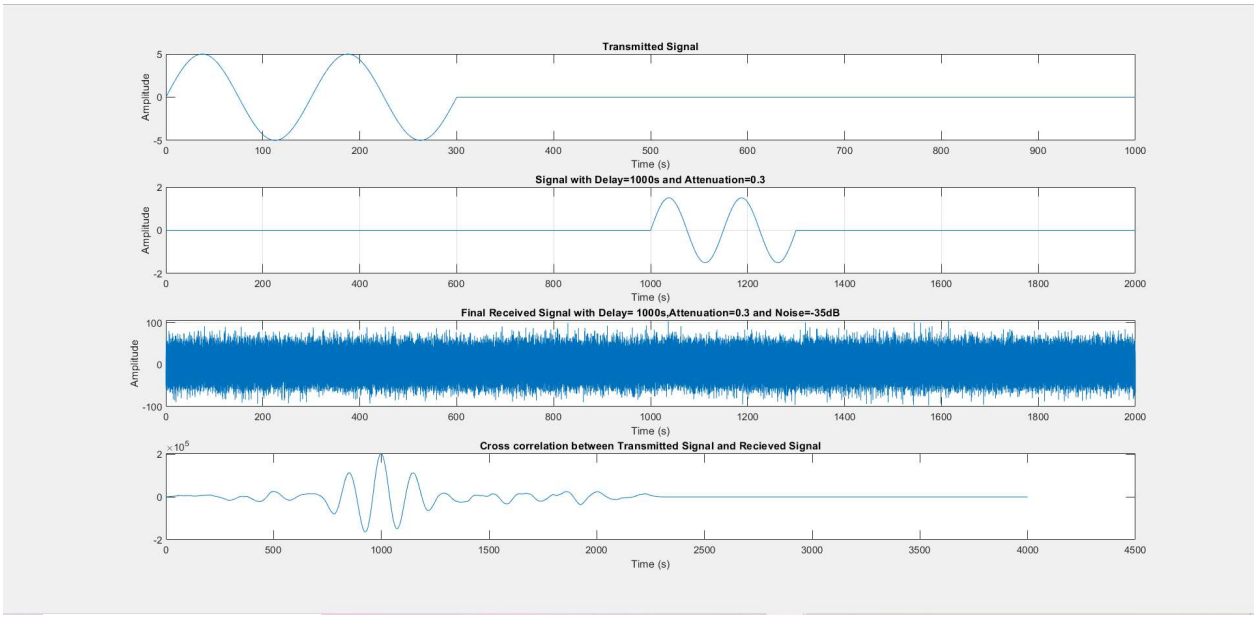
SNR=-15dB:



SNR=-25dB:

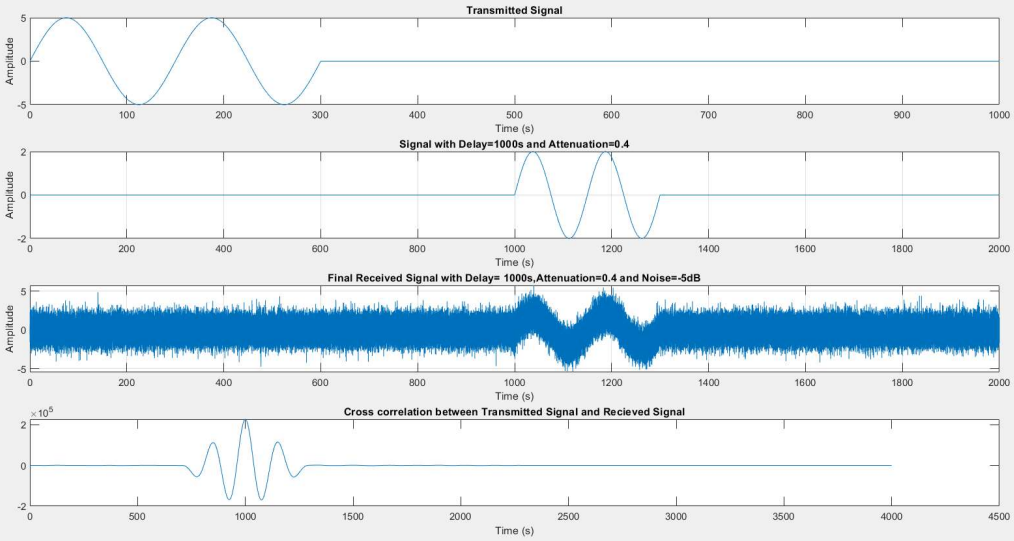


SNR=-35dB:

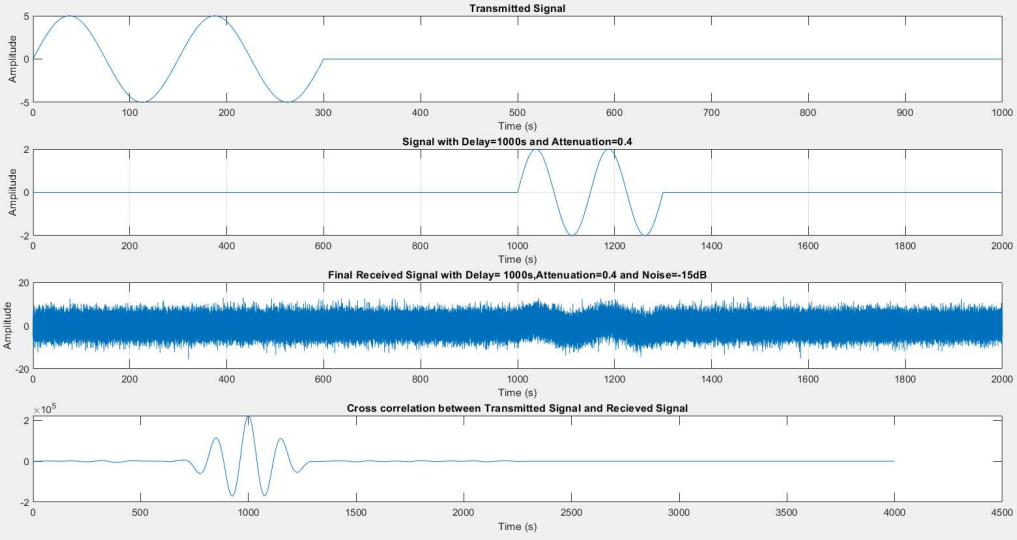


Attenuation factor (A=0.4):

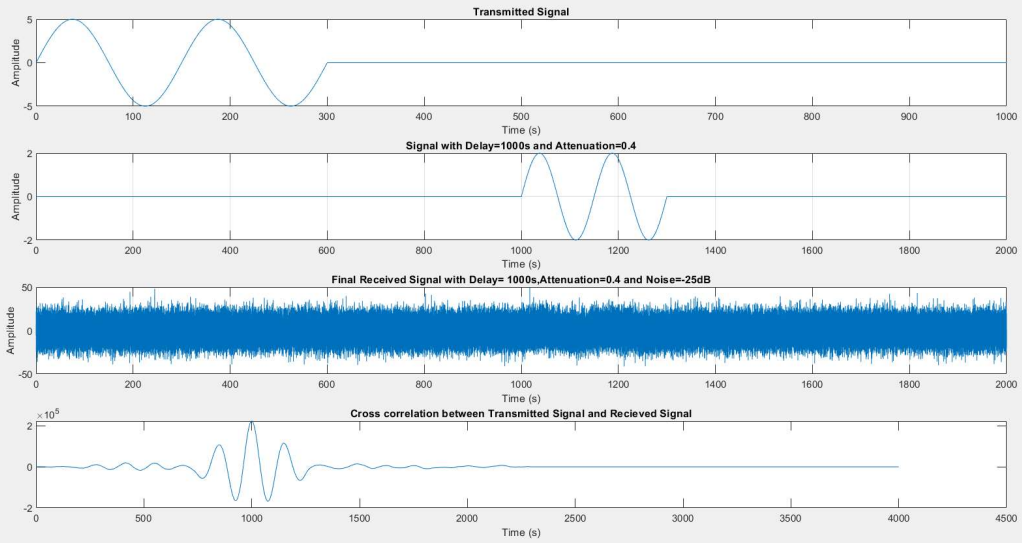
SNR=-5dB:



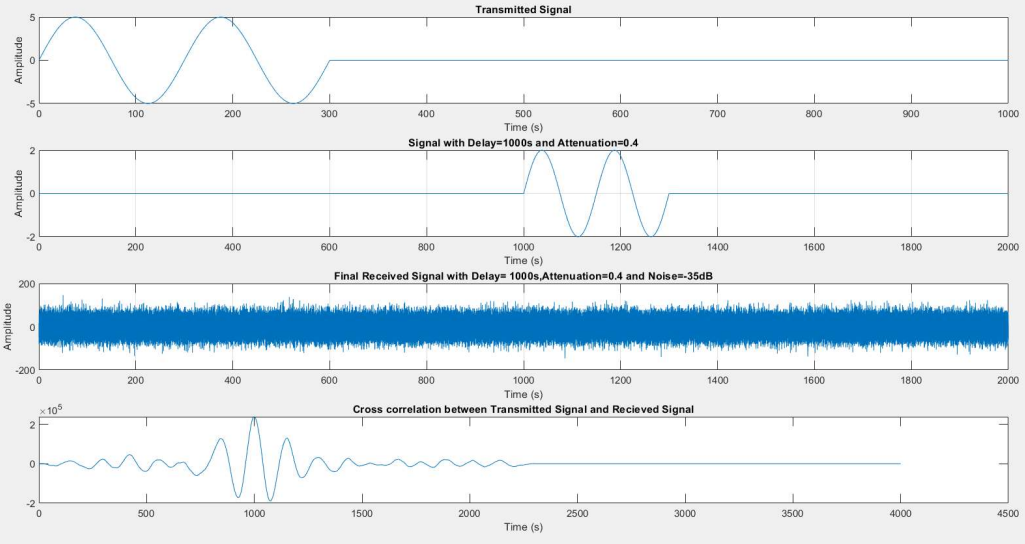
SNR=-15dB:



SNR=-25dB:

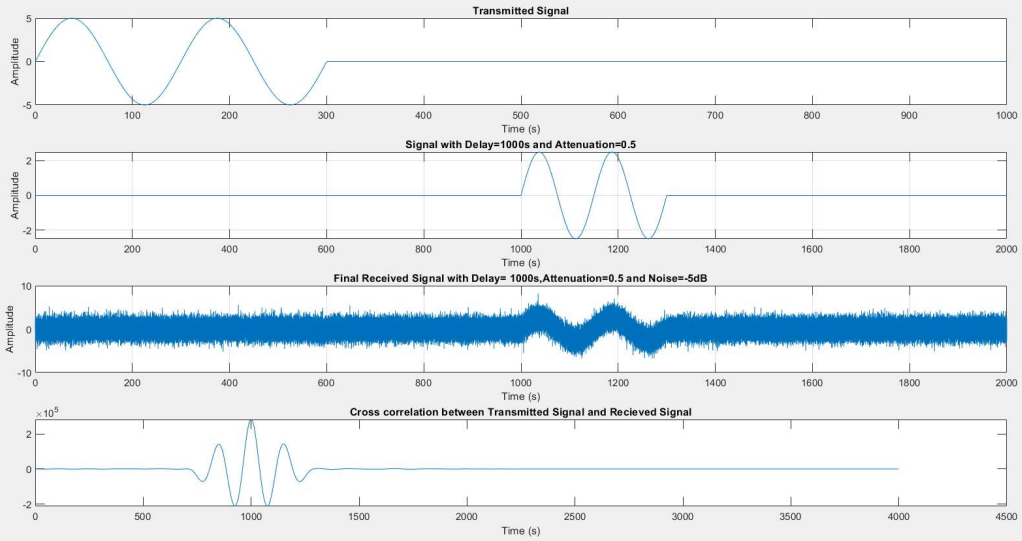


SNR=-35dB:

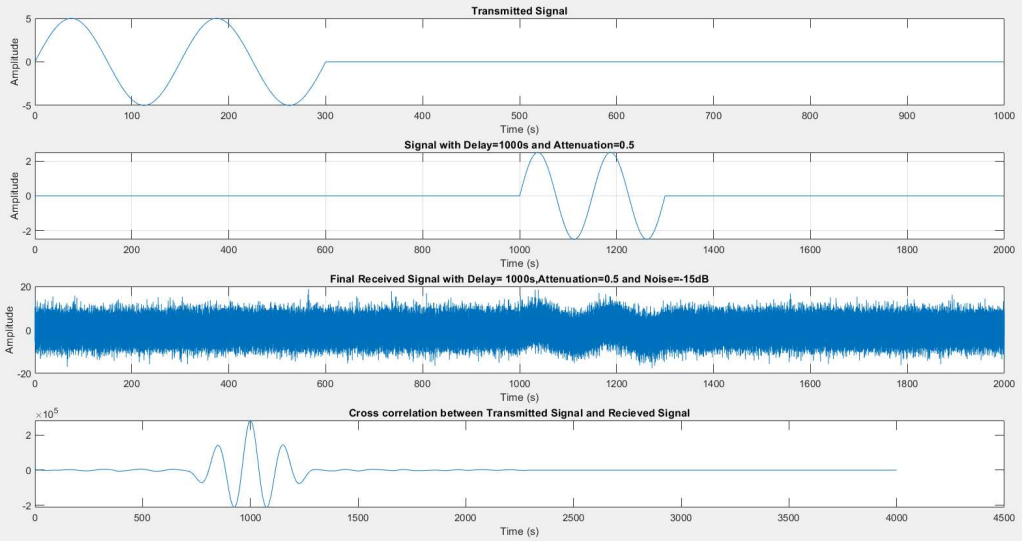


Attenuation factor (A=0.5):

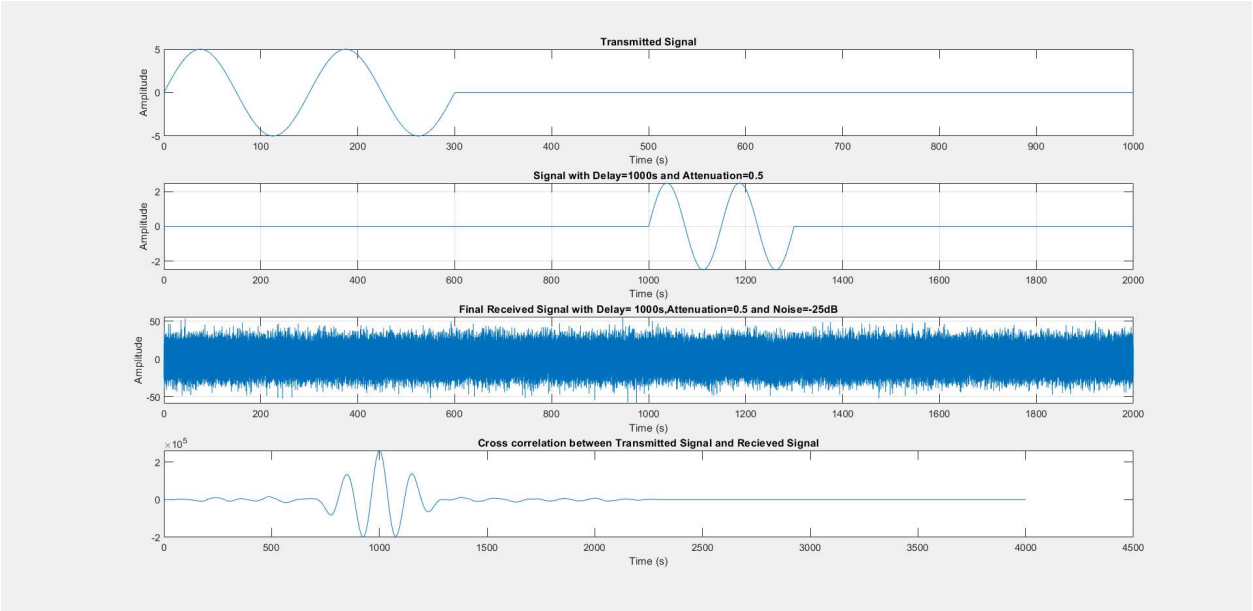
SNR=-5dB:



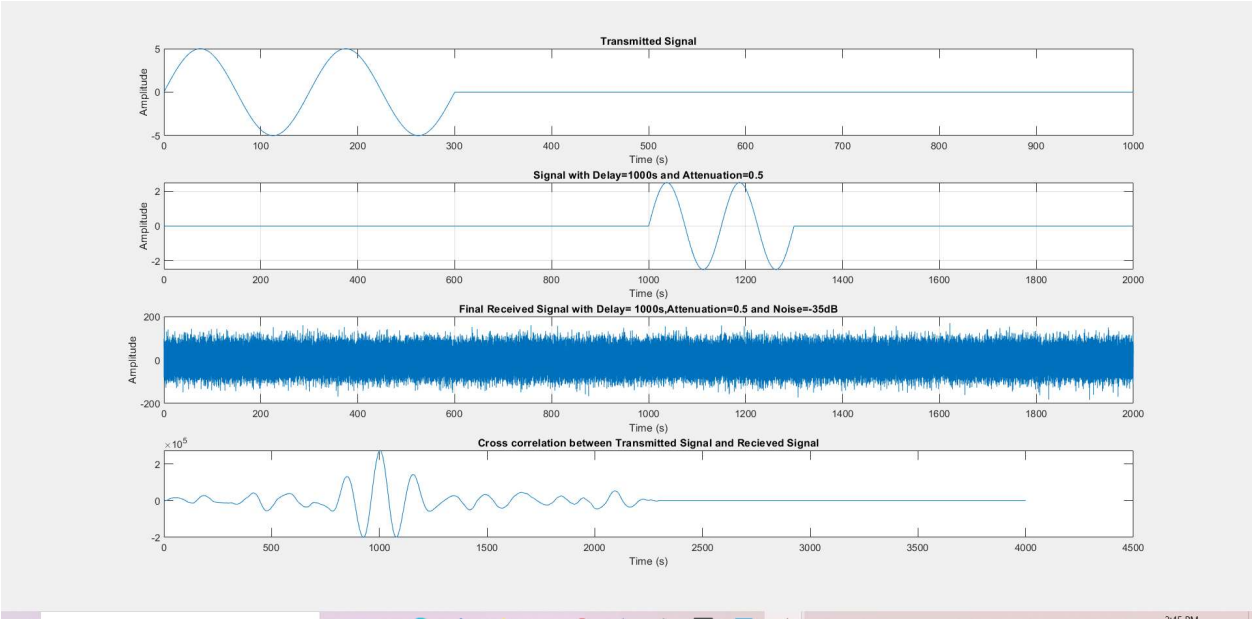
SNR=-15dB:



SNR=-25dB:

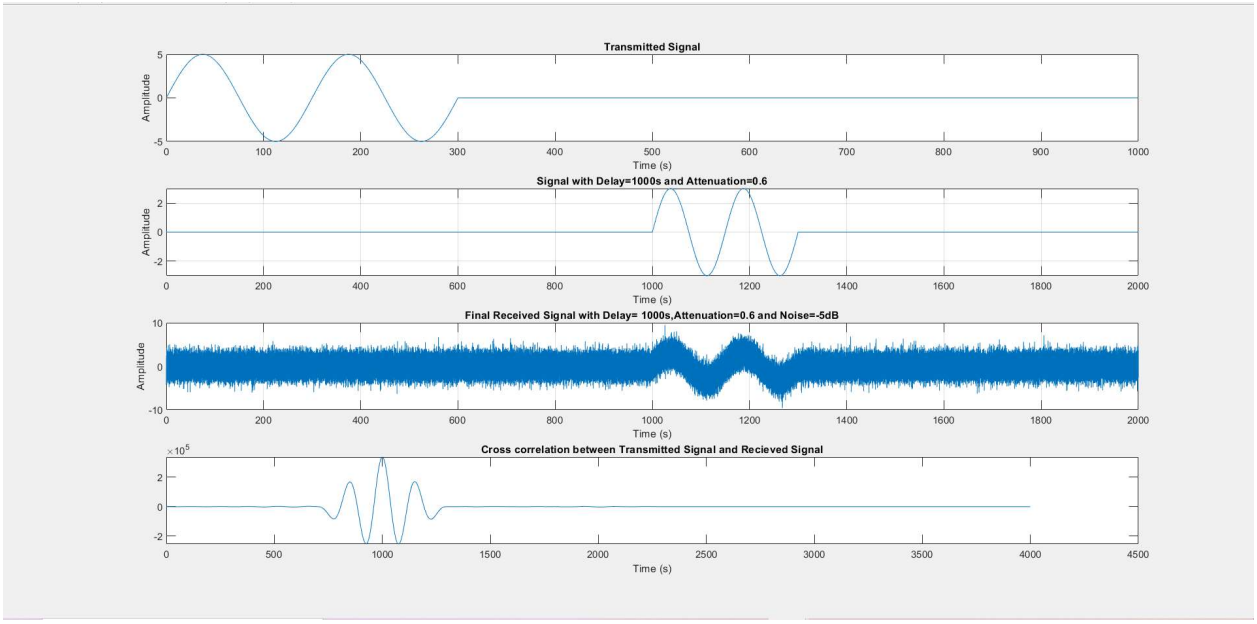


SNR=-35dB:

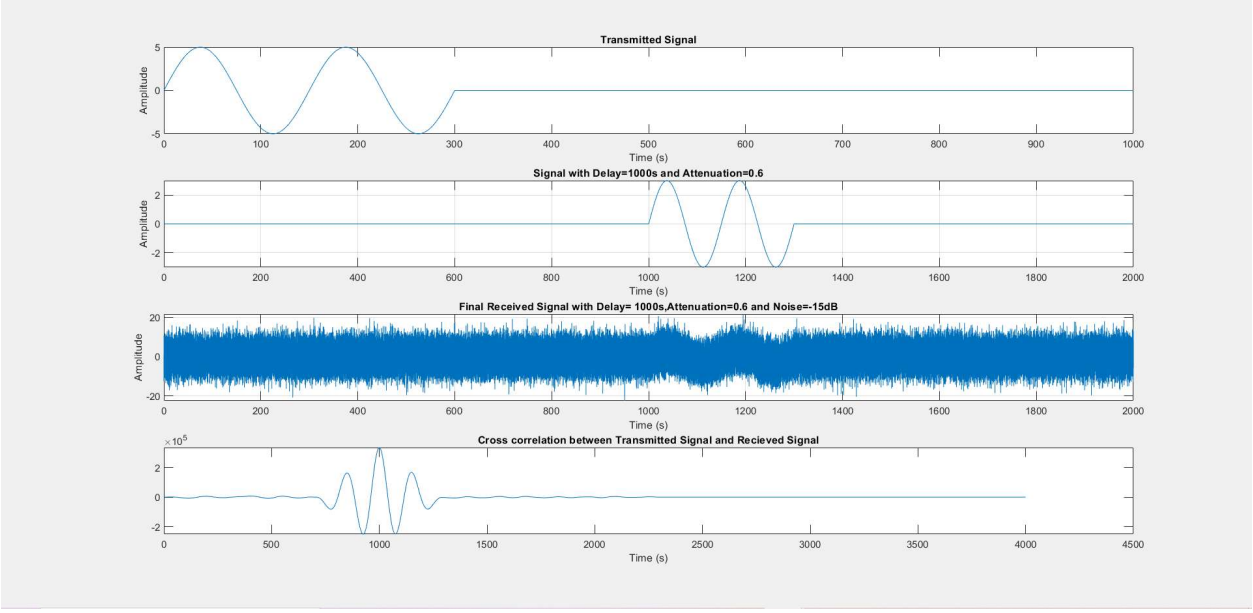


Attenuation factor (A=0.6):

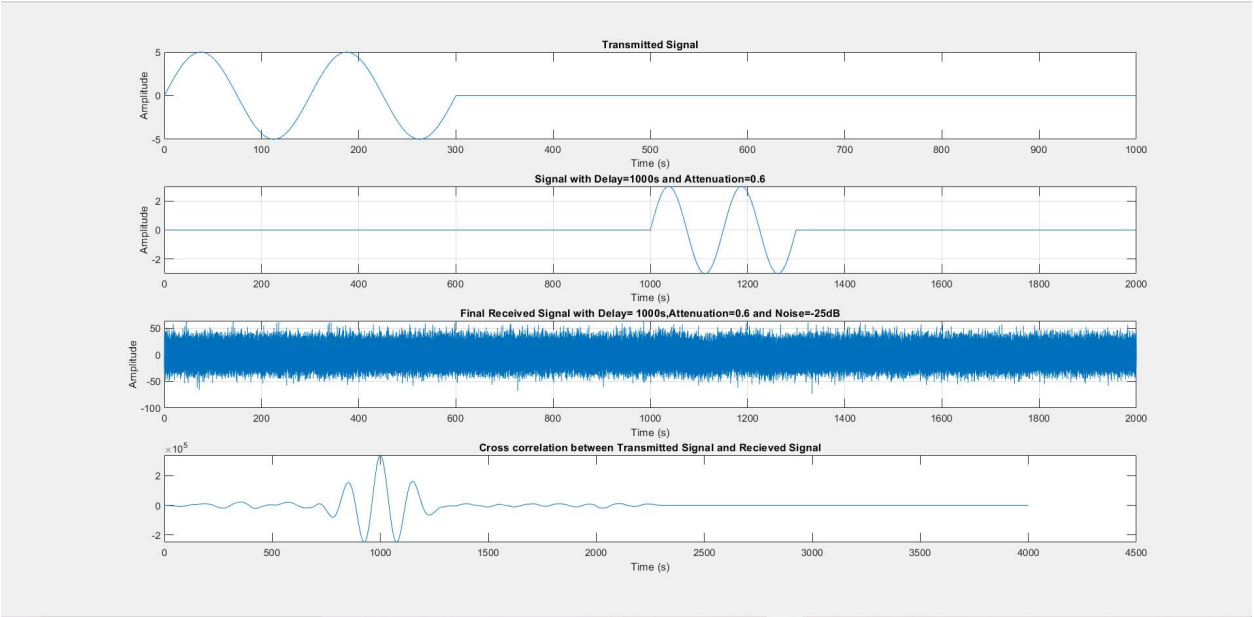
SNR=-5dB:



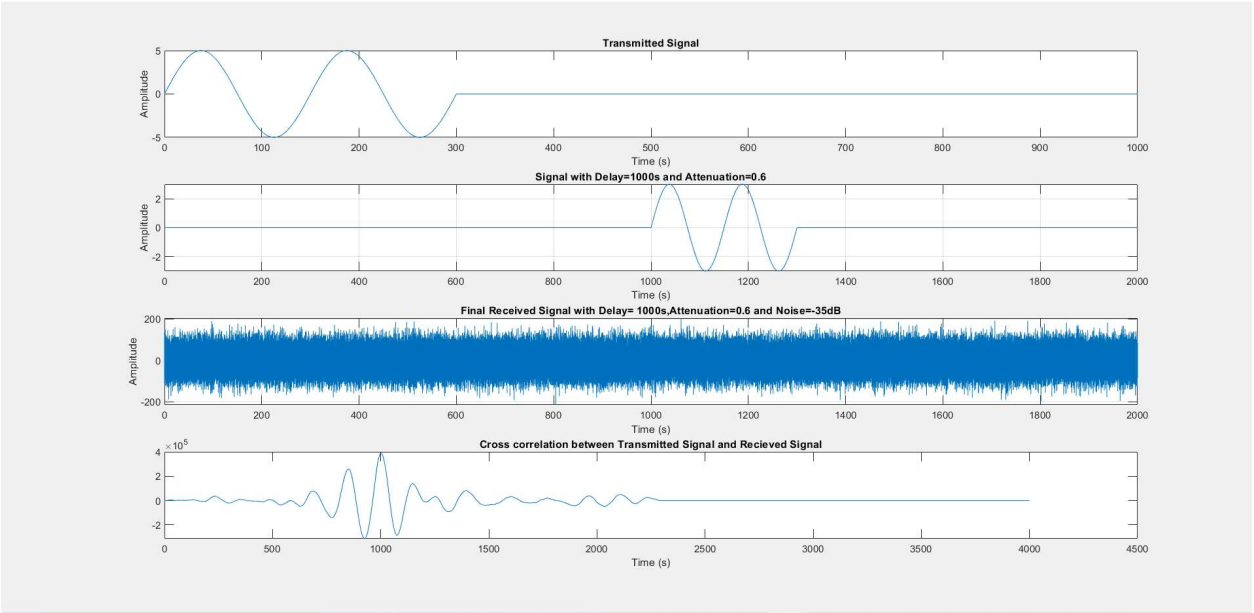
SNR=-15dB:



SNR=-25dB:

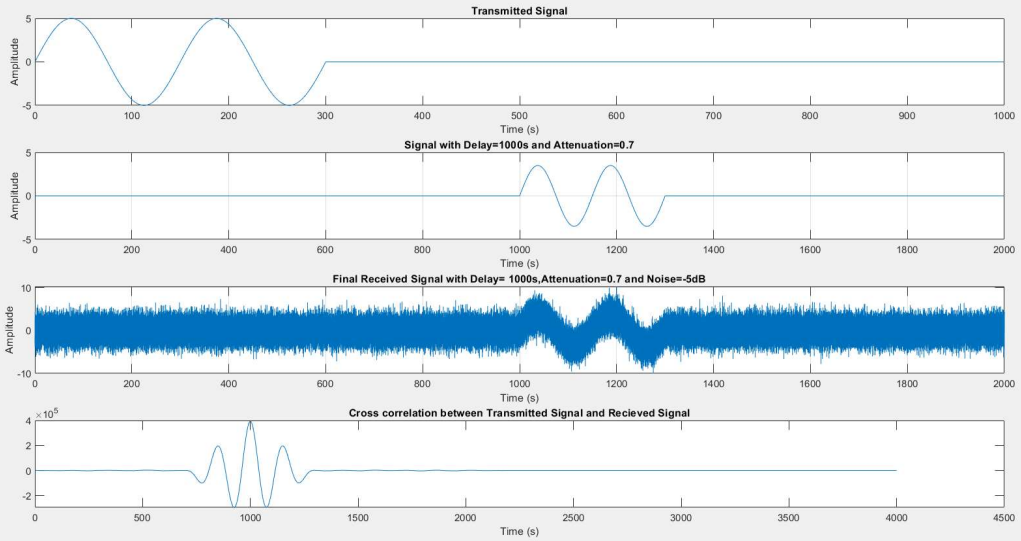


SNR=-35dB:

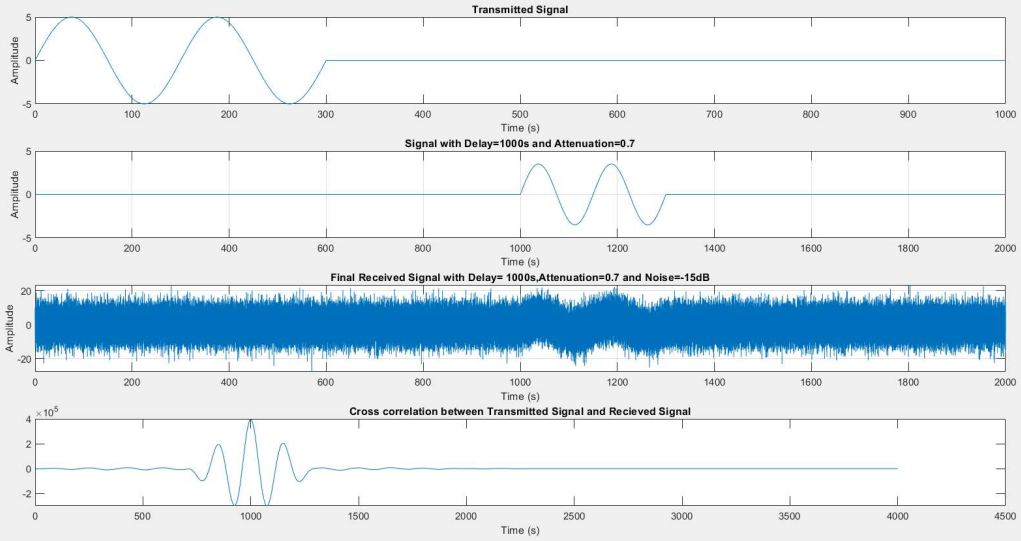


Attenuation factor (A=0.7):

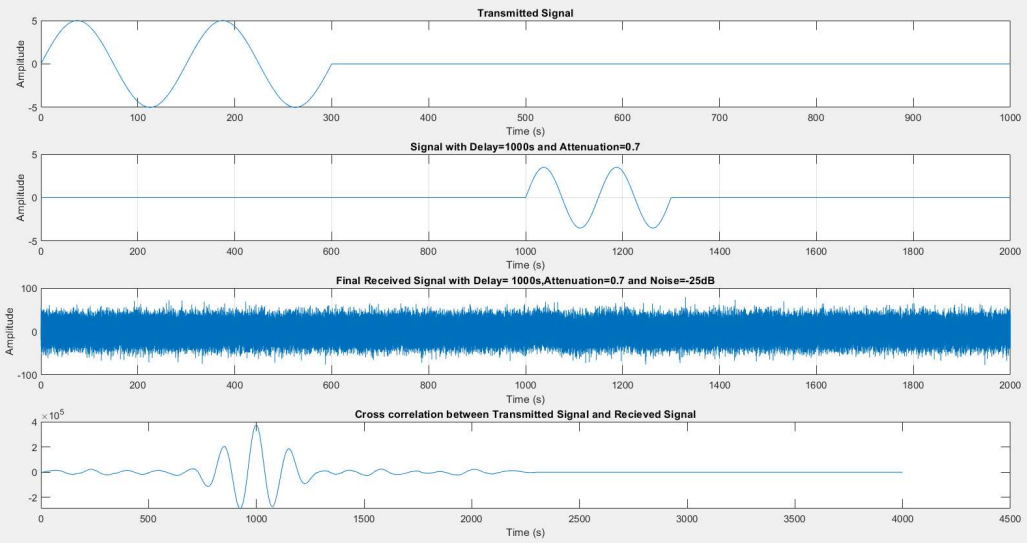
SNR=-5dB:



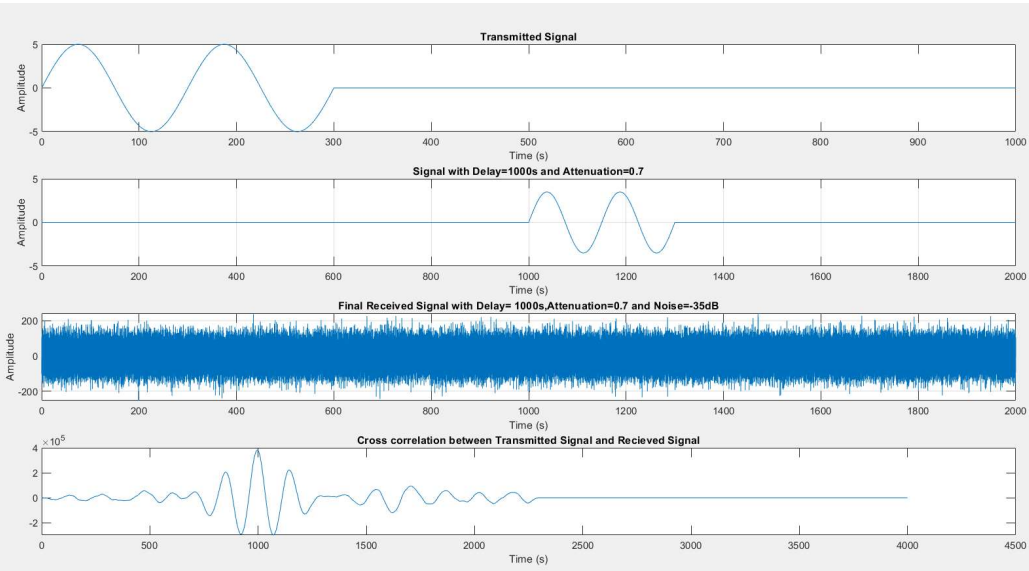
SNR=-15dB:



SNR=-25dB:

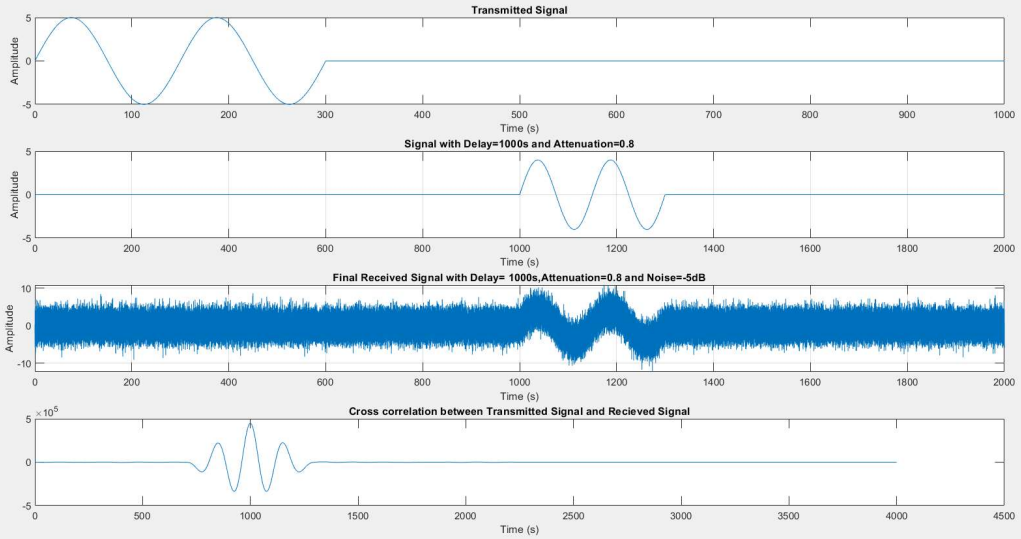


SNR=-35dB:

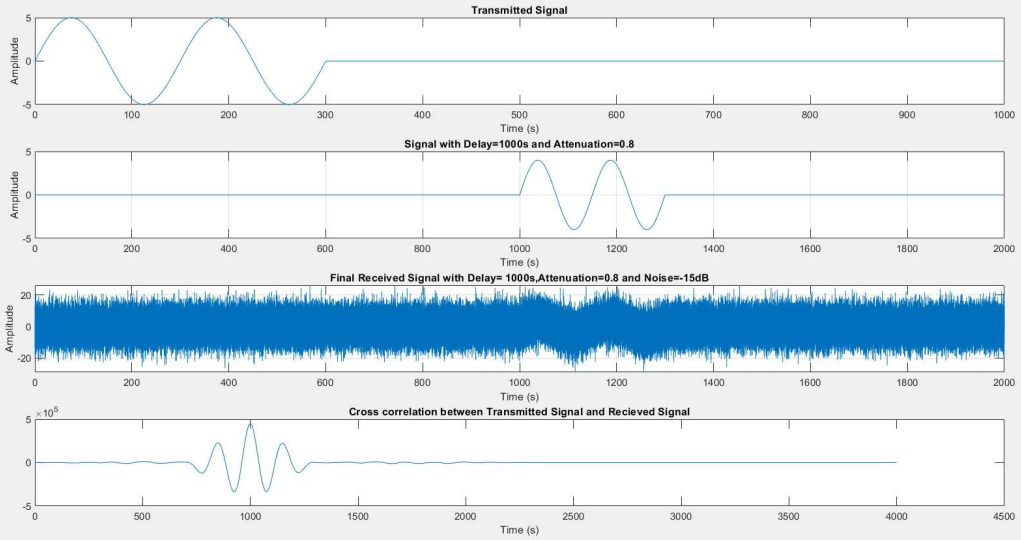


Attenuation factor (A=0.8):

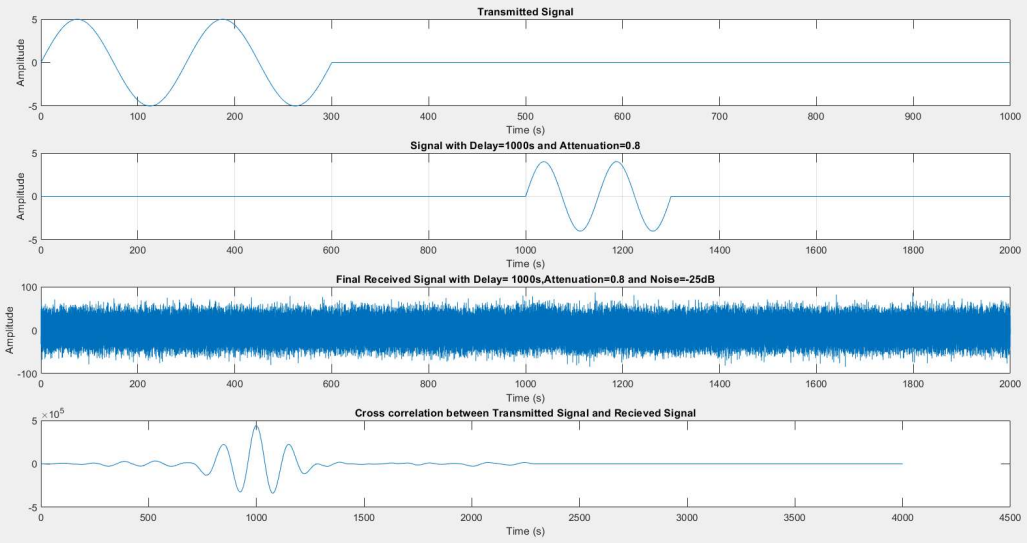
SNR=-5dB:



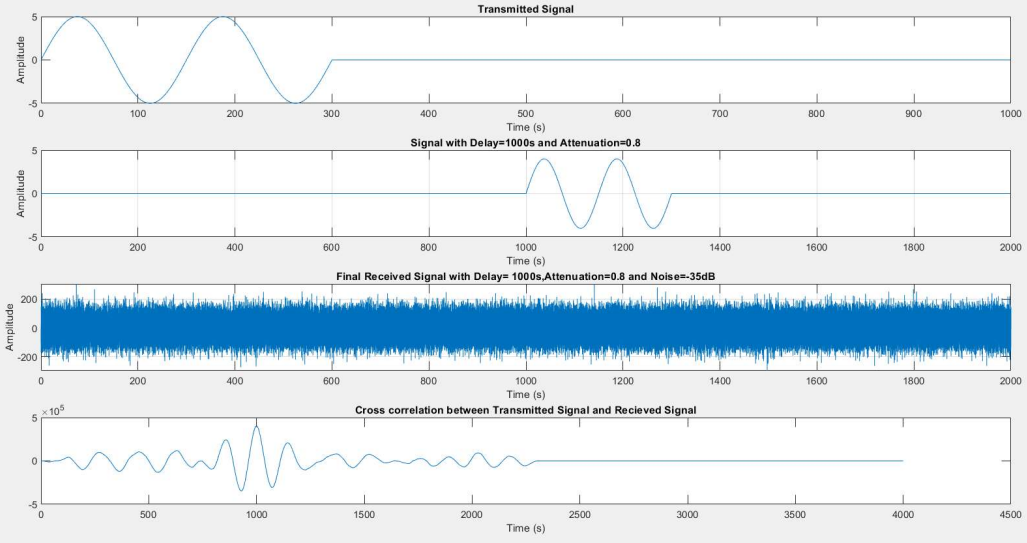
SNR=-15dB:



SNR=-25dB:

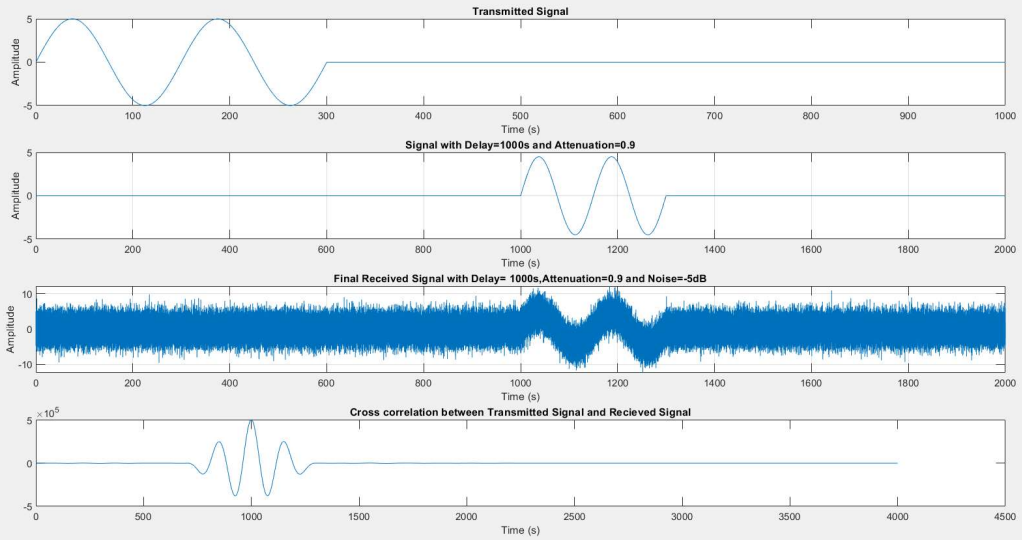


SNR=-35dB:

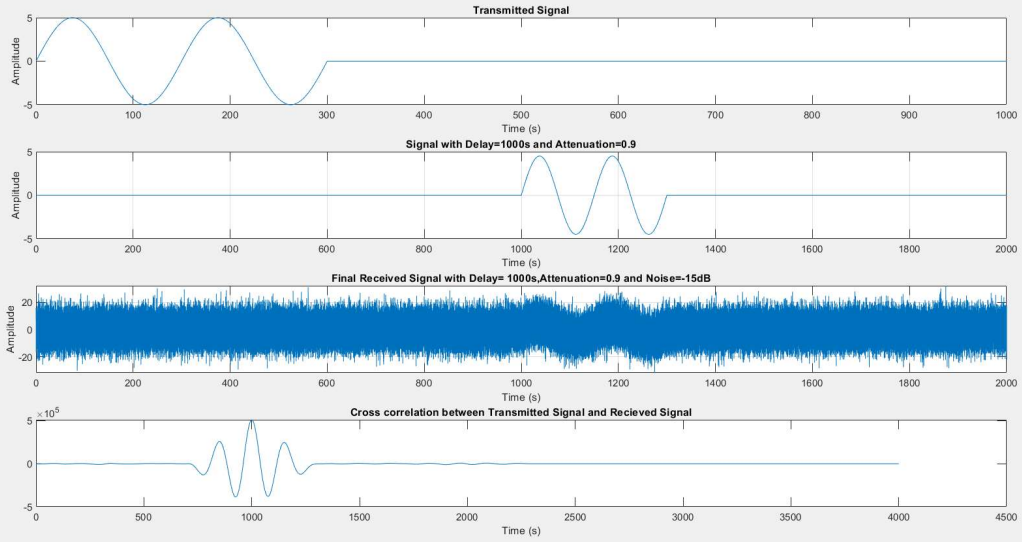


Attenuation factor (A=0.9):

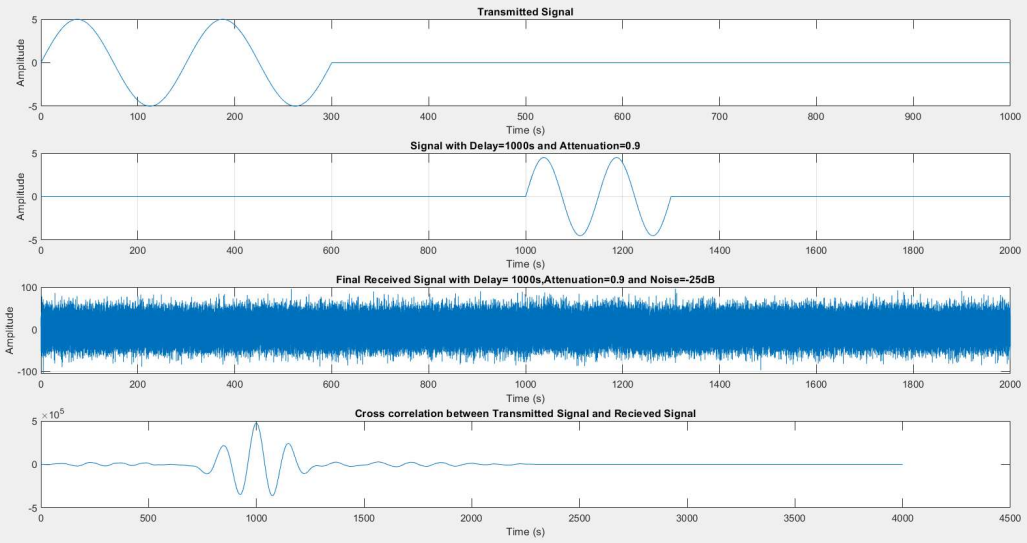
SNR=-5dB:



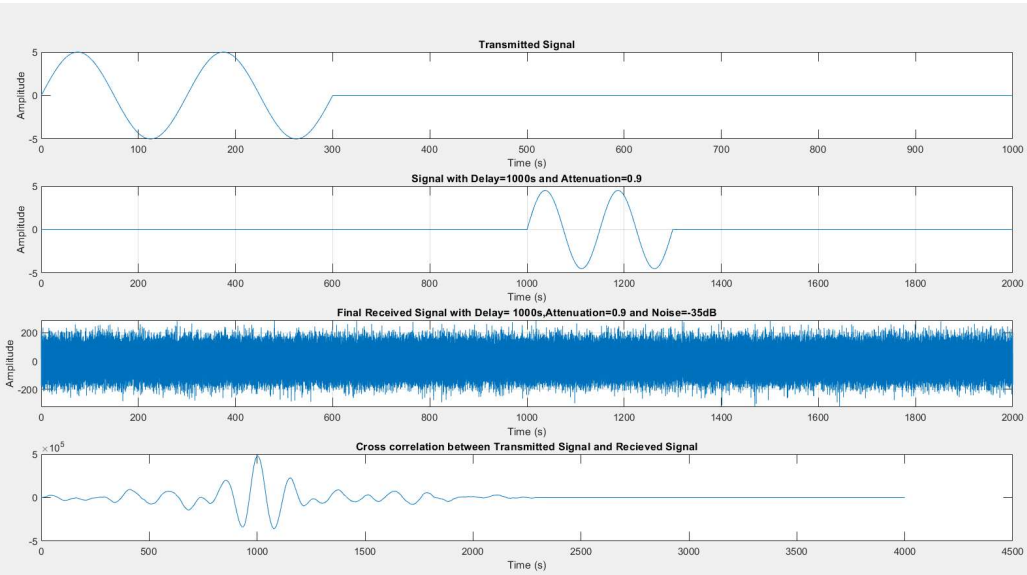
SNR=-15dB:



SNR=-25dB:



SNR=-35dB:



Analysis:

For each Attenuation factor(A) from 0.2 to 0.9 in 0.1 increments, SNR was varied from -5dB to -35dB. So a total of 32 plots are given with each plot having four graphs.

Now the first of these graphs on all of the plots is the transmitted signal (x) which is the same for all of the plots.

The second of these graphs is the attenuated and delayed signal (Ax(t+k)).Now it is worth noting that k,the time delay, is always 1000 s no matter what SNR or A has been set.This makes perfect sense, as the k has been calculated using MATLAB arithmetic and is thus a constant value.So this graph will be right shifted by 1000s always no matter what the A or SNR.
However, since the attenuation factor(A) is changed it is seen that the amplitude of the attenuated and delayed signal changes too. Now attenuation factor is just the gain of the attenuated and received signal,that is if the A=0.2 then the amplitude of the attenuated and received signal will be 0.2*5=1.

The third of these graphs, is the received signal [Ax(t+k)+noise].The received signal is a summation of the attenuated and delayed signal and white background noise.
The general trend for this graph in all of the plots regardless of A value, is that for SNR=-5dB the received signal's two pulses still can be distinguished quite easily.So the delay can be actually guessed just by looking at the graph with more or less accuracy. Of course the result will just be an approximate and cross correlation should be used to identify the delay for further accuracy.
Even for SNR=-15dB the two pulses can still be distinguished with some effort.However,it would be foolish to try and guess the delay from such a graph as it would be quite inaccurate.
However,for SNR=-25dB and -35dB the two pulses cannot be seen at all.So there is now way of recognizing the delay,not even an approximate delay, for these graphs just by looking at it and cross correlation should be used to identify the delay for these graphs.
For the third graph it should also be stated that the received signal's amplitude increases with SNR and A.

That means, for SNR=-15dB, if there are two graphs of the received signal one with A=0.3 and another with A=0.9. Then the graph of A=0.9 will have a slightly larger amplitude.
For A=0.9, if there are two graphs of the received signal one with SNR=-5dB and another with SNR=-35dB. Then the graph of SNR=-35dB will have a quite a larger amplitude (nearly 200) than SNR=-5dB graph which will have amplitude(20)

The fourth and final plot for all the plots is the cross correlation plot [rxy(t)]. Now no matter what the SNR or A values the largest peak is always at 1000 secs with surrounding smaller peaks. The number of smaller peaks and their relative size depends on the SNR.
For SNR=-5dB there are only two peaks surrounding the main peaks.After that the graphs is smooth
For SNR=-15dB, there are also two peaks surrounding the main peaks.But after that the graph is a bit jagged,showing early signs of more peaks.
For SNR=-25dB, there are more than two peaks surrounding the main-peak.(Though those peaks have quite small amplitude still they are quite noticeable and shouldn't be classified as "jagged" only)
For SNR=-35dB, there are the greatest number of peaks surrounding the main peak.(Those peaks are larger than that of SNR=-25dB graph)

Conclusion:

Challenges:

- To generate the transmitted signal x(n)
- Generating the y-axis elements of the attenuated and delayed signal [Ax(t+k)]
- Getting the indices for the xcorr() function using the traditional method.

Troubleshooting:

- The unique characteristic of the transmitted signal (x) is that it was a sinusoid with a period of T=150s and ran for only two pulses (from 0 to 300 s). Yet, the entire graph ran from 0 to 1000 secs.

  At first, we tried to generate a sinusoid from 0<t<1000 and another sinusoid from 300<t<1000.

Then we tried subtracting the two. Now obviously the issue which arose was the fact that both the sinusoids contained different numbers of elements so such a direct subtraction was not possible.

The next attempt was at zero padding the sinusoid using the zeros() function but this also did not work.

Later,it was quickly realized that a sinusoid from 0<t<300 was to be generated and a 0 amplitude sinusoid from 300<t<1000 should be somehow joined together (traditional addition doesn't work cause both sinusoids have different number of elements)

Finally, we utilized the fact that two 1D vectors(a,b) can be appended in MATLAB to make another 1D vector (c) using the syntax c=[a b].

So, a sinusoid (x1) which had period of 150s was generated for 0<t<300

Another sinusoid (x2) with 0 amplitude was generated for 300<t<1000

Then the two were appended to one another to generate the transmitted signal.

i.e x=[x1 x2].

It is worth noting that t-elements actually started from 300+ $\frac{1}{150}$ <t<1000.

This extra 1/150 needed to be added to ensure that the same number of elements existed for both y and x axis elements so that plot() can work.

- Initially, we tried to do this:

  signal= sin(2*pi*f*(t-1000))

  Now the obvious what was seen was that this actually subtract 1000 from each t-element.So wrong graph was outputed.

  Then we realized the append property from the above, i.e c=[a b] can also be used here.

  So we knew that what b had to be equal to the transmitted signal. And a must be zero amplitude sinusoid or zeros() function.

  We decided to go with the zeros() function as we already utilized 0 amplitude sinusoid before.

  ax =[zeros(1,length(tstep3)) x.*A];

  x.*A means that the attenuation factor is being multiplied with each of the elements of the transmitted signal.

zeros(1,length(tstep3)) introduces 1000 zeros before the transmitted signal causing the appropriate delay.

- The traditional method of generating the indices for xcorr() is
  id= (-max(yid)+min(xid)):(-min(yid)+max(xid)).
  The dimensions of this id did not match with the xcorr() o/p, so the number of x-elements and y-elements did not match and plot() function thus was not able to plot the graph.
  Increments were changed from 1 yet to no avail.
  Then what lab sheet 5 was checked and a new way of generating the indices for xcorr was discovered.
  f*(1:length(corr)) where corr is xcorr's output and f is steps in the x-axis for the other graphs.
  This method worked fine.

Overview and Discussion:

In this open ended lab, we were asked to model the behaviour of a radar system which sends out a signal and then receives a attenuated,delayed and noisy signal on MATLAB.Despite the signal being so altered upon receiving, the delay was still found out using cross correlation.
The different graphs were also analyzed for different attenuation factors and SNR.
Overall,this experiment provided an excellent learning opportunity at trying out several key MATLAB skills learned at the lab classes. Example: plotting and modifying graphs,generating noise power from SNR,calculating power of a signal and finding the cross correlation of two signals.
Moreover, it provided a way of applying what was learned at the class to practical scenarios like radar systems.
Analyzing all the o/p's helped us understand how subtle changes in the code end up getting magnified at the o/p.
Troubleshooting problems which had no definite answers was very challenging.
All in all, this open ended lab report was fun to solve.

Appendix(MATLAB Code):

%DEFINING THE VARIABLES

%x=transmitted signal

%tstep2 defines the t-axis for x

%d=distance between radar and target

%v=velocity of signal in air

%t_delay=time taken for signal to go to and fro bwtween radar and target

%A= Attenuation factor

%SNR= Signal to Noise ratio

%ax= attenuated signal with delay too

%t_ax=time axis for attenuated signal with delay

%SP=Signal Power

%NP=Noise Power

%noise

%y=recieved signal

%corr= o/p of xcorr

%corrid indices of corr

%MAKING THE TRANSMITTED SIGNAL

%making a signal called x1 which will be a sinusoid with a period=150 secs and will run 0<t<300

```
Am=5;
T=150;
f=1/150;
tstep=0:f:300
x1=Am*sin(2*pi*f*tstep)
```

% making a signal called x2 which will be a sinusoid of 0 amplitude and

```
% will run from 300+f <t<1000
tstep1= (300+f):f:1000
x2=0*sin(2*pi*f*tstep1)
%transmitted signal 'x'=x1+x2 and will run from 0<t<1000
tstep2=0:f:1000
x=[x1 x2]

%MAKING THE RECEIVED SIGNAL

%calculating the time delay
d=165*1000;
v=330;
t_delay=((2*d)/v)
%attenuation factor and SNR
A=0.9;
SNR=-35;
%making a variable tstep3 which will run from 0 to 1000 secs with 1/150s increments
tstep3=0:f:t_delay
%setting up the attenuated and delayed signal (ax) and its it's time axis (t_ax)
t_ax=[tstep3 tstep2+t_delay]
ax =[zeros(1,length(tstep3)) x.*A];
% calculating signal and noise power
SP=sum(ax.^2)/length(ax);
NP=SP/10^(SNR/10);
%noise signal
noise=sqrt(NP)*randn(1,length(ax));
% received signal= Attenuated and Delayed Signal + Noise signal
y=ax+noise;

%CORRELATION
```

% correlation between transmitted signal (x) and received signal (y)

```
corr=xcorr(x,y)
corrid=f*(1:length(corr))
subplot (4,1,4), plot(corrid,corr)
```

%GRAPHS

%plotting the graphs

```
legend1=sprintf('Final Received Signal with Delay= %ds,Attenuation=%.1f and
Noise=%ddB',t_delay,A,SNR)
legend2=sprintf('Signal with Delay=%ds and Attenuation=%.1f',t_delay,A)

subplot(4,1,1),plot(tstep2,x), xlabel('Time (s)'), ylabel('Amplitude'),title('Transmitted Signal')
subplot(4,1,2),plot(t_ax,ax), xlabel('Time (s)'), ylabel('Amplitude'),title(legend2),grid on
subplot(4,1,3), plot(t_ax,y) , xlabel('Time (s)'), ylabel('Amplitude'),title(legend1),grid on
subplot (4,1,4), plot(corrid,corr), xlabel('Time (s)'),title('Cross correlation between Transmitted
Signal and Recieved Signal')
```

Code Link:

https://drive.google.com/file/d/1g_xaH77P0aIGoiMIx6uuQjmt33OxiALf/view?usp=sharing