EXPERIMENT 3 PART 1
SIGNAL GENERATION, FILTERING, CROSS CORRELATION, A/D, D/A, DMA MATLAB IMPLEMENTATION

Student 1: Abdullah Canbolat, 2304244
Student 2: Güray Özgür, 2167054
Student 3: Yunus Bilge Kurt, 2232395

Task

1) Generate a sine wave **without frequency hopping**. Choose the **frequency 3 kHz**, **sampling frequency 50 kHz**, **amplitude 1**. Increase the sine wave frequency to **24kHz** in **3kHz steps**. **Comment** on the change in the waveforms. **Attach** the plot for the **24kHz** case.

Up to 9 kHz, the plot of the sinusoidal can be interpreted as a sinusoidal by the human eye. However, after 9kHz including 12 kHz, the appearance of the sinusoidal is not apparent for the human eye. This is expected, and occurs because of the comparable frequencies with the sampling frequency. Sampling frequency, 50 Hz, is comparable with 24kHz, so the number of samples are not enough to be seen as a sinusoidal, despite that it is a sinusoidal in discrete time. This can be also observed from the DFT of the signal. The signal and its FFT are shown in Figure 1.

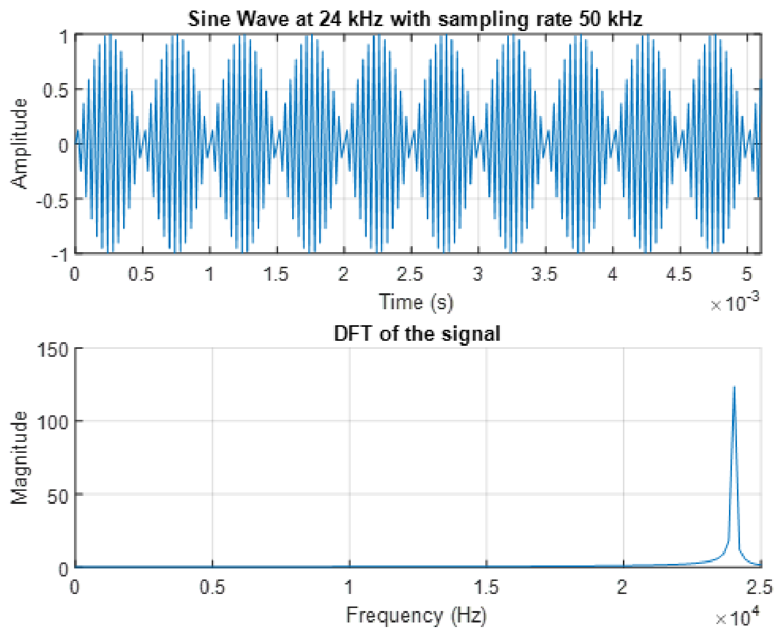


Figure 1: Sine wave at 24kHz with sampling rate 50 kHz and its DFT

2) Keep the frequency of the sine wave at 3kHz. **Increase the sampling rate from 50kHz to 500 kHz in 50kHz steps**. **Comment** on the change in the waveforms. **Attach** the plot for the **500kHz** case.

In time domain, we observe that time axis shrinks in time, while increasing sampling rate. Even though, there is an increase in sampling rate, sinusoidal can be inspected by looking at it for all sampling rates. Since we observe in smaller time intervals when the sampling rate is higher, it is easier to inspect the sinusoidal, which indicates a better time resolution in a way.

In frequency domain, we observe that the peaks of the frequency component of the sinusoidal approaches to the boundaries, since the sampling rate is increased, which means that it becomes harder and harder to inspect its frequency. So, there is a trade-off between frequency resolution and time resolution.

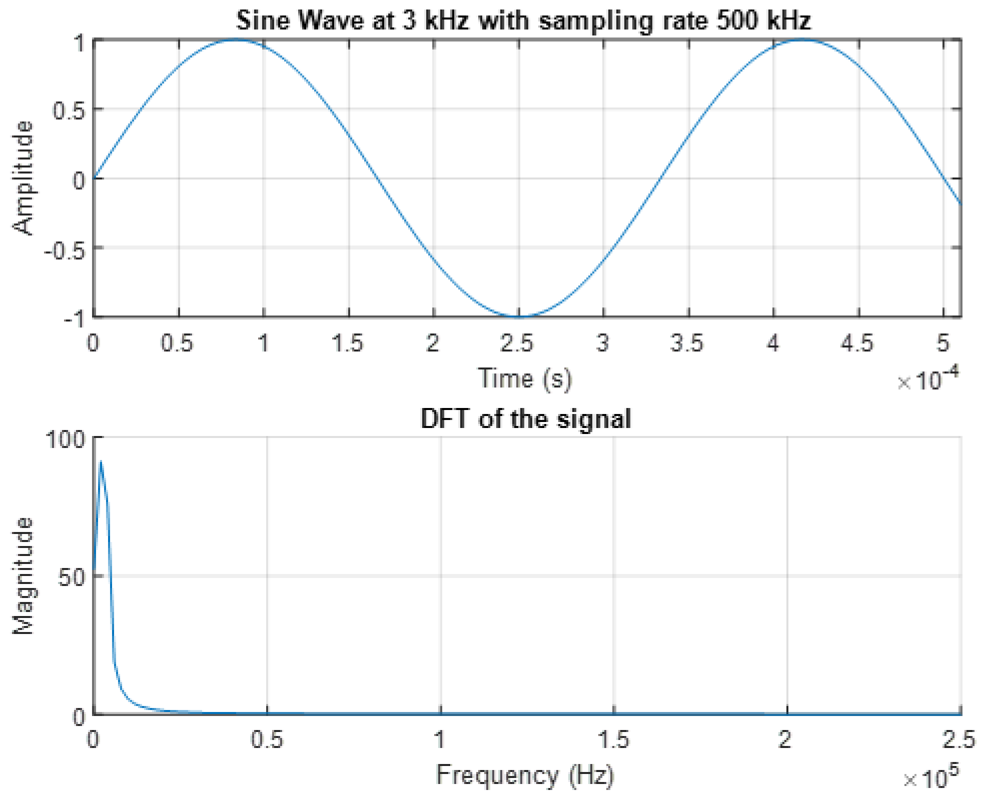


Figure 2: Sine wave at 3kHz with sampling rate 500 kHz and its DFT

3) Keep the frequency of the sine wave at 3kHz. **Decrease the sampling rate from 50kHz to 10 kHz in 10kHz steps**. **Comment** on the change in the waveforms. **Attach** the plot for the **10kHz** case.

Here, there is also a trade-off between frequency resolution and time resolution. As we decrease the sampling rate, it gets easier to understand the frequency content by looking at the 256 point DFT of the signal. On the other hand, time resolution decreases with decreasing sampling rate and it becomes much harder to understand that the signal is a sine wave at 3 kHz by looking at the time plot.

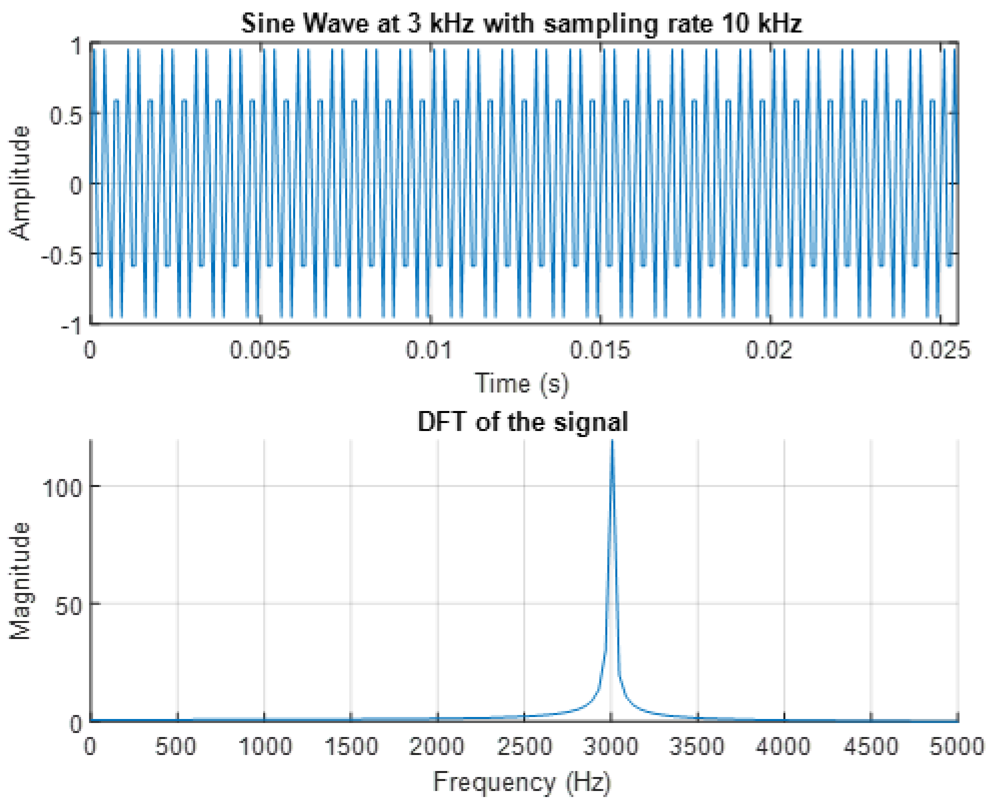


Figure 3: Sine wave at 3kHz with sampling rate 10 kHz and its DFT

4) Now, set the **sine wave frequency** and **sampling frequency** to **1 kHz and 50 kHz**, respectively. **Activate the frequency hopping**. Set the **frequency deviation to 100 Hz**. Observe the waveforms. For the frequency hopping, use five waveforms. That is, once the deviated frequency is calculated, create a signal with 256-samples. Repeat this five times, create your final signal by concatenating these five signals. Increase the frequency deviation **from 100 Hz to 500 Hz in 100 Hz steps**. **Observe** the changes in the waveforms. **Comment** on the results. **Attach** the plot for the **200 Hz deviation** case.

What we expect in this task is that in the DFT of the signal, we will see a peak at its frequency. However, since there is a deviation in the frequency, the frequency of the signal will change in a neighborhood, thus we will observe several peaks instead of one at that neighborhood. Figure 4 shows a sinusoidal signal at frequency 1 kHz with a deviation of 200 Hz, thus as expected, we observe not only a peak at 1 kHz but also peaks in a frequency range of 1 kHz ± 100 Hz. By concatenating sinusoids with different frequency deviations, the same observation is also obtained in Figure 5 in a broad range since the deviation is higher.

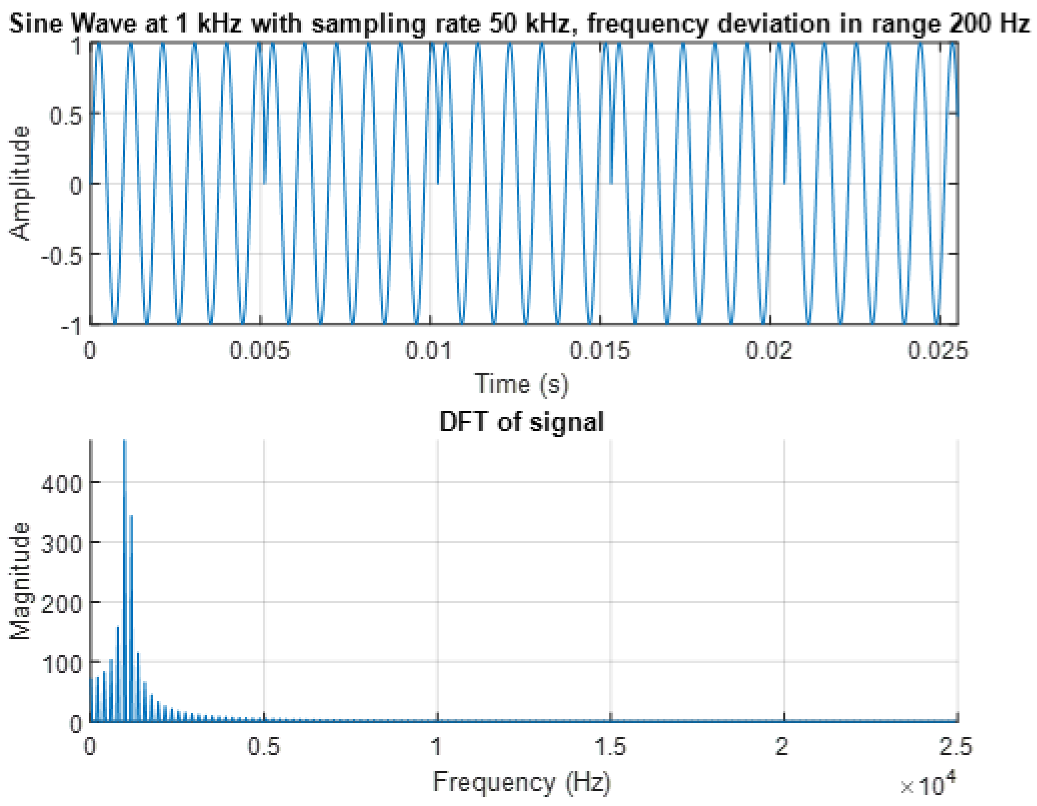


Figure 4: Sine wave at 1kHz with a deviation of 200Hz by sampling rate 50kHz and its DFT

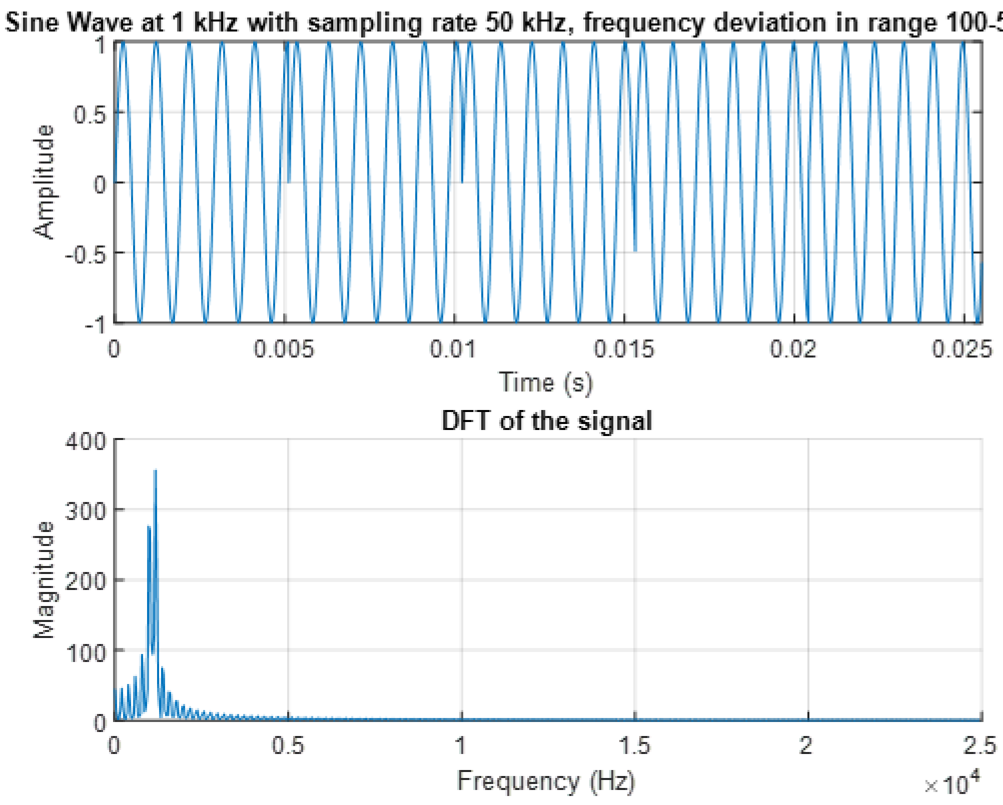


Figure 5: Concatenating sine waves at 1kHz with a deviation of 100-500Hz by sampling rate 50kHz and its DFT

5) Repeat steps 1, 2, 3, and 4 for the square wave.

A square wave can be expressed as a sum of sinusoids at odd harmonics of the wave frequency where each sinusoidal's amplitude is inversely proportional to its frequency. This wide frequency content gets harder to capture when sampling frequency gets closer to the square wave frequency. The higher order sinusoidals cause aliasing in the frequency domain. This effect can be seen in Figure 6, where the frequencies lower than 24 kHz actually does not exist in the signal but exist in the frequency spectrum. Other than this effect, the observations of time and frequency components are the same with the sinusoidal signals as this signal can be expressed as superposition of sinusoidal signals.

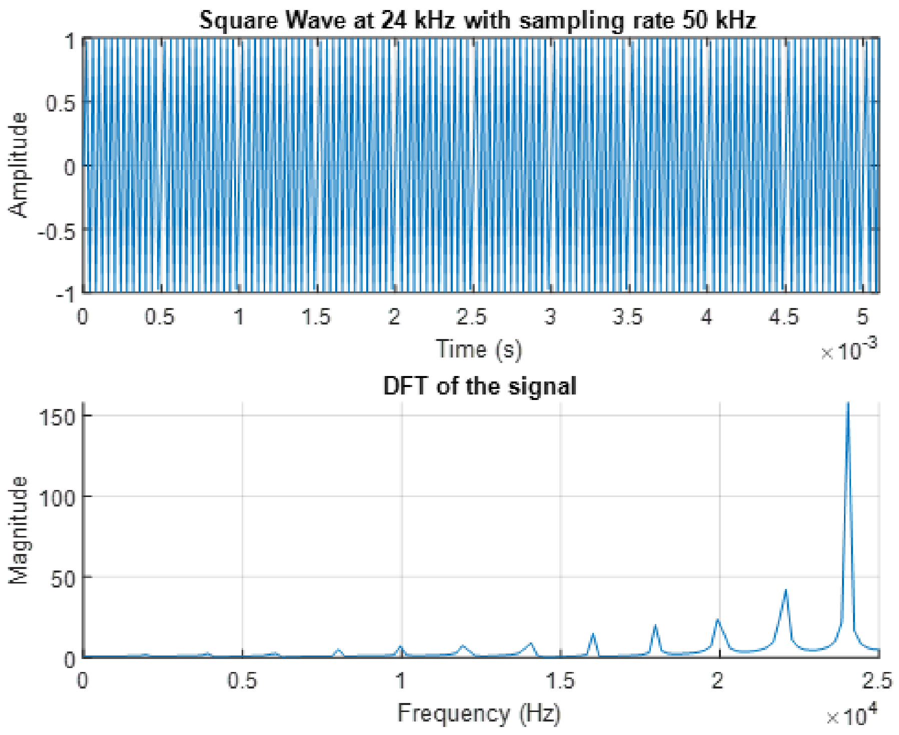


Figure 6: Square wave at 24kHz with a sampling rate 50kHz and its DFT

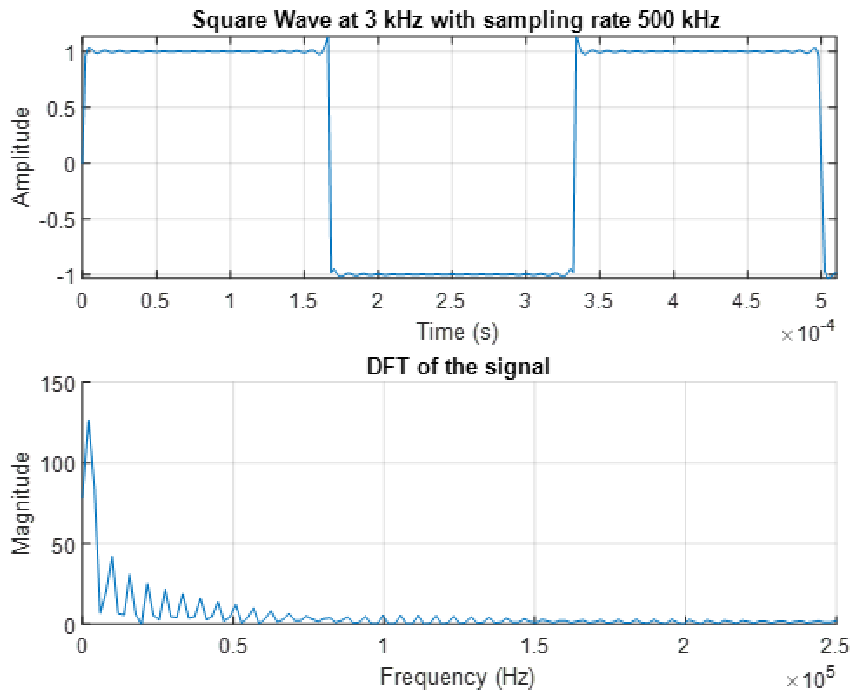


Figure 7: Square wave at 3kHz with a sampling rate 500kHz and its DFT

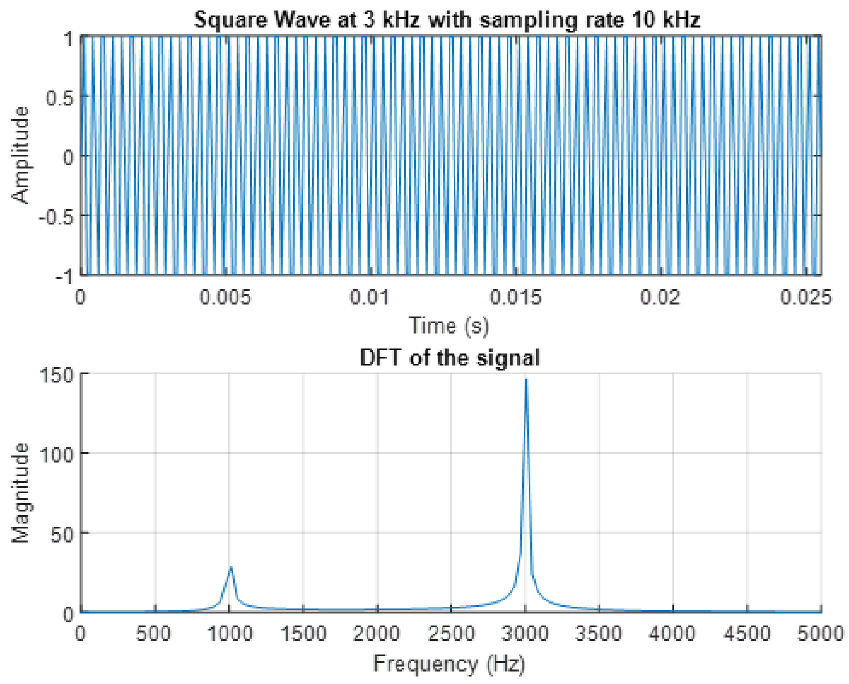


Figure 8: Square wave at 3kHz with a sampling rate 10kHz and its DFT

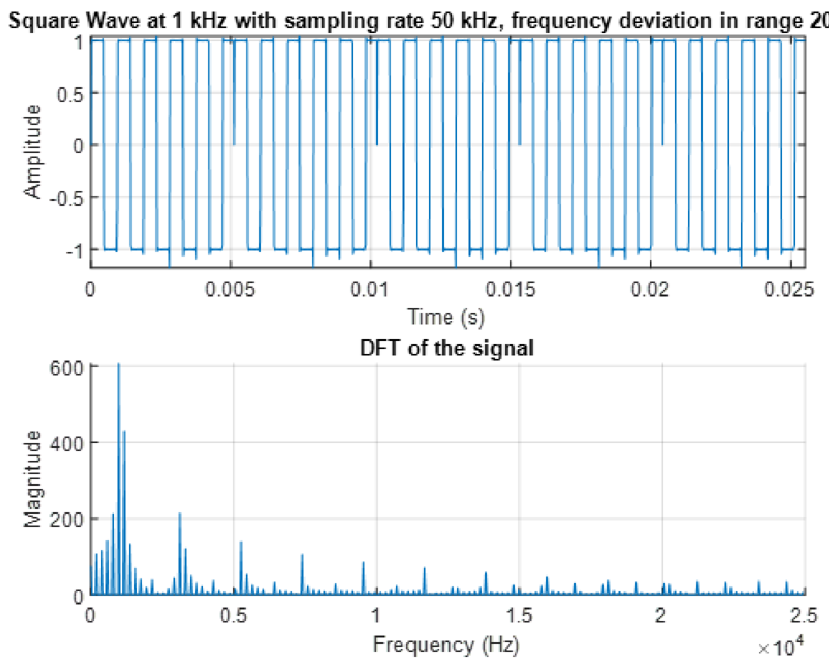


Figure 9: Square wave at 1kHz with a deviation of 200Hz by sampling rate 50kHz and its DFT

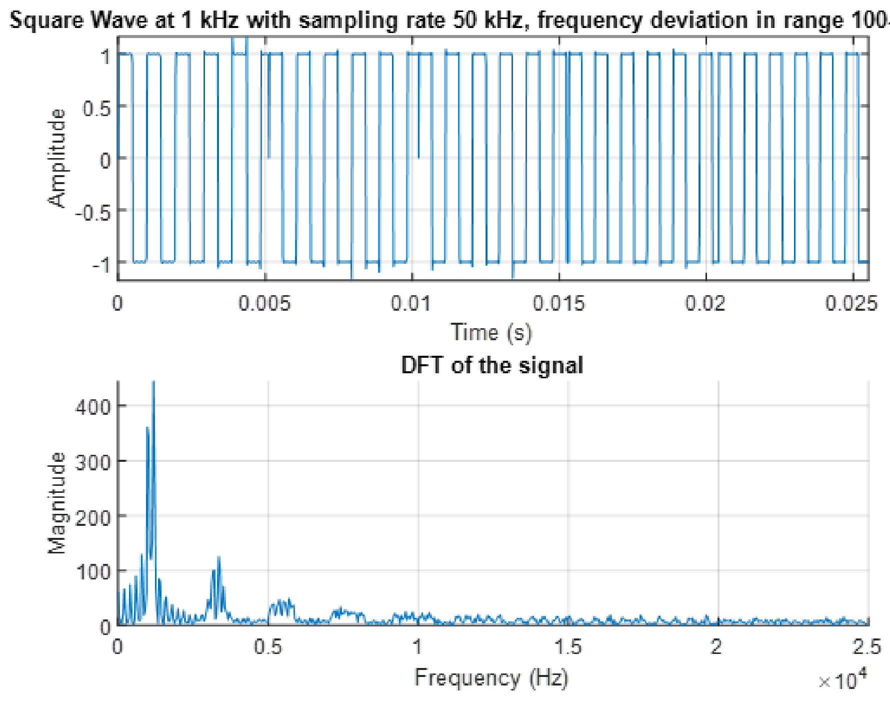


Figure 10: Square wave at 1kHz with a deviation of 100-500Hz by sampling rate 50kHz and its DFT

6) Generate a sine wave **without frequency hopping** with a **sampling frequency of 100 Hz**. Adjust the frequency of the sine wave to **101 Hz**. **What** do you **observe**? **What** is the **frequency of the observed signal**? **Comment** on the results. **Attach** the plot.

We observed a sinusoidal with a frequency of 1 Hz. The reason behind this phenomena can be explained by looking at the way sampling operation affects the signals frequency spectrum. By sampling, signal is multiplied with an impulse train with frequency equal to the sampling frequency. This operation means convolving the frequency spectrum with an impulse train with period equal to the sampling frequency. Hence the amplitude of 101 Hz frequency is carried to the 1 Hz by the effect of this shift operation and the spectrum becomes the same with the spectrum of a 1 Hz signal sampled with a sampling frequency of 100 Hz.

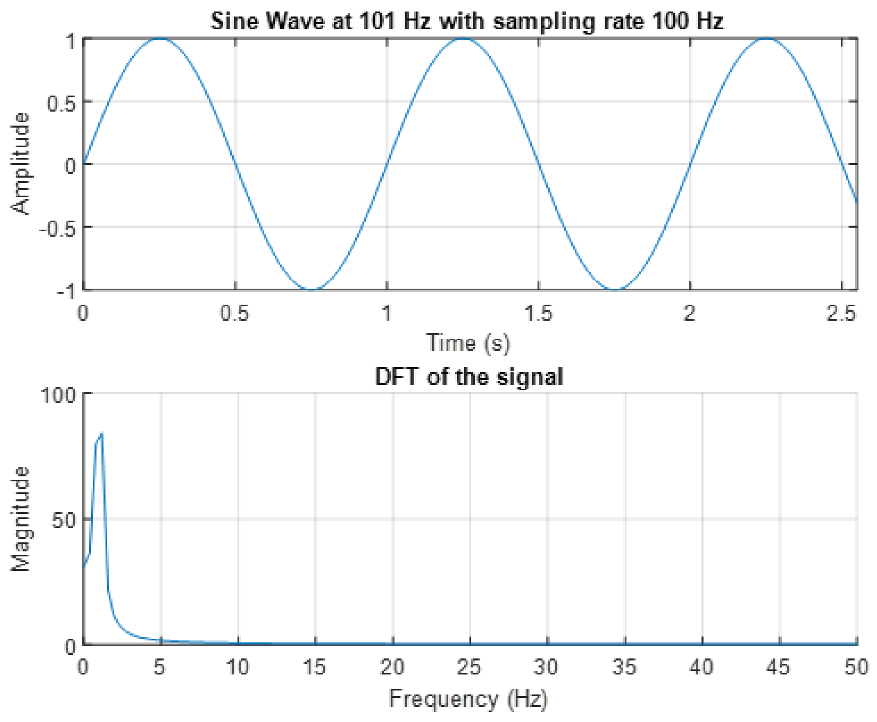


Figure 11: Sine wave at 101Hz with a sampling rate 100Hz and its DFT

7) Now increase the frequency to **105 Hz in 1 Hz steps**. **Comment on the frequencies** of the waveforms.

As in Task 6, since the frequency is higher than the sampling frequency, observed frequencies of the signals are 1Hz, 2Hz, 3Hz, 4Hz, and 5Hz, respectively.

8) Increase the sine wave frequency from **1001 Hz to 1005 Hz in 1 Hz steps** while the **sampling frequency is still 100 Hz**. Are the results **similar** to 7? **Why**? Attach the plot for **1001 Hz** case.

The result is the same with the 101 Hz sine wave. As the impulse train is infinite, from the impulse at the 1000 Hz in the spectrum, shifting carries the frequency component at the 1001 Hz to 1 Hz. For any signal with frequency k*100+1 Hz where k is a positive integer, we will observe the same spectrum.

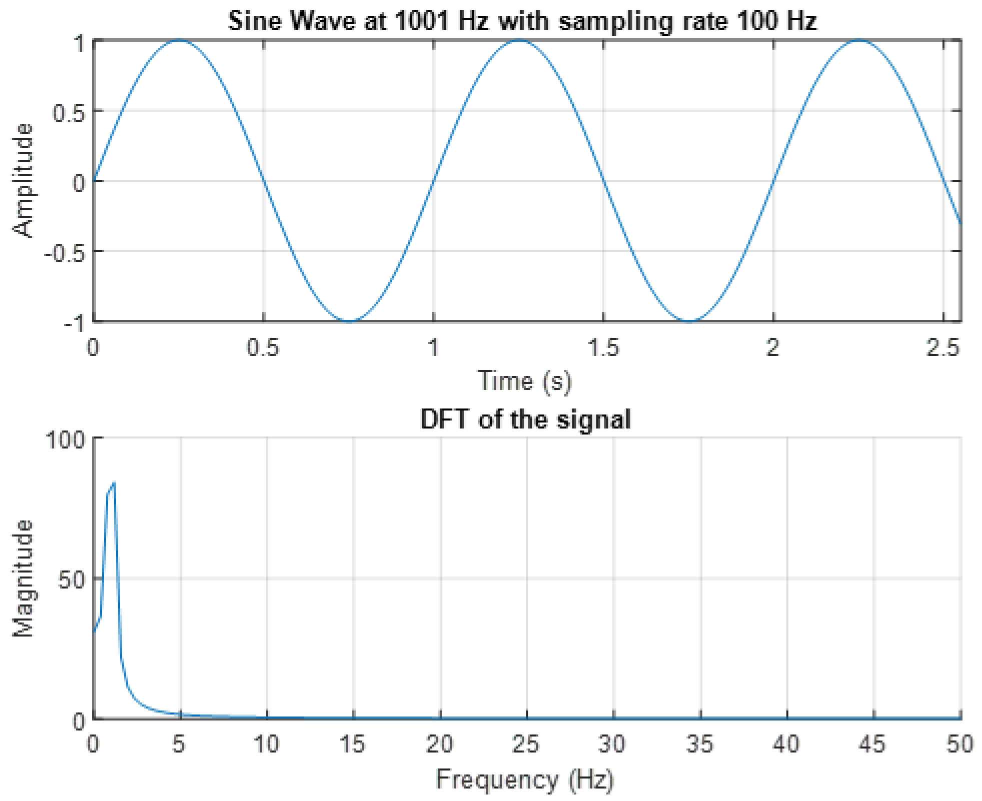


Figure 12: Sine wave at 1001Hz with a sampling rate 1000Hz and its DFT

9) Now generate a sinusoid at **1000 Hz** frequency and select the sampling frequency $f_{S1} = 16\ kHz$. Assume that we have converted this signal to analog waveform and then sampled again with $f_{s2} = 16\ kHz$. Increase and decrease **fs1** to **32kHz and 8kHz**, respectively and note the received signal's **estimated frequency**. Also, **note the differences in time and frequency between each case**. Now select **f_{s1}=16kHz** and change **f_{s2} to 32kHz and 8 kHz**, respectively. **Note the estimated frequency and differences in time and frequency.** Attach the plots for $f_{S1} = 16\ kHz$ **and** $f_{S2} = 32\ kHz$ **case.**

Signal and its DFT are the same, but time and frequency resolutions are changed when we resampled the signal. Time resolution is increased and frequency resolution is decreased when we sampled the signal with 32 kHz.

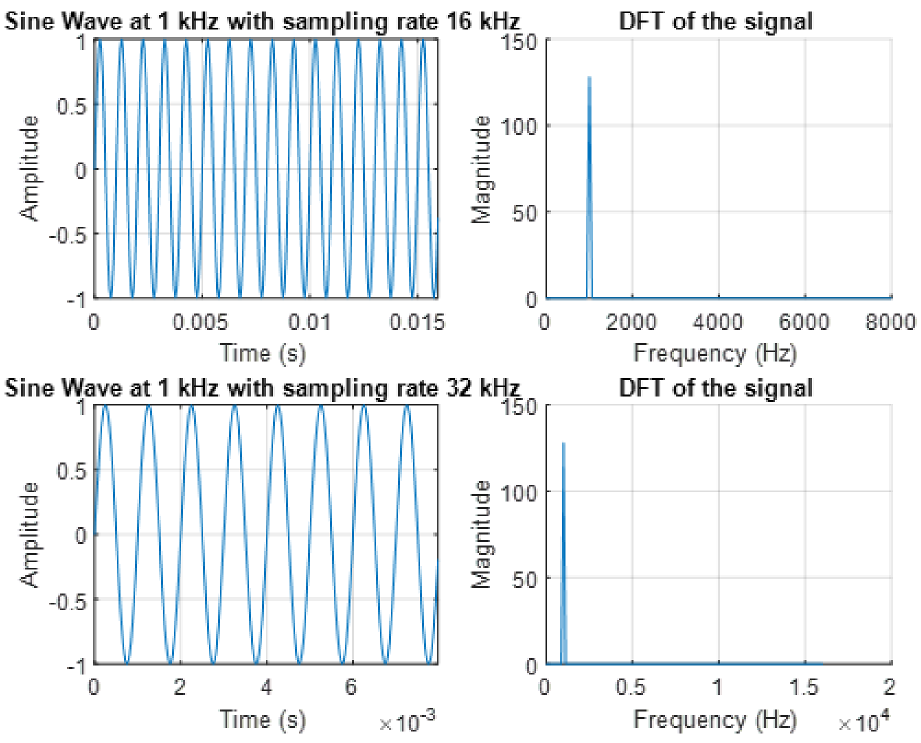
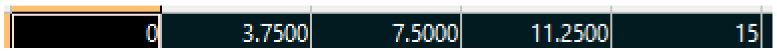


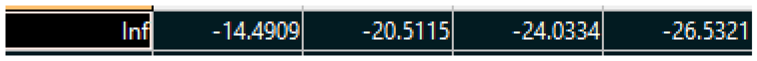
Figure 13: Sine wave at 1kHz with a sampling rate 32kHz and its DFT

10) Generate a **sinusoid at 1 kHz**. Choose the **amplitude of the signal as 10**. Choose the **sampling frequency of 16 kHz**. **Add noise** to the generated signal by changing the **noise standard deviation from 0 to 15 with an increase in 5 steps**. Note the **estimated frequency and SNR in each case**. Now **increase the noise standard deviation to a value** such that sinusoid frequency **cannot be estimated any longer**. **Note this value** and write it in your report.



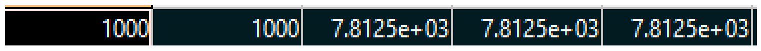
0	3.7500	7.5000	11.2500	15

Noise standard deviation values



Inf	-14.4909	-20.5115	-24.0334	-26.5321

SNR values in dB



1000	1000	7.8125e+03	7.8125e+03	7.8125e+03

Estimated frequency for each noise std

When the noise standard deviation is above 3.8875 frequency of sine wave will not be estimated accurately.

11) Continue **from 10** but select the **sampling frequency 32 kHz** and **noise standard deviation as 0.1**. Observe the **cross-correlation of the input with the chirp**. Select the chirp **f1 and f2 frequencies as 160 Hz and 6400. Increase the value of f1 in 1600 steps until 6400. Note the changes in cross-correlation. Attach the plots for all cases.**

Our sinusoidal signal has a frequency component at 1 kHz. The frequency range for the first chirp is 160-6400 Hz, for the second chirp is 1760-6400 Hz, for the third chirp is 3360-6400 Hz, for the fourth chirp is 4960-6400 Hz, thus the first chirp is more similar to our sinusoidal for frequency-wise. Since correlation implies the similarity, our first observation is that more similar means a higher correlation, which is observed in Figure 14. Correlation is higher for the first chirp than the others, and when the frequency matches, we observe a higher correlation.

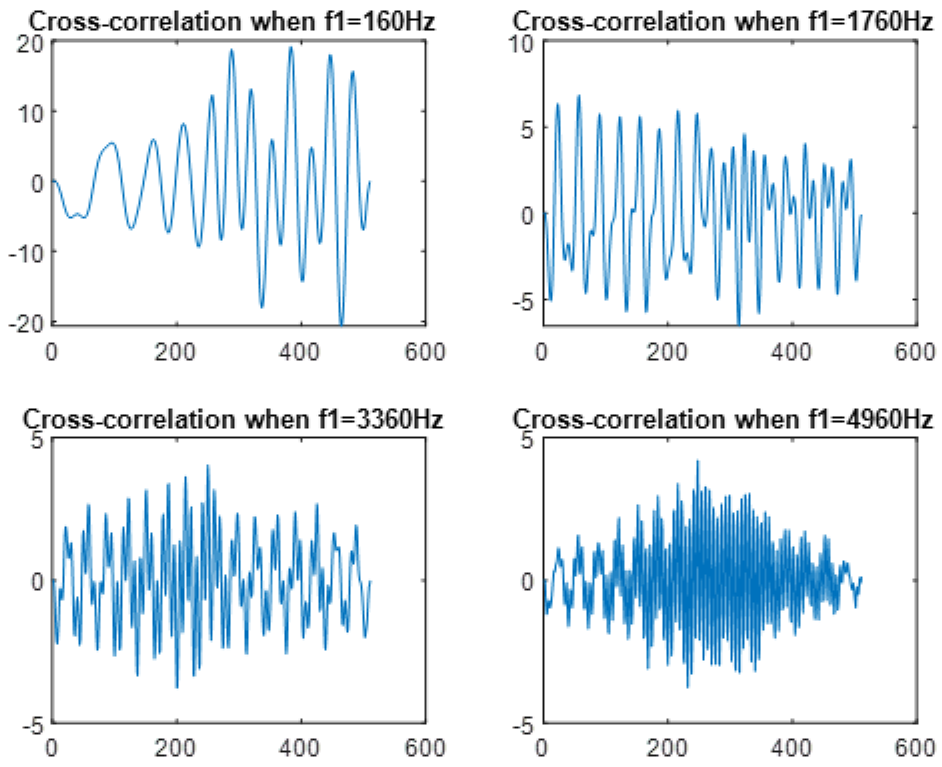
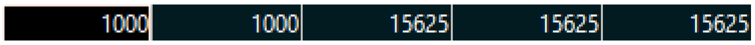


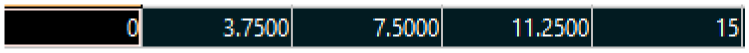
Figure 14: Cross-correlation of sine wave at 1kHz and several chirps with different start frequencies

12) **Repeat 10 using frequency hopping mode** where the **frequency deviation** is set as **50Hz**.



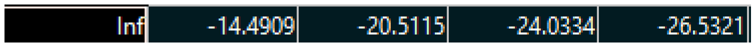
1000	1000	15625	15625	15625

Frequency estimation for each case



0	3.7500	7.5000	11.2500	15

Noise standard deviations



Inf	-14.4909	-20.5115	-24.0334	-26.5321

SNR values

When the noise standard deviation is above 2.78, frequency of sine wave will no longer be estimated.

13) Generate a **square wave** with a **period** of **T=0.5ms**. Choose the **sampling rate as 64 kHz**. Choose the **normalized low cutoff frequency of the Butterworth filter** as $\frac{f_{square}}{(\frac{f_{sampling}}{2})}$.

You can use the **butter** and **filter** functions of MATLAB. Note the **input and filtered signal waveforms** both **in time and in frequency**. Determine **the main spectrum width** for the square wave and **its relation to the period, T**. You can change the period to see its effect on the frequency spectrum. **Decrease T by 2, 4 and increase it by 2, 4** to identify the **time-frequency relation**. **For T=0.5ms, T=1ms and T= 2ms attach your time and frequency domain plots**. Also, **note the cross-correlation function** between **the input** and the **filtered output**. For **T=0.5ms attach the plot.**

Butterworth filter with order 1 is used. For a reasonable approximation to the square-wave shape, at least the fundamental and third harmonic need to be present, with the fifth harmonic being desirable. We decided to also take up the fifth harmonic for the main spectrum width. We observe that when we increase the period, the main spectrum width decreases as seen from Figure 15, 16, and 17. Observation is that time and frequency are inversely proportional.

Since Butterworth filter preserves the main spectrum, the cross-correlation plot in Figure 18 shows us that there is a similarity between input and output waveforms despite that it can not be seen from the time-domain representations as seen in Figures 15, 16, and 17.

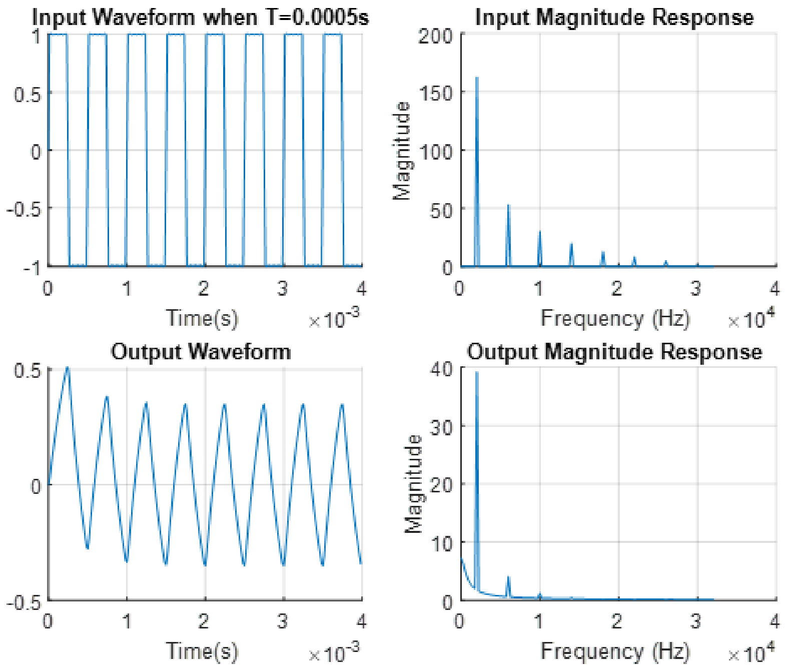


Figure 15: Input and output waveforms and their magnitude responses for T=0.5 ms

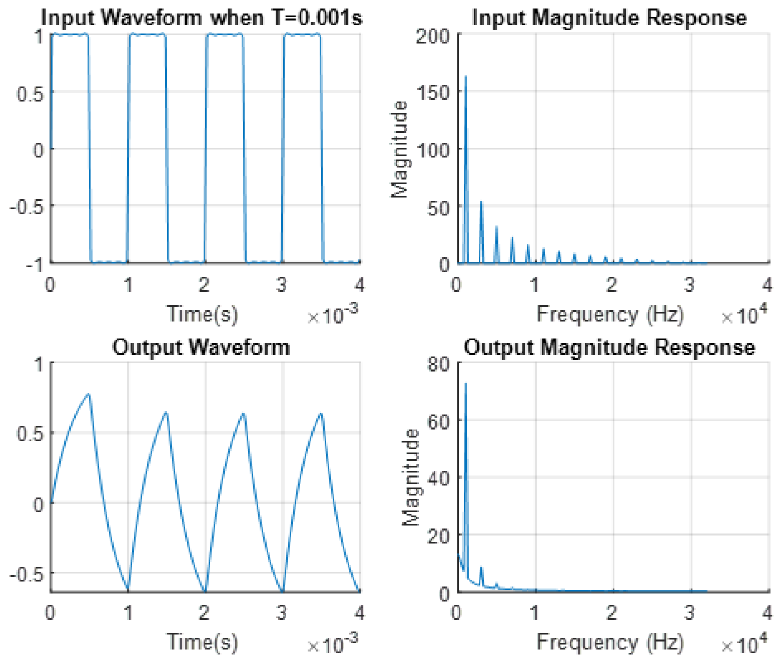


Figure 16: Input and output waveforms and their magnitude responses for T=1 ms

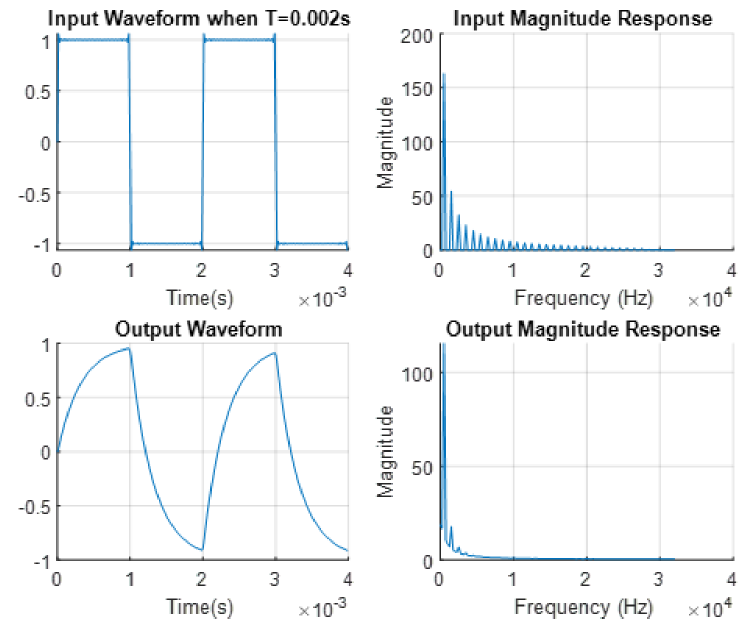


Figure 17: Input and output waveforms and their magnitude responses for T=2 ms

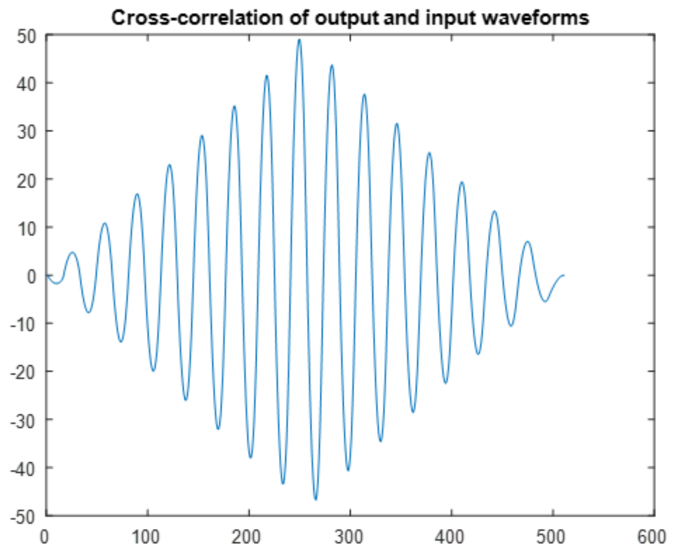


Figure 18: Cross-correlation of output and input waveforms for T=0.5 ms

14) Generate a **square wave** with a **period** of **T=0.5ms**. Choose the **sampling rate as 64 kHz**. Choose the **normalized low cutoff frequency of the Butterworth filter** as $\frac{f_{square}}{(\frac{f_{sampling}}{2})}$.

Filter your input signal with this filter. Additionally, filter your input signal using the following bandpass filters:

- 5000 Hz – 7000 Hz
- 10000 Hz – 11000 Hz
- 13000 Hz – 15000 Hz

You can use **bandpass** command of **MATLAB**. For each case, **plot the cross-correlation between the input and filtered signal**. Attach your plots. **Comment on the results**.

The cross-correlation between input and output is given in Figure 18 for Butterworth filter, while for bandpass filters they are given in Figures 19, 20, and 21. From the Figures 19, 20, 21 we see that the highest correlation occurs when the bandpass filter cut off frequencies are 5 kHz and 7 kHz, and lowest when cut off frequencies are 13 kHz and 15 kHz. Since the output only contains the higher harmonics of the square wave, we can observe that the higher the frequency of the bandpass, the lower the cross-correlation value, which is expected and observed from Figures 19, 20, and 21.

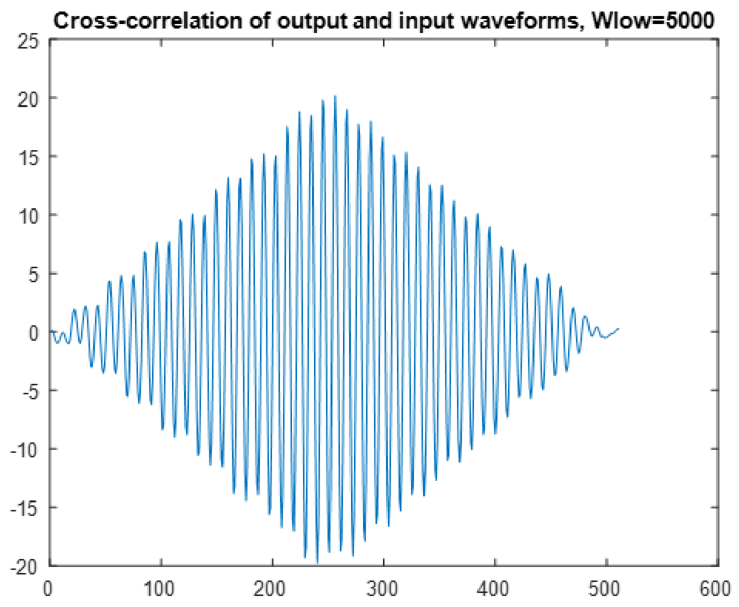


Figure 19: Cross-correlation of output and input waveforms for w_low=5000 Hz

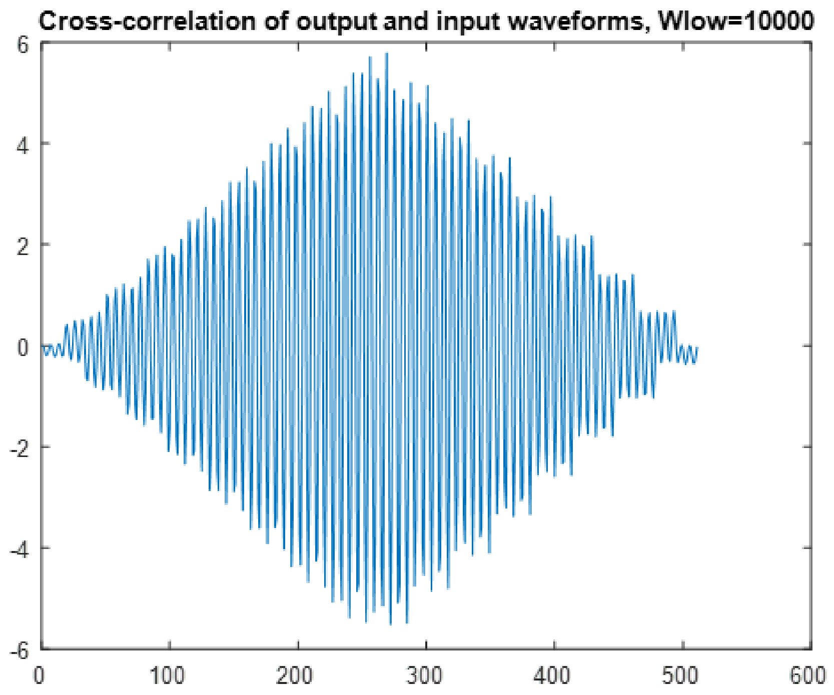


Figure 20: Cross-correlation of output and input waveforms for w_low=10000 Hz

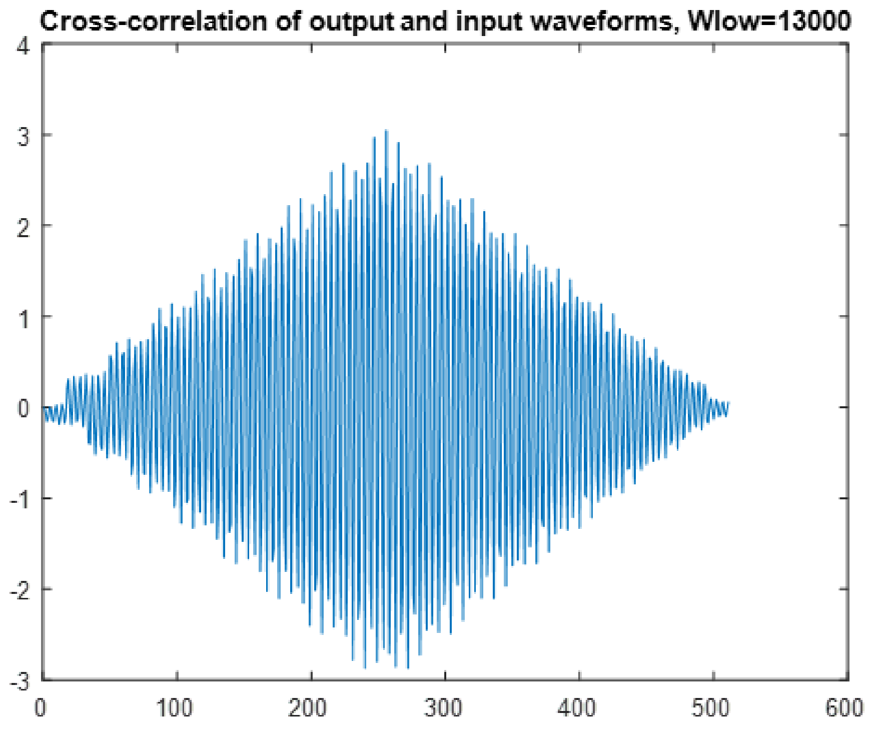


Figure 21: Cross-correlation of output and input waveforms for w_low=13000 Hz

15) **(Bonus)** Modulate **the square wave generated in step 13 by a 6kHz sinusoid**. Select the **noise standard deviation as 0.5**. **Filter the resulting signal with a bandpass filter to remove noise as much as possible while keeping the signal characteristics**. **Determine the minimum bandwidth that you can use for this purpose**. **Demodulate** the signal down to baseband by **multiplying with another sinusoid** with the same frequency and **lowpass filtering** the resulting waveform. **Plot the input and demodulated signal time and frequency characteristics in the front panel.** Note that this structure is used to transmit and receive signals, just like a standard communication system.

Carrier frequency is selected as 15 kHz and sampling frequency is 64 kHz. We decided to take first, third and fifth harmonics and filter out others as they become affected too much by the noise. w

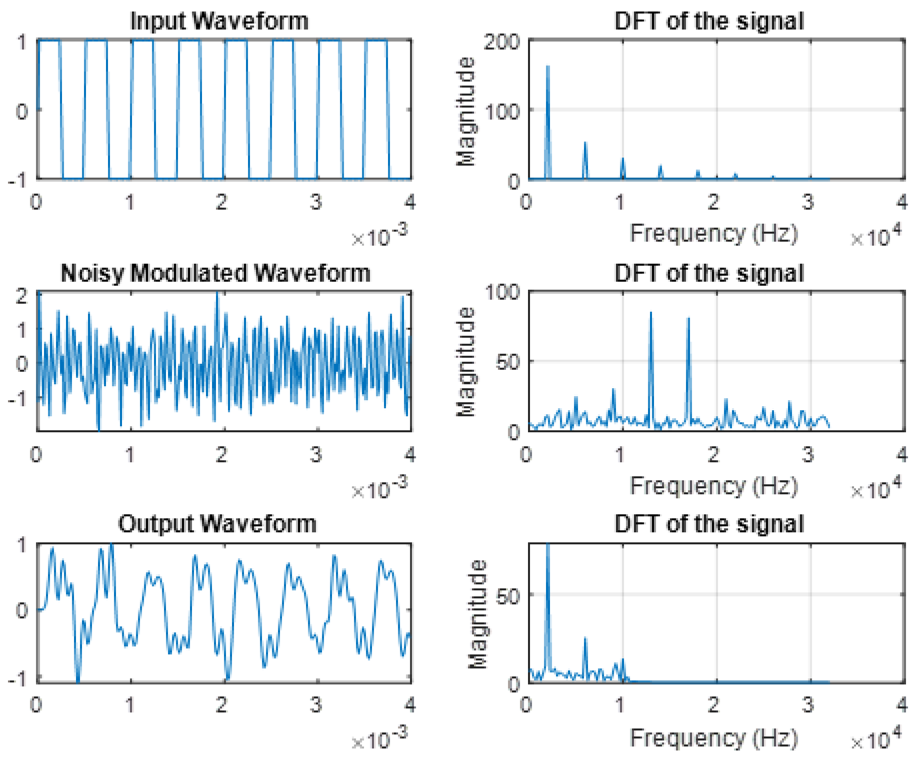


Figure 22: Input, noisy modulated, and output waveforms and their DFTs

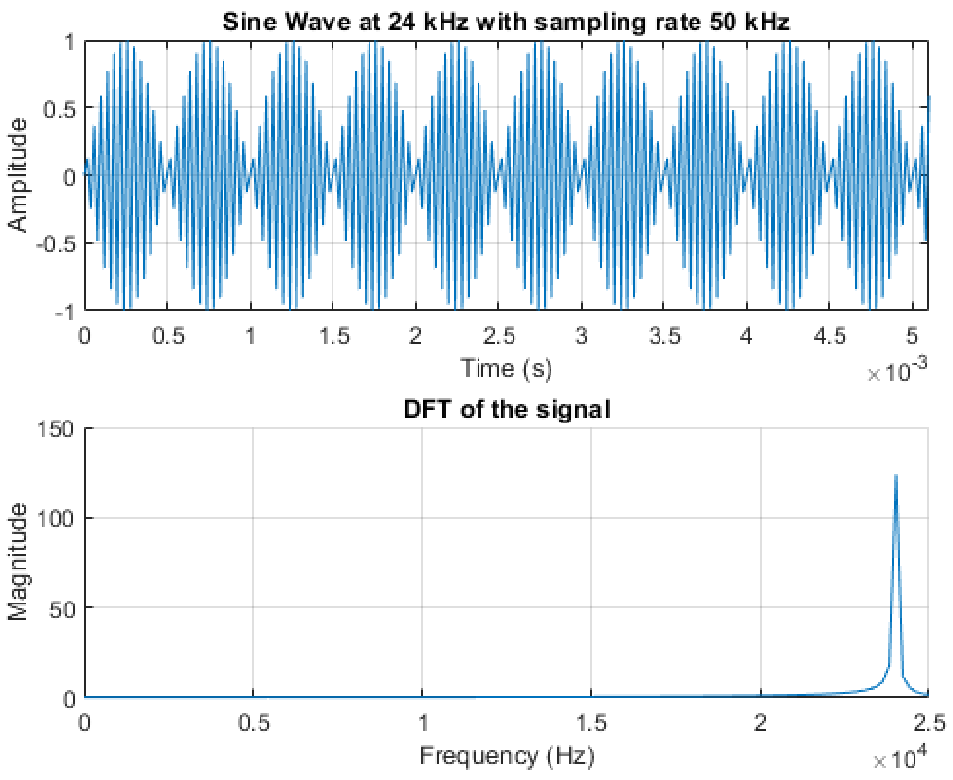
Table of Contents

```
close all;
clear;
```

Task 1

```
num_samp = 256;
frequency_hop = false;
noise = false;
amplitude = 1 ;
wavetype = 'sine';
sample_freq = 50*1e3;
wave_freq = 24*1e3;
phase = 0;

[wave,time] = wave_generator(wavetype, wave_freq, sample_freq,
 amplitude, phase, frequency_hop, noise);


figure;
subplot(2,1,1);
plot(time,wave);
hold on;
grid;
title('Sine Wave at 24 kHz with sampling rate 50 kHz');
xlim([0,time(end)]);
xlabel('Time (s)');
```

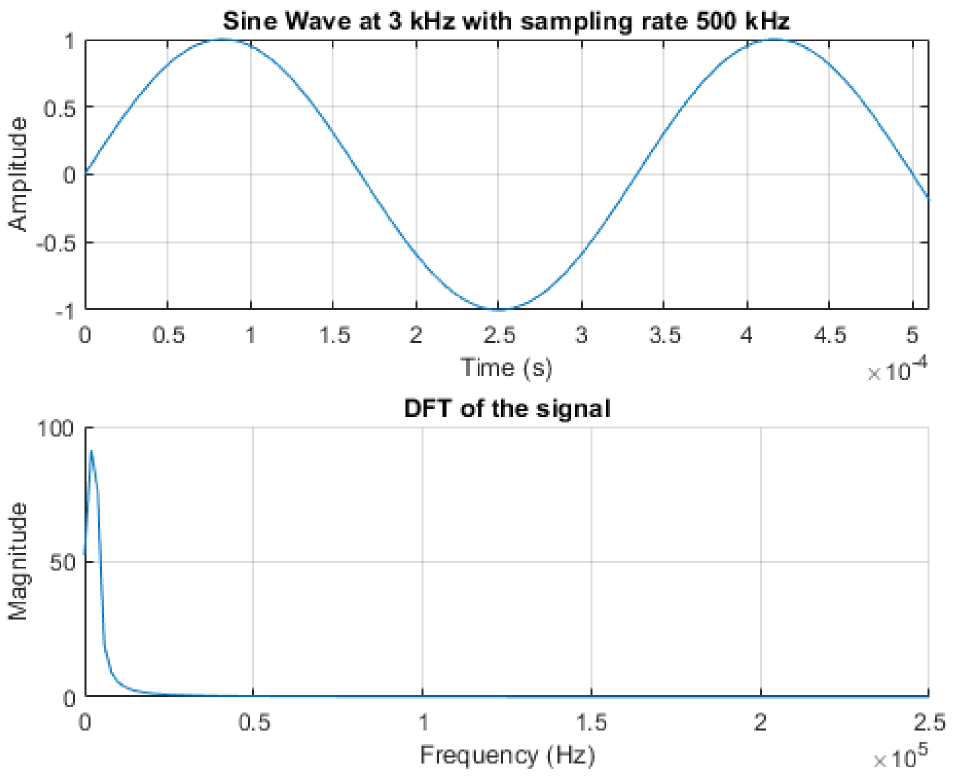
```
ylabel('Amplitude');

w = linspace(0,sample_freq*(num_samp-1)/num_samp,num_samp);

wave_fft = abs(fft(wave));
subplot(2,1,2);
hold on;
plot(w(1:num_samp/2+1),wave_fft(1:num_samp/2+1));
grid;
title('DFT of the signal')
xlabel('Frequency (Hz)');
ylabel('Magnitude');
```



Task 2

```
sample_freq = 500*1e3;
wave_freq = 3*1e3;


[wave,time] = wave_generator(wavetype, wave_freq, sample_freq,
 amplitude, phase, frequency_hop, noise);

figure;
subplot(2,1,1);
plot(time,wave);
hold on;
```

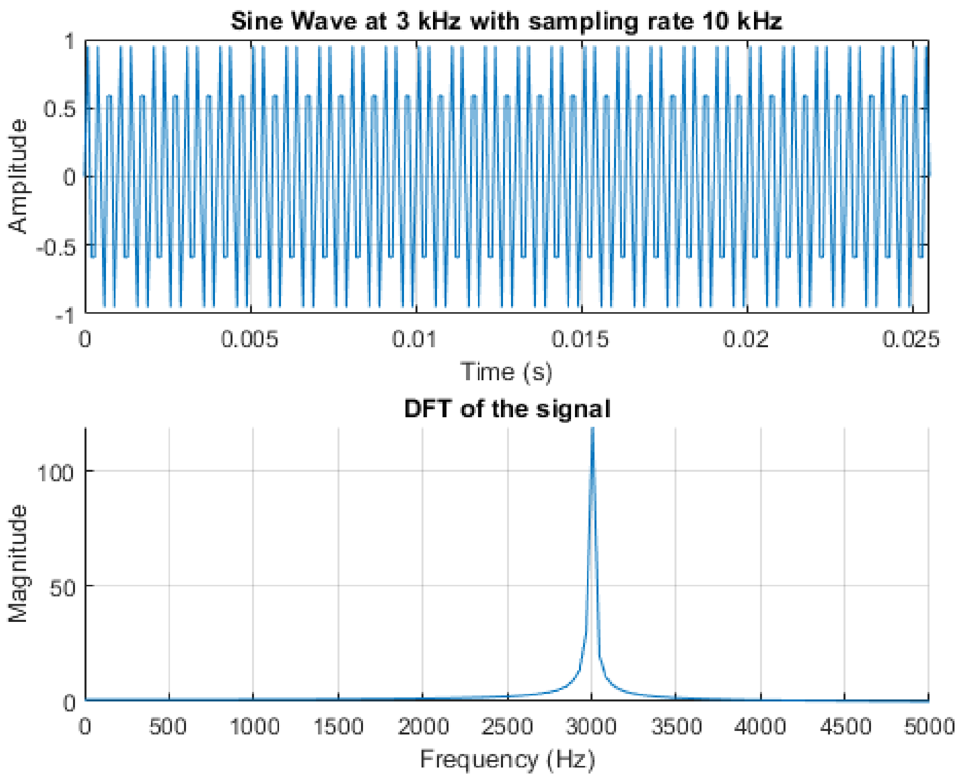
```
grid;
title('Sine Wave at 3 kHz with sampling rate 500 kHz');
xlim([0,time(end)]);
xlabel('Time (s)');
ylabel('Amplitude');


w = linspace(0,sample_freq*(num_samp-1)/num_samp,num_samp);

wave_fft = abs(fft(wave));
subplot(2,1,2);
hold on;
plot(w(1:num_samp/2+1),wave_fft(1:num_samp/2+1));
grid;
title('DFT of the signal')
xlabel('Frequency (Hz)');
ylabel('Magnitude');
```



Task 3

```
sample_freq = 10*1e3;
wave_freq = 3*1e3;


[wave,time] = wave_generator(wavetype, wave_freq, sample_freq,
 amplitude, phase, frequency_hop, noise);
```

```
figure;
subplot(2,1,1);
plot(time,wave);
hold on;
grid;
title('Sine Wave at 3 kHz with sampling rate 10 kHz');
xlim([0,time(end)]);
xlabel('Time (s)');
ylabel('Amplitude');

w = linspace(0,sample_freq*(num_samp-1)/num_samp,num_samp);

wave_fft = abs(fft(wave));
subplot(2,1,2);
hold on;
plot(w(1:num_samp/2+1),wave_fft(1:num_samp/2+1));
grid;
title('DFT of the signal')
xlabel('Frequency (Hz)');
ylabel('Magnitude');
```

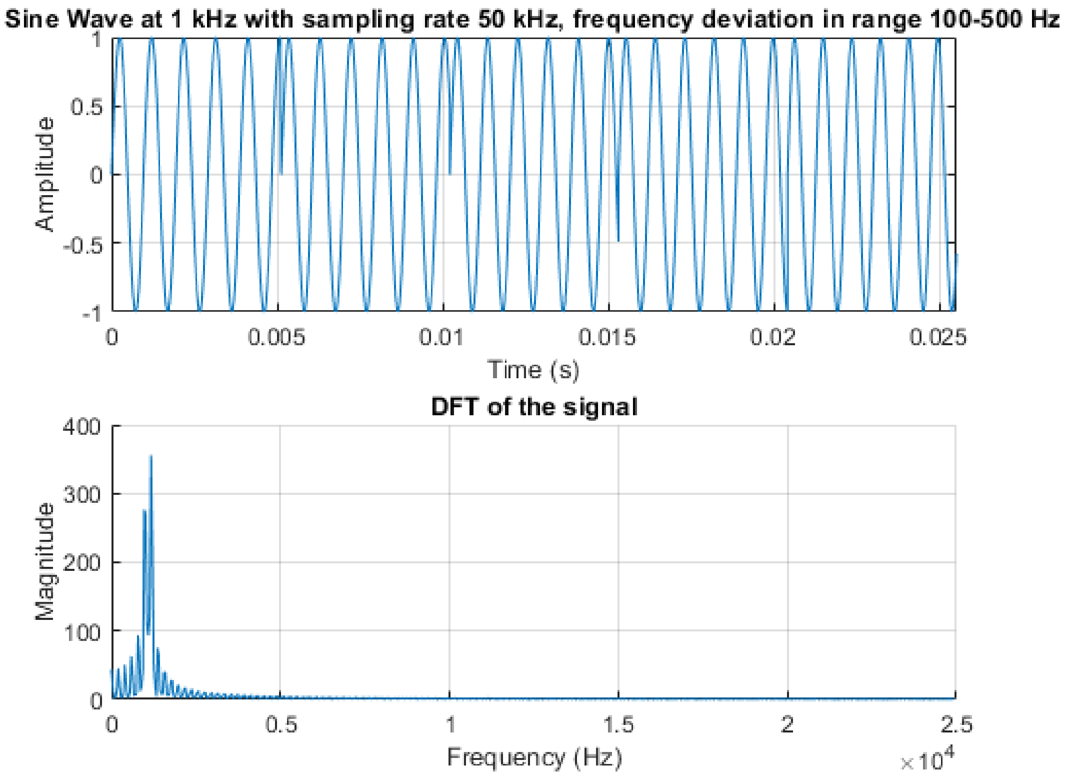


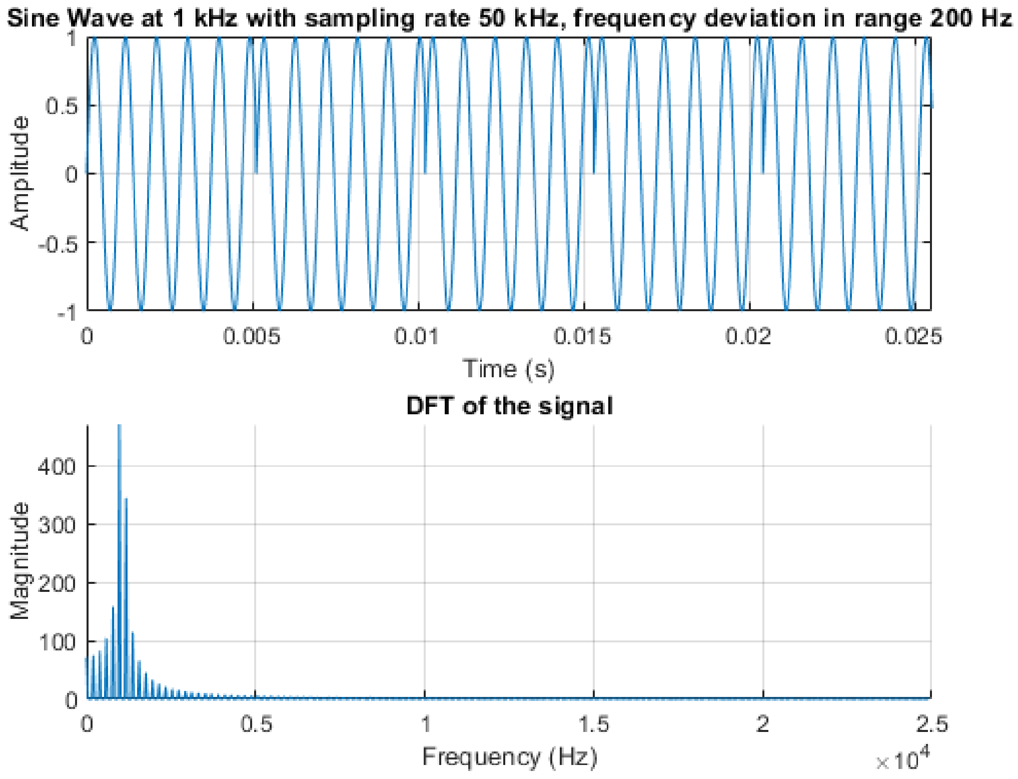
Task 4

```
frequency_hop = true;
sample_freq = 50*1e3;
```

```
wave_freq = 1*1e3;


total_wave = [];
total_time = [];
for deviate = 100:100:500
    [wave,time] = wave_generator(wavetype, wave_freq, sample_freq,
 amplitude, phase, frequency_hop, noise,0,deviate);
    total_wave = [total_wave,wave];
    if deviate-100 == 0
        total_time = [total_time,time];
    end
    if deviate - 100 > 0
       total_time =  [total_time,time+(deviate-100)/100*time(end)+1/
sample_freq];
    end
end


figure;
subplot(2,1,1);
plot(total_time,total_wave);
hold on;
grid;
title('Sine Wave at 1 kHz with sampling rate 50 kHz, frequency
 deviation in range 100-500 Hz');
xlim([0,total_time(end)]);
xlabel('Time (s)');
ylabel('Amplitude');


w = linspace(0,sample_freq*(num_samp-1)/num_samp,5*num_samp);

wave_fft = abs(fft(total_wave));
subplot(2,1,2);
hold on;
plot(w(1:5*(num_samp/2)+1),wave_fft(1:5*(num_samp/2)+1));
grid;
title('DFT of the signal')
xlabel('Frequency (Hz)');
ylabel('Magnitude');

frequency_deviation = 200;

total_wave = [];
total_time = [];
for i= 1:5
    [wave,time] = wave_generator(wavetype, wave_freq, sample_freq,
 amplitude, phase, frequency_hop, noise,0,frequency_deviation);
    total_wave = [total_wave,wave];
    if i == 1
        total_time = [total_time,time];
    end
    if i > 1
```

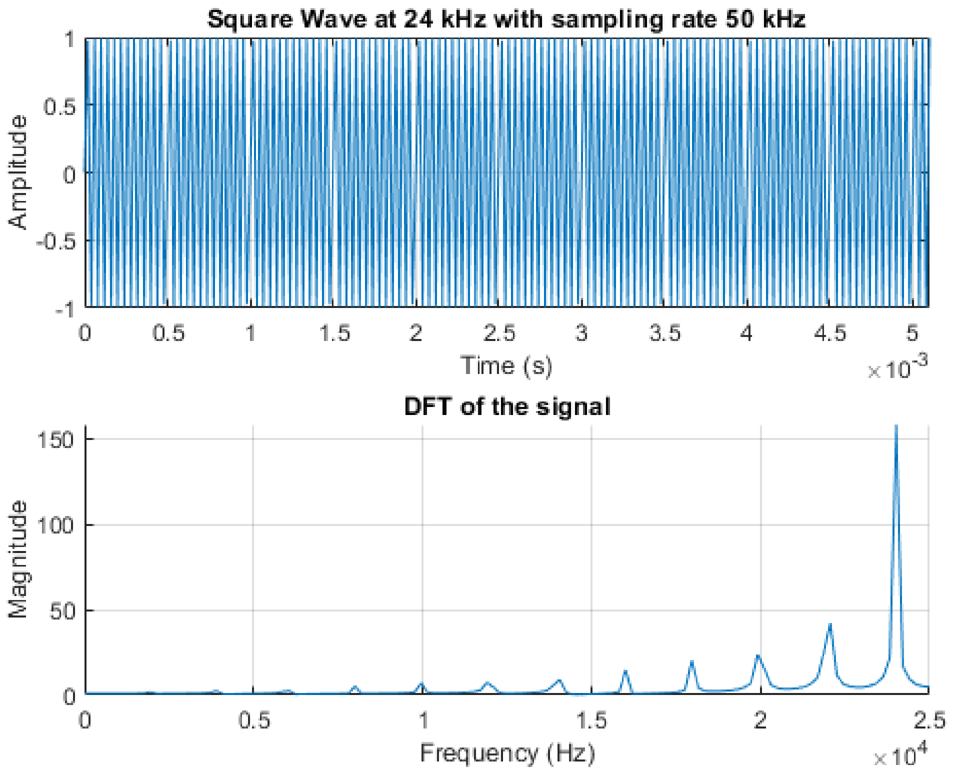
```
        total_time =  [total_time,time+(i-1)*time(end)+1/sample_freq];
    end
end



figure;
subplot(2,1,1);
plot(total_time,total_wave);
hold on;
grid;
title('Sine Wave at 1 kHz with sampling rate 50 kHz, frequency
 deviation in range 200 Hz');
xlim([0,total_time(end)]);
xlabel('Time (s)');
ylabel('Amplitude');


w = linspace(0,sample_freq*(num_samp-1)/num_samp,5*num_samp);

wave_fft = abs(fft(total_wave));
subplot(2,1,2);
hold on;
plot(w(1:5*(num_samp/2)+1),wave_fft(1:5*(num_samp/2)+1));
grid;
title('DFT of the signal')
xlabel('Frequency (Hz)');
ylabel('Magnitude');
```


Sine Wave at 1 kHz with sampling rate 50 kHz, frequency deviation in range 100-500 Hz
Amplitude
Time (s)
DFT of the signal
Magnitude
Frequency (Hz)
$\times 10^4$


Sine Wave at 1 kHz with sampling rate 50 kHz, frequency deviation in range 200 Hz
Amplitude
Time (s)
DFT of the signal
Magnitude
Frequency (Hz)
$\times 10^4$

Task 5

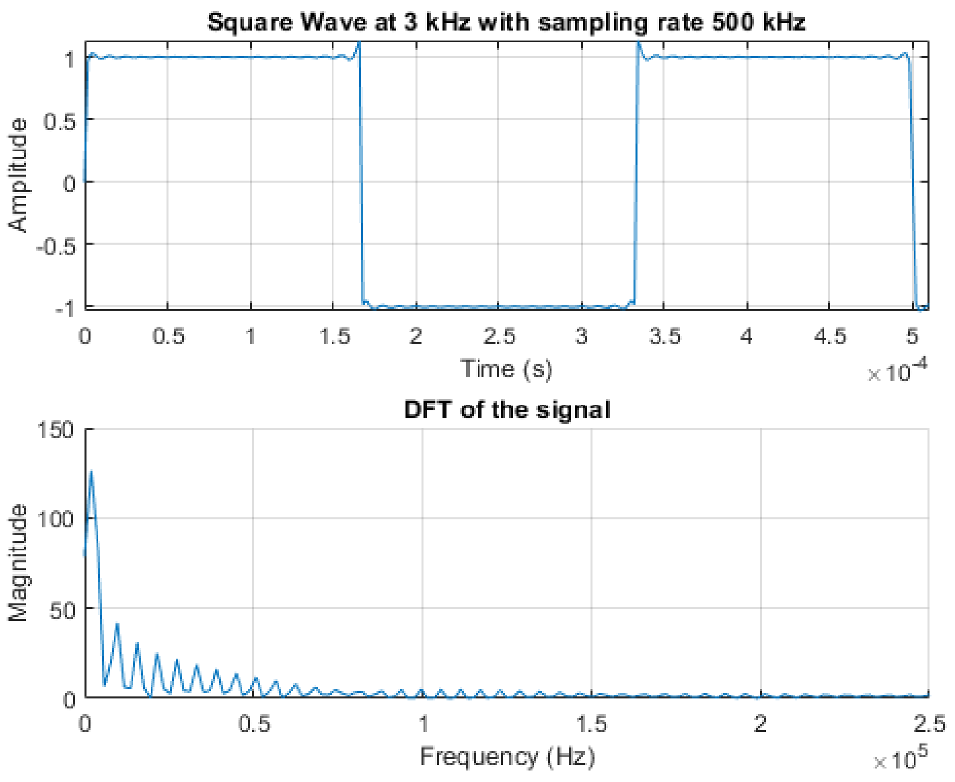
```
wavetype = 'square';
frequency_hop = false;

sample_freq = 50*1e3;
wave_freq = 24*1e3;

[wave,time] = wave_generator(wavetype, wave_freq, sample_freq,
 amplitude, phase, frequency_hop, noise);

figure;
subplot(2,1,1);
plot(time,wave);
hold on;
grid;
title('Square Wave at 24 kHz with sampling rate 50 kHz');
xlim([0,time(end)]);
xlabel('Time (s)');
ylabel('Amplitude');

w = linspace(0,sample_freq*(num_samp-1)/num_samp,num_samp);

wave_fft = abs(fft(wave));
subplot(2,1,2);
hold on;
plot(w(1:num_samp/2+1),wave_fft(1:num_samp/2+1));
grid;
title('DFT of the signal')
xlabel('Frequency (Hz)');
ylabel('Magnitude');
```



Task 5-2

```
sample_freq = 500*1e3;
wave_freq = 3*1e3;


[wave,time] = wave_generator(wavetype, wave_freq, sample_freq,
 amplitude, phase, frequency_hop, noise);

figure;
subplot(2,1,1);
plot(time,wave);
hold on;
grid;
title('Square Wave at 3 kHz with sampling rate 500 kHz');
xlim([0,time(end)]);
xlabel('Time (s)');
ylabel('Amplitude');


w = linspace(0,sample_freq*(num_samp-1)/num_samp,num_samp);

wave_fft = abs(fft(wave));
subplot(2,1,2);
hold on;
```

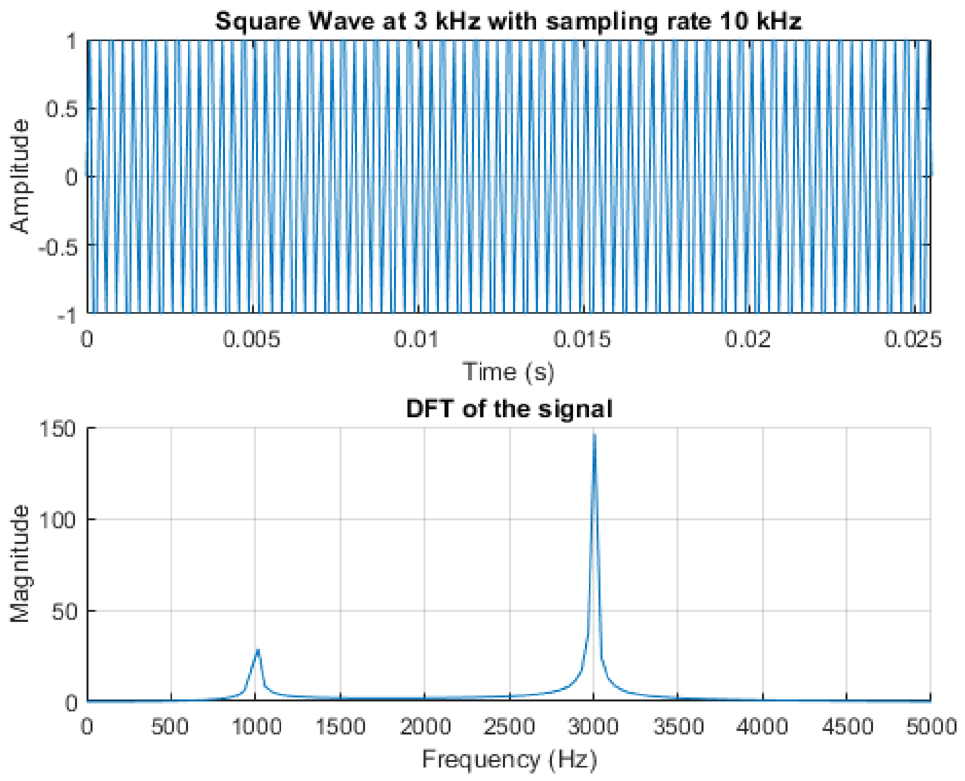
```
plot(w(1:num_samp/2+1),wave_fft(1:num_samp/2+1));
grid;
title('DFT of the signal')
xlabel('Frequency (Hz)');
ylabel('Magnitude');
```



Task 5-3

```
sample_freq = 10*1e3;
wave_freq = 3*1e3;


[wave,time] = wave_generator(wavetype, wave_freq, sample_freq,
 amplitude, phase, frequency_hop, noise);

figure;
subplot(2,1,1);
plot(time,wave);
hold on;
grid;
title('Square Wave at 3 kHz with sampling rate 10 kHz');
xlim([0,time(end)]);
xlabel('Time (s)');
ylabel('Amplitude');

w = linspace(0,sample_freq*(num_samp-1)/num_samp,num_samp);
```

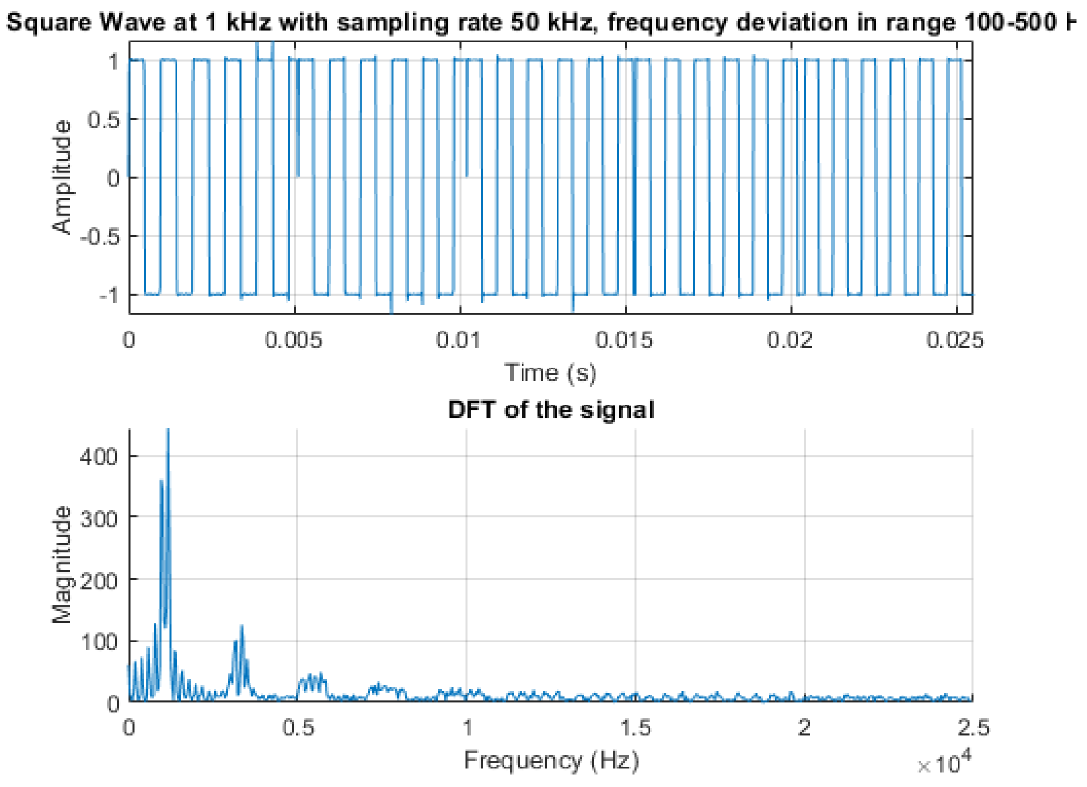
```
wave_fft = abs(fft(wave));
subplot(2,1,2);
hold on;
plot(w(1:num_samp/2+1),wave_fft(1:num_samp/2+1));
grid;
title('DFT of the signal')
xlabel('Frequency (Hz)');
ylabel('Magnitude');
```

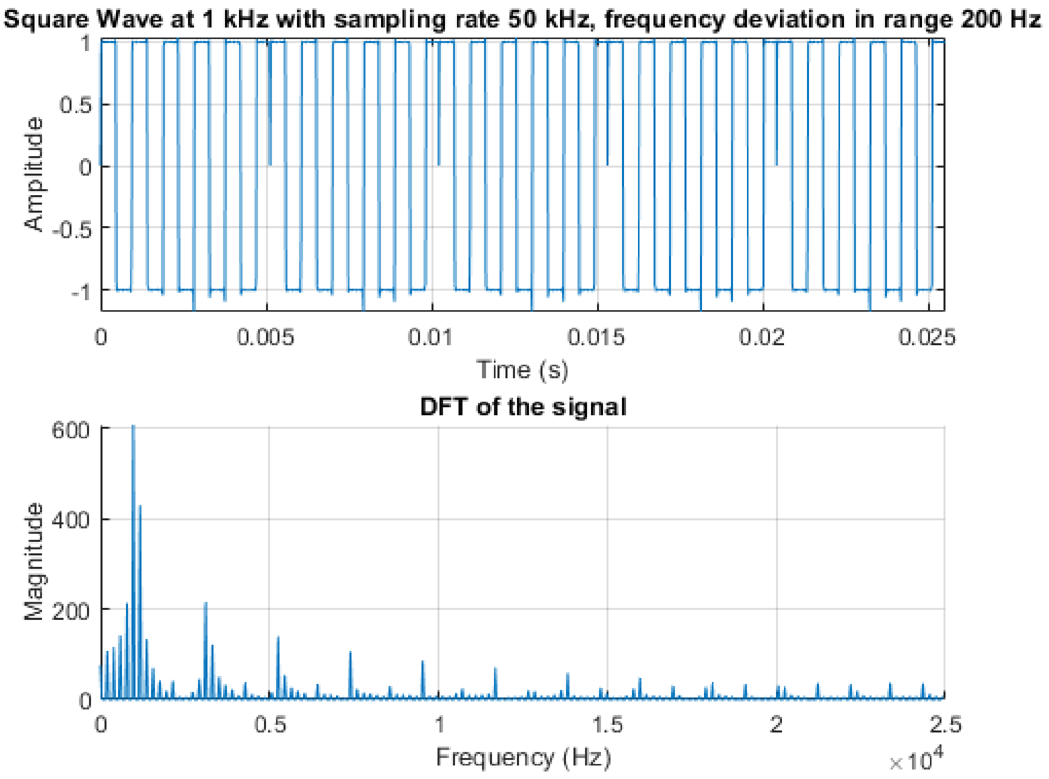


Task 5-4

```
frequency_hop = true;
sample_freq = 50*1e3;
wave_freq = 1*1e3;


total_wave = [];
total_time = [];
for deviate = 100:100:500
    [wave,time] = wave_generator(wavetype, wave_freq, sample_freq,
 amplitude, phase, frequency_hop, noise,0,deviate);
    total_wave = [total_wave,wave];
    if deviate-100 == 0
        total_time = [total_time,time];
    end
    if deviate - 100 > 0
       total_time =  [total_time,time+(deviate-100)/100*time(end)+1/
sample_freq];
    end
end


figure;
subplot(2,1,1);
plot(total_time,total_wave);
hold on;
grid;
title('Square Wave at 1 kHz with sampling rate 50 kHz, frequency
 deviation in range 100-500 Hz');
xlim([0,total_time(end)]);
xlabel('Time (s)');
ylabel('Amplitude');


w = linspace(0,sample_freq*(num_samp-1)/num_samp,5*num_samp);

wave_fft = abs(fft(total_wave));
subplot(2,1,2);
hold on;
plot(w(1:5*(num_samp/2)+1),wave_fft(1:5*(num_samp/2)+1));
grid;
title('DFT of the signal')
xlabel('Frequency (Hz)');
ylabel('Magnitude');

frequency_deviation = 200;

total_wave = [];
total_time = [];
for i= 1:5
    [wave,time] = wave_generator(wavetype, wave_freq, sample_freq,
 amplitude, phase, frequency_hop, noise,0,frequency_deviation);
    total_wave = [total_wave,wave];
    if i == 1
        total_time = [total_time,time];
    end
    if i > 1
       total_time =  [total_time,time+(i-1)*time(end)+1/sample_freq];
    end
end



figure;
subplot(2,1,1);
plot(total_time,total_wave);
hold on;
grid;
```

```
title('Square Wave at 1 kHz with sampling rate 50 kHz, frequency
 deviation in range 200 Hz');
xlim([0,total_time(end)]);
xlabel('Time (s)');
ylabel('Amplitude');


w = linspace(0,sample_freq*(num_samp-1)/num_samp,5*num_samp);

wave_fft = abs(fft(total_wave));
subplot(2,1,2);
hold on;
plot(w(1:5*(num_samp/2)+1),wave_fft(1:5*(num_samp/2)+1));
grid;
title('DFT of the signal')
xlabel('Frequency (Hz)');
ylabel('Magnitude');
```

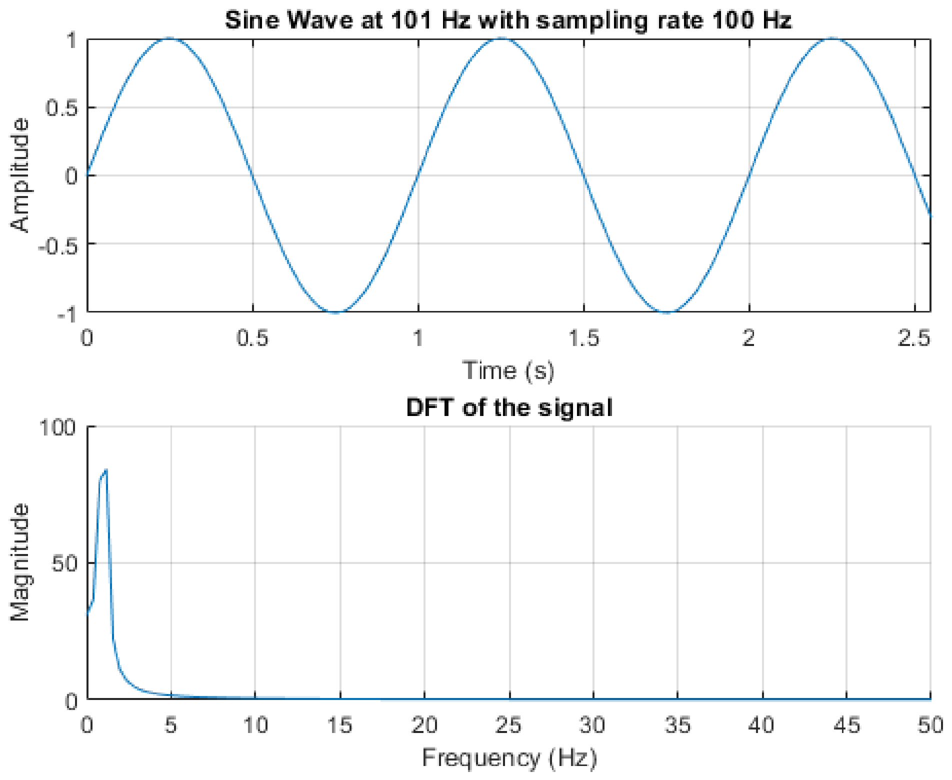




Task 6

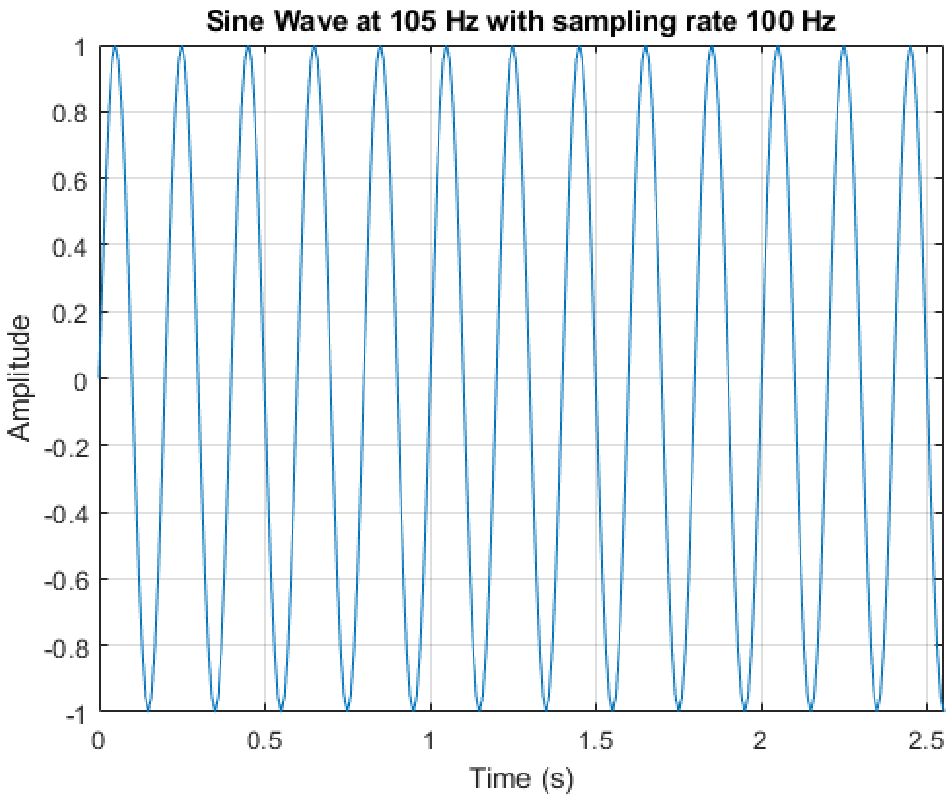
```
wavetype = 'sine';
sample_freq = 100;
wave_freq = 101;
frequency_hop = false;

[wave,time] = wave_generator(wavetype, wave_freq, sample_freq,
 amplitude, phase, frequency_hop, noise);

figure;
subplot(2,1,1);
plot(time,wave);
hold on;
grid;
title('Sine Wave at 101 Hz with sampling rate 100 Hz');
xlim([0,time(end)]);
xlabel('Time (s)');
ylabel('Amplitude');


w = linspace(0,sample_freq*(num_samp-1)/num_samp,num_samp);

wave_fft = abs(fft(wave));
subplot(2,1,2);
```

```
hold on;
plot(w(1:num_samp/2+1),wave_fft(1:num_samp/2+1));
grid;
title('DFT of the signal')
xlabel('Frequency (Hz)');
ylabel('Magnitude');
```



Task 7

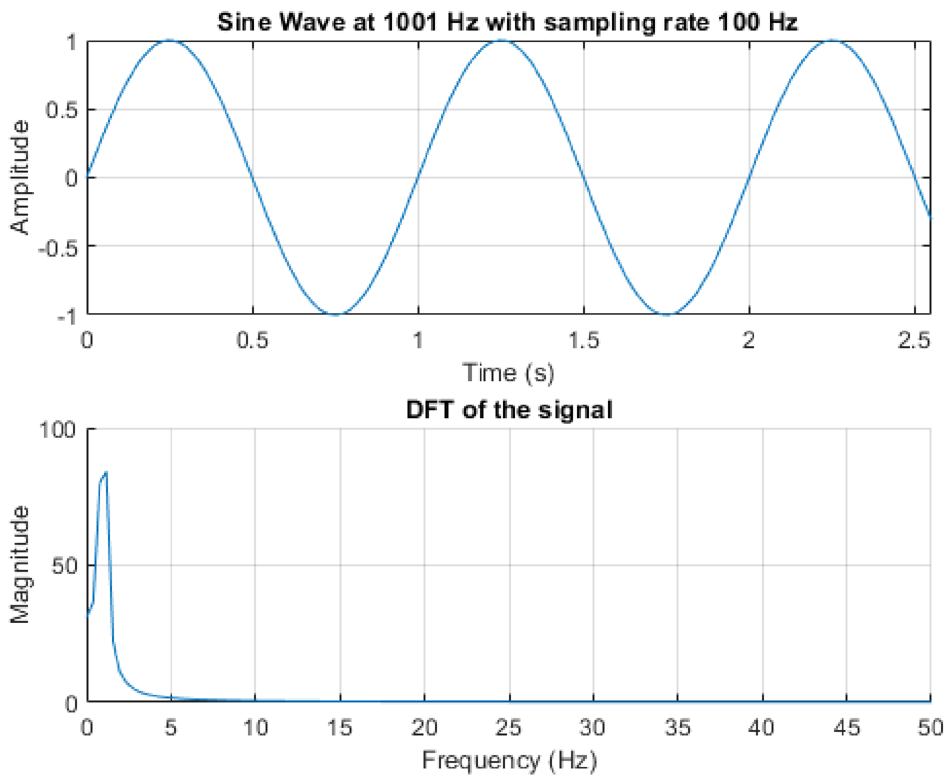
```
sample_freq = 100;
wave_freq = 105;


[wave,time] = wave_generator(wavetype, wave_freq, sample_freq,
 amplitude, phase, frequency_hop, noise);

figure;
plot(time,wave);
hold on;
grid;
title('Sine Wave at 105 Hz with sampling rate 100 Hz');
xlim([0,time(end)]);
xlabel('Time (s)');
ylabel('Amplitude');
```



Task 8

```
sample_freq = 100;
wave_freq = 1001;


[wave,time] = wave_generator(wavetype, wave_freq, sample_freq,
 amplitude, phase, frequency_hop, noise);

figure;
subplot(2,1,1);
plot(time,wave);
hold on;
grid;
title('Sine Wave at 1001 Hz with sampling rate 100 Hz');
xlim([0,time(end)]);
xlabel('Time (s)');
ylabel('Amplitude');


w = linspace(0,sample_freq*(num_samp-1)/num_samp,num_samp);

wave_fft = abs(fft(wave));
subplot(2,1,2);
hold on;
```

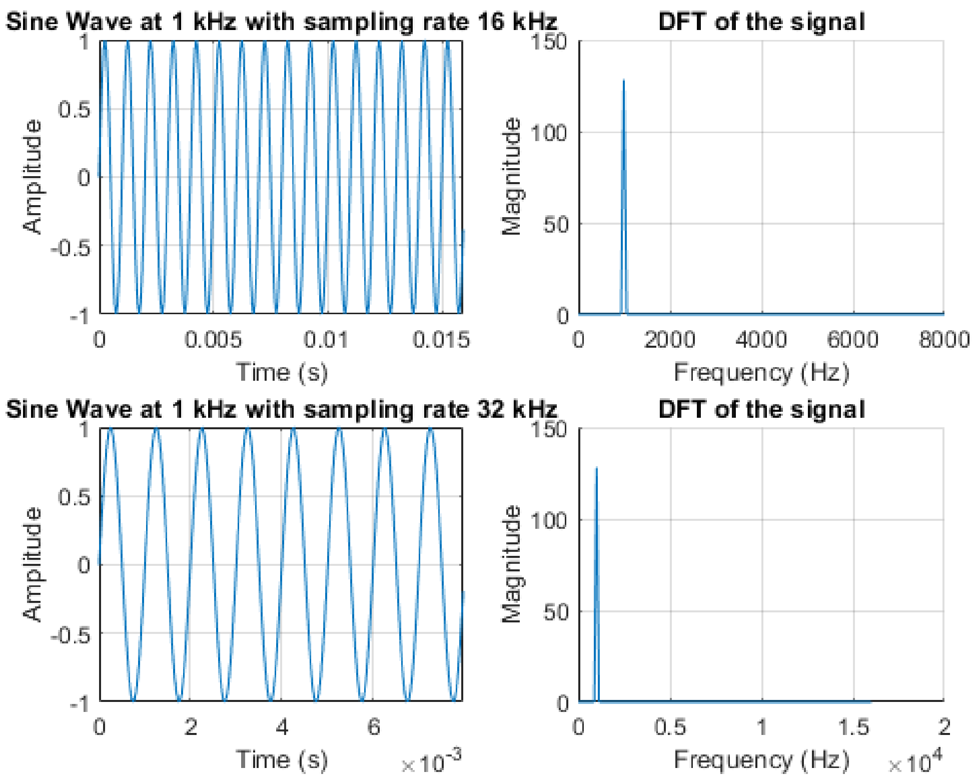
```
plot(w(1:num_samp/2+1),wave_fft(1:num_samp/2+1));
grid;
title('DFT of the signal')
xlabel('Frequency (Hz)');
ylabel('Magnitude');
```



Task 9

```
sample_freq = 8*1e3;
wave_freq = 1*1e3;


[wave,time] = wave_generator(wavetype, wave_freq, sample_freq,
 amplitude, phase, frequency_hop, noise);
wave_freq91 =  estimate_frequency(wave,sample_freq)


sample_freq = 16*1e3;
wave_freq = 1*1e3;


[wave,time] = wave_generator(wavetype, wave_freq, sample_freq,
 amplitude, phase, frequency_hop, noise);
wave_freq92 =  estimate_frequency(wave,sample_freq)

figure;
```

```
subplot(2,2,1);
plot(time,wave);
hold on;
grid;
title('Sine Wave at 1 kHz with sampling rate 16 kHz');
xlim([0,time(end)]);
xlabel('Time (s)');
ylabel('Amplitude');


w = linspace(0,sample_freq*(num_samp-1)/num_samp,num_samp);

wave_fft = abs(fft(wave));
subplot(2,2,2);
hold on;
plot(w(1:num_samp/2+1),wave_fft(1:num_samp/2+1));
grid;
title('DFT of the signal')
xlabel('Frequency (Hz)');
ylabel('Magnitude');

sample_freq = 32*1e3;
wave_freq = 1*1e3;


[wave,time] = wave_generator(wavetype, wave_freq, sample_freq,
 amplitude, phase, frequency_hop, noise);
wave_freq93 =  estimate_frequency(wave,sample_freq)



subplot(2,2,3);
plot(time,wave);
hold on;
grid;
title('Sine Wave at 1 kHz with sampling rate 32 kHz');
xlim([0,time(end)]);
xlabel('Time (s)');
ylabel('Amplitude');


w = linspace(0,sample_freq*(num_samp-1)/num_samp,num_samp);

wave_fft = abs(fft(wave));
subplot(2,2,4);
hold on;
plot(w(1:num_samp/2+1),wave_fft(1:num_samp/2+1));
grid;
title('DFT of the signal')
xlabel('Frequency (Hz)');
ylabel('Magnitude');


wave_freq91 =
        1000


wave_freq92 =

        1000


wave_freq93 =

        1000
```



Task 10

```
sample_freq = 16*1e3;
wave_freq = 1*1e3;
noise = true;

noise_std = linspace(0,15,5);
wave_freq_arr = zeros(size(noise_std));
SNR = zeros(size(noise_std));

for i = 1:length(noise_std)
```

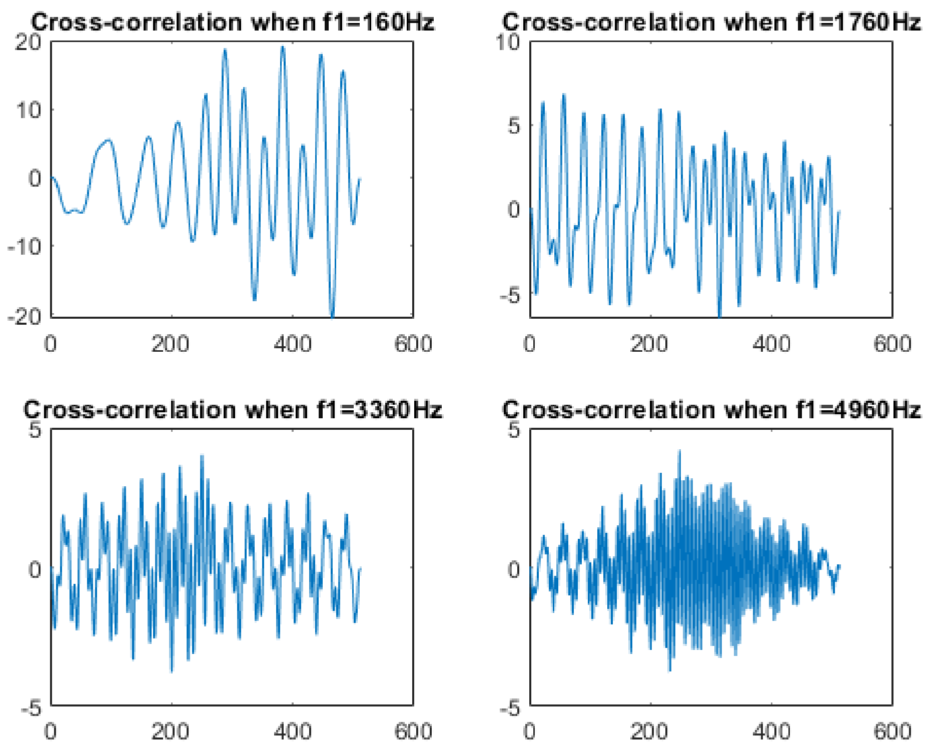
```
    [wave,time] = wave_generator(wavetype, wave_freq, sample_freq,
 amplitude, phase, frequency_hop, noise,noise_std(i));
    wave_freq_arr(i)=  estimate_frequency(wave,sample_freq)
    SNR(i) = estimate_SNR(wavetype,noise_std(i))

end


wave_freq_arr =

        1000           0           0           0           0
```

Task 10-2

```
noise_std = 3.8874;

[wave,time] = wave_generator(wavetype, wave_freq, sample_freq,
 amplitude, phase, frequency_hop, noise,noise_std);
wave_freq_arr102=  estimate_frequency(wave,sample_freq)
SNR102 = estimate_SNR(wavetype,noise_std)


wave_freq_arr102 =

        1000


SNR102 =

  -14.8035
```

Task 11

```
noise_std = 0.1;
end_freq = 6400;
sample_freq = 32*1e3;
noise_std = 0.1;


figure;
i=1;
for start_freq = 160:1600:6400

    [wave,time] = wave_generator(wavetype, wave_freq, sample_freq,
 amplitude, phase, frequency_hop, noise,noise_std);
    chirp_wave = sin(2*pi*start_freq*time+pi*(end_freq-start_freq)/
(80*1e-3)*time.^2);

    subplot(2,2, i);
    plot(xcorr(chirp_wave,wave));
    hold on;
    title(['Cross-correlation when f1=',num2str(start_freq),'Hz']);

    i= i + 1;
end
```



Task 12

```
frequency_hop = true;

total_wave = [];
total_time = [];


noise_std = linspace(0,15,5);
wave_freq_arr12 = zeros(size(noise_std));
SNR12 = zeros(size(noise_std));

wave_freq = 1*1e3;

frequency_deviation = 50;

for i = 1:length(noise_std)


    for j= 1:5
```

```
        [wave,time] = wave_generator(wavetype,
 wave_freq, sample_freq, amplitude, phase, frequency_hop,
 noise,noise_std(i),frequency_deviation);
        total_wave = [total_wave,wave];
        if j == 1
            total_time = [total_time,time];
        end
        if j> 1
            total_time =  [total_time,time+(j-1)*time(end)+1/
sample_freq];
        end
    end


    wave_freq_arr12(i)=  estimate_frequency(total_wave,sample_freq)
    SNR12(i) = estimate_SNR(wavetype,noise_std(i))

end


wave_freq_arr12 =

        1000            0            0            0            0


SNR12 =

   Inf      0      0      0      0


wave_freq_arr12 =

        1000         1000            0            0            0


SNR12 =

       Inf   -14.4909         0         0         0


wave_freq_arr12 =

        1000         1000        15625            0            0


SNR12 =

       Inf   -14.4909   -20.5115         0         0


wave_freq_arr12 =

        1000         1000        15625        15625            0
```

```
SNR12 =

       Inf   -14.4909   -20.5115   -24.0334          0


wave_freq_arr12 =

        1000         1000        15625        15625        15625


SNR12 =

       Inf   -14.4909   -20.5115   -24.0334   -26.5321
```

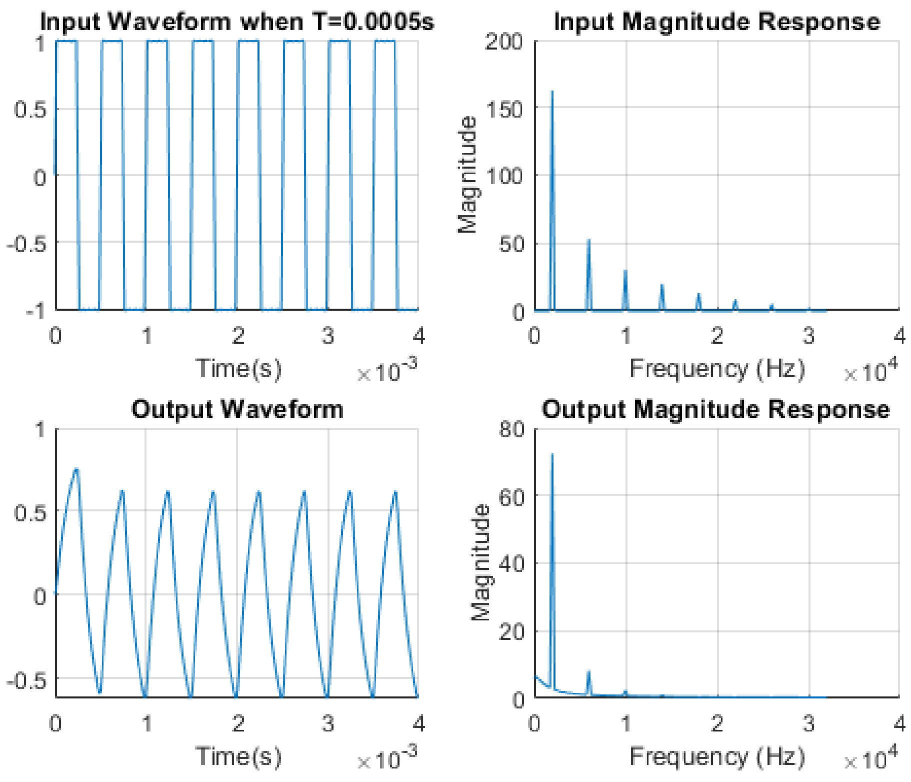
Task 12-2

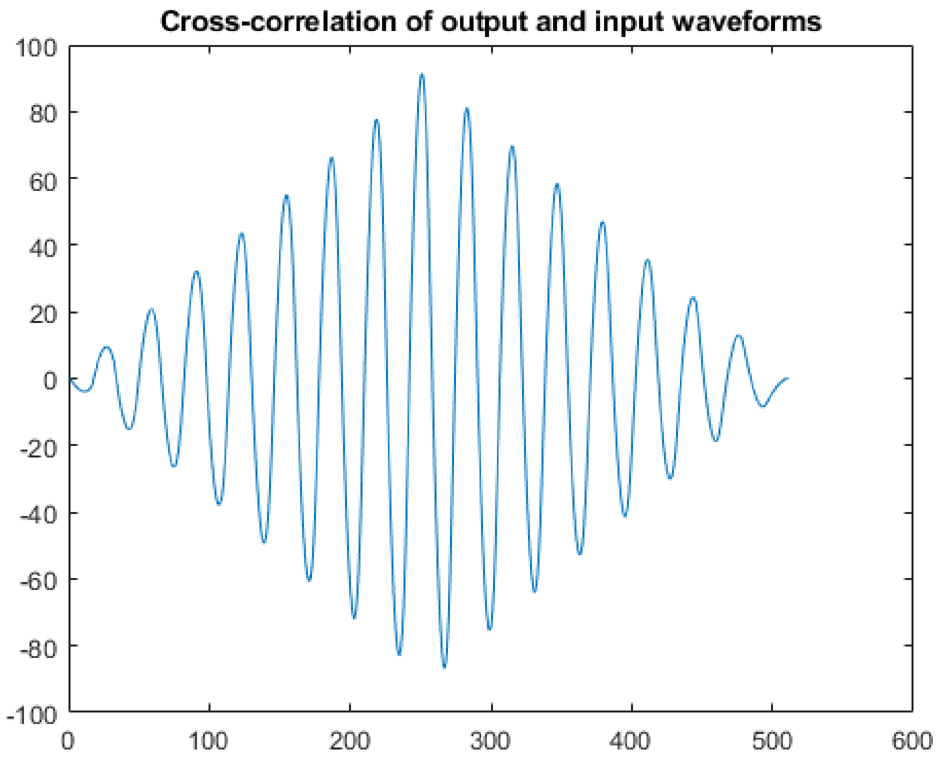
```
noise_std = 2.77


total_wave = [];
total_time = [];
for j= 1:5
    [wave,time] = wave_generator(wavetype, wave_freq,
 sample_freq, amplitude, phase, frequency_hop,
 noise,noise_std,frequency_deviation);
    total_wave = [total_wave,wave];
    if j == 1
        total_time = [total_time,time];
    end
    if j> 1
        total_time =  [total_time,time+(j-1)*time(end)+1/sample_freq];
    end
end

wave_freq_arr122=  estimate_frequency(total_wave,sample_freq)
SNR122 = estimate_SNR(wavetype,noise_std)


noise_std =

    2.7700


wave_freq_arr122 =

        1000


SNR122 =

  -11.8599
```

Task 13

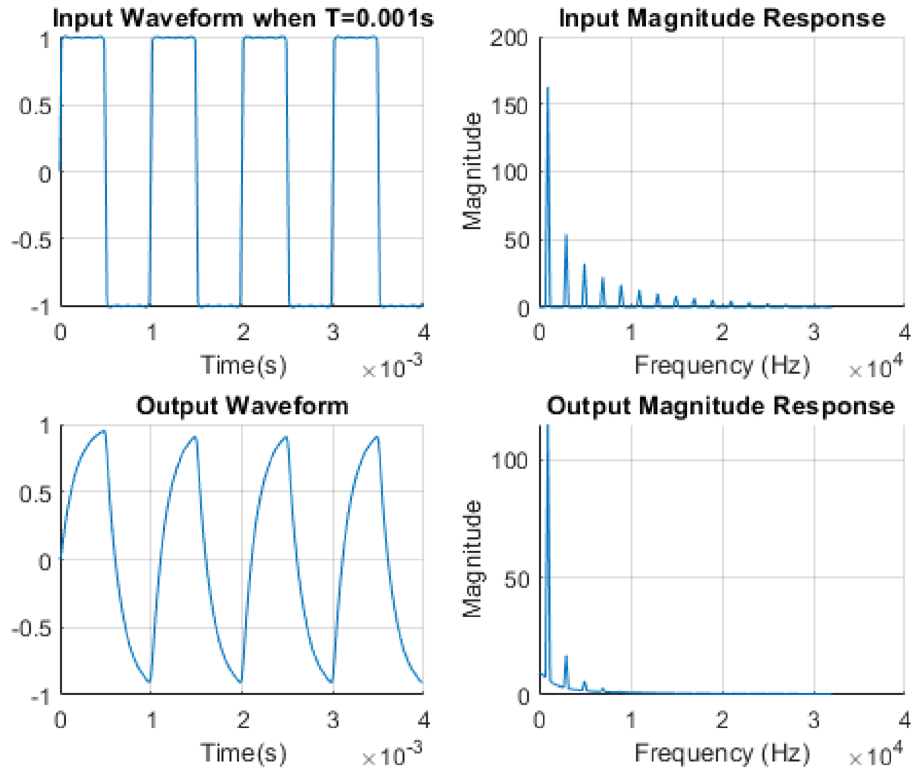
```
wavetype = 'square';

sample_freq = 64*1e3;
amplitude = 1;
phase = 0;
frequency_hop = false;
noise = false;

order = 1;
cutoff = wave_freq/(sample_freq/2);

T_arr = [0.5,1,2]*1e-3;
[B,A] = butter(order,cutoff,'low');

for T =T_arr
    wave_freq = 1/T;
    [wave,time] = wave_generator(wavetype, wave_freq, sample_freq,
 amplitude, phase, frequency_hop, noise);


    output = filter(B,A,wave);

    figure;
    subplot(2,2,1);
    title(['Input Waveform when T=',num2str(T),'s']);
    hold on;
    grid;
    plot(time,wave);
    xlabel('Time(s)');

    w = linspace(0,sample_freq*(num_samp-1)/num_samp,num_samp);

    wave_fft = abs(fft(wave));
    subplot(2,2,2);
    hold on;
    plot(w(1:num_samp/2+1),wave_fft(1:num_samp/2+1));
    grid;
    title('Input Magnitude Response')
    xlabel('Frequency (Hz)');
    ylabel('Magnitude');


    subplot(2,2,3);
    title('Output Waveform');
    hold on;
    grid;
    plot(time,output);
    xlabel('Time(s)');
```

```
        w = linspace(0,sample_freq*(num_samp-1)/num_samp,num_samp);

        wave_fft = abs(fft(output));
        subplot(2,2,4);
        hold on;
        grid;
        plot(w(1:num_samp/2+1),wave_fft(1:num_samp/2+1));

        title('Output Magnitude Response')
        xlabel('Frequency (Hz)');
        ylabel('Magnitude');



        if T == 0.5*1e-3

            figure;
            plot(xcorr(wave,output));
            hold on;
            title('Cross-correlation of output and input waveforms');


        end
    end
```

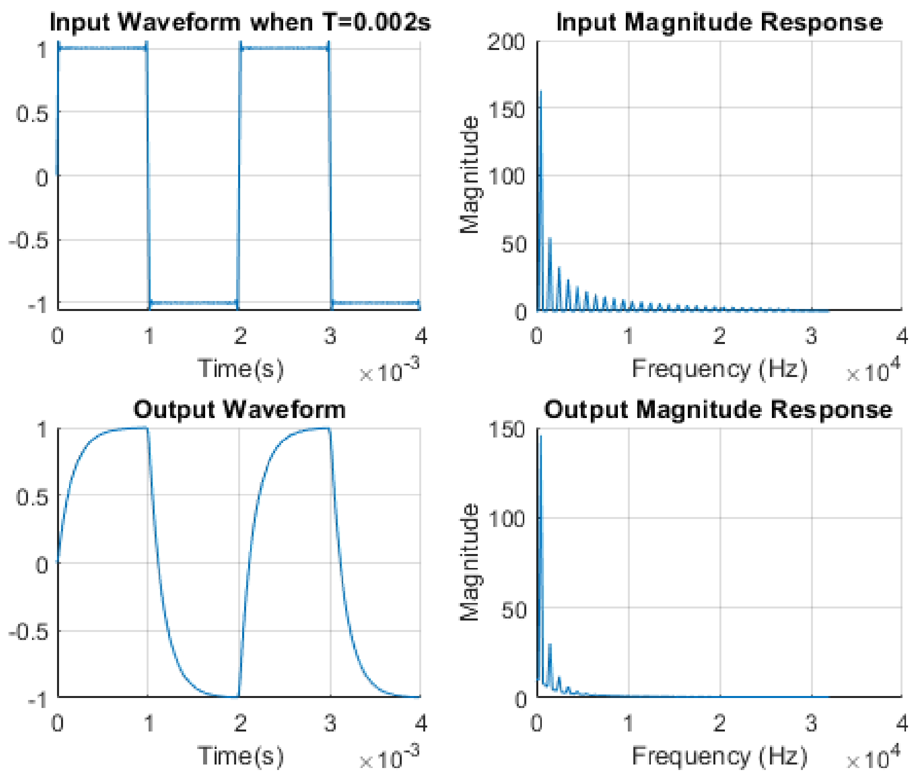




Cross-correlation of output and input waveforms
100
80
60
40
20
0
-20
-40
-60
-80
-100
0
100
200
300
400
500
600

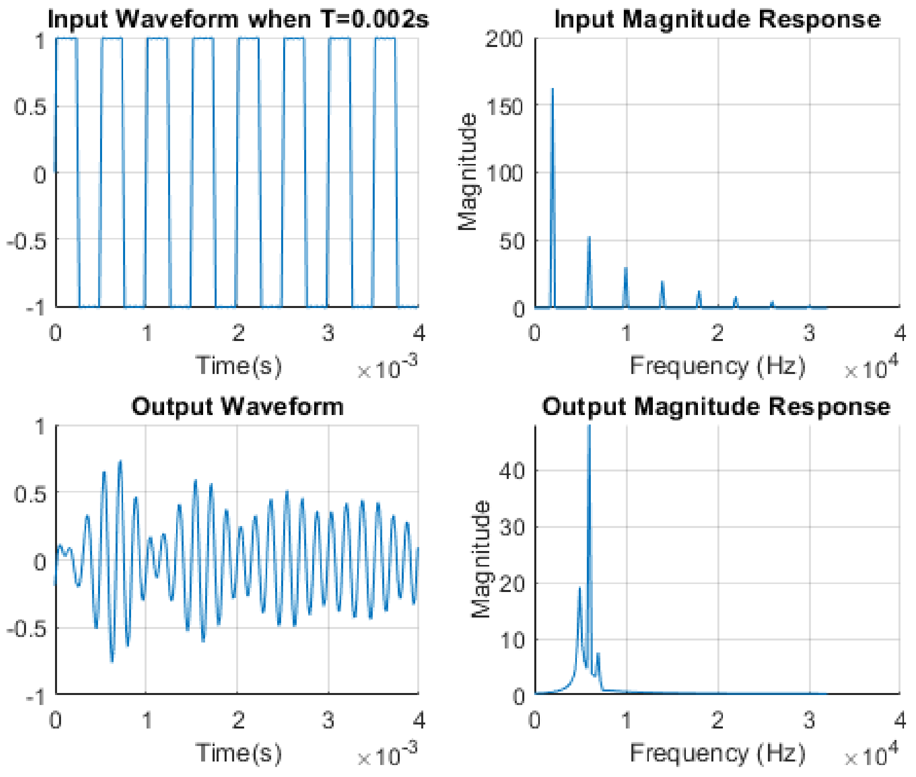


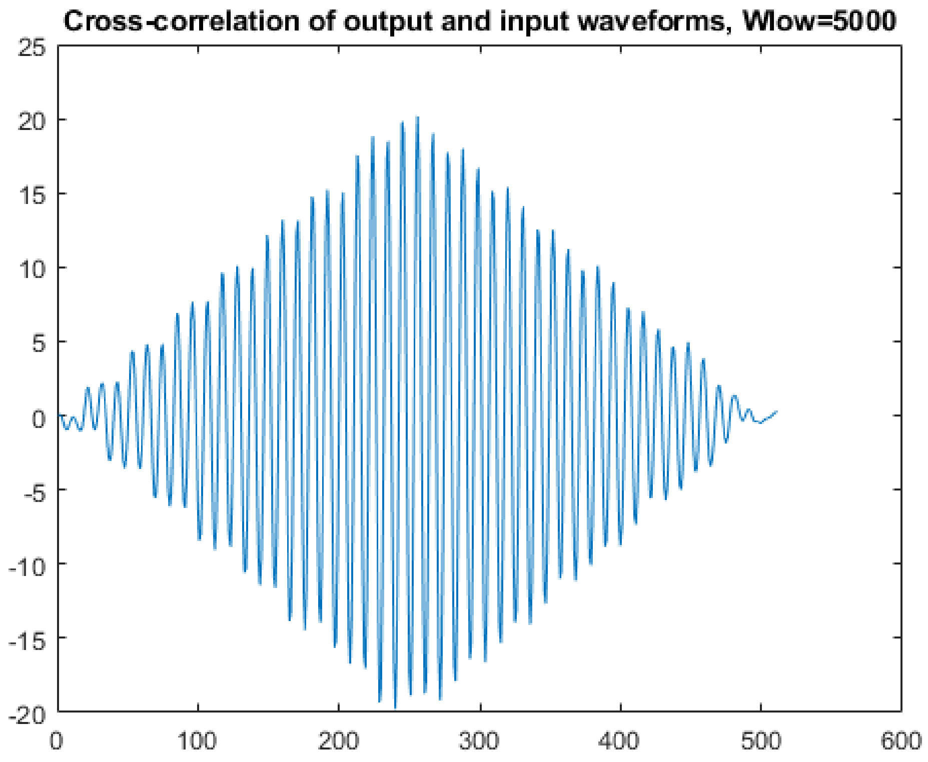
Input Waveform when T=0.001s
Time(s)
×10^-3
Input Magnitude Response
Magnitude
Frequency (Hz)
×10^4
Output Waveform
Time(s)
×10^-3
Output Magnitude Response
Magnitude
Frequency (Hz)
×10^4

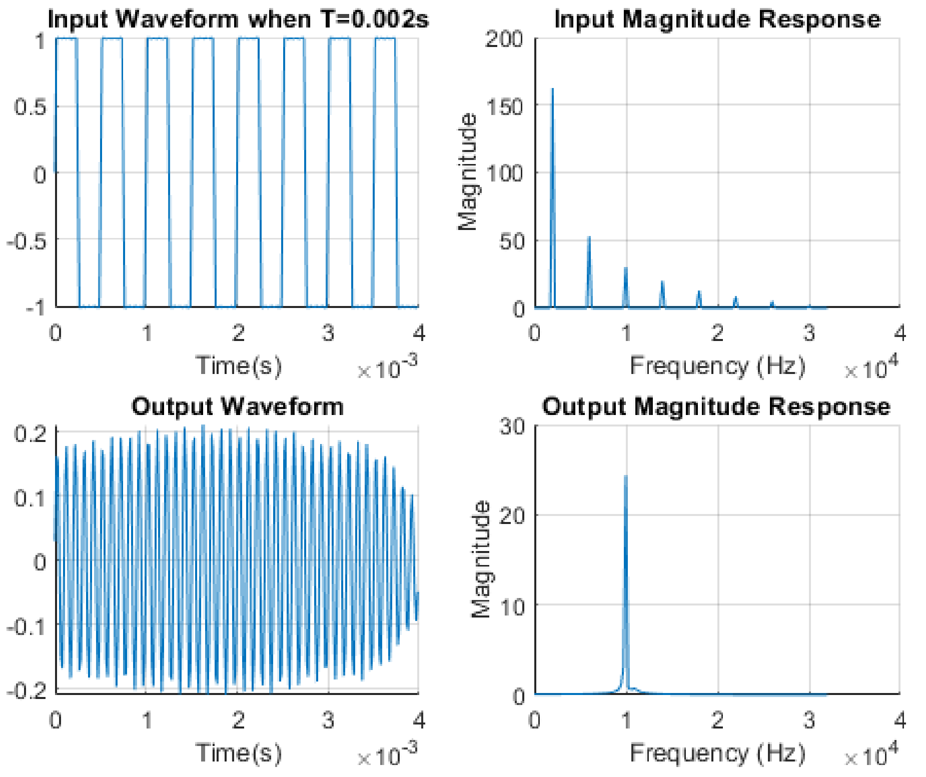


Task 14

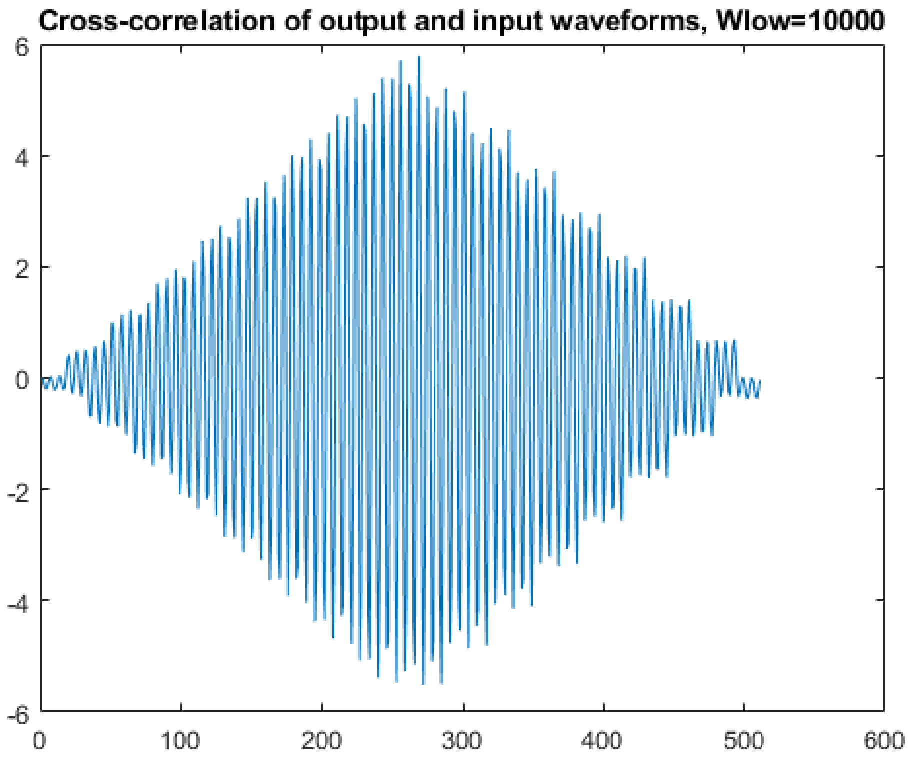
```
noise = true;
wave_freq = 2*1e3;
noise_std = 0;

order = 1;
cutoff = wave_freq/(sample_freq/2);

[wave,time] = wave_generator(wavetype, wave_freq, sample_freq,
 amplitude, phase, frequency_hop, noise,noise_std);

Wlow = [5,10,13]*1e3;
Whigh = [7,11,15]*1e3;



for i = 1 : length(Whigh)

    output= bandpass(wave,[Wlow(i),Whigh(i)],sample_freq);

    figure;
    subplot(2,2,1);
    title(['Input Waveform when T=',num2str(T),'s']);
    hold on;
    grid;
    plot(time,wave);
    xlabel('Time(s)');

    w = linspace(0,sample_freq*(num_samp-1)/num_samp,num_samp);

    wave_fft = abs(fft(wave));
    subplot(2,2,2);
    hold on;
    plot(w(1:num_samp/2+1),wave_fft(1:num_samp/2+1));
    grid;
    title('Input Magnitude Response')
    xlabel('Frequency (Hz)');
    ylabel('Magnitude');


    subplot(2,2,3);
    title('Output Waveform');
    hold on;
    grid;
    plot(time,output);
    xlabel('Time(s)');

    w = linspace(0,sample_freq*(num_samp-1)/num_samp,num_samp);

    wave_fft = abs(fft(output));
    subplot(2,2,4);
    hold on;
    grid;
    plot(w(1:num_samp/2+1),wave_fft(1:num_samp/2+1));

    title('Output Magnitude Response')
    xlabel('Frequency (Hz)');
    ylabel('Magnitude');


    figure;
    plot(xcorr(wave,output));
    hold on;
    title(['Cross-correlation of output and input waveforms,
 Wlow=',num2str(Wlow(i))]);


end
```


Input Waveform when T=0.002s
Time(s)
×10^-3
Input Magnitude Response
Magnitude
Frequency (Hz)
×10^4
Output Waveform
Time(s)
×10^-3
Output Magnitude Response
Magnitude
Frequency (Hz)
×10^4

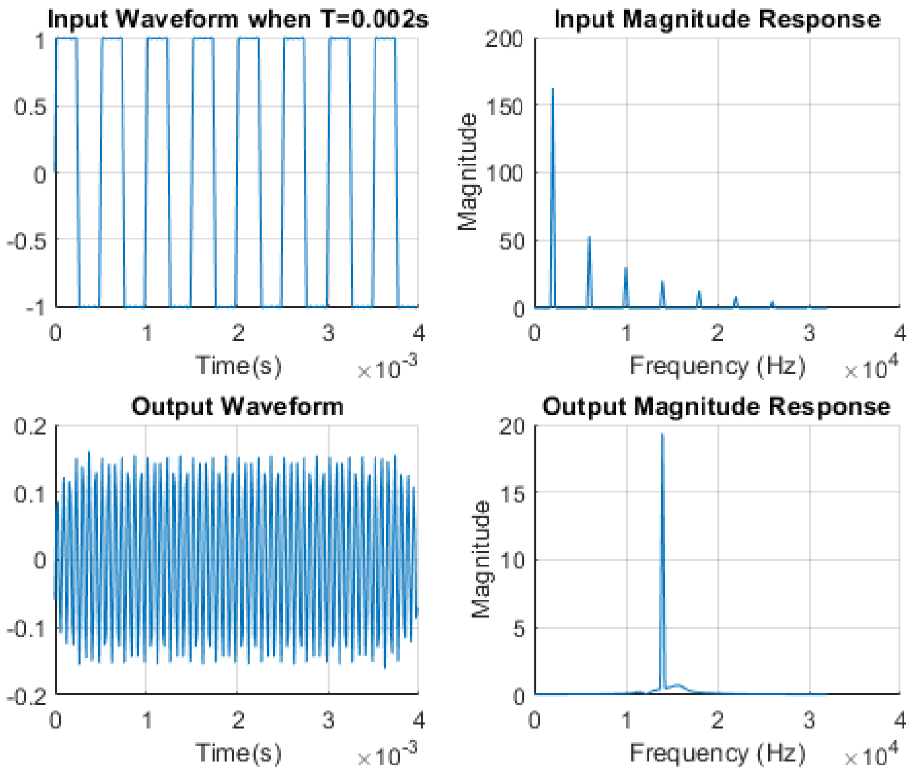

Cross-correlation of output and input waveforms, Wlow=5000

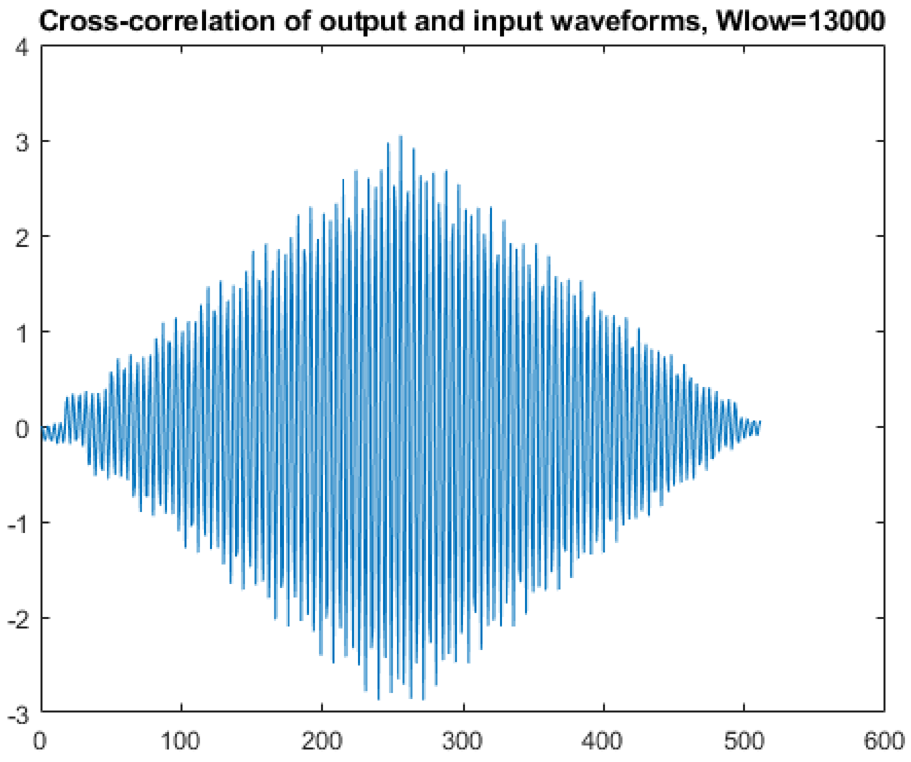


Input Waveform when T=0.002s
Time(s)
×10^-3
Input Magnitude Response
Magnitude
Frequency (Hz)
×10^4
Output Waveform
Time(s)
×10^-3
Output Magnitude Response
Magnitude
Frequency (Hz)
×10^4



Cross-correlation of output and input waveforms, Wlow=10000

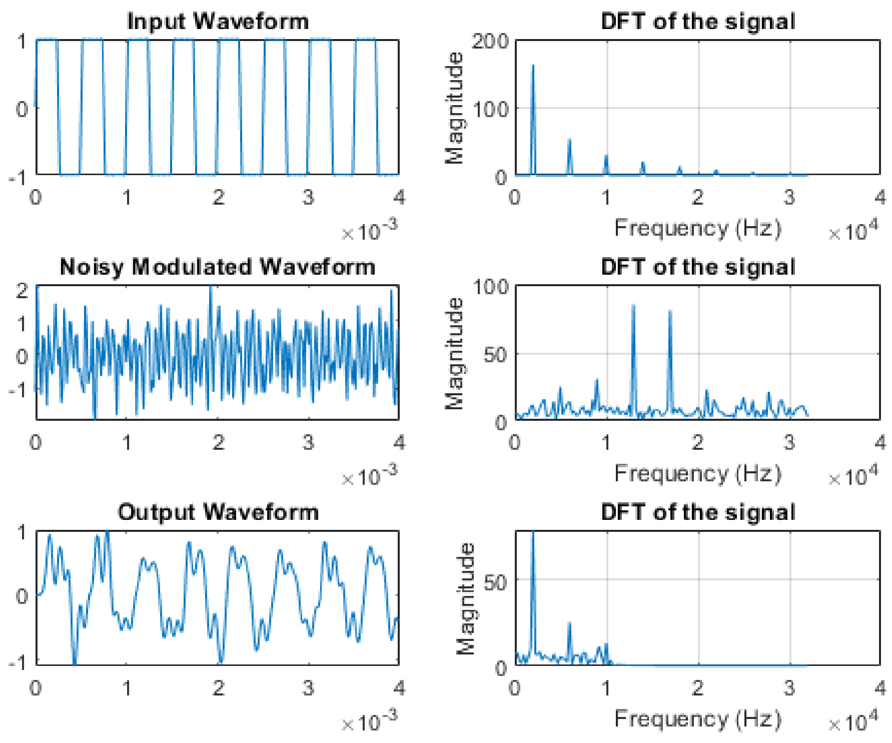

Input Waveform when T=0.002s
Time(s)
×10^-3
Input Magnitude Response
Magnitude
Frequency (Hz)
×10^4
Output Waveform
Time(s)
×10^-3
Output Magnitude Response
Magnitude
Frequency (Hz)
×10^4


Cross-correlation of output and input waveforms, Wlow=13000

Task 15

```
wavetype = 'square';
wave_freq = 2*1e3;
sample_freq = 64*1e3;
amplitude = 1;
phase = 0;
frequency_hop = false;
noise = false;

[wave,~] = wave_generator(wavetype, wave_freq, sample_freq, amplitude,
 phase, frequency_hop, noise);




figure;
subplot(3,2,1);
plot(time,wave);
hold on;
title('Input Waveform');

w = linspace(0,sample_freq*(num_samp-1)/num_samp,num_samp);

wave_fft = abs(fft(wave));
hold on;

subplot(3,2,2);
plot(w(1:num_samp/2+1),wave_fft(1:num_samp/2+1));
grid;
title('DFT of the signal')
xlabel('Frequency (Hz)');
ylabel('Magnitude');


wavetype = 'sine';
wave_freq = 15*1e3;
noise = false;
noise_std = 0;

[wave_sine,time] = wave_generator(wavetype, wave_freq, sample_freq,
 amplitude, phase, frequency_hop, noise,noise_std);
add_noise = 0.5* randn(size(wave));




wave = wave_sine.*wave + add_noise;

subplot(3,2,3);
plot(time,wave);
hold on;
title('Noisy Modulated Waveform');
```

```
w = linspace(0,sample_freq*(num_samp-1)/num_samp,num_samp);

wave_fft = abs(fft(wave));
hold on;
subplot(3,2,4);
plot(w(1:num_samp/2+1),wave_fft(1:num_samp/2+1));
grid;
title('DFT of the signal')
xlabel('Frequency (Hz)');
ylabel('Magnitude');



order = 12;
output = bandpass(wave,[4500,25500],sample_freq);

output = output .*wave_sine;
[B,A] = butter(order,20002/sample_freq,'low');
output = filter (B,A,output);


subplot(3,2,5);
plot(time,output);
hold on;
title('Output Waveform');

w = linspace(0,sample_freq*(num_samp-1)/num_samp,num_samp);

wave_fft = abs(fft(output));
hold on;
subplot(3,2,6);
plot(w(1:num_samp/2+1),wave_fft(1:num_samp/2+1));
grid;
title('DFT of the signal')
xlabel('Frequency (Hz)');
ylabel('Magnitude');
```



Auxiliary Functions

```
function SNR = estimate_SNR(wavetype,noise_std)

if strcmp(wavetype,'sine')
    SNR = 10*log10(0.5*1/noise_std^2);
end

if strcmp(wavetype,'square')
    SNR = 10*log10(1/noise_std^2);
end
end

function wave_freq = estimate_frequency(wave,sample_freq)

dft_wave = abs(fft(wave));

dex = 0:(length(dft_wave)-1);
dex = dex(max(dft_wave)==dft_wave);
dex= dex(1);

wave_freq = dex/length(dft_wave)*sample_freq;


end
```