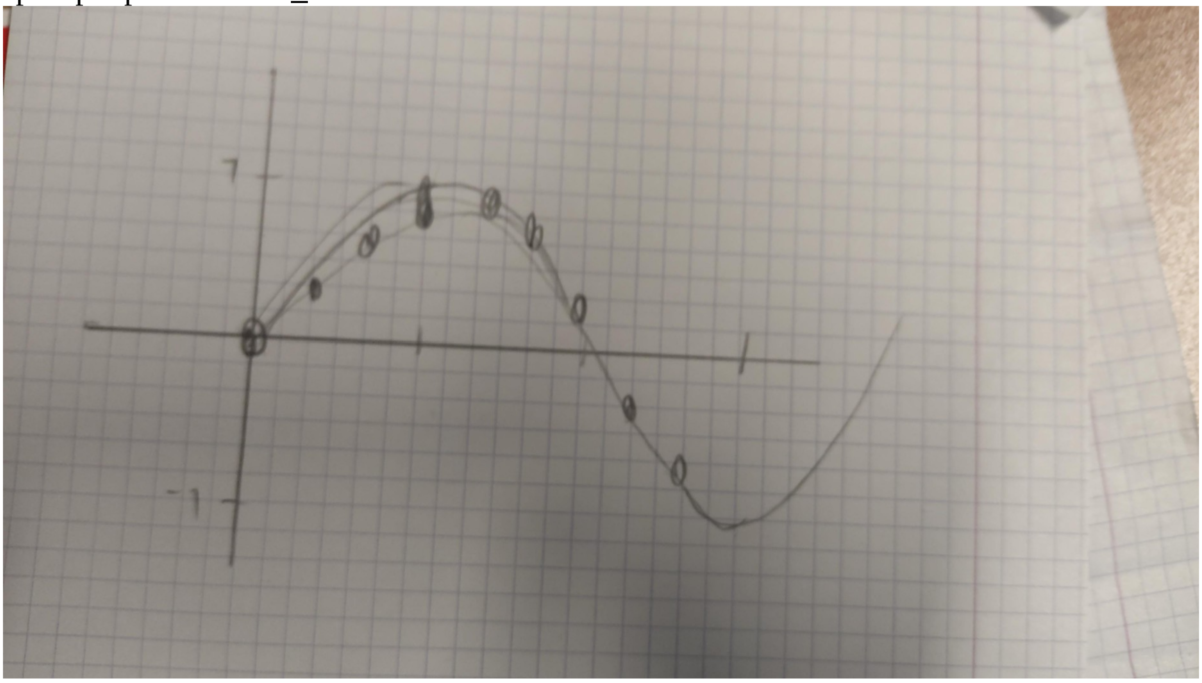
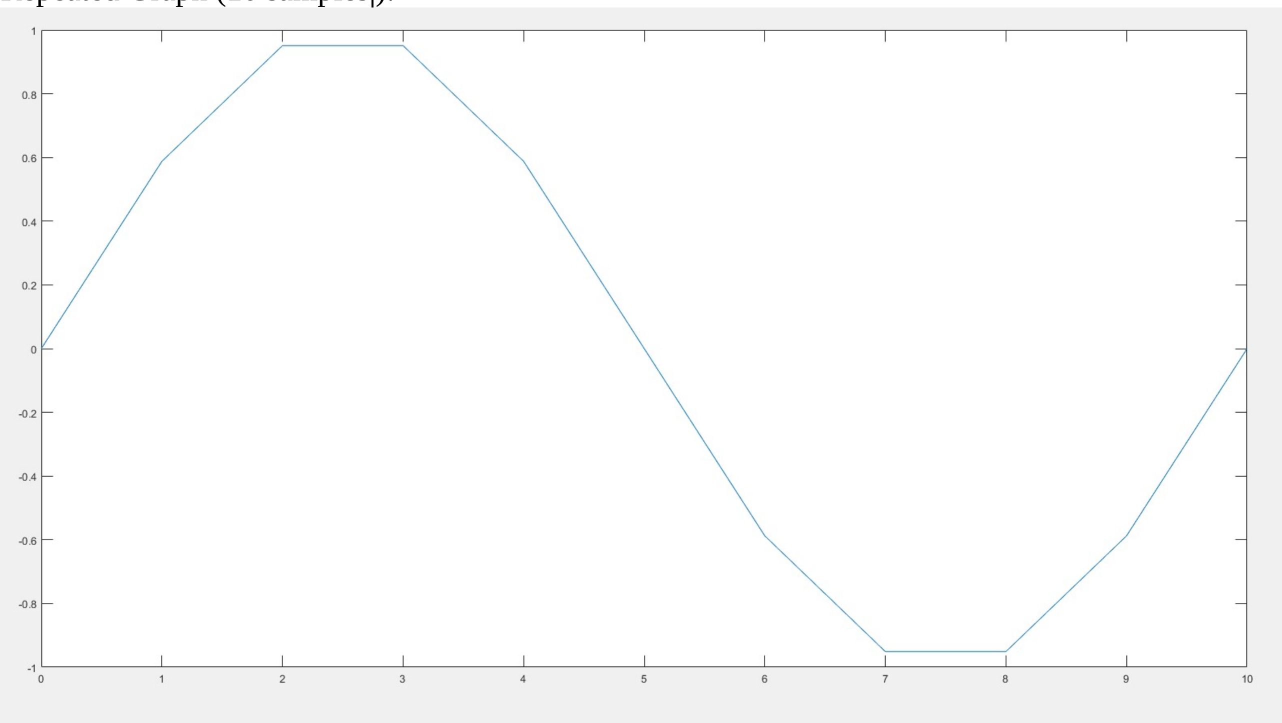
Krzysztof Rudnicki, 307585

Task 1
For f = 4800 Hz
f_s = 48 kHz
F_n = f / f_s = 4800 / 48000 = 0.1
Samples per period = 1 / f_n = 10

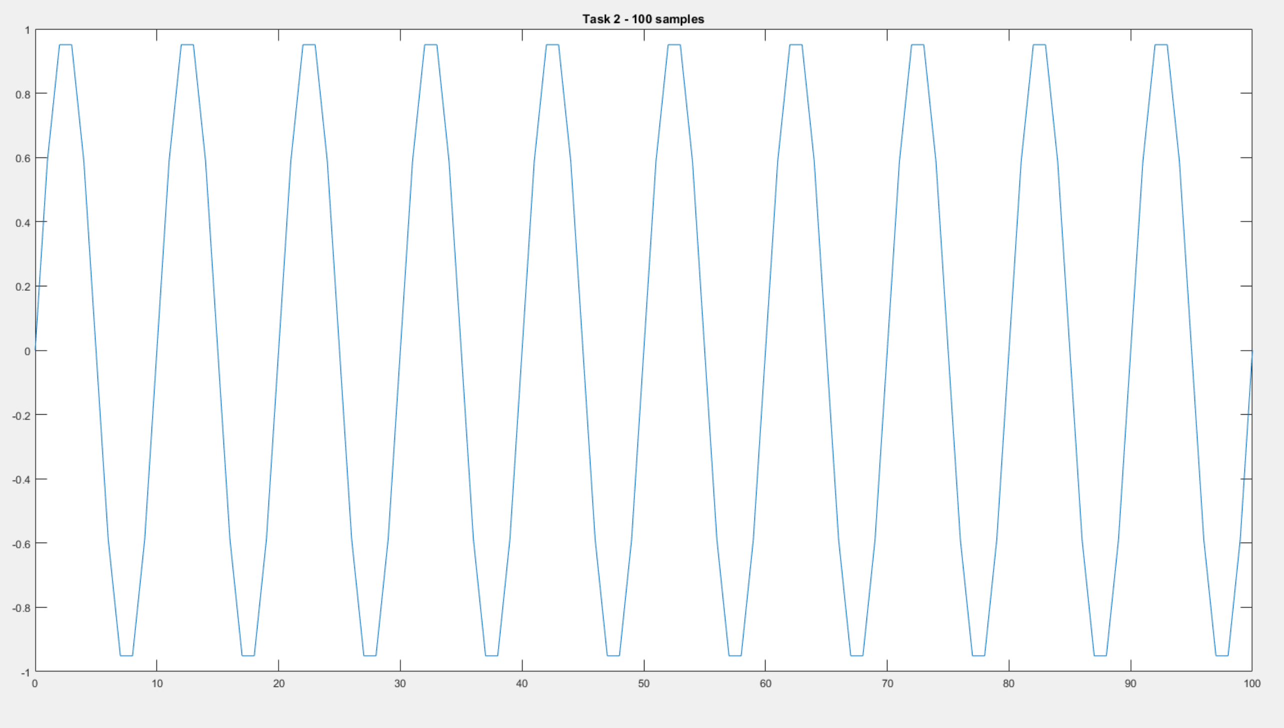


Task 2

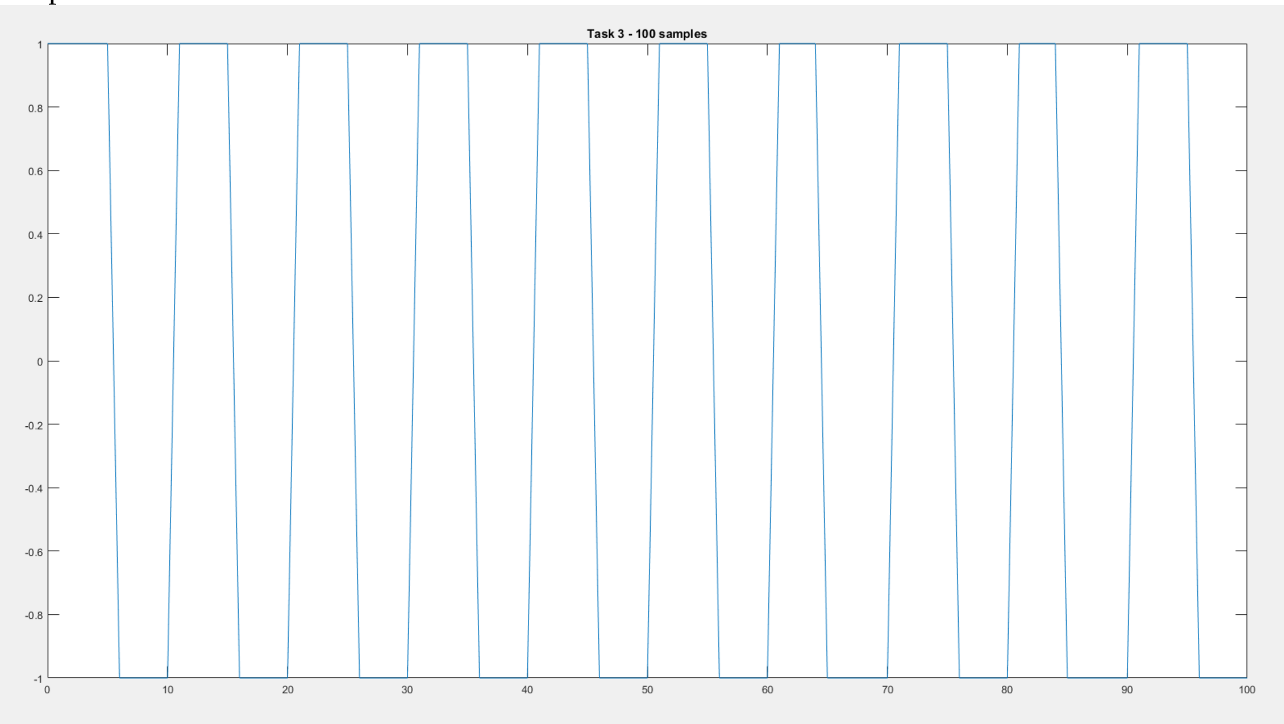
Repeated Graph (10 samples|):



Repeated Picture (100 samples|):

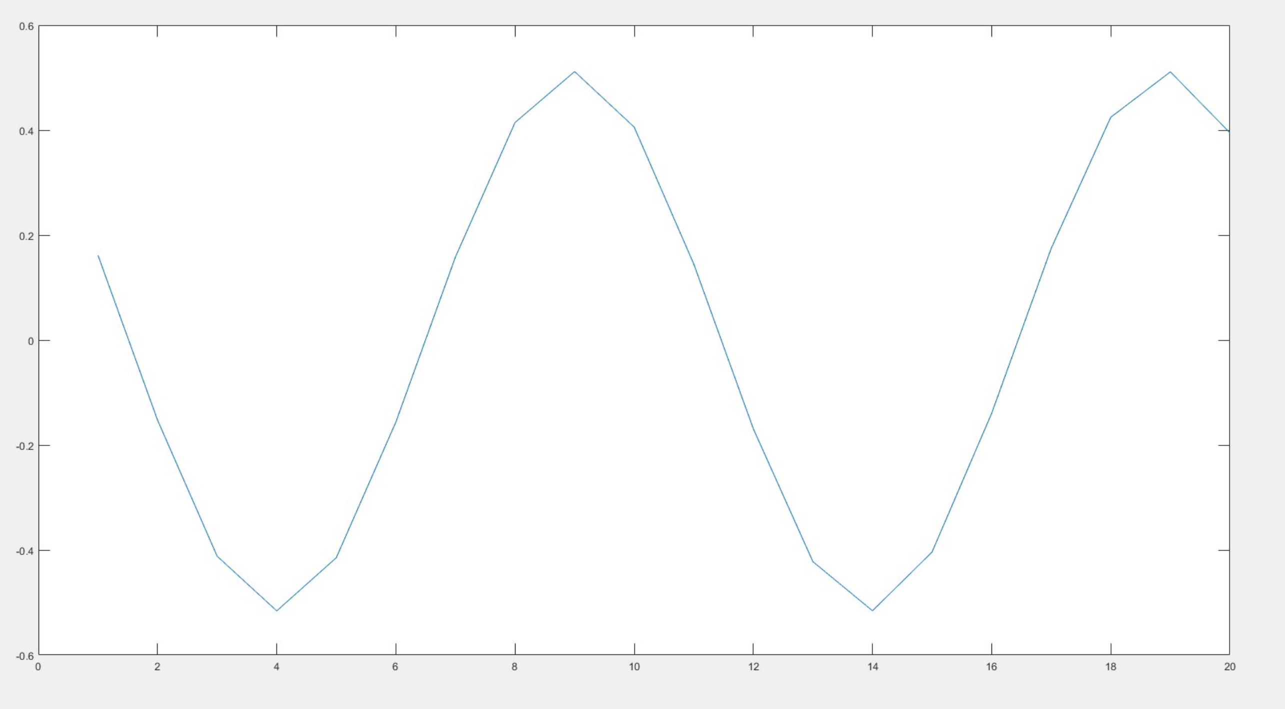


Task 3:
Graph:

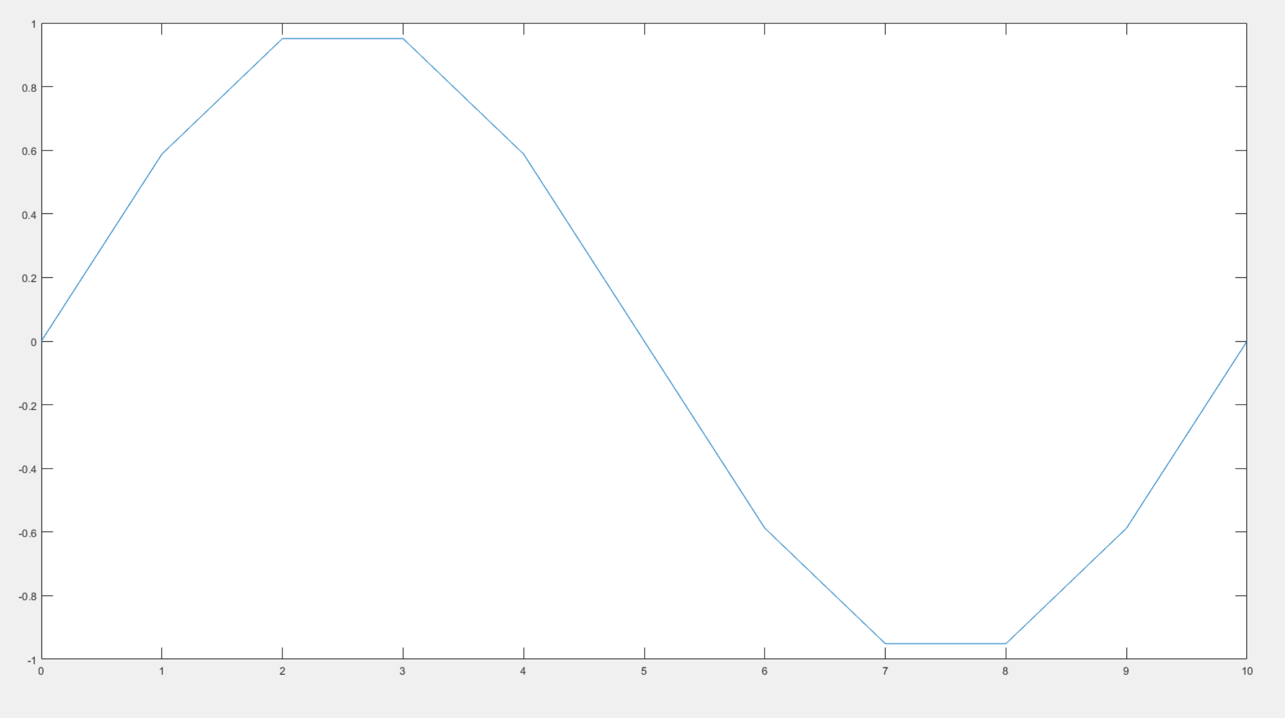


As we can see the width of the „summits” is not the same

Task 4:
20 Samples, Sampled at T_s = 1 / 48 kHz (as in the first task)
Real-world plot:

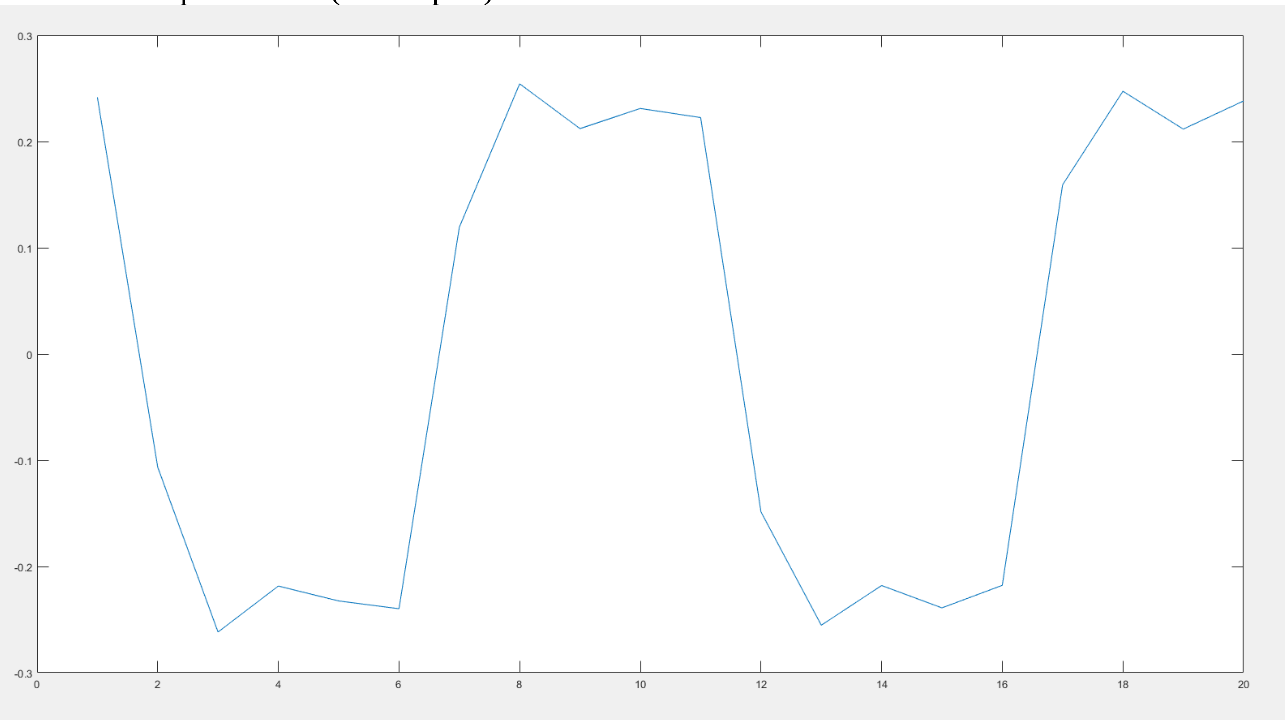


Difference between simulated and real-world plots:

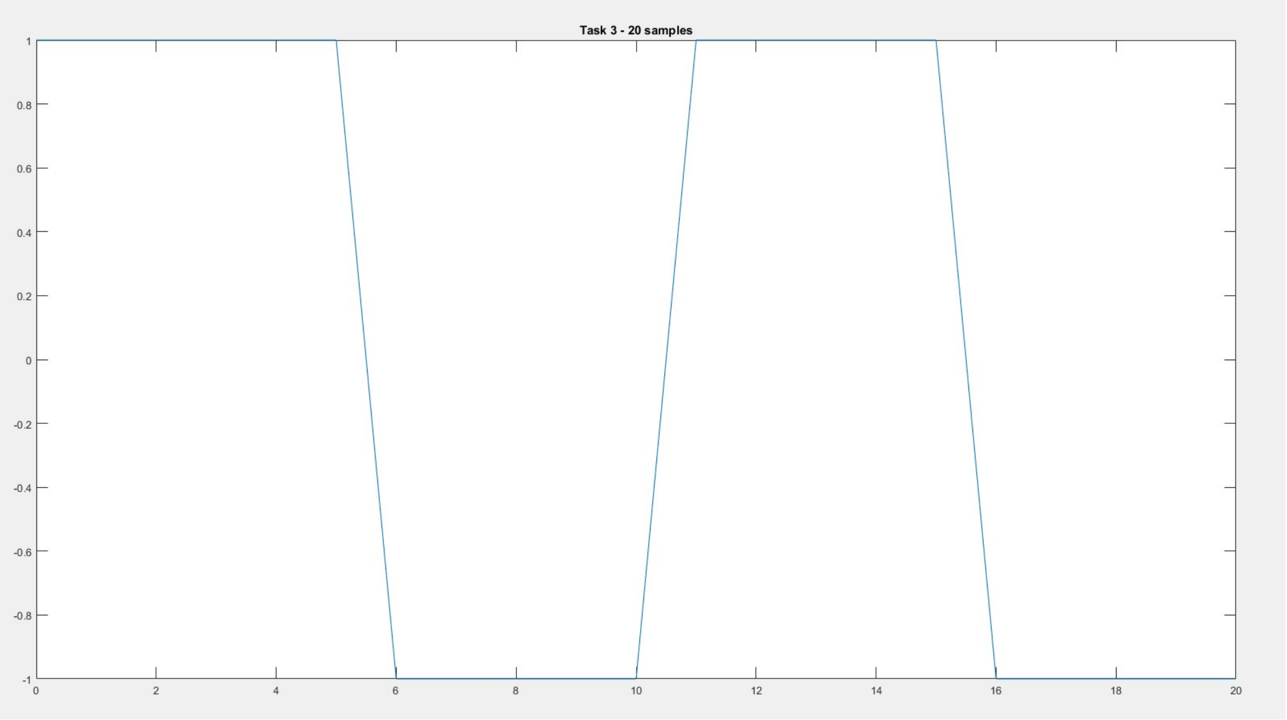


As we can see in the real world plot we DO NOT know when we will start sampling, (in this example

Real World Square Wave (20 samples):

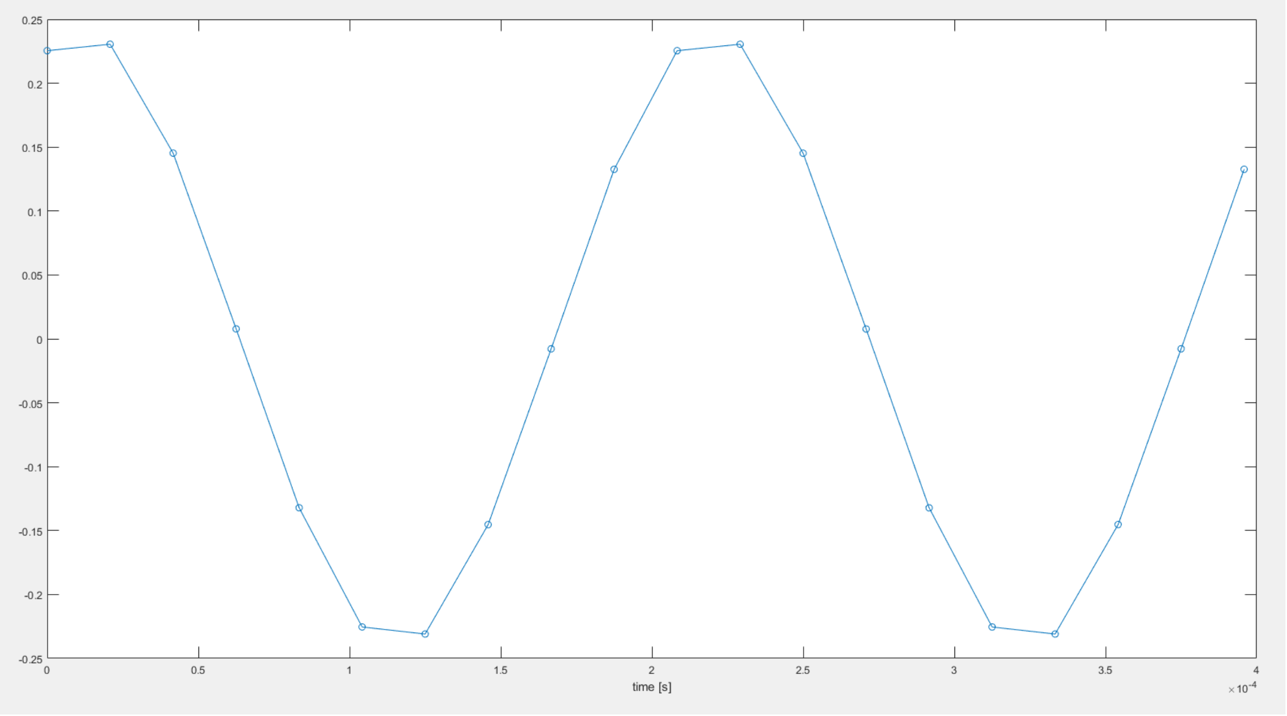


Simulated Square Wave (20 samples):



As we can see simulated square wave is much „cleaner” it gets to ‘1’ and ‘-1’ value much faster and is much more stable, as expected real world one behaves in less predicted way.

Task 5

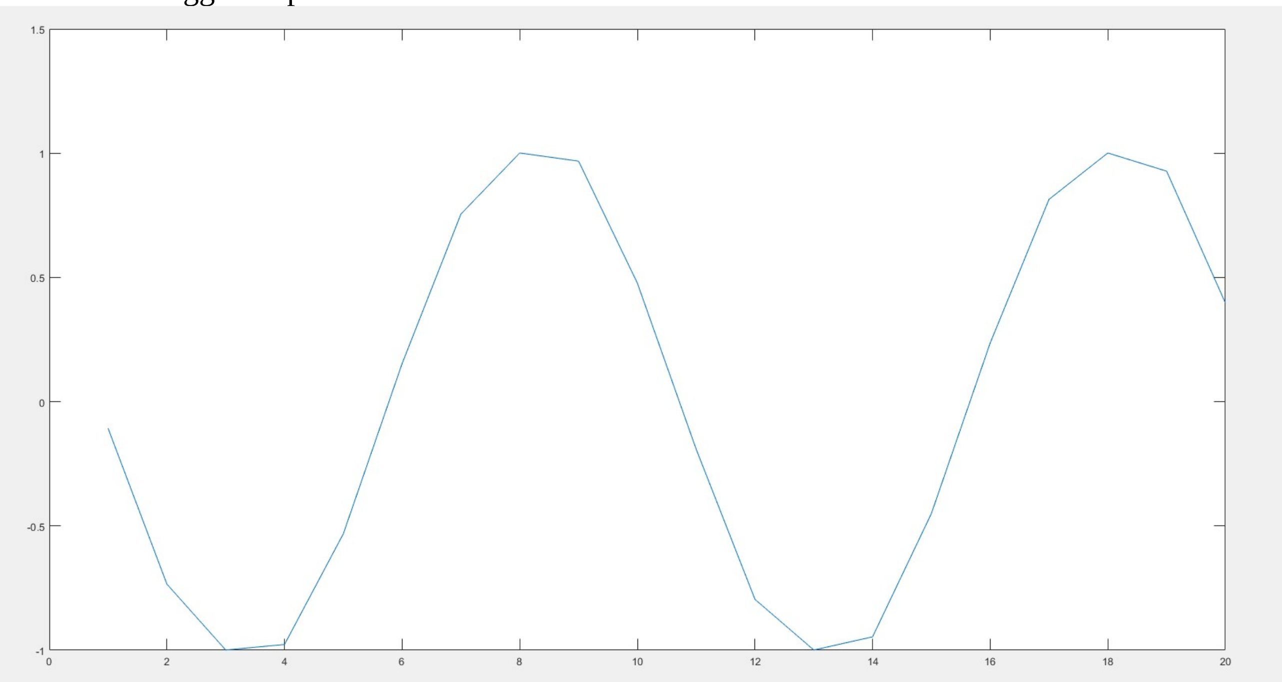


Matlab commands used

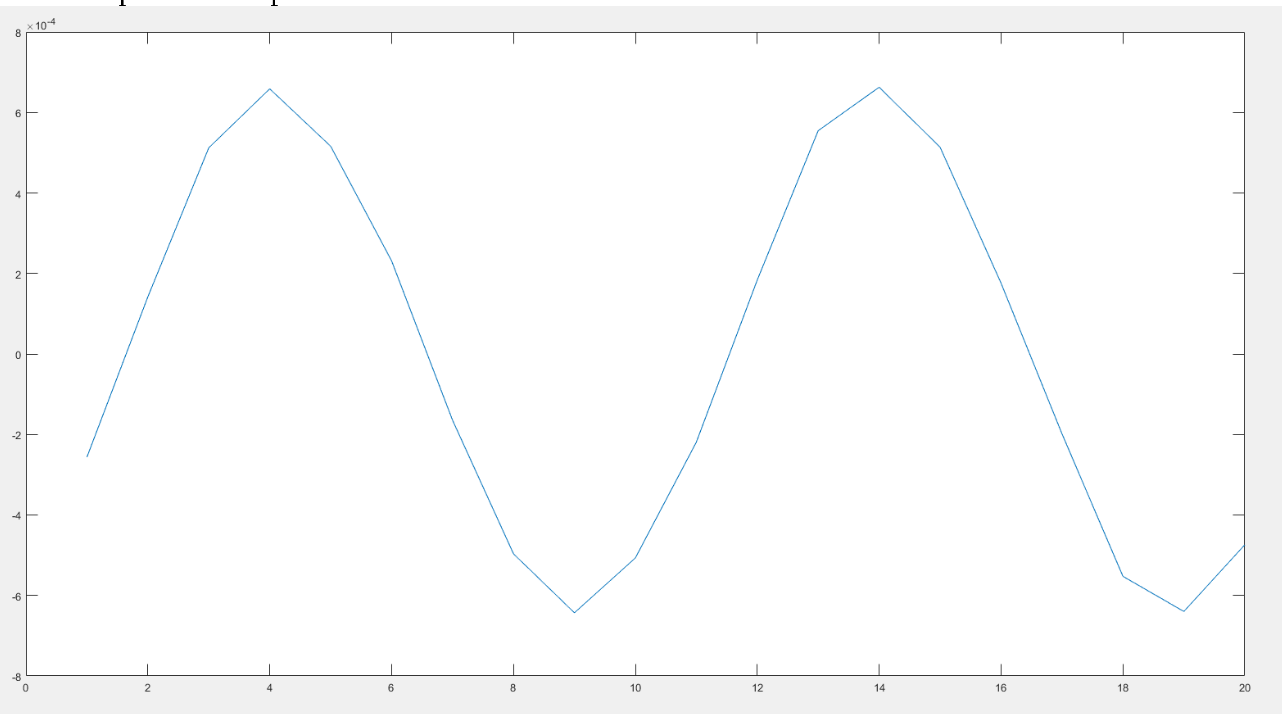
```
f_s = 48000;
N = 20;
n = 0: N - 1;
T_s = 1 / f_s;
x = LCPS_getdata(N, 1, T_s);
figure(2);

plot(n * T_s, x, '-o')
xlabel('time [s]')
```

Task 6

Much bigger amplitude:

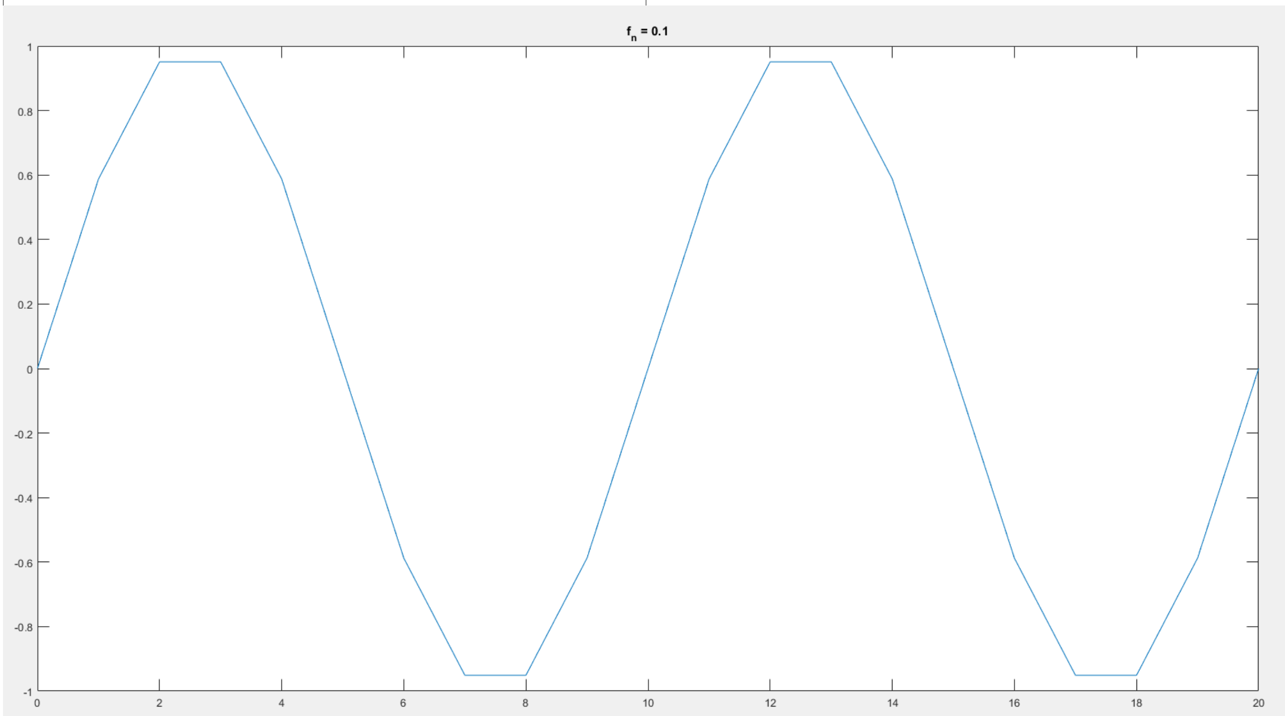


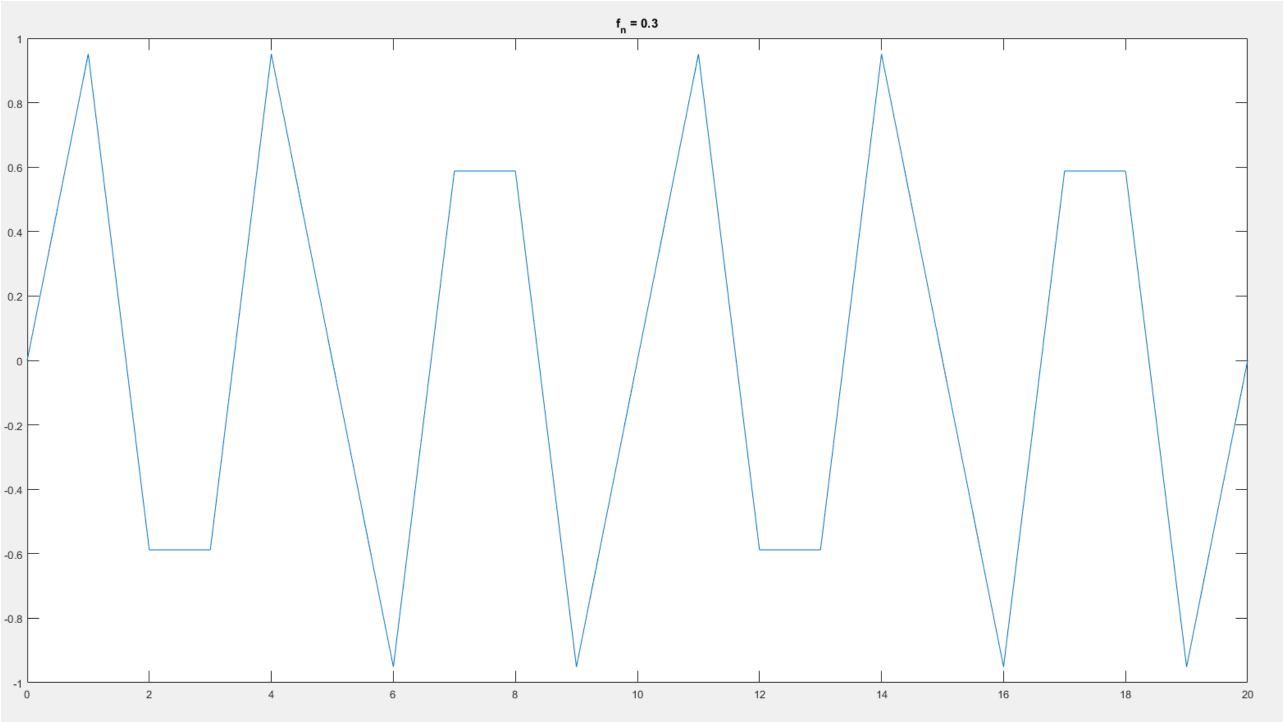
Smallest possible amplitude:

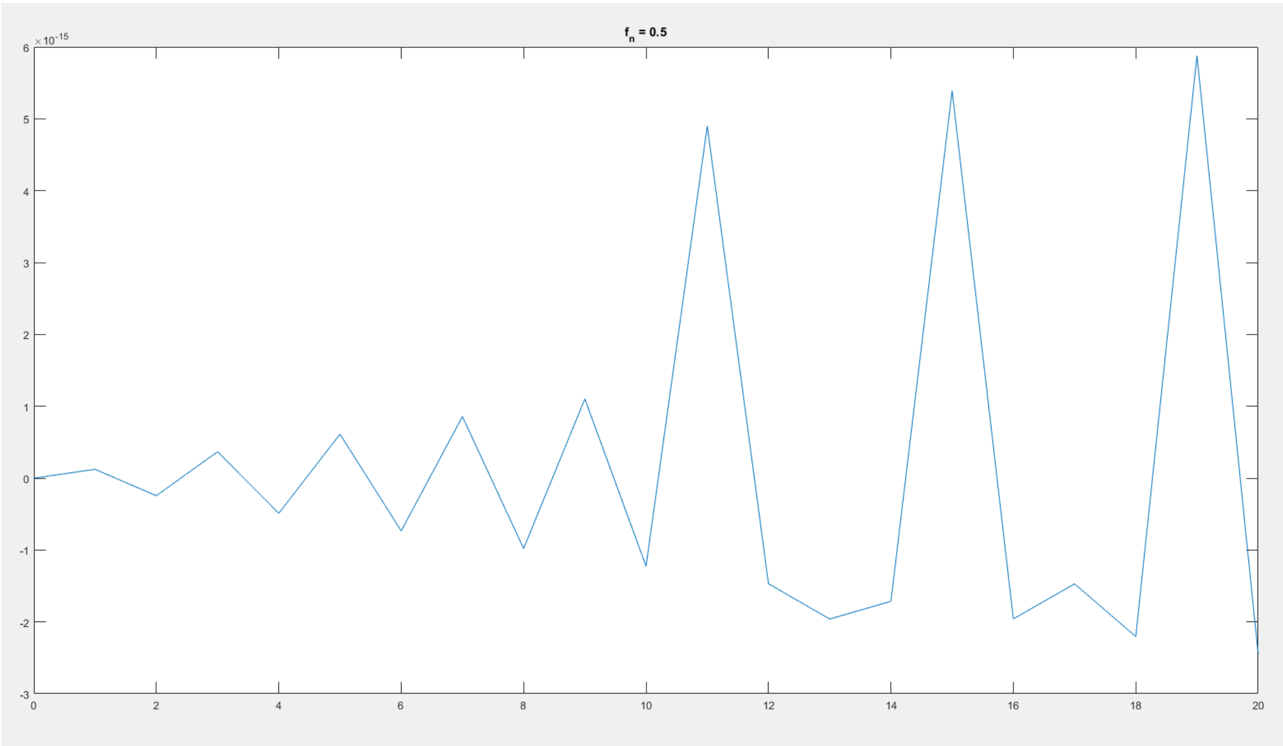


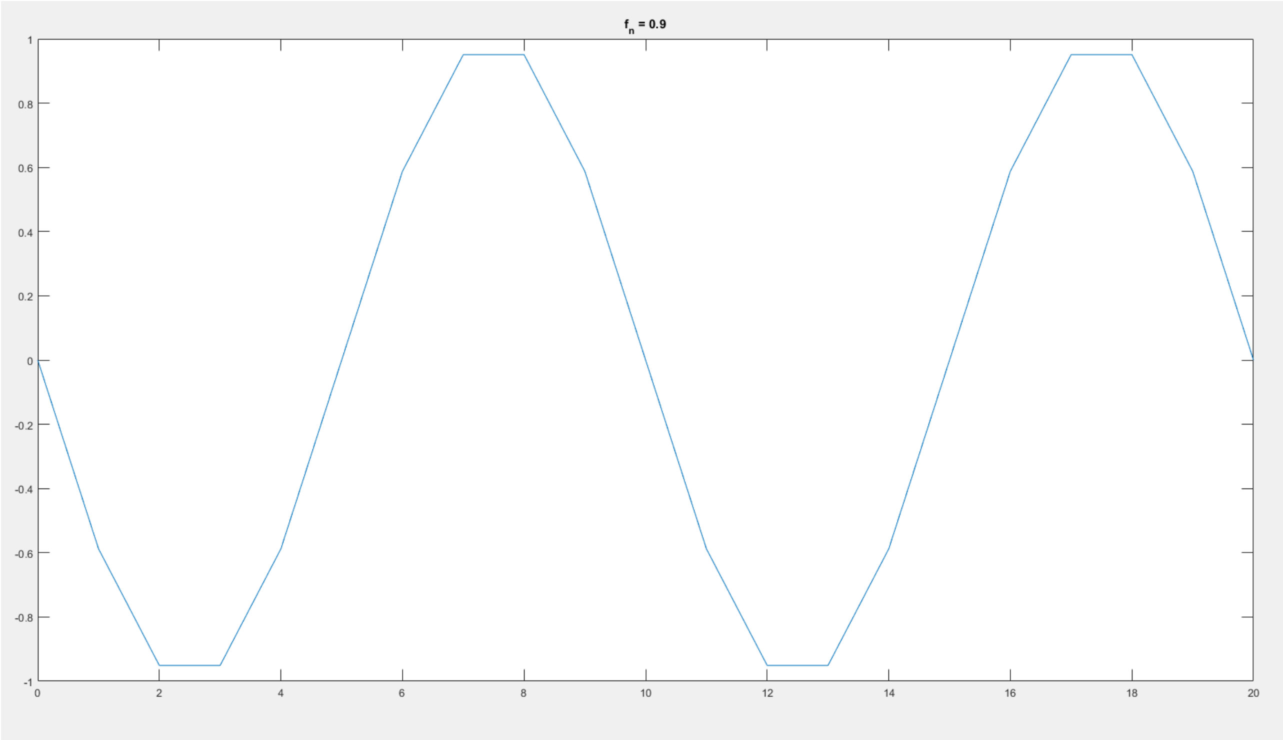
Task 7

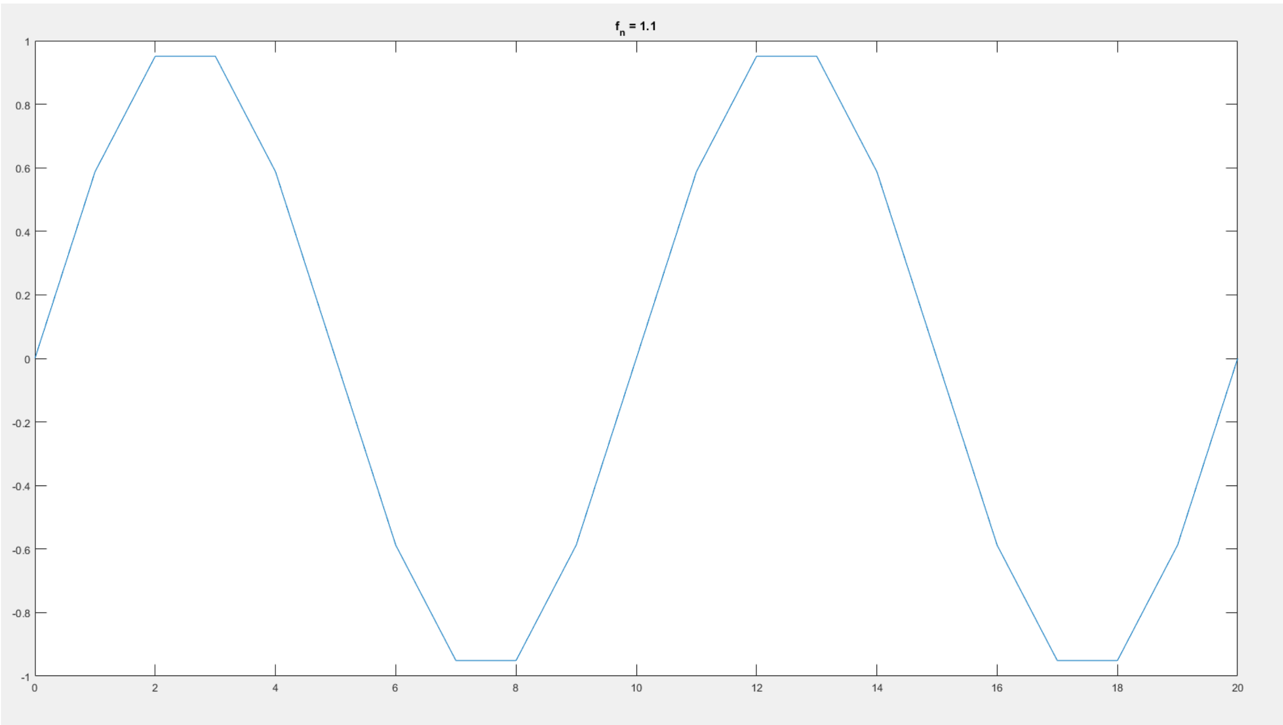
f_n	Number of samples in period
0.1	10
0.3	10 / 3
0.5	2
0.9	10 / 9
1.1	10 / 11
2.1	10 / 21

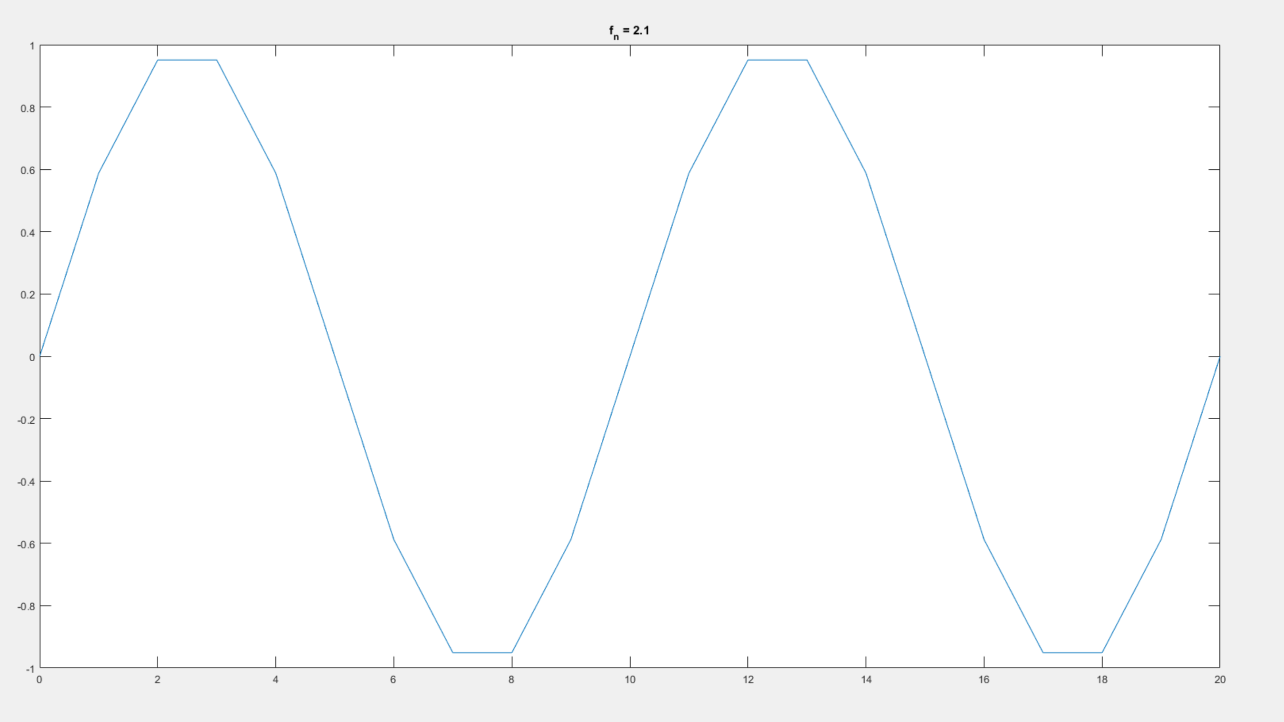



f_n = 0.3
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
2
4
6
8
10
12
14
16
18
20


f_n = 0.5
$\times 10^{-15}$
6
5
4
3
2
1
0
-1
-2
-3
0
2
4
6
8
10
12
14
16
18
20


f_n = 0.9
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
2
4
6
8
10
12
14
16
18
20


f_n = 1.1
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
2
4
6
8
10
12
14
16
18
20



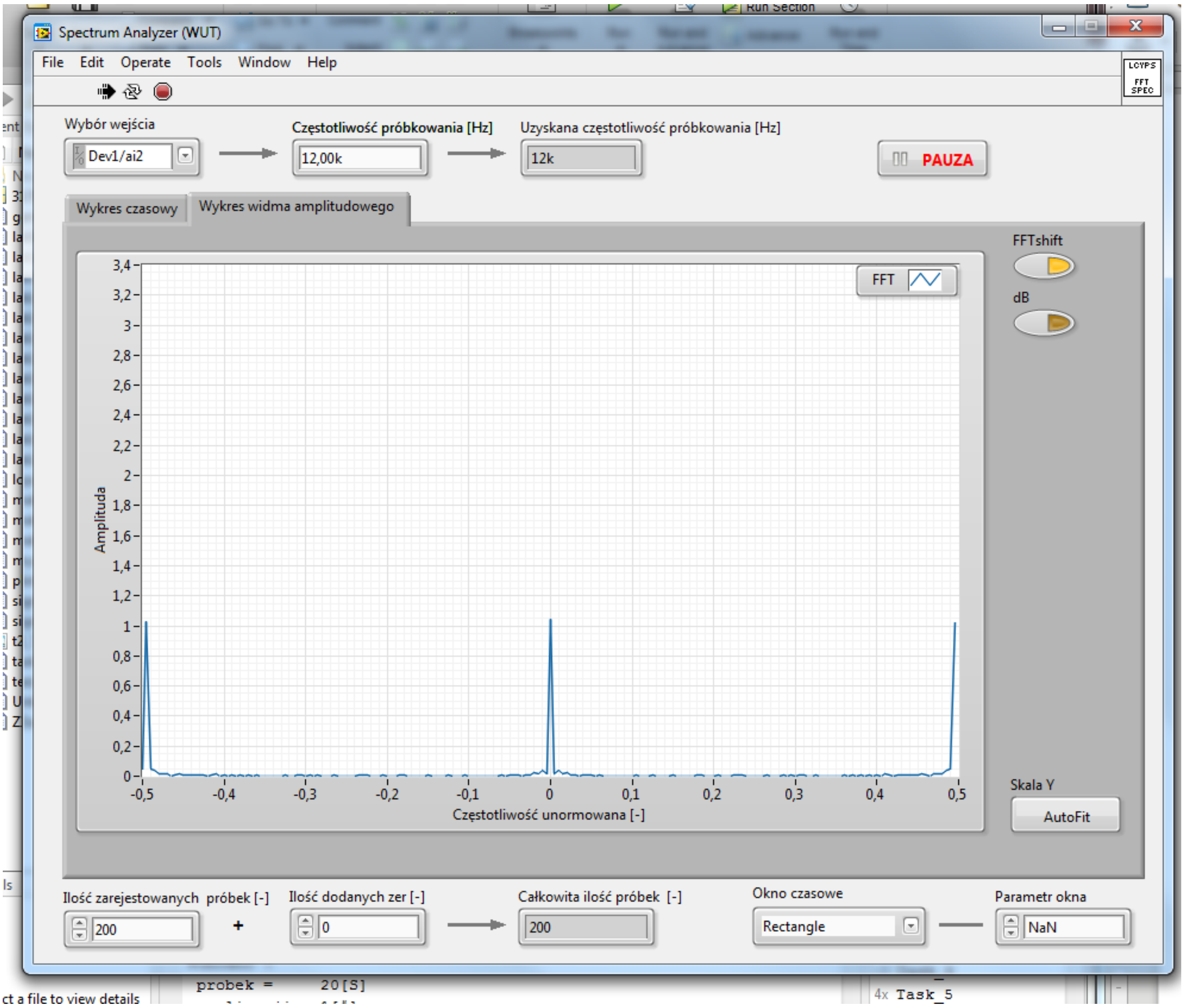
In some cases different f_n result in identical plots because the f_n is multiplied by a number that makes the graph ‘shift’ by the period of the sin wave so even though the ‘x’ position of sinus will change the value of it will be exactly the same since sinus is periodic function.

Task 8

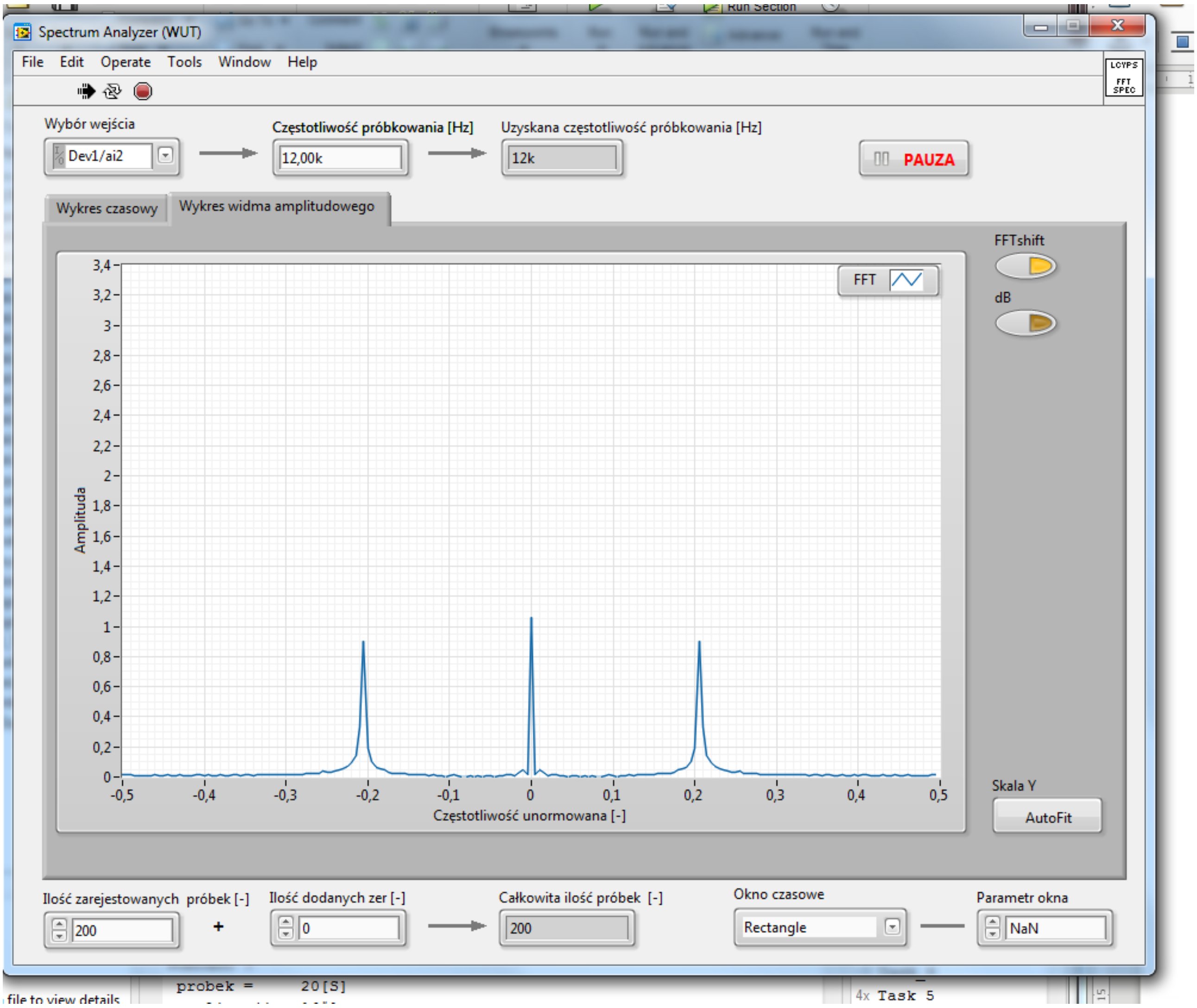
Nyquist Frequency is equal to 0.5 * f_s where f_s is equal to 12 kHz

In our case Nyquist Frequency is equal to 6 kHz

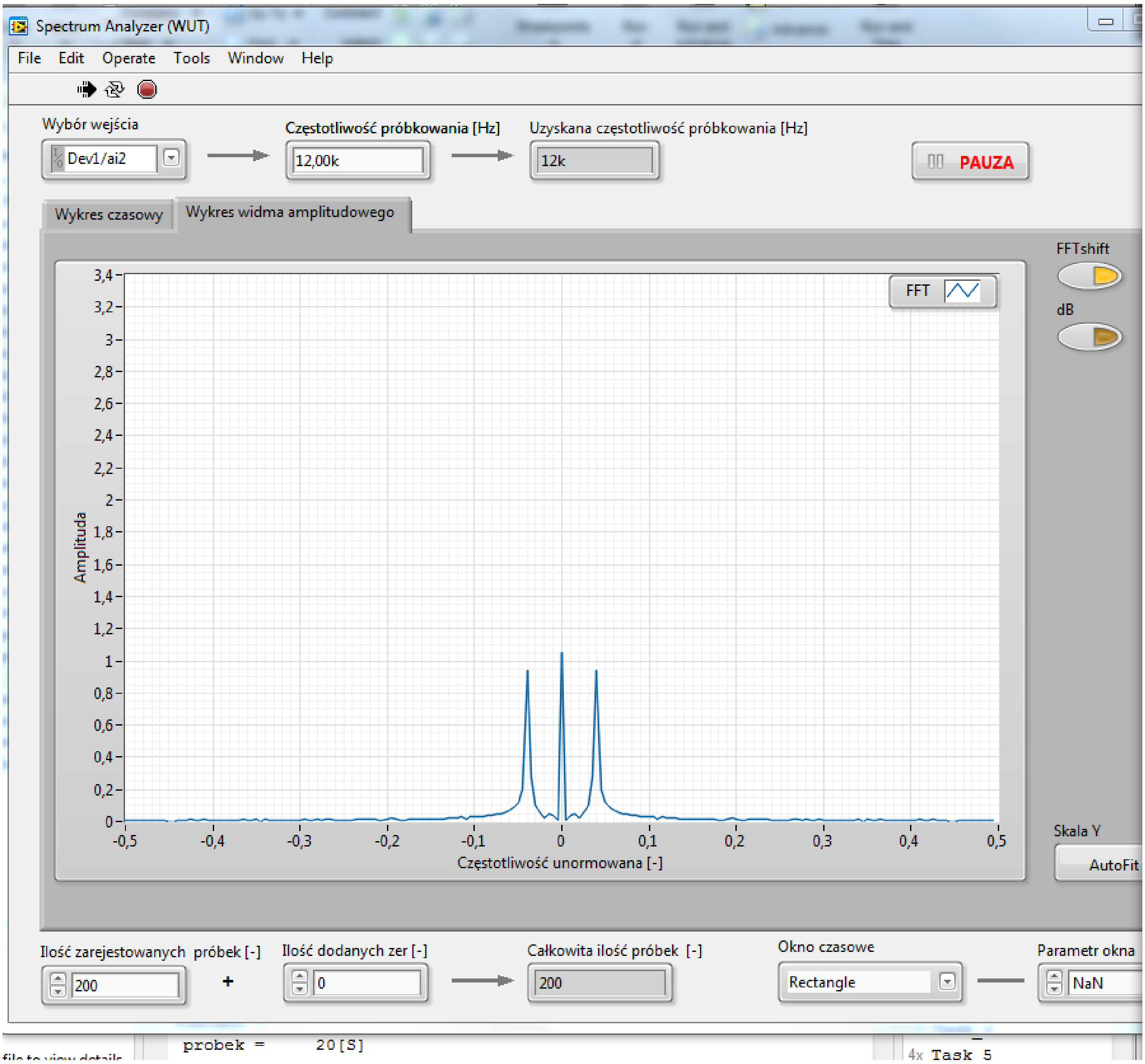
After setting frequency of the signal to the Nyquist Frequency (6 kHz):



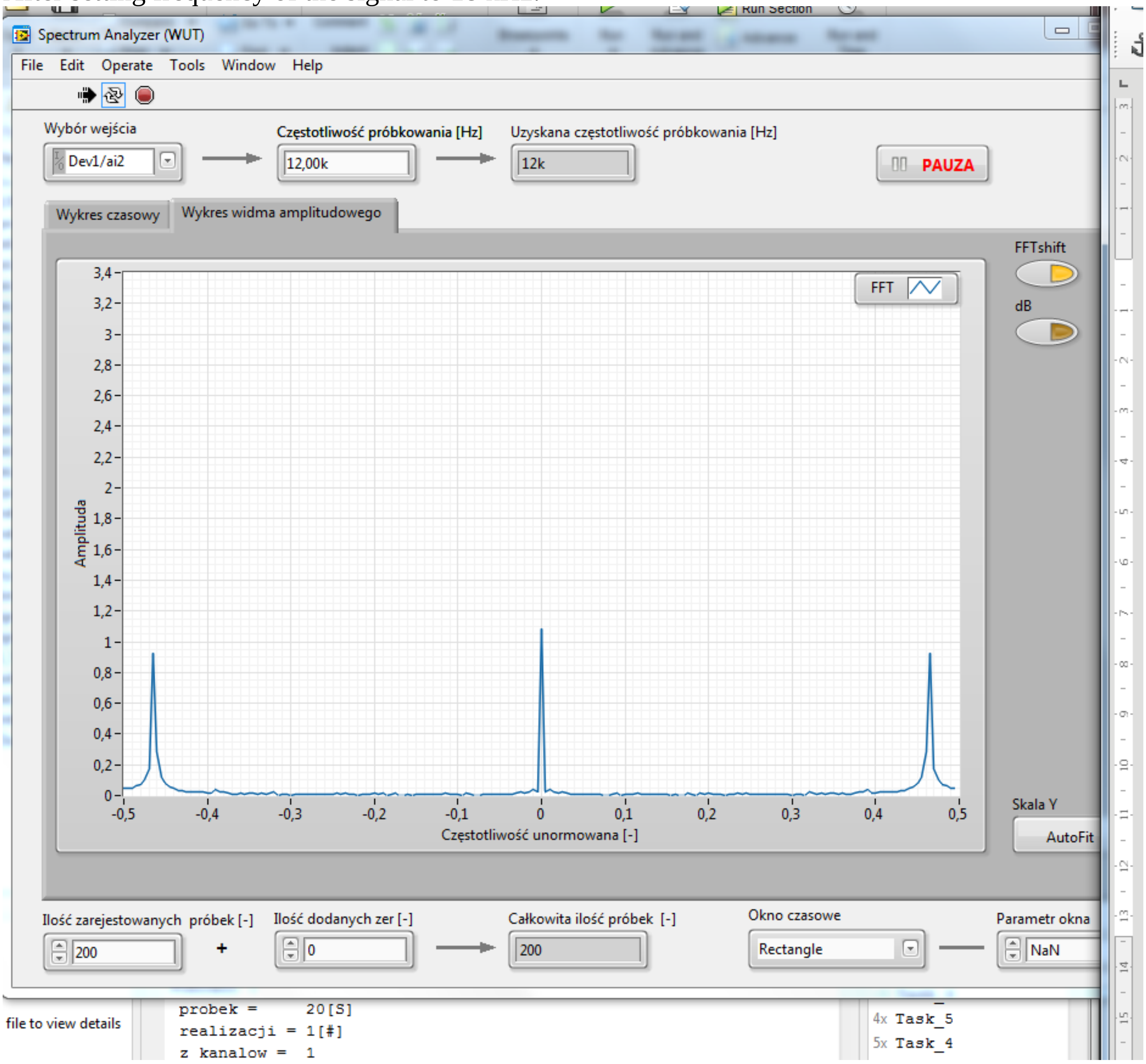
After setting frequency of the signal to 9 kHz:



After setting frequency of the signal to the Sampling Frequency (12 kHz):

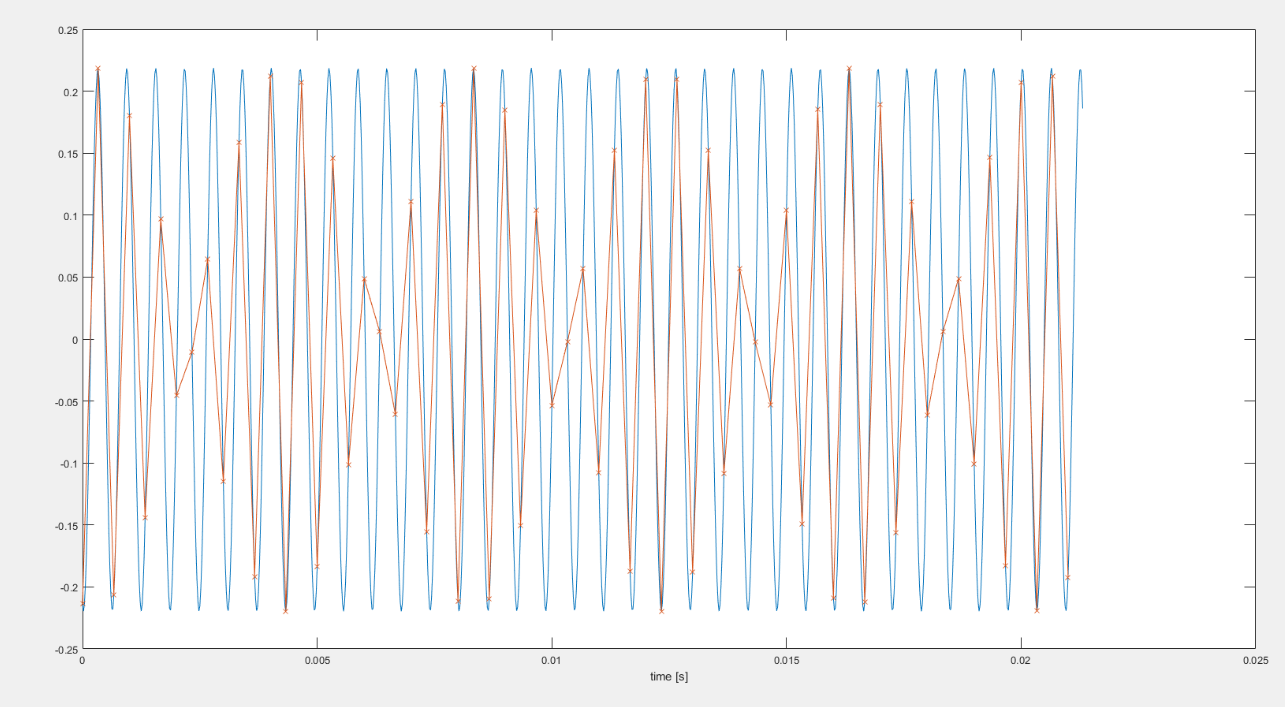


After setting frequency of the signal to 18 kHz:



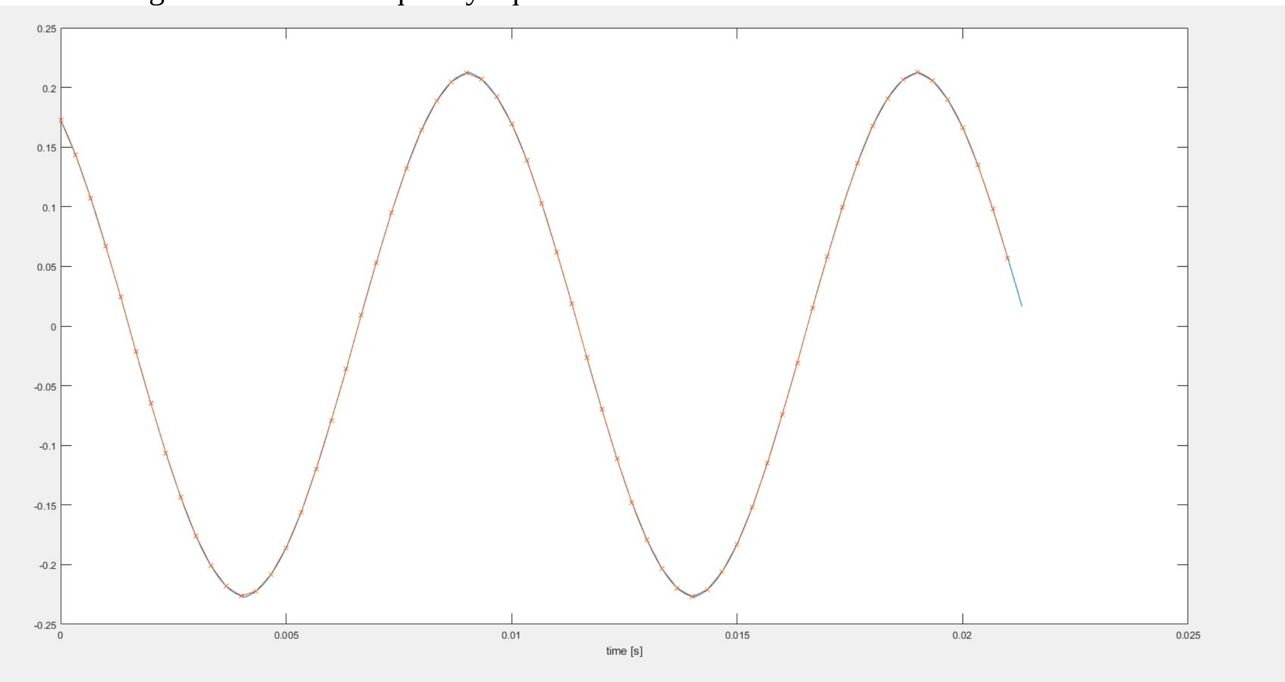
When the signal frequency exceeds the Nyquist frequency signals ‘come closer’ to each other
Once the signal frequency exceeds te sampling frequency and we increase it still signals ‘go apart’ from each other
We can also observe aliasing of signals.

Task 9

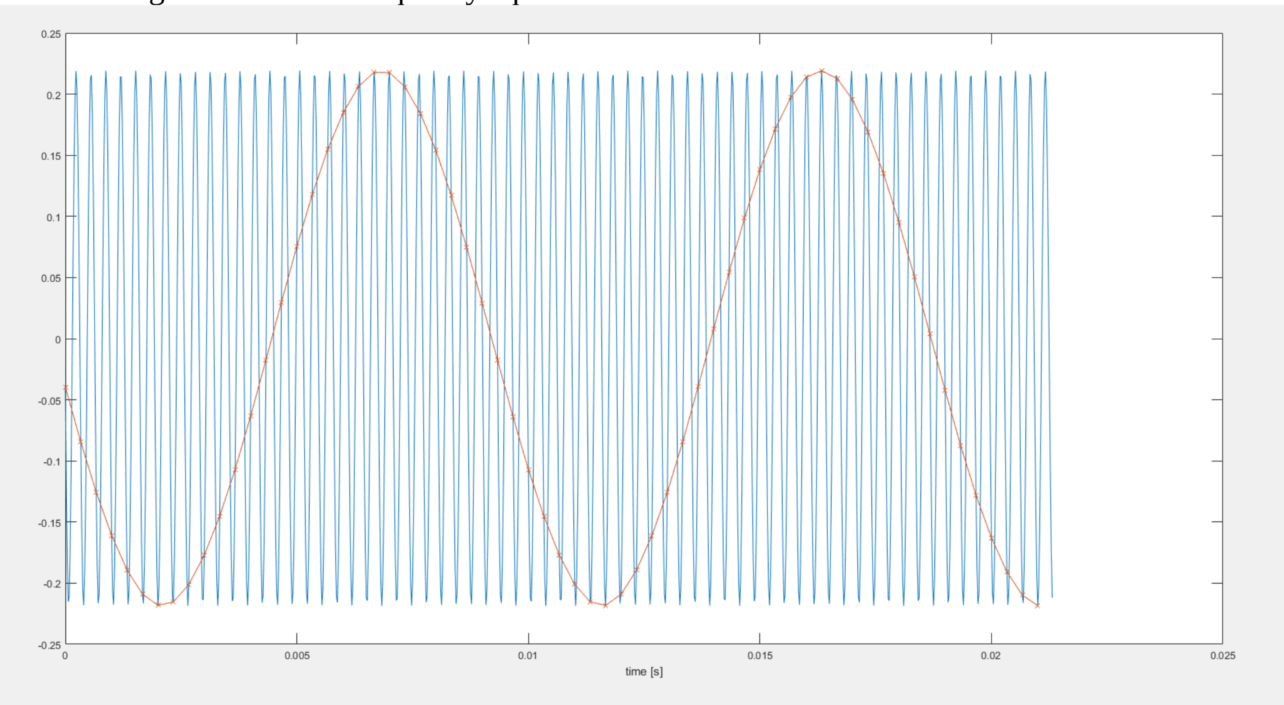


Sampling frequency

Undersampling means sampling signal at a sample rate below Nyquist rate. We basically get rid of every sample except 16th.
This means that the resulting sampling freqency will be equal to 48 kHz / 16 = 3 kHz!
Since we violated the nyquist criteria the sampling we got accomodated a lot of noise.
Task 10
Same settings as above but frequency equal to 0.1 kHz:

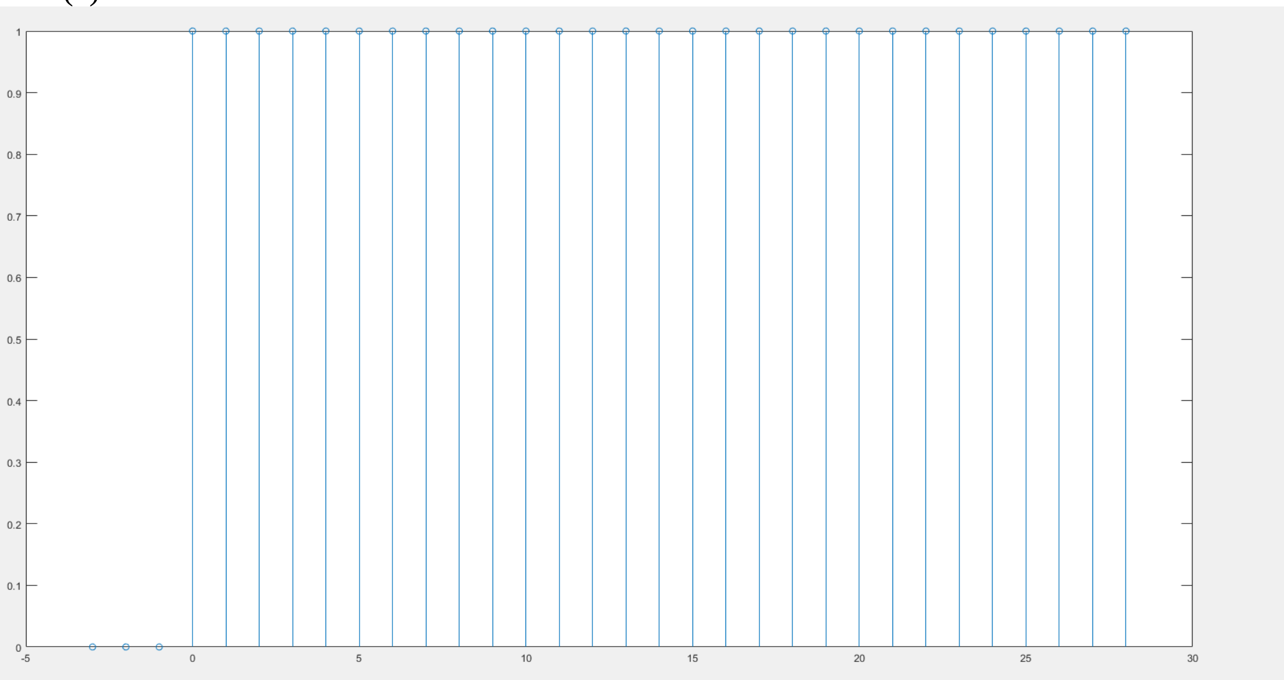


Same settings as above but frequency equal to 3.1 kHz:

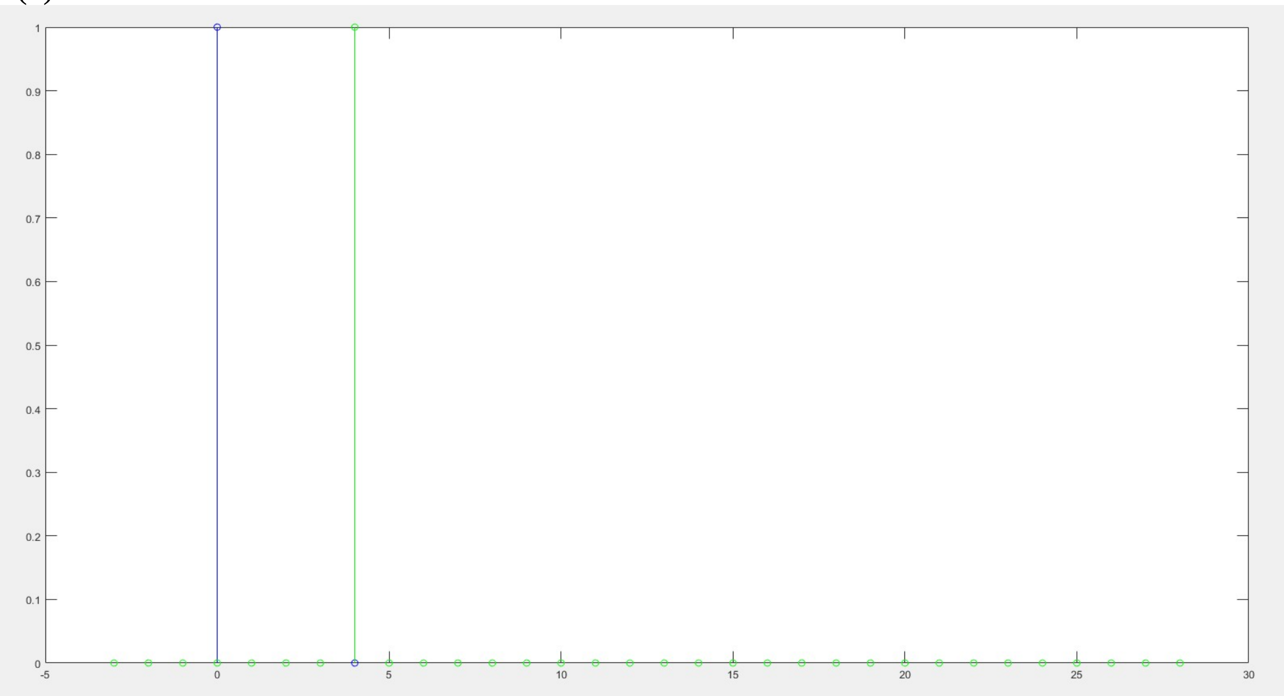


As we can see the results align to the sin function much more nicely, especially for frequency 0.1 kHz. For frequency 0.1 we got so many samples that even after undersampling we still retained the shape of the sinus wave, we also did not violate Nyquist criteria which made the signal much closer to sinus wave.

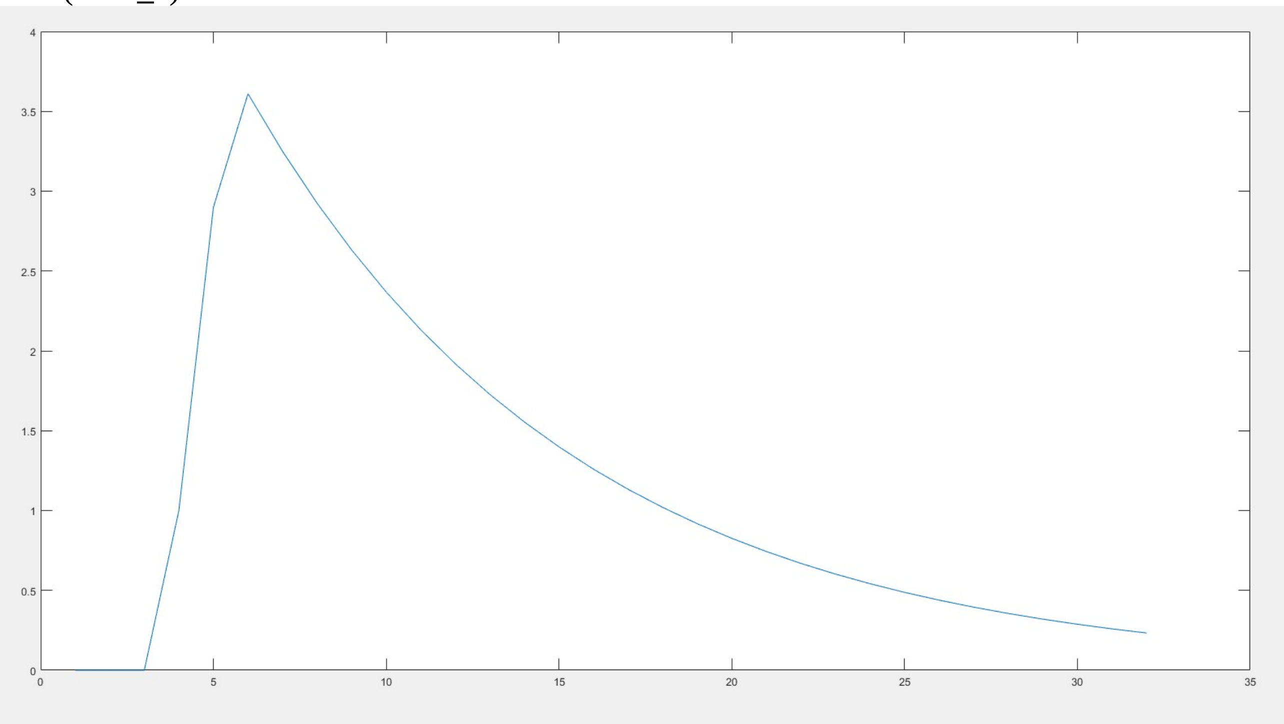
Task 11 and 12:
delta(n)



u(n):


1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-5
0
5
10
15
20
25
30

delta(n – n_0):


4
3.5
3
2.5
2
1.5
1
0.5
0
0
5
10
15
20
25
30
35