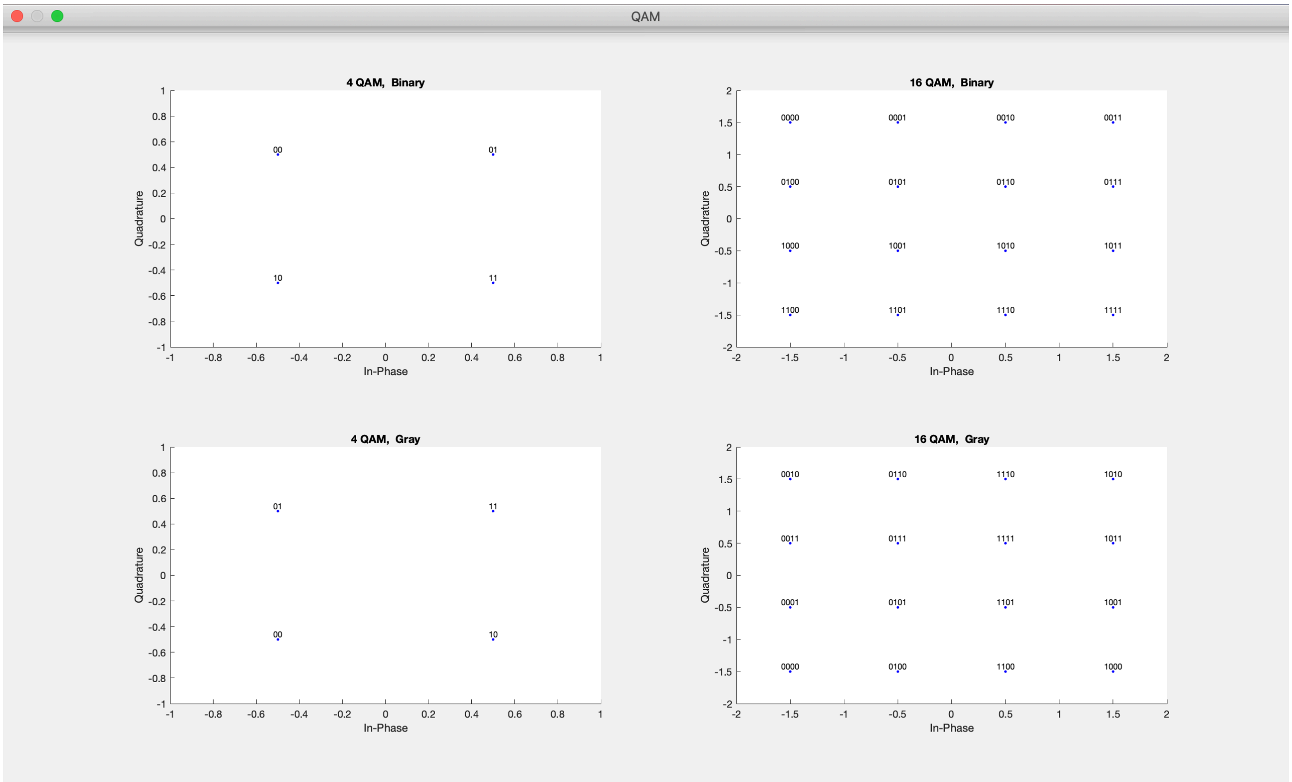
Lab 3: Digital Modulation and Demodulation

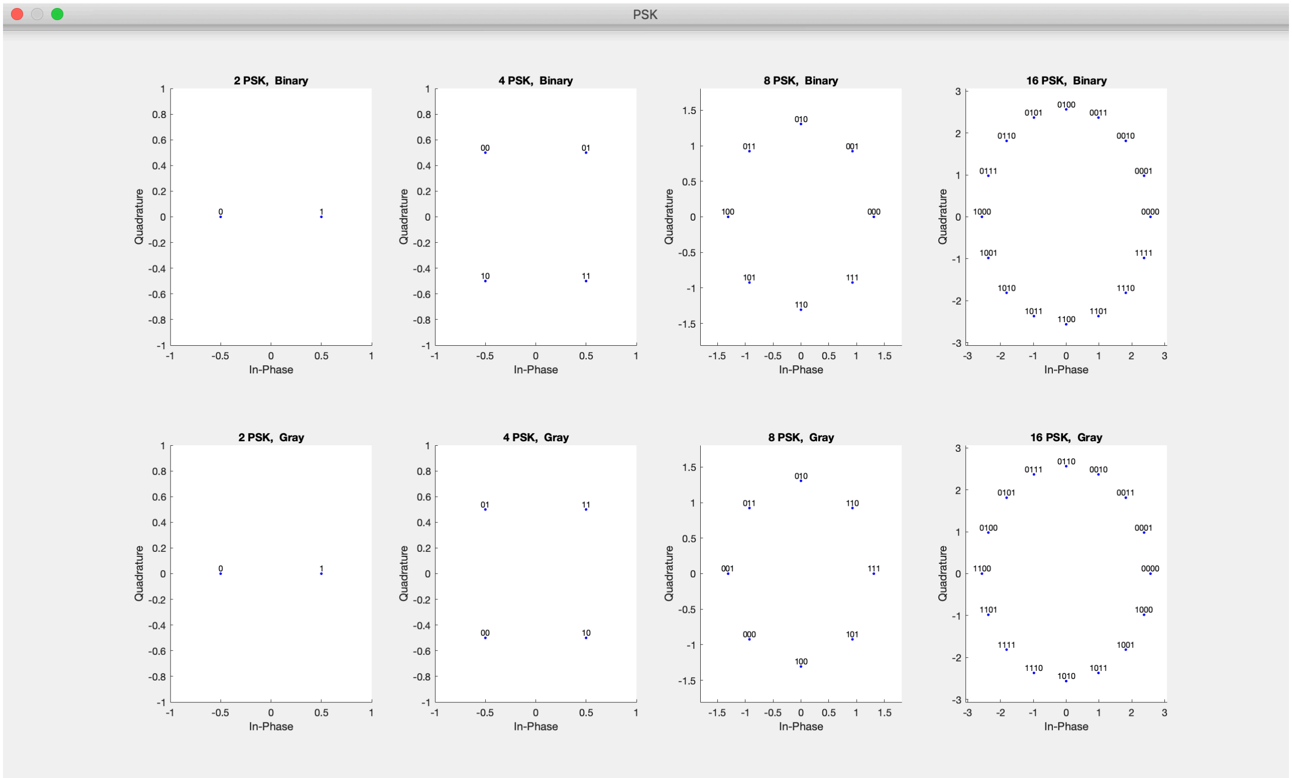
B04901067 電機四 陳博彥

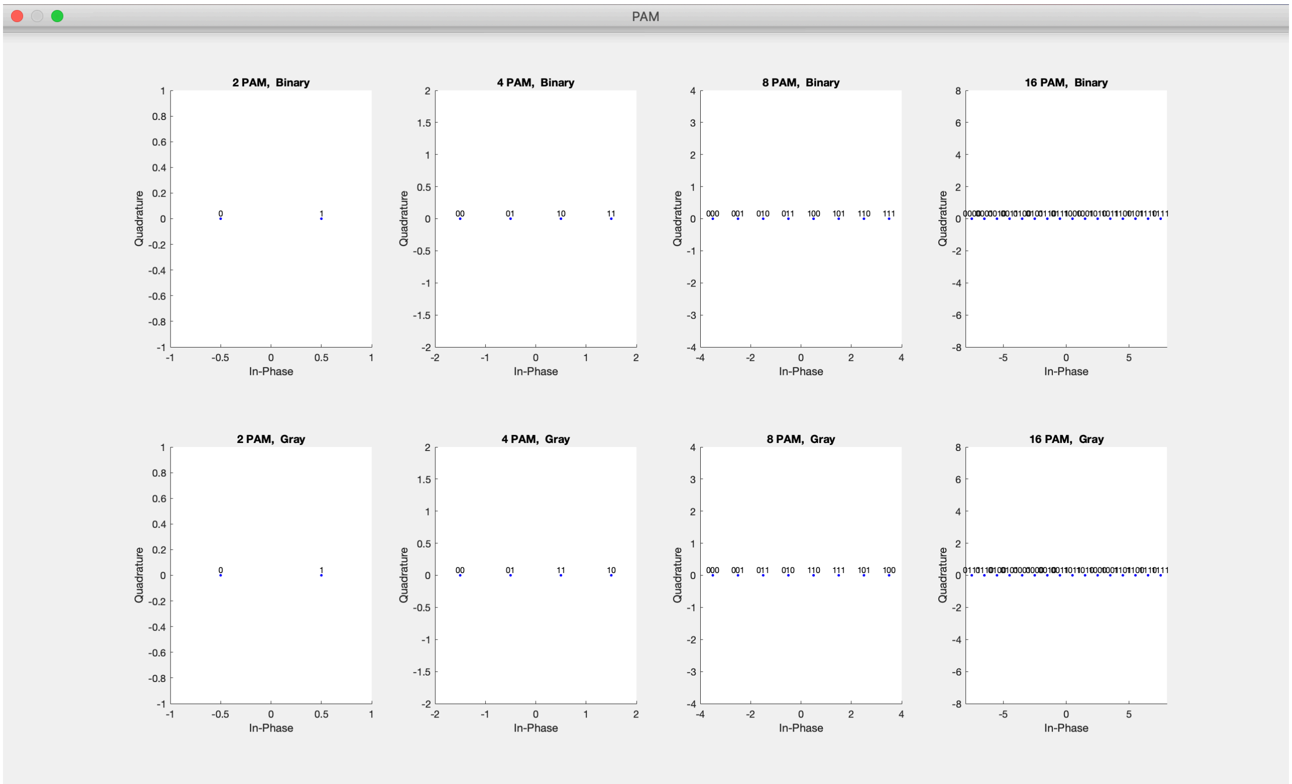
All the codes for this report is stored in https://github.com/Andy19961017/Communication_System_Lab/tree/master/Lab3

1.

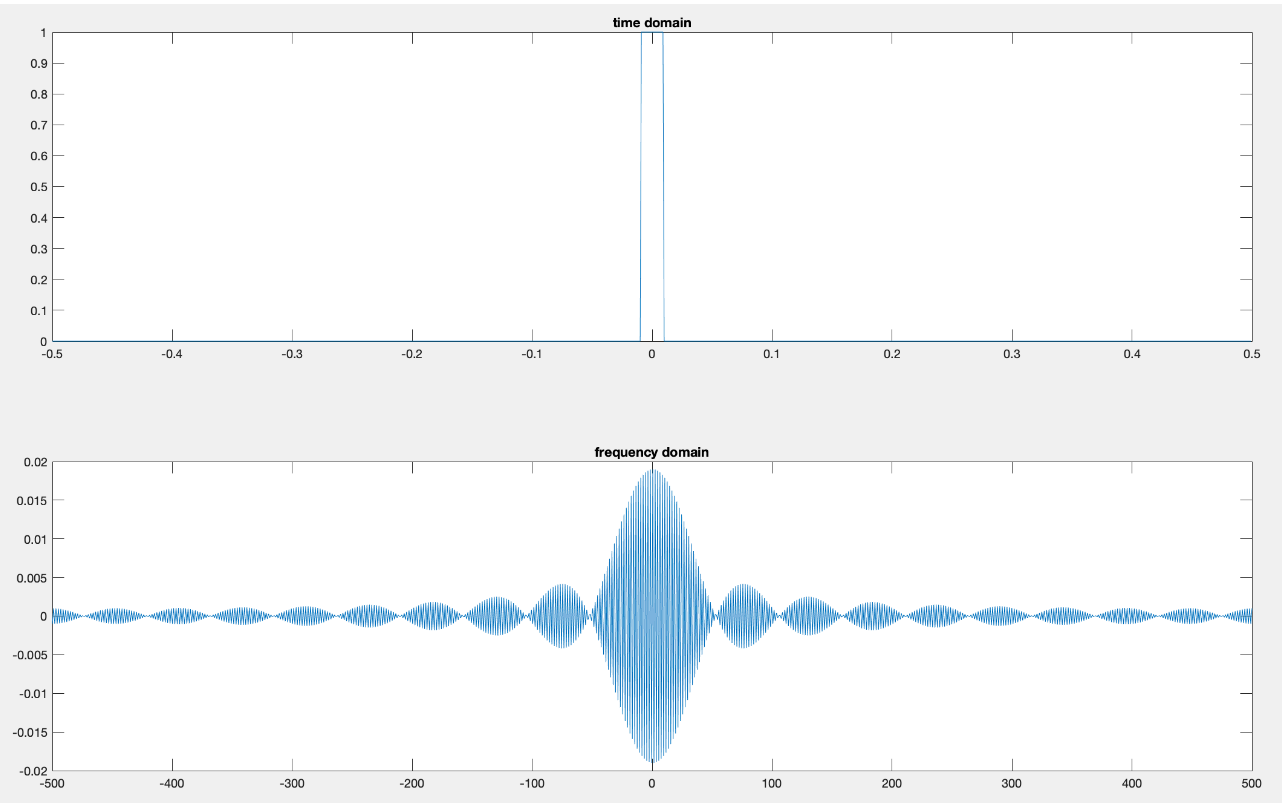
The function is defined in symbol_mapper.m.



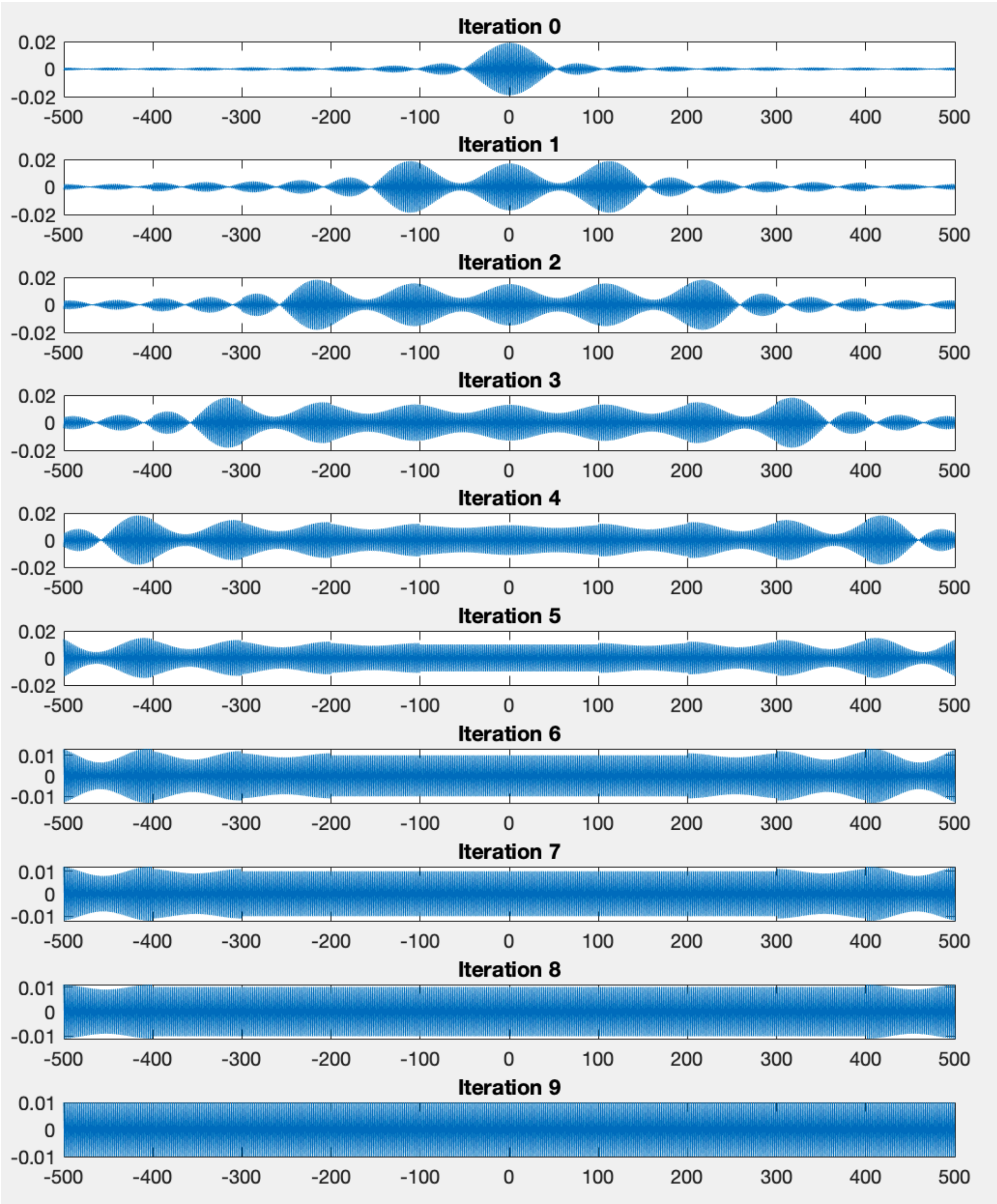




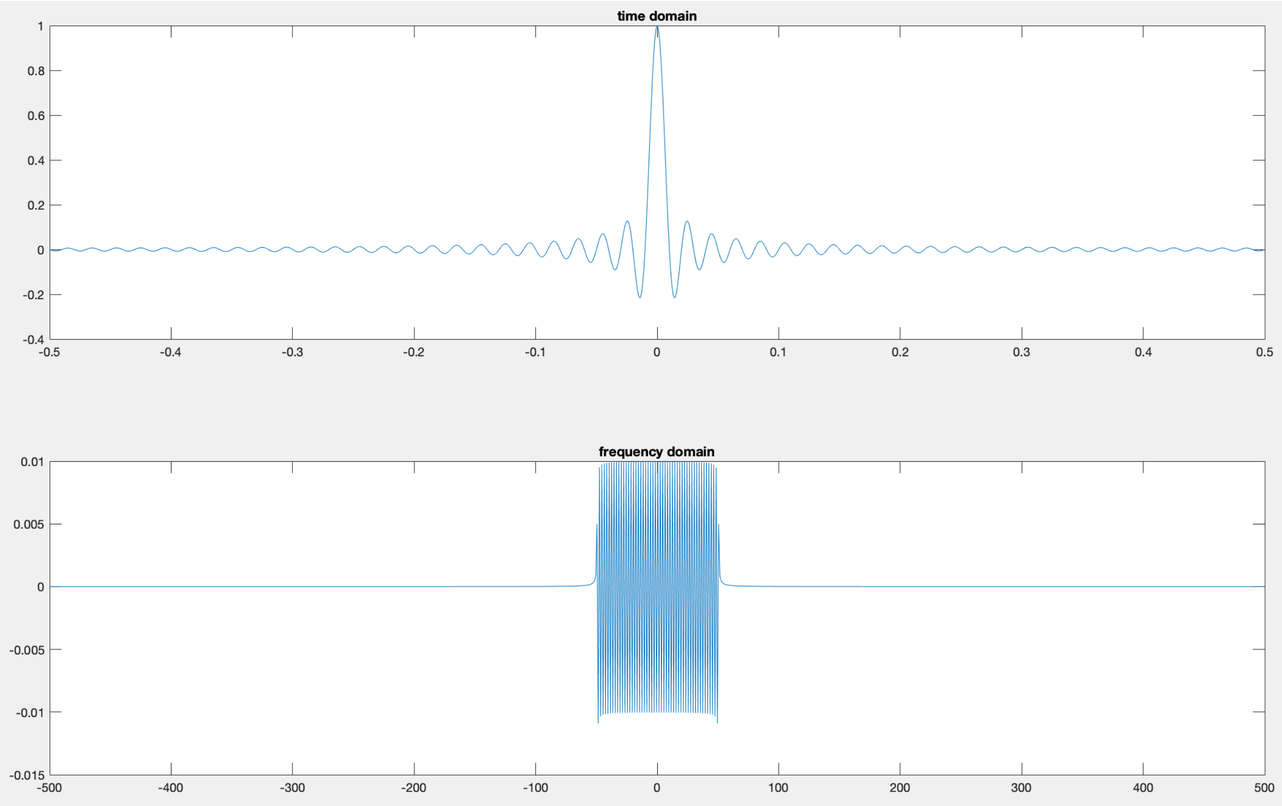
2.
(a)
The followings are the time domain and frequency domain signal of g(t).
Observing the time domain signal, we can see that g(n*T)=1 for n=0 and g(n*T)=1 for n!=0, where T=0.01 at this case. This validates the Nyquist criterion.
Since we are using DFT to implement CTFT, we can not get the true CTFT transformation (ideally, a sinc function).



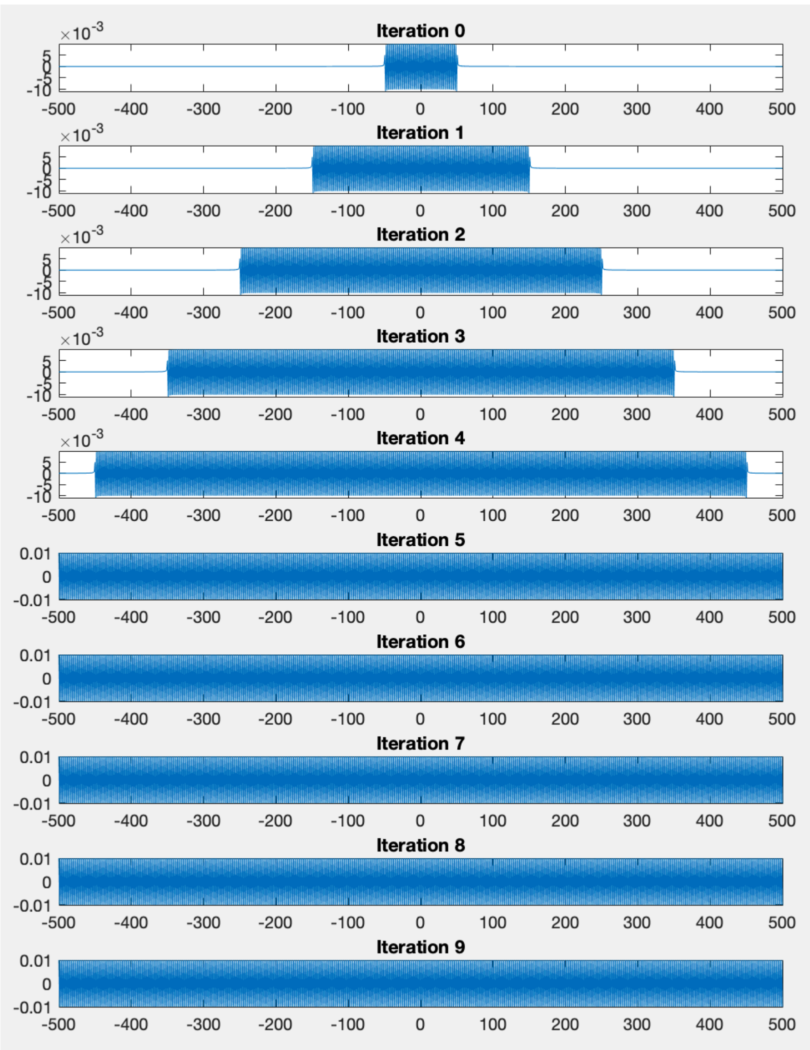
Starting from the original frequency domain signal at iteration 0, at each iteration, shift the signal 1/T units to the left and 1/T units to the right and add them to the original signal. We get the following results.
The signal converges to constant 0.01 gradually, while T=0.01 at this case. This validates the Nyquist criterion.



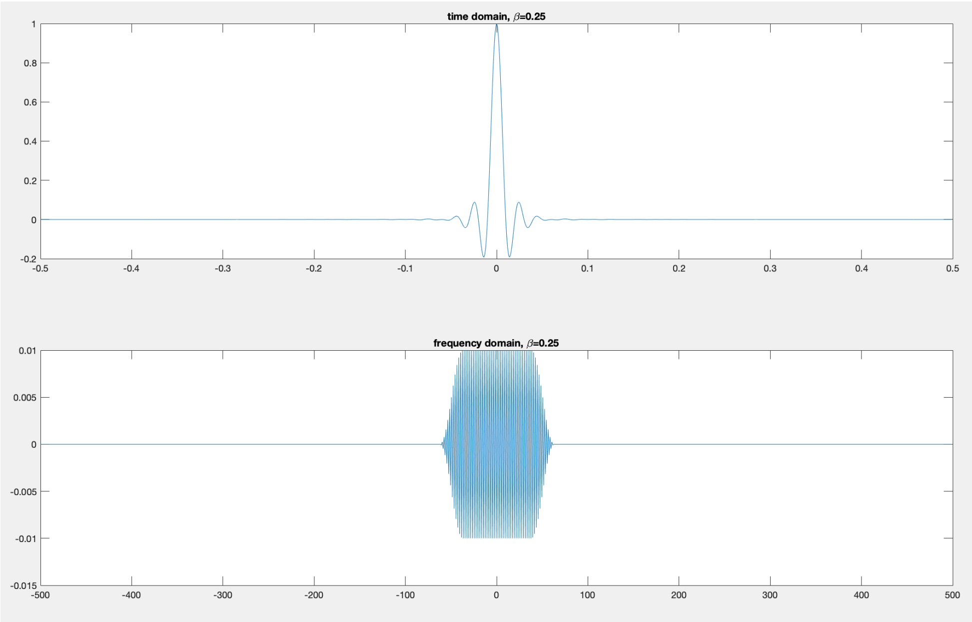
(b)
The followings are the time domain and frequency domain signal of g(t).
Observing the time domain signal, we can see that g(n*T)=1 for n=0 and g(n*T)=1 for n!=0, where T=0.01 at this case. This validates the Nyquist criterion.
Since we are using DFT to implement CTFT, we can not get the true CTFT transformation (ideally, a rectangular function).



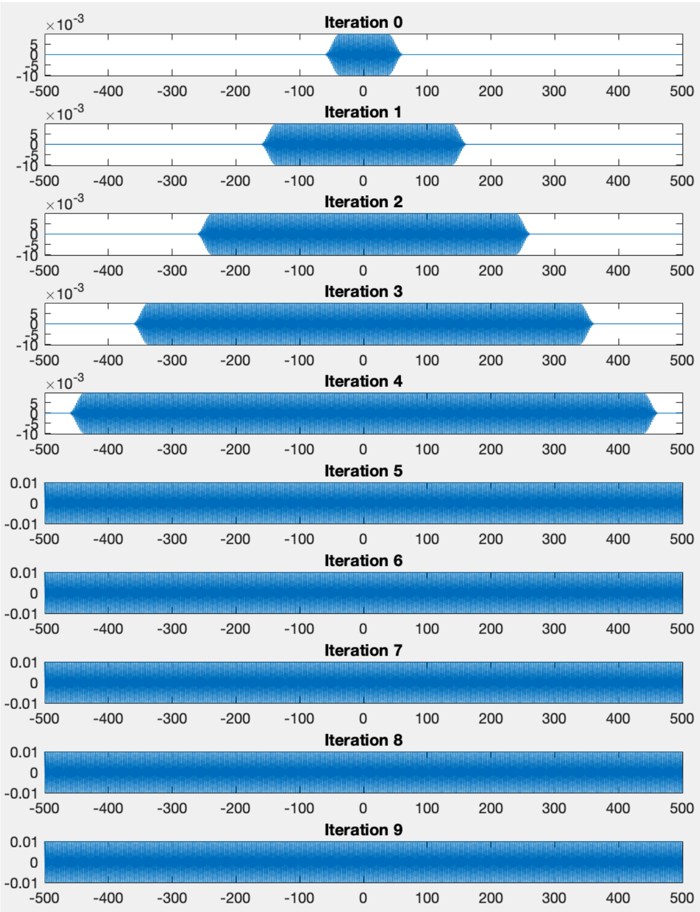
Starting from the original frequency domain signal at iteration 0, at each iteration, shift the signal 1/T units to the left and 1/T units to the right and add them to the original signal. We get the following results.
The signal converges to constant 0.01 gradually, while T=0.01 at this case. This validates the Nyquist criterion.



(c)
The followings are the time domain and frequency domain signal of g(t).
Observing the time domain signal, we can see that g(n*T)=1 for n=0 and g(n*T)=1 for n!=0, where T=0.01 at this case. This validates the Nyquist criterion.
Since we are using DFT to implement CTFT, we can not get the true CTFT transformation.
ps. the roll-off factor at this case is 0.25.



Starting from the original frequency domain signal at iteration 0, at each iteration, shift the signal 1/T units to the left and 1/T units to the right and add them to the original signal. We get the following results.
The signal converges to constant 0.01 gradually, while T=0.01 at this case. This validates the Nyquist criterion.



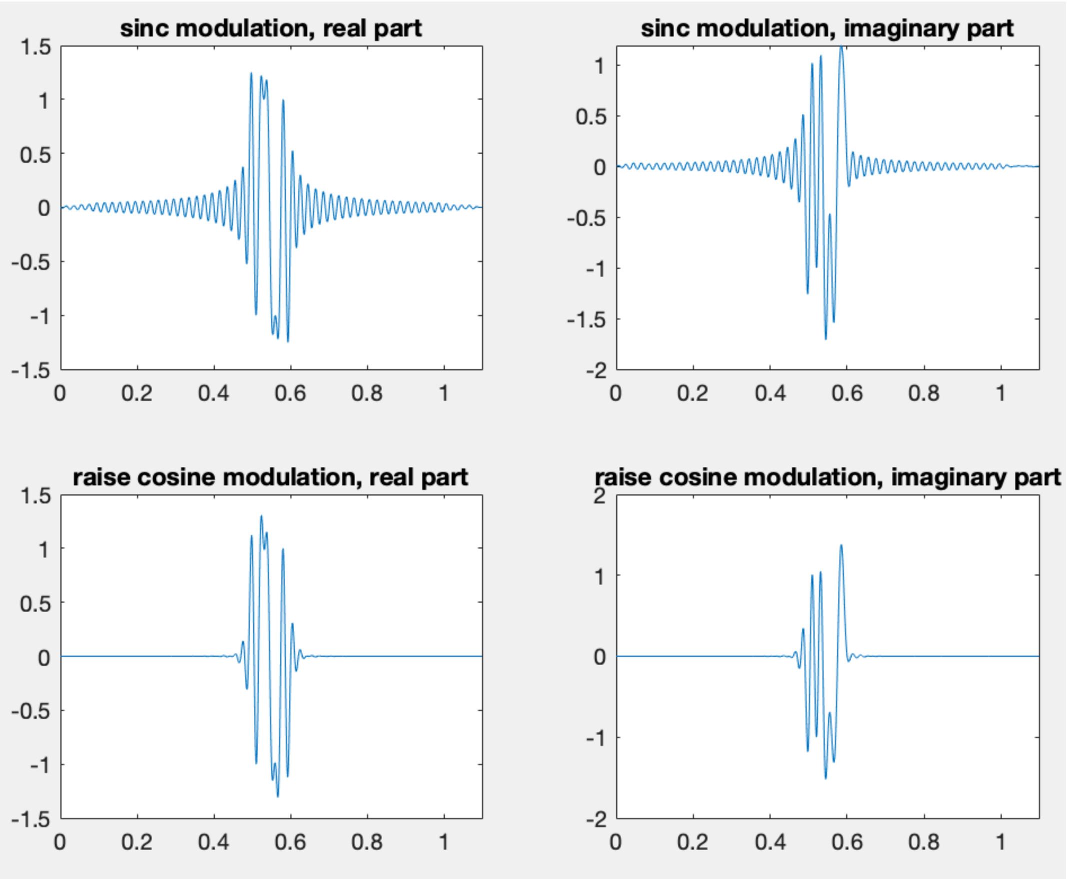
(d)

Time domain signal	Time domain decay	Frequency domain decay
Rectangular function	O(0) (limited)	O(1/f)
Sinc function	O(1/t)	O(0) (limited)
Raised cosine	O(1/t^3)	O(0) (limited)

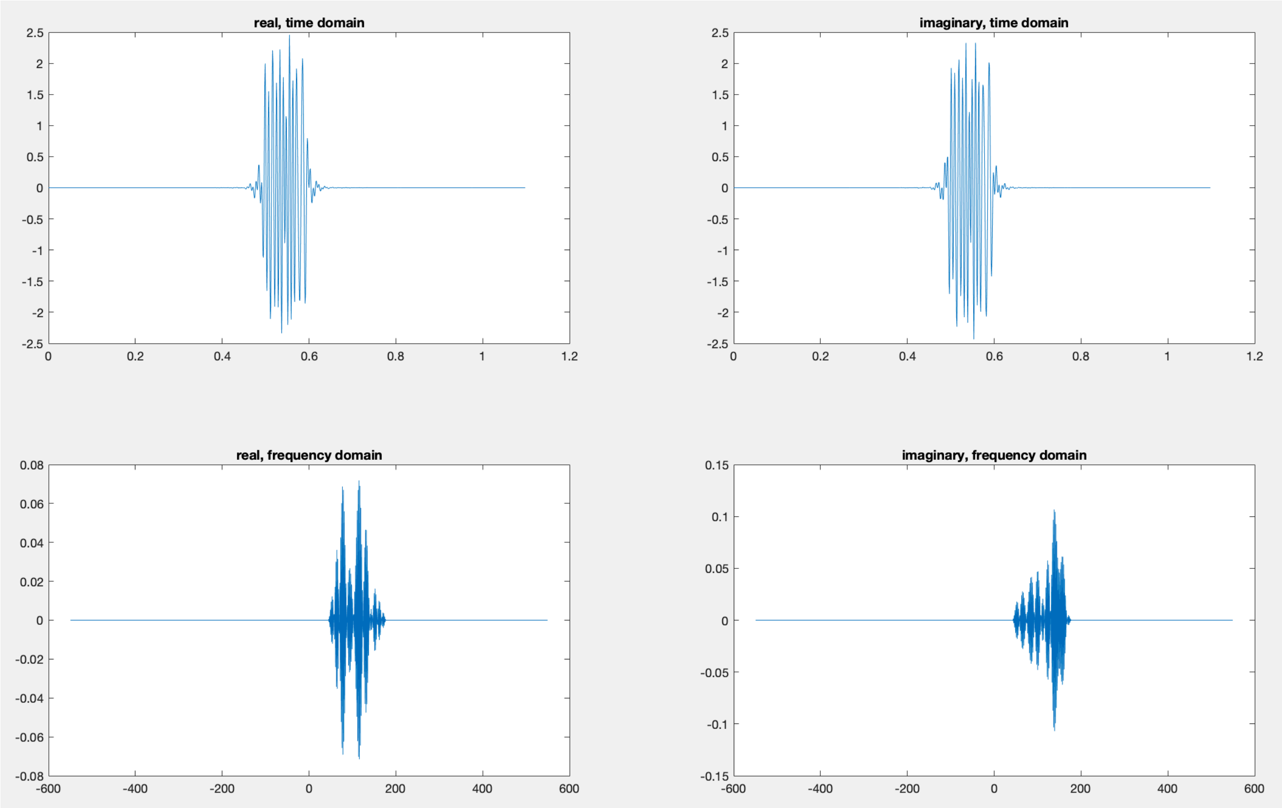
Slow decay in time domain leads to signal truncation, causing error at the receiver end. Slow decay in frequency domain requires large bandwidth. Considering both, raised cosine is the best.

(e)
The function is defined in pulse_shaper.m

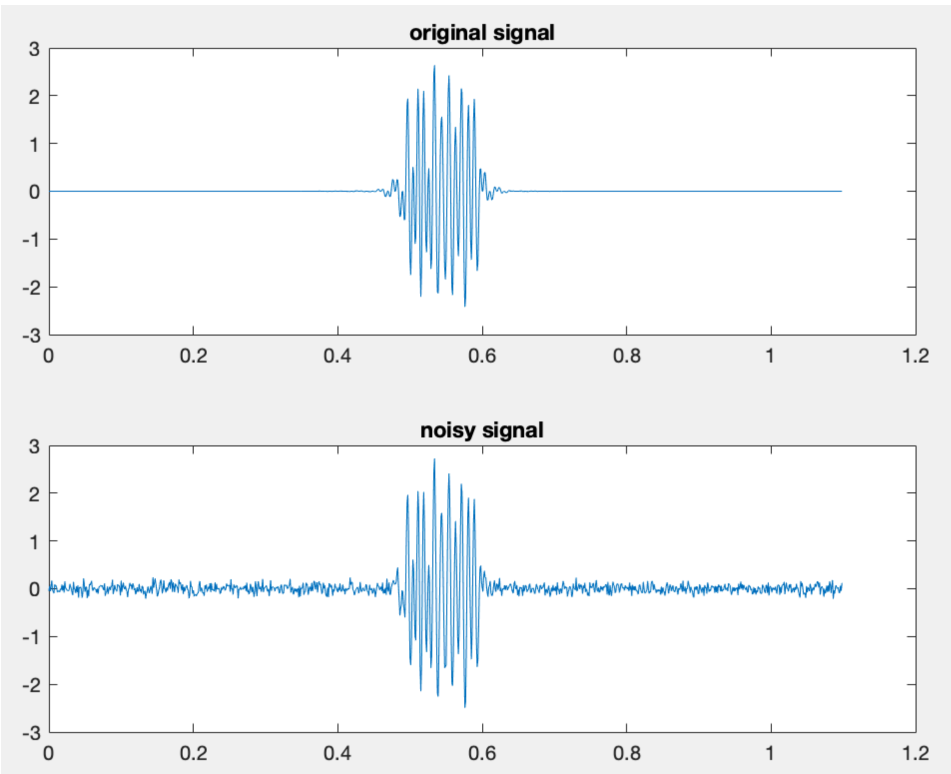
(f)
ps. The pulse duration I used is 1 sec and T=0.01. Thus, sending 20 bits generates signal with 1.2 sec duration.



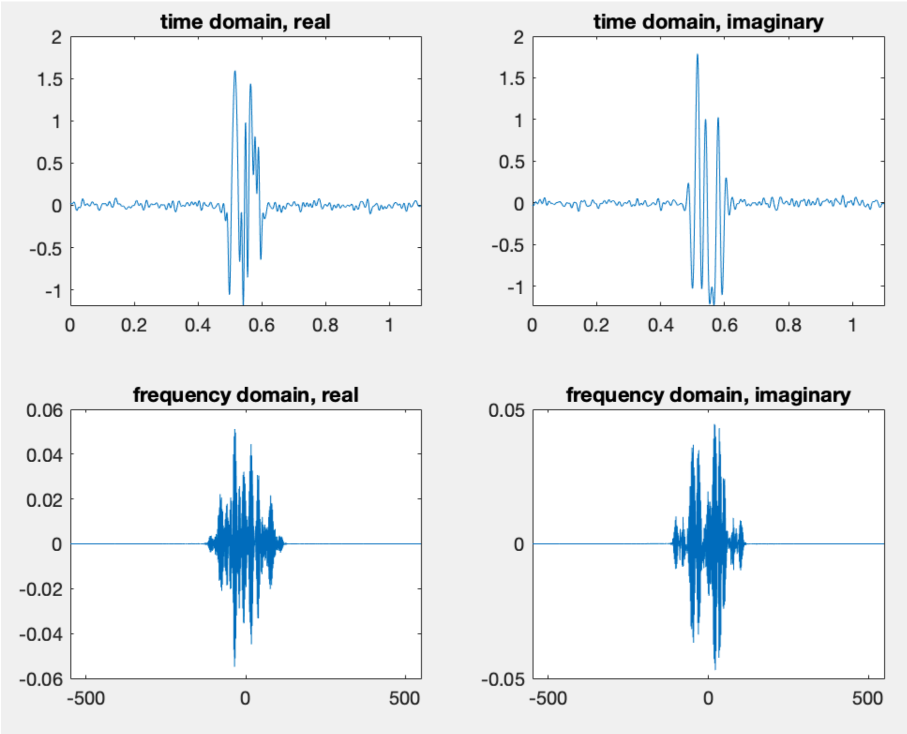
3.
(a)
I used raise cosine with roll-off factor 0.25 for pulse shaping. The signal, both in time and frequency domain, is as following.
We can see that the frequency is indeed centered around 100Hz.
ps.Since the signal is generated by a random binary sequence, we get different results each time. Thus, the signal in 2(f) is not the same as that in this part.



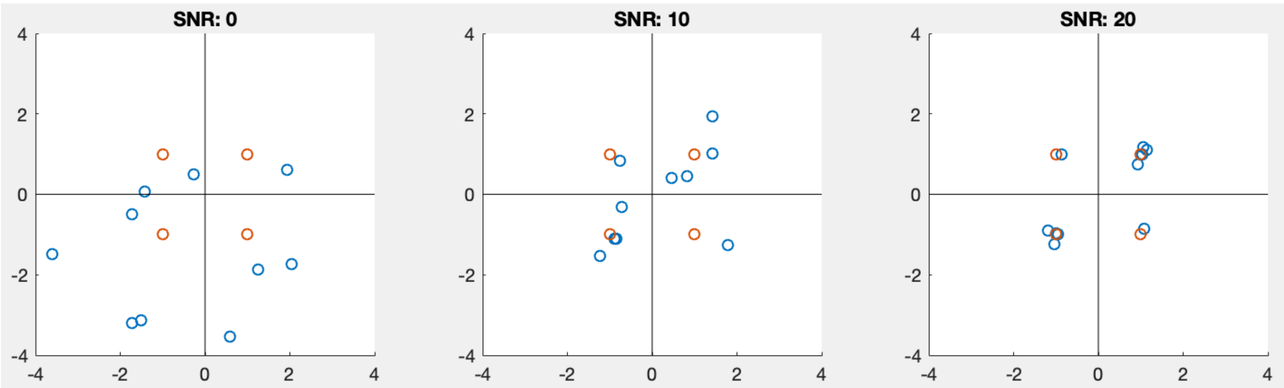
(b)
Since d=2 at this case, the average symbol energy is 2. With SNR 25dB, the variance of the AWGN is 6.3*10^(-3).
Since in real world, only the real part of the signal is sent, only the real part is plotted.



(c)
As showed in the following figure, the baseband signal is reconstructed. The real part is the in-phase part and the imaginary part is the quadrature part. Both of them are noisy.
We can observe form the frequency domain plot that the signal is indeed baseband.

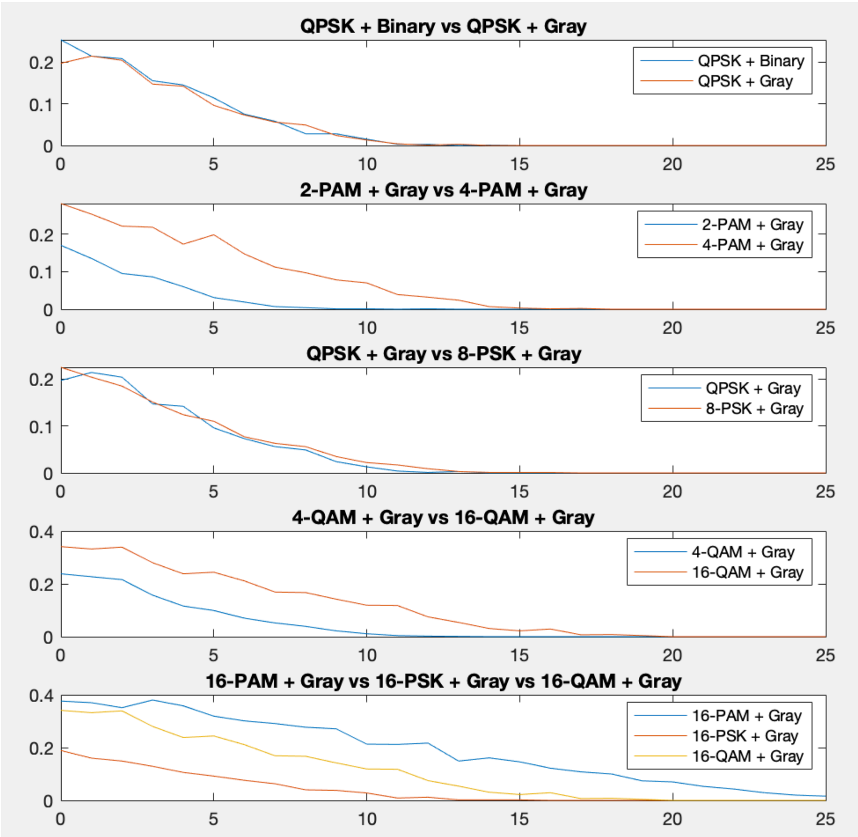


4.
(a)
I used 4QAM gray mapping. The red dots are the symbols without noise.

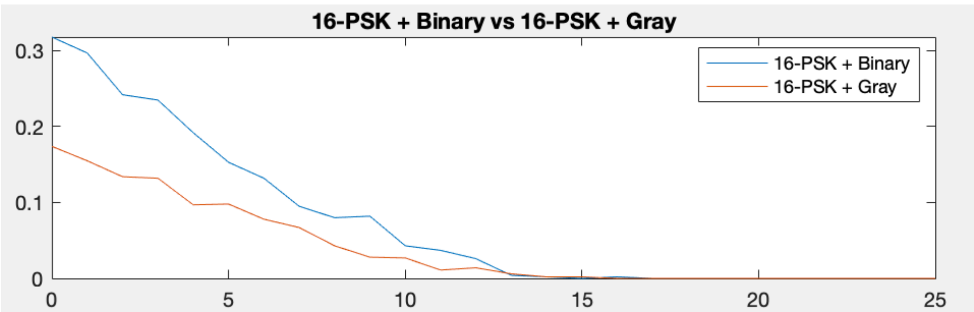


(b)
The function is defined in symbol_demapper.m.

(c)
I generated 1000 bits with uniform probability to do the following experiments.
From the first plot, we can see that QPSK binary code is as good as QPSK gray code while, usually, gray code should perform better than binary code. This is because for QPSK, the constellation set with binary code assignment is also a gray code. It can be observed from the second plot of this report. QPSK binary code is also a gray code.



To show that gray code is indeed better for most cases, I did the following extra experiment. It shows that, for 16-PSK, gray code is indeed better than binary.



5.
I used the following to complete the process.
(i) 4 bits quantization
(ii) PCM code
(iii) 4-QAM constellation mapping
(iv) SNR=0, 10, 20

From the following result, we can see that the recovered signal when SNR=20 is quite similar to the original quantized signal except for some extreme points. When SNR decreases, the signal becomes more complex and also sounds noisier.
PS. I tried to use Huffman code to implement. But with noise, Huffman code sometimes gets decode error.