ECE 233 Wireless Communications System Design, Modeling, and Implementation.

WiFi Protocol Simulation

By:

Akshay Joshi, UCLA

MATLAB Code:

```
fs = 20e6;      %Sampling rate (Hz)
Ts = 1/fs;      %Sampling time (s)
N = 64;         %No. of sub-carriers
delta_f = fs/N; %Sub-carrier spacing (Hz)
j= sqrt(-1);
S_26_26 = sqrt(1/2)* [0,0,1+j, 0,0,0,-1-j,0,0,0,1+j,0,0,0,-1-j,0,0,0,-1-j,0,0,0,1+j,0,0,0, ...
        0,0,0,0,-1-j,0,0,0,-1-j,0,0,0,1+j,0,0,0,1+j,0,0,0,1+j,0,0,0,1+j,0,0];
x_stf = zeros(1,160); % Time domain signal (samples)
for n=0:159
   temp_sum = 0;
   for k=-26:26
      temp_sum = temp_sum + S_26_26(k+27)*exp(1j*2*pi*k*delta_f*n*Ts);
   end
   x_stf(n+1) = sqrt(1/12)* temp_sum;
end

figure, plot(1:160, real(x_stf)), title('STF - Real'), xlim([1 160])

%% Generate LTF

L_26_26 = [1,1,-1,-1,1,1,-1,1,-1,1,1,1,1,1,1,-1,-1,1,1,-1,1,-1,1,1,1,1,0, ...
        1,-1,-1,1,1,-1,1,-1,1,-1,-1,-1,-1,-1,1,1,-1,-1,1,-1,1,-1,1,1,1,1];
x_ltf =zeros(1,160);
for n=0:159
   temp_sum = 0;
   for k=-26:26
      temp_sum = temp_sum + L_26_26(k+27)*exp(1j*2*pi*k*delta_f*n*Ts);
   end
   x_ltf(n+1) = sqrt(1/52)* temp_sum;
end

figure, plot(1:160, real(x_ltf)), title('LTF - Real'), xlim([1 160])

%% Packet with x_stf and x_ltf
x = [x_stf, x_ltf];
figure, plot(1:320, real(x)), title('x - Real'), xlim([1 320])
%% Part 2a,b
z_stf=x_stf(1:16);
angle_z=unwrap(angle(z_stf));
```

```
figure, plot(abs(z_stf), 'linewidth', 2), title('\fontsize{14}Magnitude of z_s_t_f'), xlim([1 16]),
xlabel('\fontsize{14}n'), ylabel('\fontsize{12}Magnitude')
figure, plot(angle_z, 'linewidth', 2), title('\fontsize{14}Angle of z_s_t_f'), xlim([1 16]),
xlabel('\fontsize{14}n'), ylabel('\fontsize{12}Angle')
%% Part 2c
SNR_dB=0:2:20;
h=1;
probability=zeros(1,length(SNR_dB));
for s=1:length(SNR_dB)
   no_of_detections=0;
   SNR_lin=10^(SNR_dB(s)/10);
   for m=1:10000
      noise = sqrt(rms(conv(x,h))^2/(2*SNR_lin))*(randn(size(x)) + 1j* randn(size(x)));
      y = conv(x,h) + noise;
      r=xcorr(z_stf,y,160);
      r=r(16:176);
      thres=mean(abs(r))+2*std(abs(r));
      peak=abs(r)>thres;
      distance=find(peak==1);
      d=1;
      if length(distance)>=9
         for i=1:length(distance)-1
            if distance(i+1)-distance(i)~=16
               d=0;
               break
            end
         end
         if d==1
            no_of_detections=no_of_detections+1;
         end
      end
   end
   probability(s)=no_of_detections/10000;
end
%%
figure, plot(SNR_dB, probability, 'linewidth',2), title('\fontsize{14}Probability of packet
detection vs SNR'), xlabel('\fontsize{12}SNR in dB'), ylabel('\fontsize{12}Probability')
%% Part 3a
h=[1, 0.9, 0.5];
Hk=fft(h,64);
SNR_dB=0:2:20;
k_set=[10,22];
```

```
Hk_cap=zeros(2,10000,11);
errors=zeros(2,length(SNR_dB));
for k=1:length(k_set)
    for s=1:length(SNR_dB)
        SNR_lin=10^(SNR_dB(s)/10);
        norm_error = 0;
        for m=1:10000
            noise = sqrt(rms(conv(x,h))^2/(2*SNR_lin))*(randn(size(conv(x,h))) + 1j*
randn(size(conv(x,h))));
            y = conv(x,h) + noise;
            y_ltf=y(161:320);
            v=y_ltf(33:96);
            temp_sum = 0;
            for n=0:63
                temp_sum = temp_sum + v(n+1)*exp(-1j*2*pi*k_set(k)*delta_f*n*Ts);
            end
            Lk_cap = 1/8 * temp_sum;
            Hk_cap(k,m,s) = sqrt(52/64)*Lk_cap/L_26_26(k_set(k)+27);
            norm_error = norm_error + (abs(Hk_cap(k,m,s)-Hk(k_set(k)))^2)/abs(Hk(k_set(k)))^2;
        end
        errors(k,s)=norm_error/10000;
    end
end
%%
figure
plot(SNR_dB, errors(1,:), 'linewidth', 2)
hold on
plot(SNR_dB, errors(2,:), 'linewidth', 2)
title('\fontsize{14}Channel estimation error for subcarriers 10 and 22')
legend('Subcarrier 10', 'Subcarrier 22', 'FontSize', 12)
xlabel('\fontsize{12}SNR in dB'), ylabel('\fontsize{12}Mean normalized squared channel
estimation error')
%% Part 3b
h=[1, 0.9, 0.5];
hfft=fft(h,64);
hfftshift=fftshift(hfft);
figure
plot(-32:31, abs(hfftshift), 'linewidth', 2), title('\fontsize{14} Magnitude of FFT of h (Shifted)'),
xlabel('\fontsize{12}Carrier'), ylabel('\fontsize{12}Magnitude')
figure
plot(abs(hfft), 'linewidth', 2), title('\fontsize{14} Magnitude of FFT of h'),
xlabel('\fontsize{12}Carrier'), ylabel('\fontsize{12}Magnitude')
```

```
%% Part 3c
M=4;
Xk_trial=zeros(12,64);
Xkifft_trial=zeros(12,64);
h=[1, 0.9, 0.5];
bit_error_10_trial=zeros(1,11);
bit_error_22_trial=zeros(1,11);
for s=1:length(SNR_dB)
    SNR_lin=10^(SNR_dB(s)/10);
    bit_error_10m_trial=0;
    bit_error_22m_trial=0;
    for m=1:10000
        for o=1:2:23
            for k=1:64
                b_in_trial(o:o+1,k) = randi([0 1], log2(M)*1, 1);
                Xk_trial((o+1)/2,k) = qammod(b_in_trial(o:o+1,k), M, 'InputType', 'bit',
'UnitAveragePower', true);
            end
        end
        for o=1:12
            Xkifft_trial(o,:)= ifft(Xk_trial(o,:), 64);
            xdatacp_trial(o,:)=[Xkifft_trial(o,end-16+1:end) Xkifft_trial(o,:)];
        end
        xdata_trial=[];
        for i=1:12
            xdata_trial=[xdata_trial xdatacp_trial(i,:)];
        end
        noise = sqrt(rms(conv(xdata_trial,h))^2/(2*SNR_lin))*(randn(size(conv(xdata_trial,h))) +
1j* randn(size(conv(xdata_trial,h))));
        y_trial=conv(xdata_trial,h)+noise;
        y_trial=y_trial(1:960);
        for d=0:11
            ydata_trial(d+1,:)=y_trial((d*80)+1:(d*80)+80);
            y_cp_rem_trial(d+1,:)=ydata_trial(d+1,17:80);
            yfft_trial(d+1,:)=fft(y_cp_rem_trial(d+1,:),64);
            y_cap_10_trial(d+1)=yfft_trial(d+1,10)/Hk_cap(1,m,s);
            y_cap_22_trial(d+1)=yfft_trial(d+1,22)/Hk_cap(2,m,s);
        end
        for d=1:2:23

y_10_demod_trial(d:d+1)=qamdemod(y_cap_10_trial((d+1)/2),M,'OutputType','bit','UnitAverag
ePower', true);
```

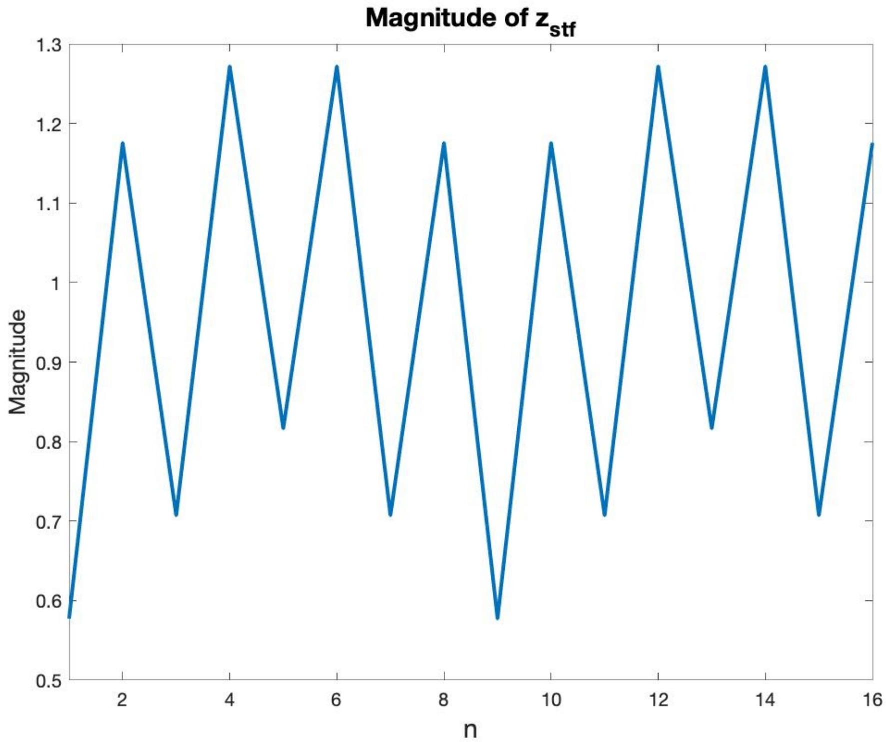
```
y_22_demod_trial(d:d+1)=qamdemod(y_cap_22_trial((d+1)/2),M,'OutputType','bit','UnitAverag
ePower', true);
        end
        bit_error_10m_trial=bit_error_10m_trial + sum(reshape(y_10_demod_trial,[24,1]) ~=
b_in_trial(:,10))/24;
        bit_error_22m_trial=bit_error_22m_trial + sum(reshape(y_22_demod_trial,[24,1]) ~=
b_in_trial(:,22))/24;
        fprintf('Iteration # %d for SNR value %d \n', m, s)
    end
    bit_error_10_trial(s) = bit_error_10m_trial/10000;
    bit_error_22_trial(s) = bit_error_22m_trial/10000;
end
%%
figure
plot(SNR_dB, 2*bit_error_10_trial, 'linewidth', 2)
xlabel('\fontsize{12}SNR in dB'), ylabel('\fontsize{12}Symbol error rate')
hold on
plot(SNR_dB, 2*bit_error_22_trial, 'linewidth', 2)
title('\fontsize{14}Symbol error rate for carriers 10 and 22 in linear scale')
legend('Carrier 10', 'Carrier 22', 'FontSize', 12)
%%
figure
semilogy(SNR_dB, 2*bit_error_10_trial, 'linewidth', 2)
xlabel('\fontsize{12}SNR in dB'), ylabel('\fontsize{12}Symbol error rate')
hold on
semilogy(SNR_dB, 2*bit_error_22_trial, 'linewidth', 2)
title('\fontsize{14}Symbol error rate for carriers 10 and 22 in log-scale')
legend('Carrier 10', 'Carrier 22', 'FontSize', 12)
%% Part 4a
M_set=[2,4,16,64];
alpha=0:0.01:1;
sum_rate_array=zeros(11,101);
for s=1:5:11
    for a=1:101
        sum_rate=0;
        for m=1:10000
            SNR=10^(SNR_dB(s)/10);
            SE_10=log2(1+(abs(Hk_cap(1,m,s))^2)*alpha(a)*SNR);
            SE_22=log2(1+(abs(Hk_cap(2,m,s))^2)*(1-alpha(a))*SNR);
            for mo=4:-1:1
                if SE_10>=log2(M_set(mo)) && SE_22>=log2(M_set(mo))
```

```
                M=M_set(mo);
                break
            elseif SE_10<=log2(M_set(mo)) || SE_22<=log2(M_set(mo))
                if mo==1
                    M=1;
                else
                    M=M_set(mo-1);
                end
            elseif SE_10<=log2(M_set(mo)) && SE_22<=log2(M_set(mo))
                M=1;
            end
        end
        sum_rate = sum_rate + (3*2*log2(M)*delta_f)/(4*10^6);
    end
    sum_rate_array(s,a) = sum_rate/10000;
  end
end
%%
figure
plot(alpha, sum_rate_array(1,:), 'linewidth', 2)
xlabel('\fontsize{12}?'), ylabel('\fontsize{12}Mean sum rate in Mbps')
hold on
plot(alpha, sum_rate_array(6,:), 'r', 'linewidth', 2)
hold on
plot(alpha, sum_rate_array(11,:), 'k', 'linewidth', 2)
title('\fontsize{14}Mean sum rate vs ? (Same modulation)')
legend('SNR = 0 dB','SNR = 10 dB','SNR = 20 dB', 'FontSize',12)
%% Part 4b
M_set=[2,4,16,64];
alpha=0:0.01:1;
sum_rate_array_4b=zeros(11,101);
for s=1:5:11
  for a=1:101
    sum_rate=0;
    for m=1:10000
        SNR=10^(SNR_dB(s)/10);
        SE_10=log2(1+(abs(Hk_cap(1,m,s))^2)*alpha(a)*SNR);
        SE_22=log2(1+(abs(Hk_cap(2,m,s))^2)*(1-alpha(a))*SNR);
        for mo=4:-1:1
            if SE_10>=log2(M_set(mo))
                M00=M_set(mo);
                break
```

```
            elseif SE_10<=log2(M_set(mo))
                if mo==1
                    M10=1;
                else
                    M10=M_set(mo-1);
                end
            end
        end
        for mo=4:-1:1
            if SE_22>=log2(M_set(mo))
                M22=M_set(mo);
                break
            elseif SE_22<=log2(M_set(mo))
                if mo==1
                    M22=1;
                else
                    M22=M_set(mo-1);
                end
            end
        end
        sum_rate = sum_rate + (3*(log2(M10)+log2(M22))*delta_f)/(4*10^6);
    end
    sum_rate_array_4b(s,a) = sum_rate/10000;
  end
end
%%
figure
plot(alpha, sum_rate_array_4b(1,:), 'linewidth', 2)
xlabel('\fontsize{12}?'), ylabel('\fontsize{12}Mean sum rate in Mbps')
hold on
plot(alpha, sum_rate_array_4b(6,:), 'r', 'linewidth', 2)
hold on
plot(alpha, sum_rate_array_4b(11,:), 'k', 'linewidth', 2)
title('\fontsize{14}Mean sum rate vs ? (Different modulation)')
legend('SNR = 0 dB','SNR = 10 dB','SNR = 20 dB', 'FontSize',12)
alpha(find(sum_rate_array(11,:)==max(sum_rate_array(11,:))))
alpha(find(sum_rate_array_4b(11,:)==max(sum_rate_array_4b(11,:))))
max(sum_rate_array(11,:))
max(sum_rate_array_4b(11,:))
```

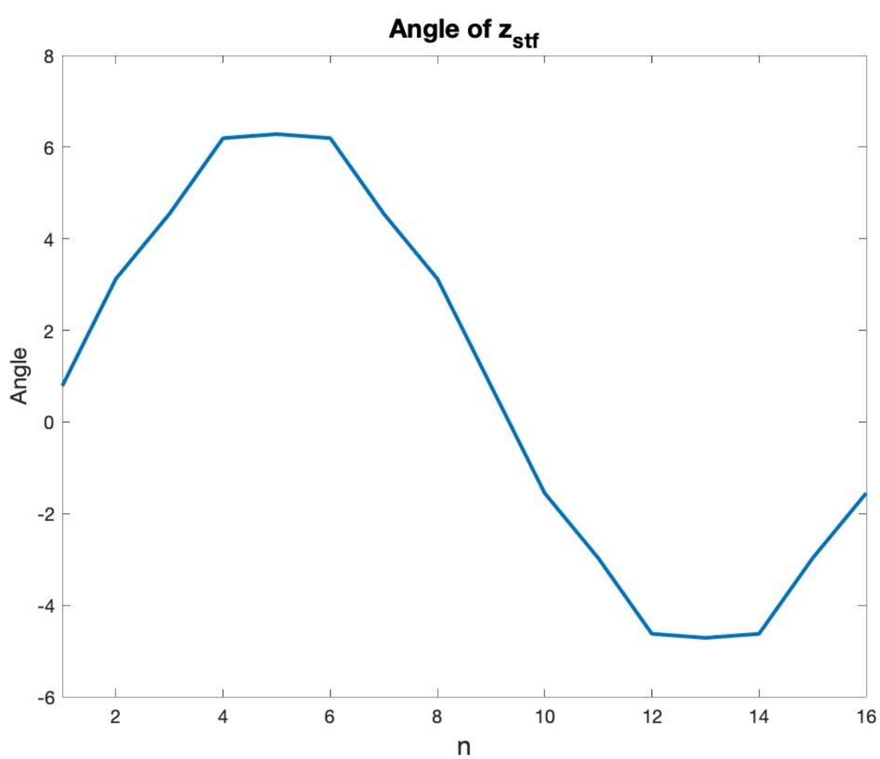
Part 2

a)



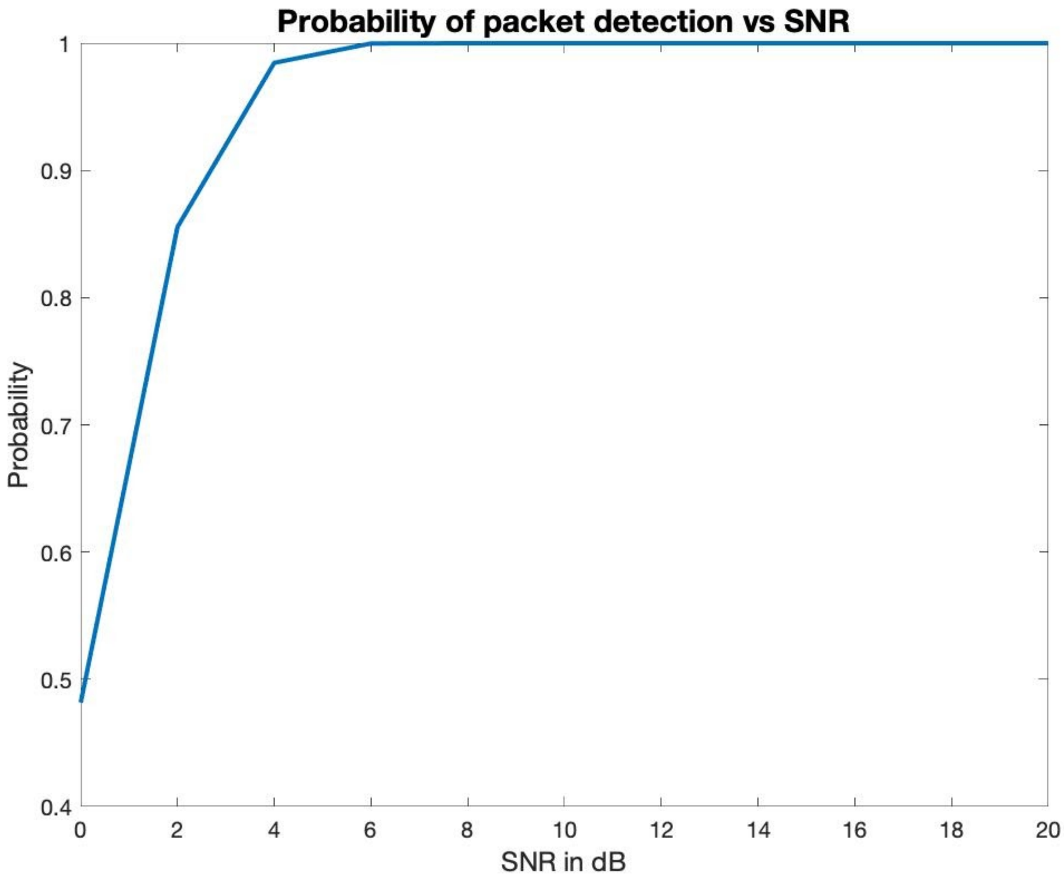
Magnitude of z_{stf}
Magnitude
n
1.3
1.2
1.1
1
0.9
0.8
0.7
0.6
0.5
2
4
6
8
10
12
14
16

b)



Angle of z_{stf}
Angle
n
8
6
4
2
0
-2
-4
-6
2
4
6
8
10
12
14
16

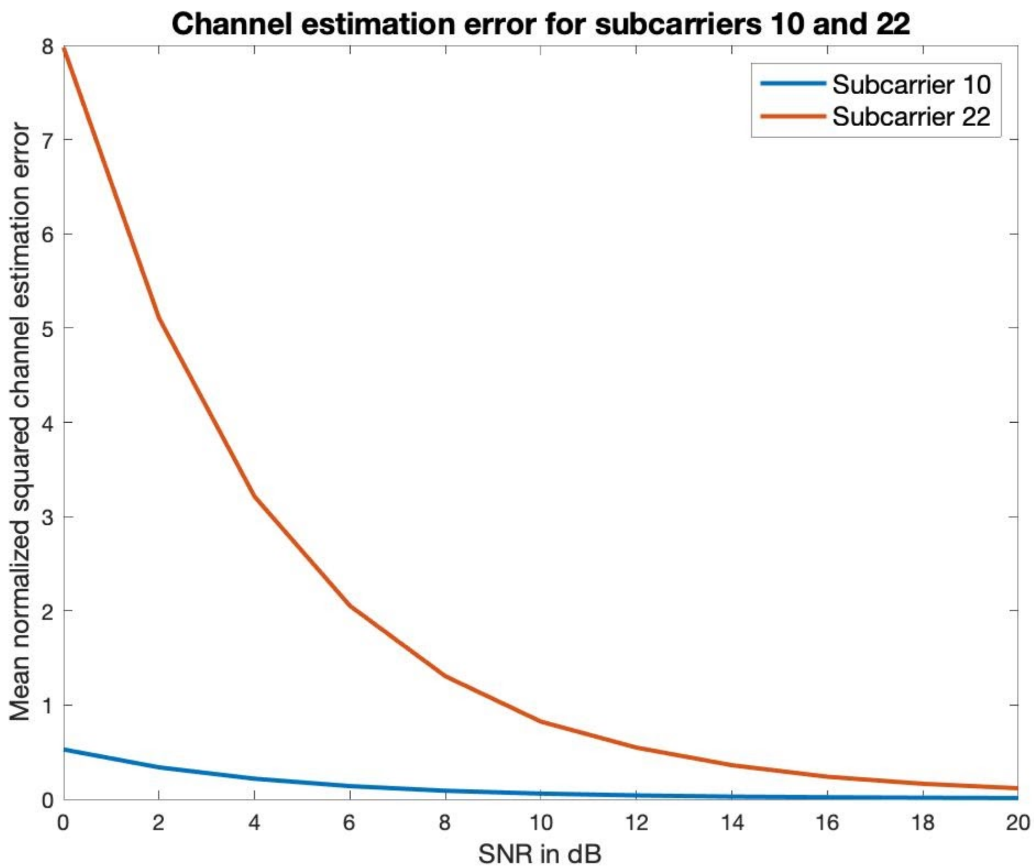
c)



Probability of packet detection vs SNR
Probability
SNR in dB
1
0.9
0.8
0.7
0.6
0.5
0.4
0
2
4
6
8
10
12
14
16
18
20

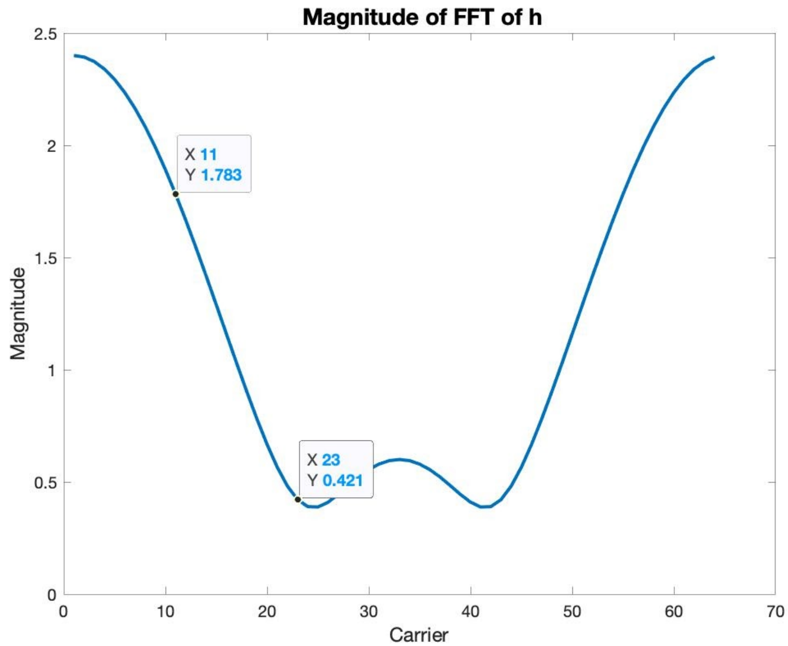
Part 3

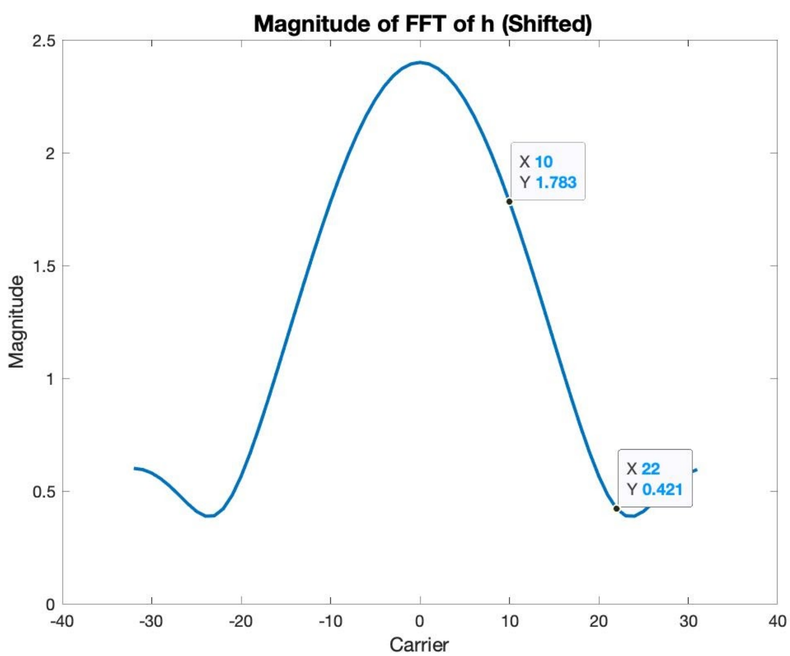
a)



Channel estimation error for subcarriers 10 and 22
Subcarrier 10
Subcarrier 22
Mean normalized squared channel estimation error
SNR in dB
0
1
2
3
4
5
6
7
8
0
2
4
6
8
10
12
14
16
18
20

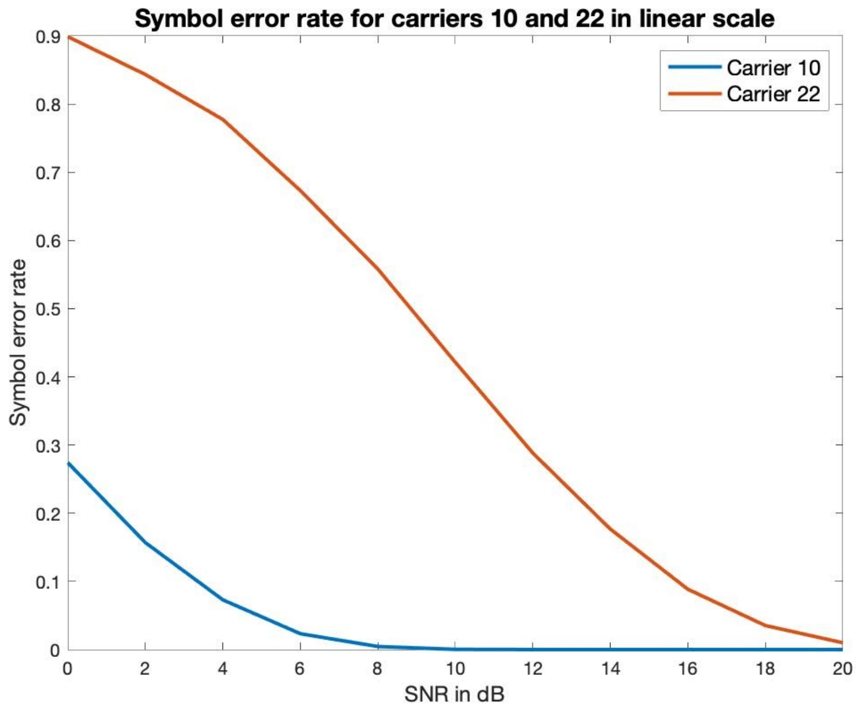
b)

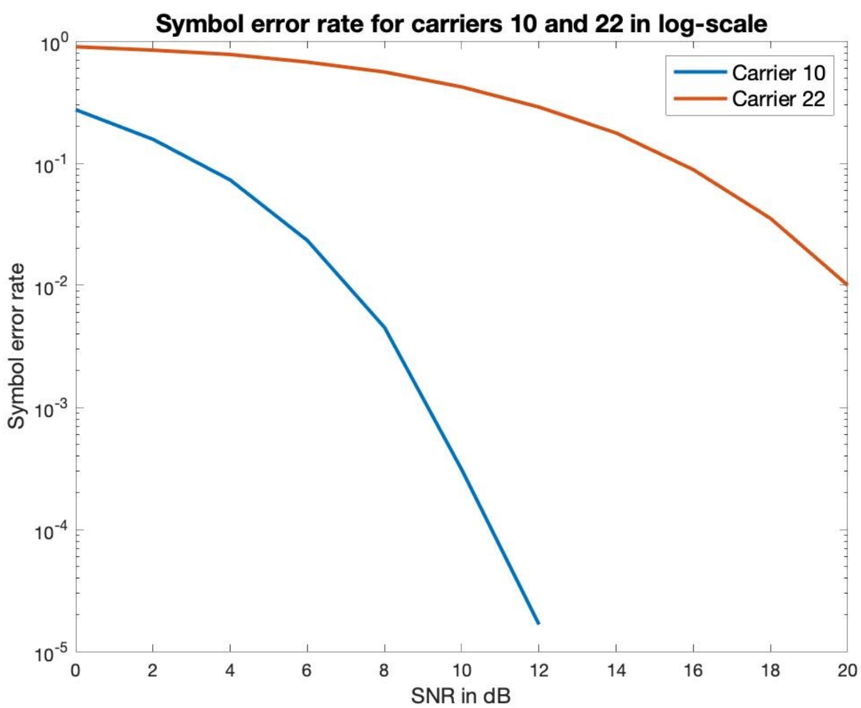




The received symbol on a subcarrier k is given by $R_k = H_k . X_k + N$. In this case, the signal power is scaled by $|H_k|^2$ and the noise power is constant (N^2). Hence, based on $|H_k|$, each subcarrier will have different SNRs. Higher the value of $|H_k|$, higher will be the SNR for subcarrier k. Thus, as depicted in the plots above, subcarrier 10 has higher $|H|$ i.e. higher SNR than subcarrier 22. Therefore, subcarrier 22 shows worse channel estimation error than subcarrier 10.

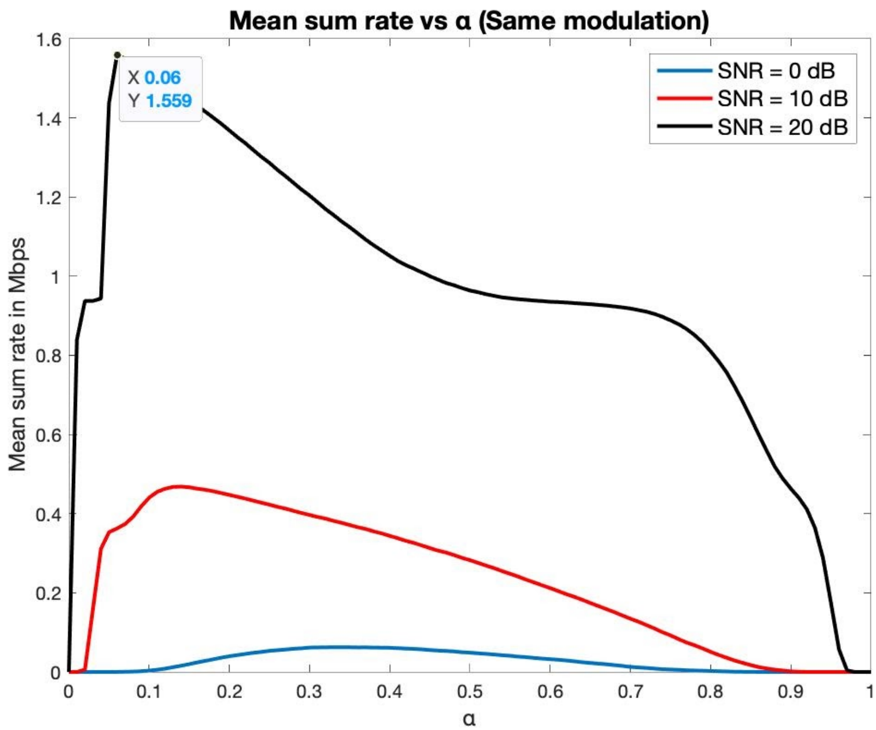
c)


Symbol error rate for carriers 10 and 22 in linear scale
Carrier 10
Carrier 22
Symbol error rate
SNR in dB
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
2
4
6
8
10
12
14
16
18
20

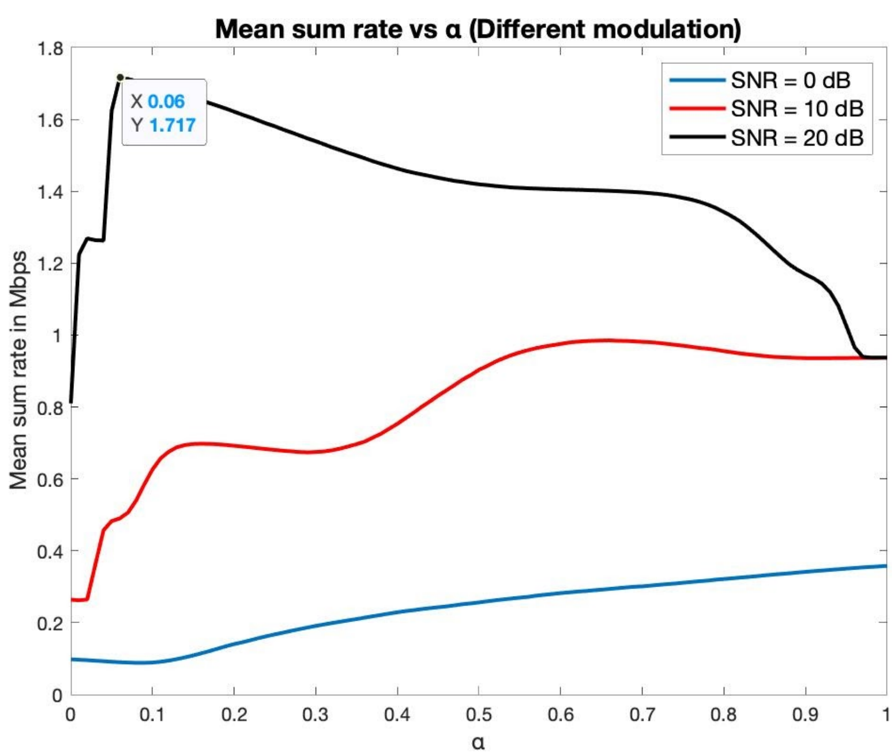

Symbol error rate for carriers 10 and 22 in log-scale
Carrier 10
Carrier 22
Symbol error rate
SNR in dB
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
0
2
4
6
8
10
12
14
16
18
20

Part 4

a)


Mean sum rate vs α (Same modulation)
X 0.06
Y 1.559
SNR = 0 dB
SNR = 10 dB
SNR = 20 dB
Mean sum rate in Mbps
α
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
0.1
0.3
0.5
0.7
0.9

b)


Mean sum rate vs α (Different modulation)
X 0.06
Y 1.717
SNR = 0 dB
SNR = 10 dB
SNR = 20 dB
Mean sum rate in Mbps
α
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
0.1
0.3
0.5
0.7
0.9

Considering bit loading first, in 3a (fixed modulation size for both carriers), both the subcarriers are restricted to use the same modulation scheme. Naturally, the subcarrier with lower spectral efficiency supports lower modulation size which leads to the subcarrier with higher efficiency to support only that modulation size even though it can transmit at a higher rate. This significantly reduces the overall rate that can be achieved. On the other hand, in 3b, when different modulation sizes for every subcarrier are allowed, higher rates are achieved since the better subcarrier can transmit at a higher rate than the not-so-better subcarrier, thus increasing the total mean sum rate.

Further, alpha is used to divide the total power budget between the two subcarriers. For example, subcarrier 10 could be given a higher fraction of the power, thus increasing its spectral efficiency. But it comes with a tradeoff; the subcarrier 22 will have a lesser amount of power resulting in lower spectral efficiency. Thus, the overall sum rate is reduced. Therefore, the optimal value of alpha is where the maximum overall sum rate is achieved. For both 3a and 3b, the optimal value of alpha was found out to be 0.06. The rate obtained when using the same modulation size was 1.559 Mbps. It was improved to 1.717 Mbps when different modulation sizes were employed. As alpha was increased, 3a showed a decrease in the mean sum rate even for 20 dB SNR. On the other hand, 3b showed a relative increase since different modulation sizes enabled the better subcarrier to transmit at a higher rate leading to a better mean sum rate.