1 Theory Problems

1. *Frequency offset correction using a periodic sequence*
   Consider the preamble structure for shown below, which is used for IEEE802.11.

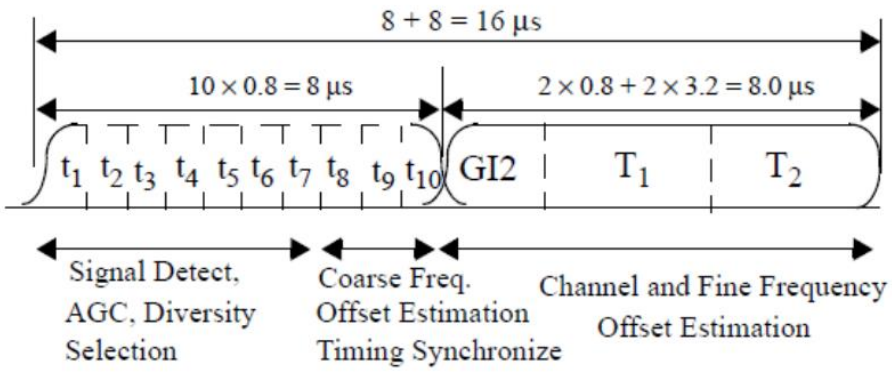


Figure 17-4—OFDM training structure

Figure 1: PLCP (Physical Layer Convergence Procedure) preamble structure

The PLCP preamble field is used for synchronization. It consists of 10 *identical* short sequences and two *identical* long sequences. The sampling rate for a this system is $f_s = 20$ MHz and the length of the short and long sequences are $T_{\text{short}} = 0.8\mu s$ and $T_{\text{long}} = 3.2\mu s$.

a) Calculate L_{short} and L_{long} in samples.

L_{short} = T_{short}*Fs = 0.8us*20MHz = 16samples/short sequence

L_{long} = T_{long}*Fs = 3.2us*20MHz = 64samples/long sequence

b) Consider the received time-series $r[n]$, which consists of the above preamble time-series $x[n]$ with a frequency offset f_0 and initial phase offset ϕ_0. Assume the following

- Index $n = n_1$ corresponds to the start-time of *short sequence* "t_1".
- Index $n = N_1$ corresponds to the start-time of *long sequence* "T_1".
- The received time series can be written as $r[n] = e^{j(2\pi f_0 n + \phi_0)} x[n]$, and the frequency offset f_0 stays constant for the duration of the preamble. Note that f_0 is a normalized frequency given by $f_0 = \epsilon/N = f_e T_s$ where f_e is the absolute frequency error and T_s is the sample time (see Lecture 8A).
- There is no additive noise in the measurements of $r[n]$.

i. Calculate $Z_1 = r^*[n_1] r[n_1 + L_{\text{short}}]$.

ii. Calculate $Z_2 = r^*[N_1] r[N_1 + L_{\text{long}}]$.

r[n1] = $e^{j(2\pi f0 n1 + \phi 0)}$ x[n1], r[n1+L_{short}] = $e^{j(2\pi f0 n1 + L_{short} + \phi 0)}$ x[n1+L_{short}]

Z1 = r^* [n1]r[n1 + L_{long}]= $e^{-j(2\pi f0 n1 + \phi 0)} e^{j(2\pi f0 n1 + L_{short} + \phi 0)} x^*$[n1]x[n1 + L_{short}]

$= e^{jL_{short}} x^*[n1]x[n1 + L_{short}] = e^{jL_{short}} R(n1)$

r[N1] = $e^{j(2\pi f0N1+\phi0)}$ x[N1], r[N1+L_{long}] = $e^{j(2\pi f0n1+L_{short}+\phi0)}$ x[N1+L_{long}]

Z2 = r^* [N1]r[N1 + L_{long}]= $e^{-j(2\pi f0N1+\phi0)} e^{j(2\pi f0N1+L_{short}+\phi0)} x^*$[N1] x[N1+$L_{long}$]

$= e^{jL_{long}} x^*$[N1] x[N1+L_{long}] = $e^{jL_{long}} R(N1)$

c) Use your results from part (b) to find an expression for the frequency offset f_0 for:

i. The short sequence.

$e^{jL_{short}}$[n1][n1 + L_{short}] = $e^{jL_{short}} R\ (n1)$

$\hat{\phi} = tan^{-1}(\frac{Im[e^{jL_{short}} R\ (n1)]}{Re[e^{jL_{short}} R\ (n1)]})$

$\hat{f}_0 = \hat{\phi}/\ 2\pi T_{short}$

ii. The long sequence.

$e^{jL_{long}}$[N1][N1 + L_{long}] = $e^{jL_{long}} R\ (N1)$

$\hat{\phi} = tan^{-1}(\frac{Im[e^{jL_{long}} R\ (N1)]}{Re[e^{jL_{long}} R\ (N1)]})$

$\hat{f}_0 = \hat{\phi}/\ 2\pi T_{long}$

d) What is the maximum estimation range (in Hertz) of:

i. The short sequence.

ii. The long sequence.

(Hint1: What is the range of the $arg(.)$ function?)
(Hint2: Use the following expression to convert from normalized frequency to hertz, $f_0 = f_e T_s$)

Max phase can accurately estimate is π

So, max f_e < 1/$2T_{short}$ *for sort sequence,*

max f_e < 1/$2T_{long}$ for long sequence.

2. *Forward error correction across OFDM subchannels*
Consider the Hamming $(7,4)$ code used in conjunction with an $N = 8$ subcarrier OFDM scheme in order to mitigate frequency selective fading. This system operates using an $N = 8$ FFT, but nulls out the "DC" subcarrier (i.e. subcarrier index $k = 0$) which makes for 7 "effective" sub-channels.

- *Case I:* The probability of code-bit error is $p_b = 10^{-4}$ for all 7 sub-channels.
- *Case II:* The probabilty of code-bit error is $p_b = 10^{-4.5}$ for 6 of the 7 sub-channels, and $p_{b'} = 10^{-1}$ for the remaining sub-channel.

a) Compute the coded block-error rate for *Case I*.

CBER: the probability that a codeword is not correctly decoded at the receiver.

Case I: Using the Hamming (7,4) code, the codeword can detect and correct any single-bit error. So, the probability of a codeword being decoded incorrectly is the probability that 2 or more bit errors.

The probability of 2 or more bit errors:

$$P_{2\_or\_more\_error} = \sum_{i=2}^{7} {P_b}^{i} * (1 - P_b)^{(7-i)} C(7,i)$$

The CBER = $P_{2\_or\_more\_error}$ = 1.657e-10 when plug in pb=10^-4

b) Compute the coded block-error rate for *Case II*.

In Case II, the probability of k subchannels having errors

$$P_{2\_or\_more\_error} = \sum_{k=2}^{7} P_k * {P_1}^{k} * {P_1}'^{(7-k)}$$

Where Pk = C(7,k) * ${P_b}^{k}$ * (1-P_b)$^{(7-k)}$ * (1-P_b')$^{(7-k)}$

If Plug in pb=10^-4.5 and pb'=10^-1,:

p1 = 0.0004205

p1' = 0.4783

Pk for k=2 to 7 are: [3.027e-10, 2.019e-8, 5.332e-7, 5.892e-6, 3.499e-5, 0.001278]

The CBER = $P_{2\_or\_more\_error}$ = 0.4798

c) Under which case does the system perform better? Why?

Case I is better, the overall BER of the system is very lower than case II as approximately 7x10^-4 and the CBER for Case I is also very low at approximately 1.7 x 10^-10 .

: It means case I system is able to provide reliable transmission of the data, even in the presence of channel noise and fading.

3. *Peak to Average Power Ratio*
Consider a classic OFDM system that utilizes an N point IFFT/FFT for its modulator/demodulator. Assume that the system uses $\frac{N}{16}$ pilots subcarriers and nulls out $\frac{N}{16}$ subcarriers (i.e. not-utilized).

$$x[n] = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} X_k e^{j\frac{2\pi}{N}nk}$$

a) Assuming QPSK modulated subcarriers, calculate the worst-case peak-to-average power ratio

$$\text{PAPR} = \frac{max\left(|x[n]|^2\right)}{E\left(|x[n]|^2\right)}$$

the worst-case peak-to-average power ratio (PAPR) occurs when all subcarriers are in phase and have equal amplitude(assume the pilot subcarriers are all in the same phase/equal amplitude): 15N/16 subcarriers are in the same phase/equal amplitude.

|x[n]| = $1/\sqrt{N}\,(\sqrt{2}A)(\frac{15N}{16})$, A: the amplitude of each QPSK symbol.

max(|x[n]|^2) = 2*$\frac{A^2}{N} * (\frac{15N}{16})^2$

E(|x[n]|^2) = $(\frac{1}{N})$ * SUM(|X[n]|^2),

X[k] : frequency-domain representation of the QPSK symbols, and the sum is over all subcarriers.

the average power of each symbol(QPSK) = A^2,

the average power of each I/Q = (1/2) A^2,

so, the average power of the time-domain signal is

E(|x[n]|^2) = (1/N) * SUM(|X[k]|^2) = $\left(\frac{15N}{16}\right) A^2/N$

PAPR = |x[n]|^2/ E(|x[n] |^2) = (2*$\frac{A^2}{N} * (\frac{15N}{16})^2$)/($\frac{\left(\frac{15N}{16}\right)A^2}{N}$) = 15N/8

b) Given that the maximum tolerated PAPR (dB) of your radio is $10dB$, calculate the maximum number of active subcarriers N_A such that the probability that the PAPR is below $10dB$ is greater than 99%. The following expression may be useful

$$Pr\left\{\text{PAPR}\left(x[n]\right) > \gamma\right\} = 1 - \left(1 - e^{-\gamma}\right)^{N_A}$$

where γ is the value of the PAPR.

The probability of the PAPR exceeding a given threshold gamma (γ) for an OFDM system with NA active subcarriers can be expressed as:

Pr{PAPR(x[n]) > 10} = 1 - (1 - e $^{-10}$)^{N_A}

i.e. Pr{PAPR(x[n]) $\leq$ 10 } = (1 - e $^{-10}$)^{N_A} $>$ 0.99

(1 - e $^{-10}$)^{N_A} $>$ 0.99

If take the natural logarithm of both sides:

N_A $*$ln (1 - e $^{-10}$) $>$ ln(0.99)

If solve for NA:

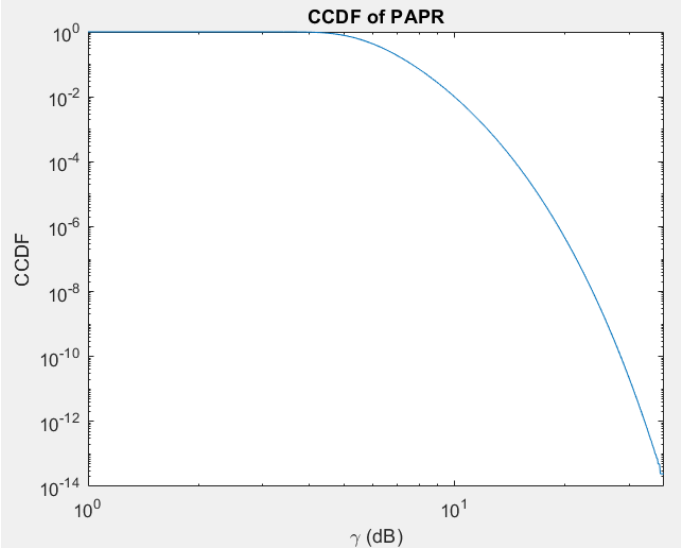
N_A $<$ ln(0.99)/ ln (1 – e $^{-10}$) as ln (1 – e $^{-10}$) $<$ 0

N_A < 221.368354

Max(N_A) = 221

c) For your calculated value of N_A, plot the theoretical complementary cumulative distribution function of the PAPR in log-log scale as a function of γ.

```
NA = 221; % maximum number of active subcarriers
gamma = logspace(0, 2, 1000); % range of gamma values to plot
ccdf = 1 - (1 - exp(-gamma)).^NA; % calculate the CCDF
loglog(gamma, ccdf); % plot the CCDF in log-log scale
xlabel('\gamma (dB)');
ylabel('CCDF');
title('CCDF of PAPR');
```



2 Simulation

1. *Delay Spread Model*

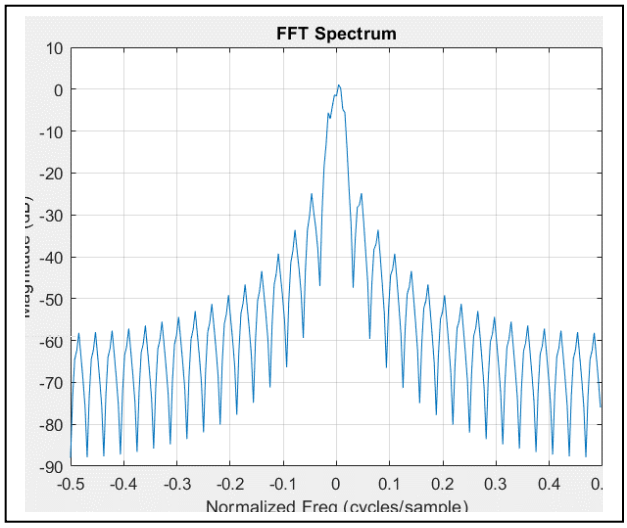
   a) Recall the channel model we created in Lab 4 (i.e. the two-path model) which took an input data sequence and simulated a delayed path with adjustable gain. We will be using a similar model for this simulation, but without any noise or adjustable gain. Using the 32 subcarrier data from the previous simulation problem in Prelab 6, simulate the channel given by the discrete impulse response

   $$h[n] = \delta[n] - \delta[n - 16].$$

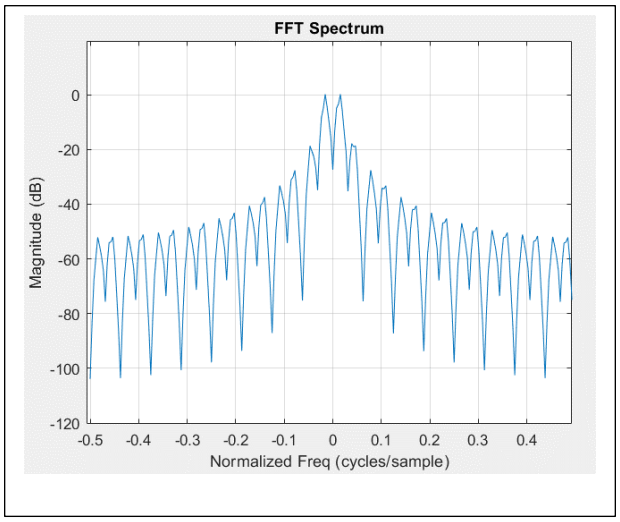
   For this impulse response, the block of 32 values is shifted by half the length of the block and added to the original block.

   i. Plot the power density spectrum for the received signal given this channel model.

< without fading >

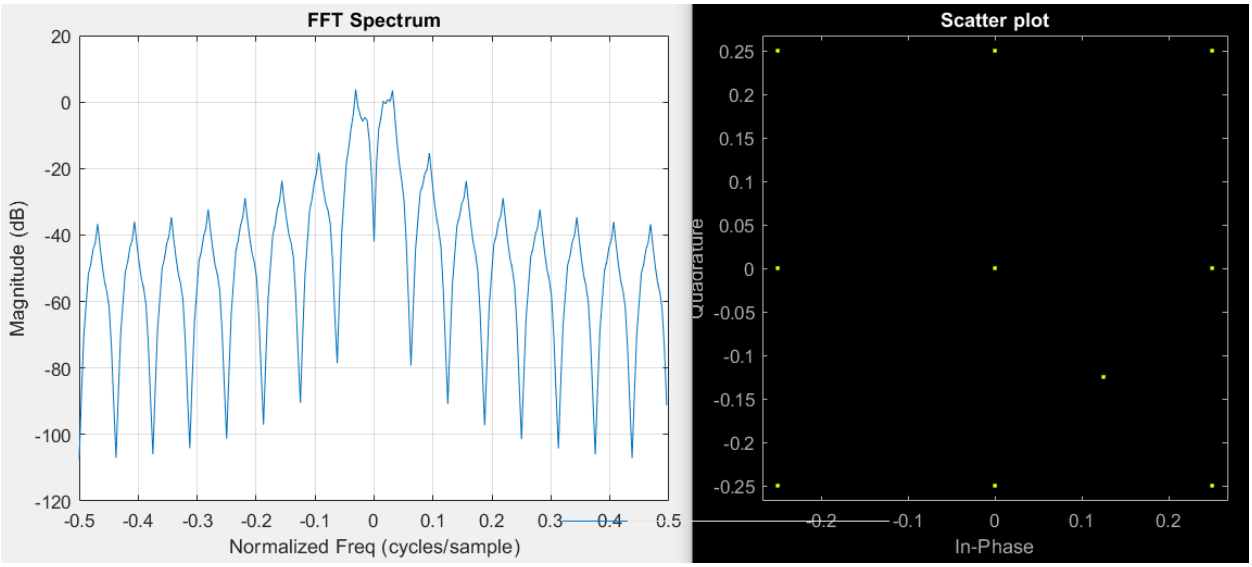


< with fading >

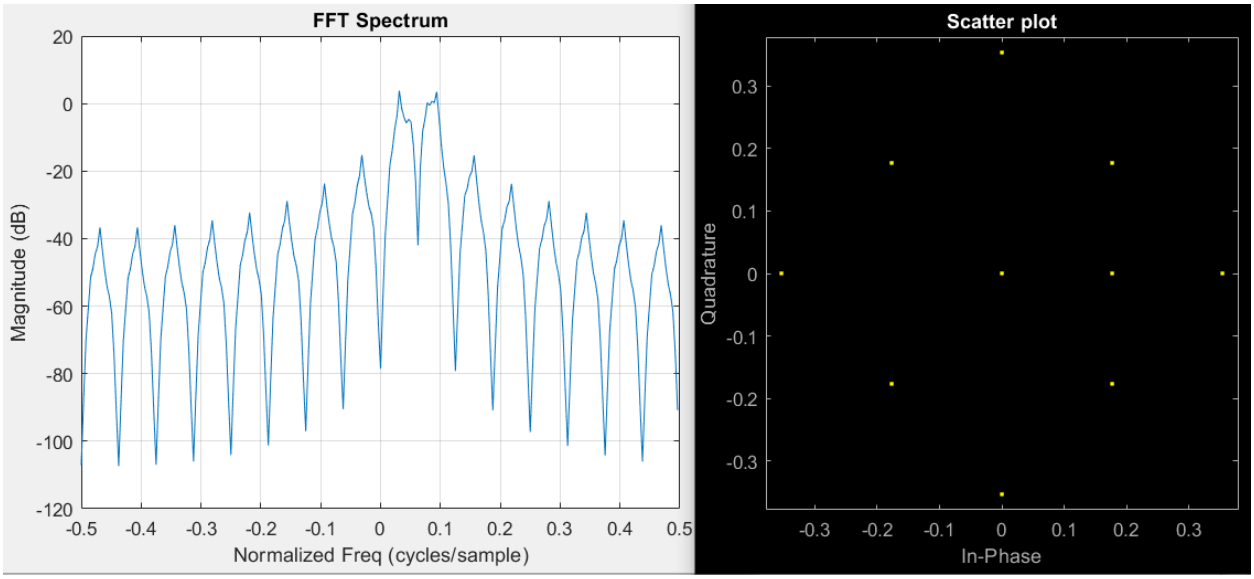


ii. Plot the constellation of several subcarriers and comment on the result. Are all the QPSK signal constellations for each subcarrier the same?

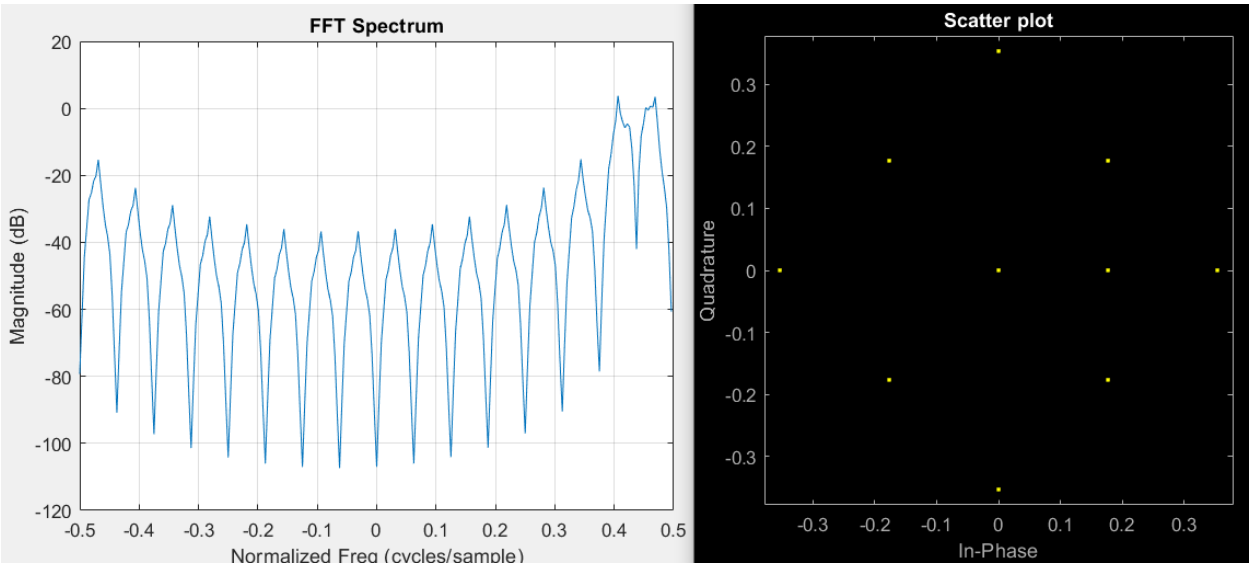
< subcarrier f0 only enabled without CP>



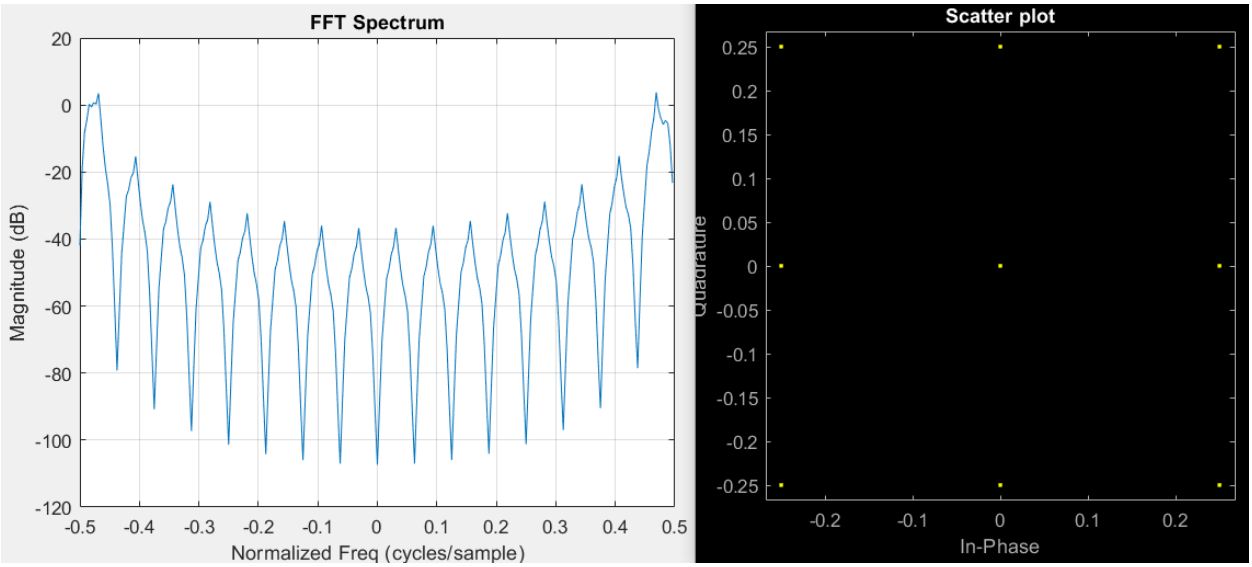
< subcarrier f2 only enabled without CP >



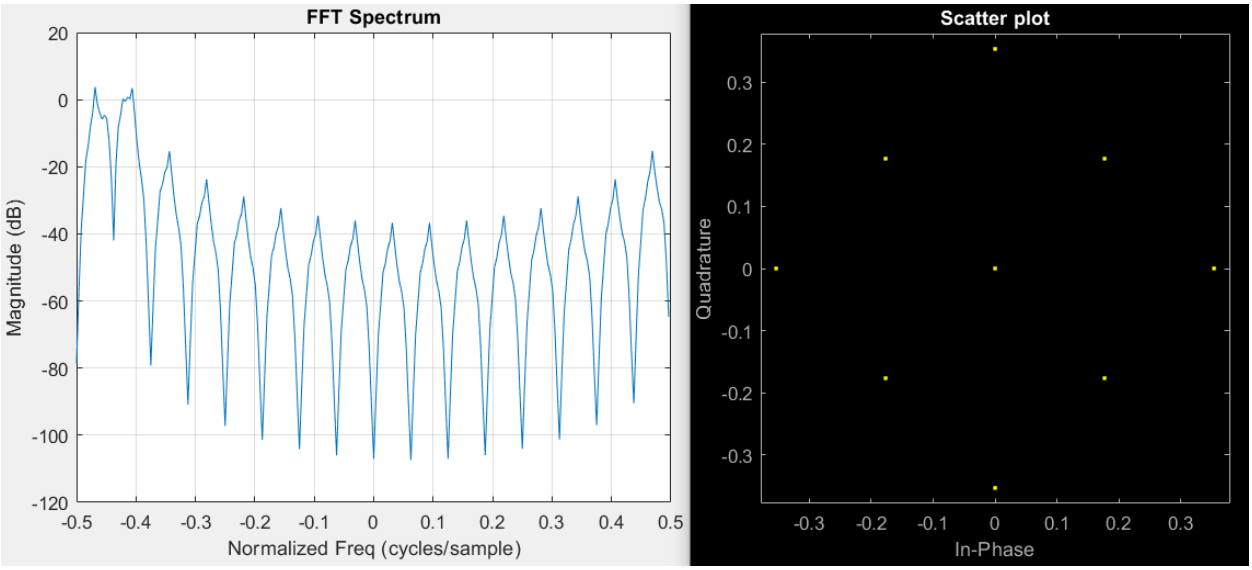
< subcarrier f14 only enabled without CP >



< subcarrier f16 only enabled without CP >



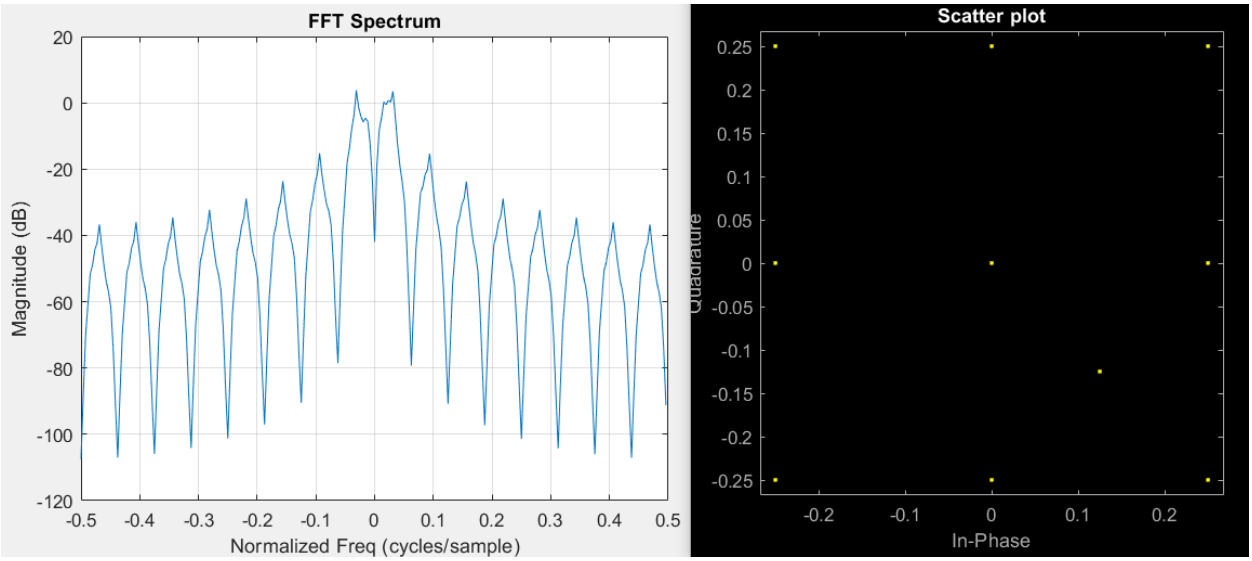
< subcarrier f18 only enabled without CP >



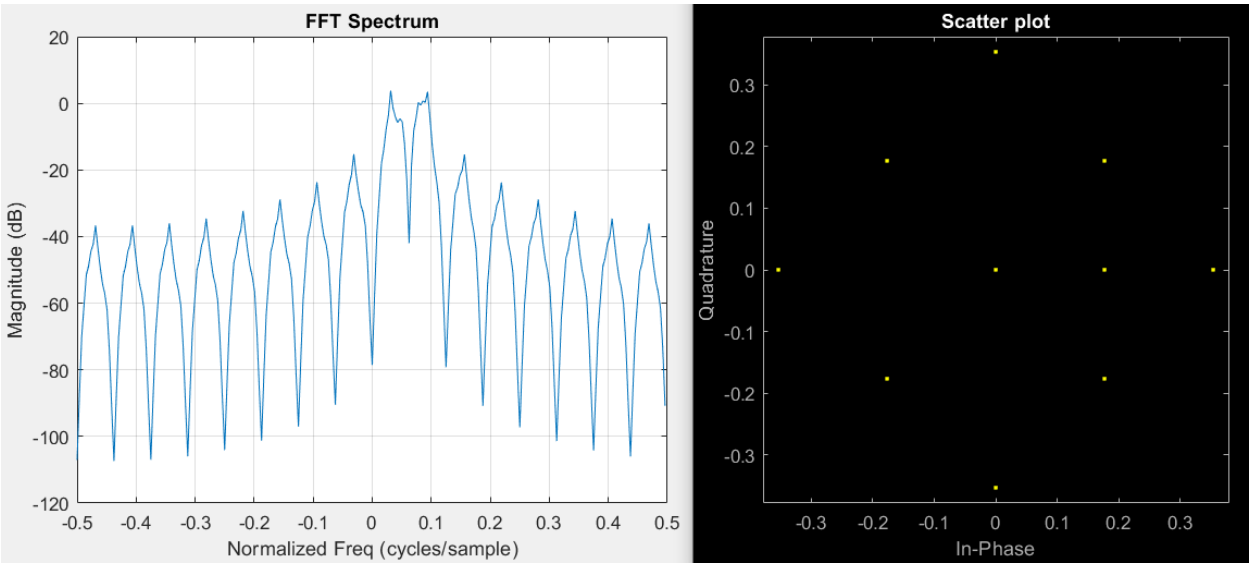
After passing through channel it has more constellation due to multi-fading.

b) Now add cyclic prefix (CP) to the data that is longer than the delay in the channel model. (A Matlab-based simulation of how a cyclic prefix works is here). Repeat part (a) and comment on the result. Now reduce the length of the CP so that it is shorter than the delay spread and repeat part (a). Comment on the result.

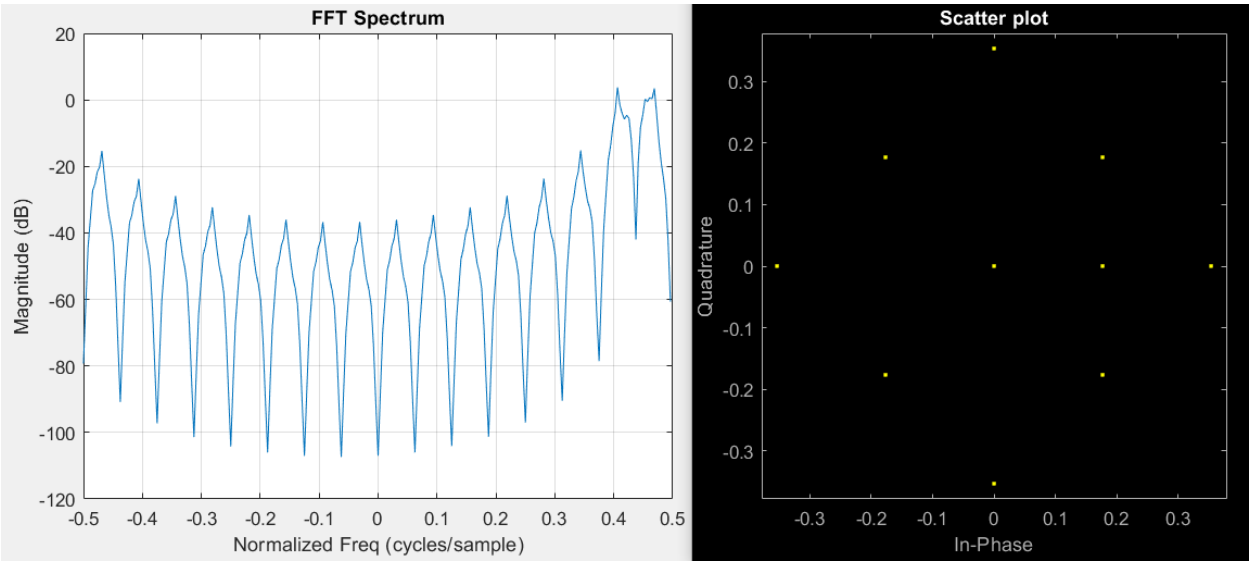
<f0 only enabled with CP=16>



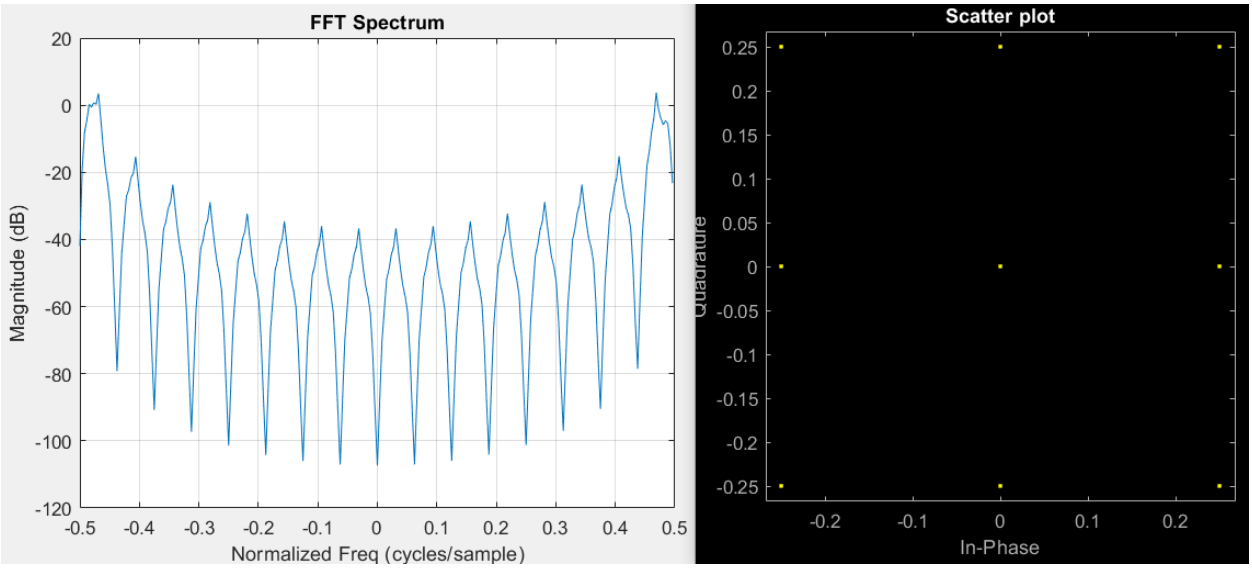
<f2 only enabled with CP=16>



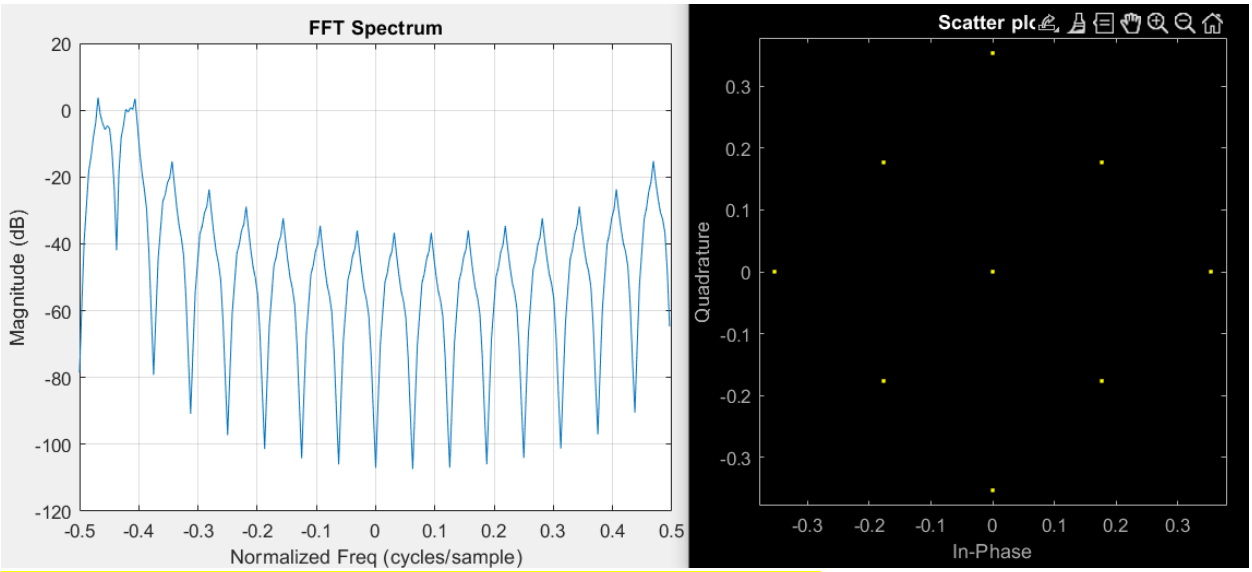
<f14 only enabled with CP=16>



<f16 only enabled with CP=16>



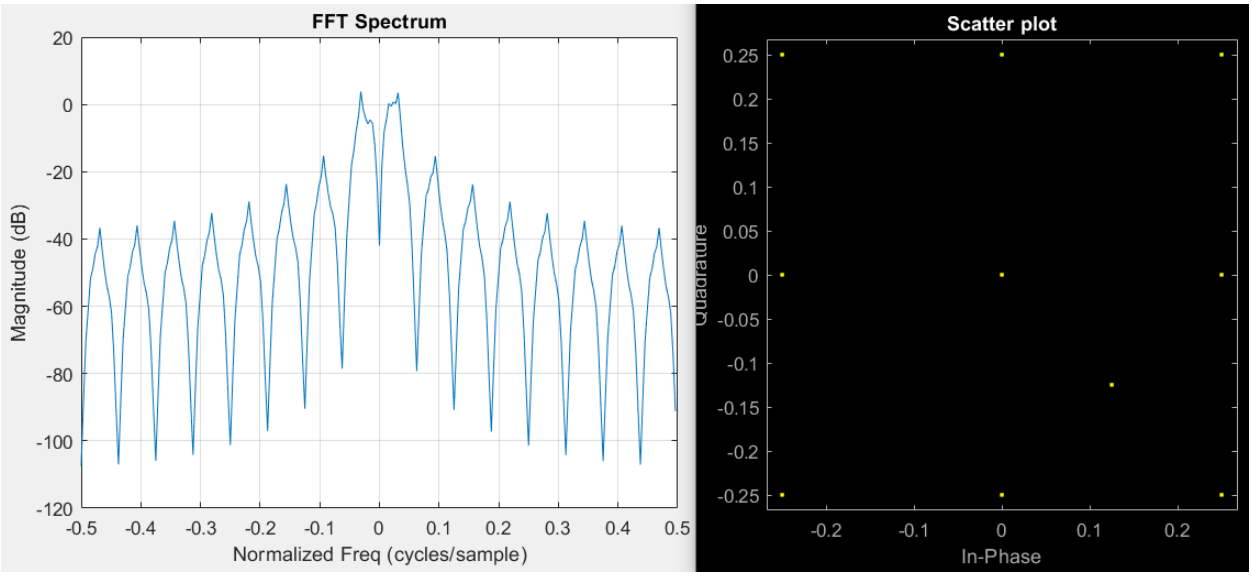
<f18 only enabled with CP=16>



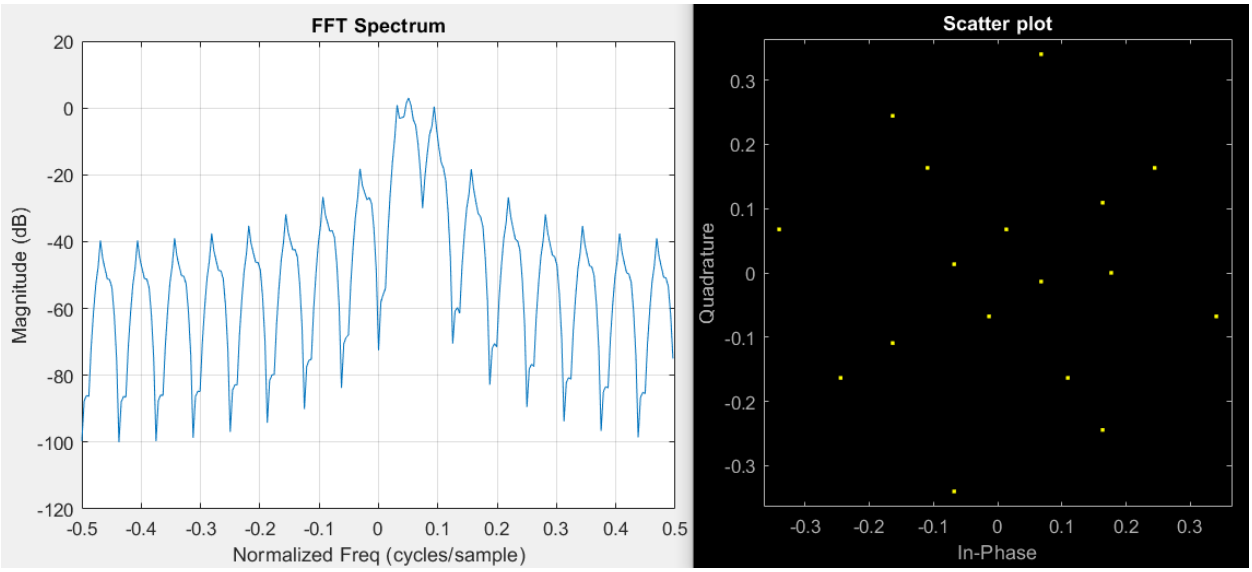
With or without CP doesn`t show difference for the same data point.

It seems without equalization it doesn't have original constellation.

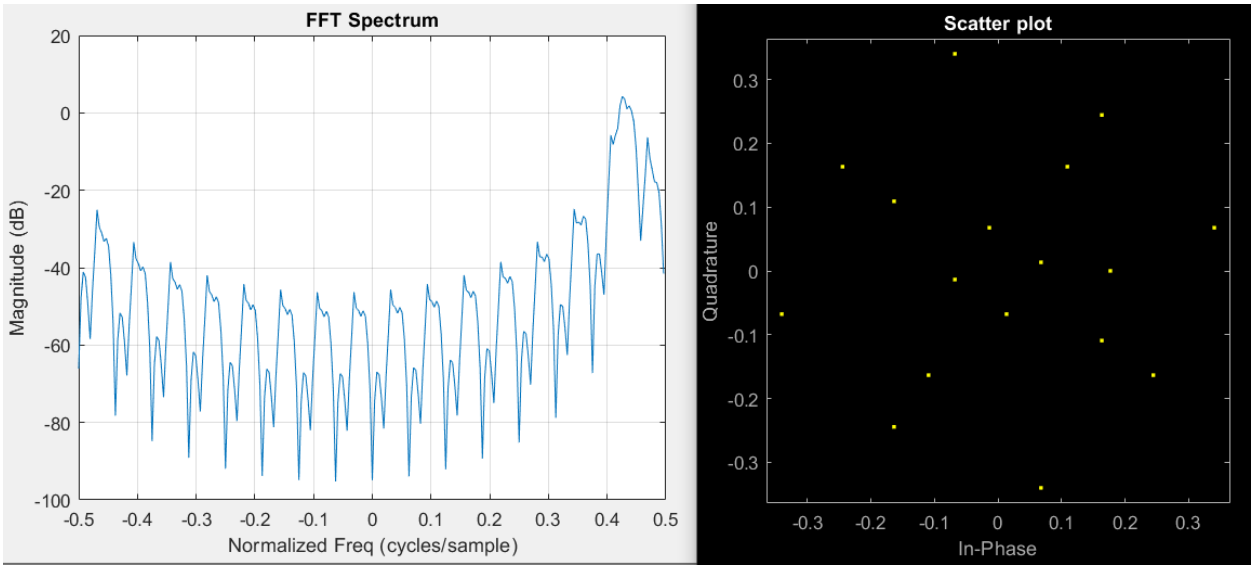
<f0 only enabled with CP=3>



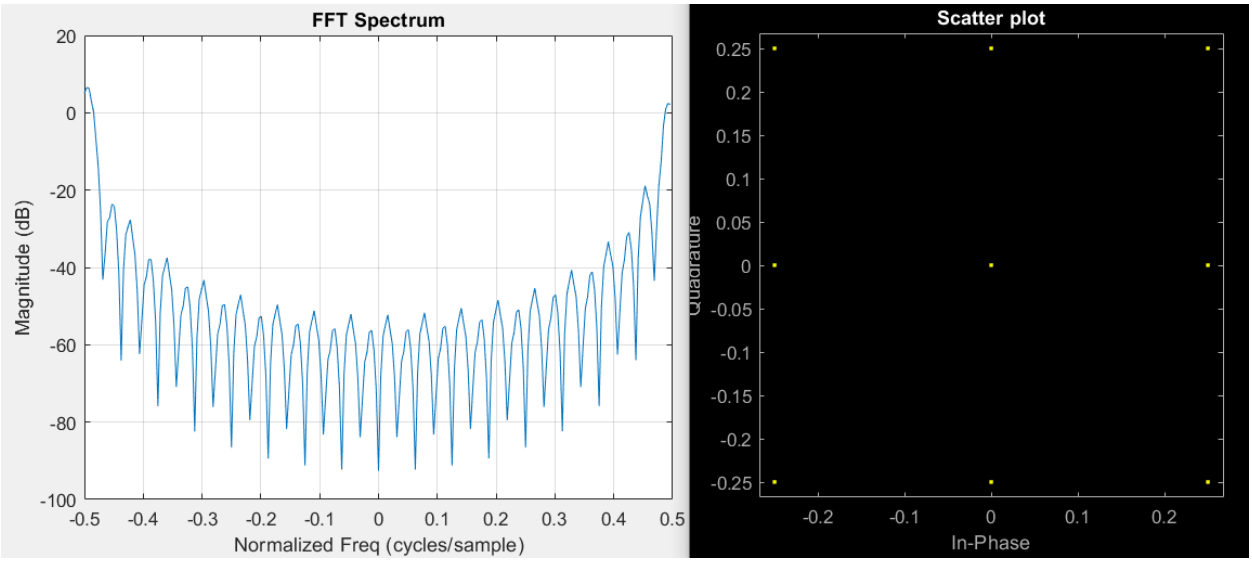
<f2 only enabled with CP=3>



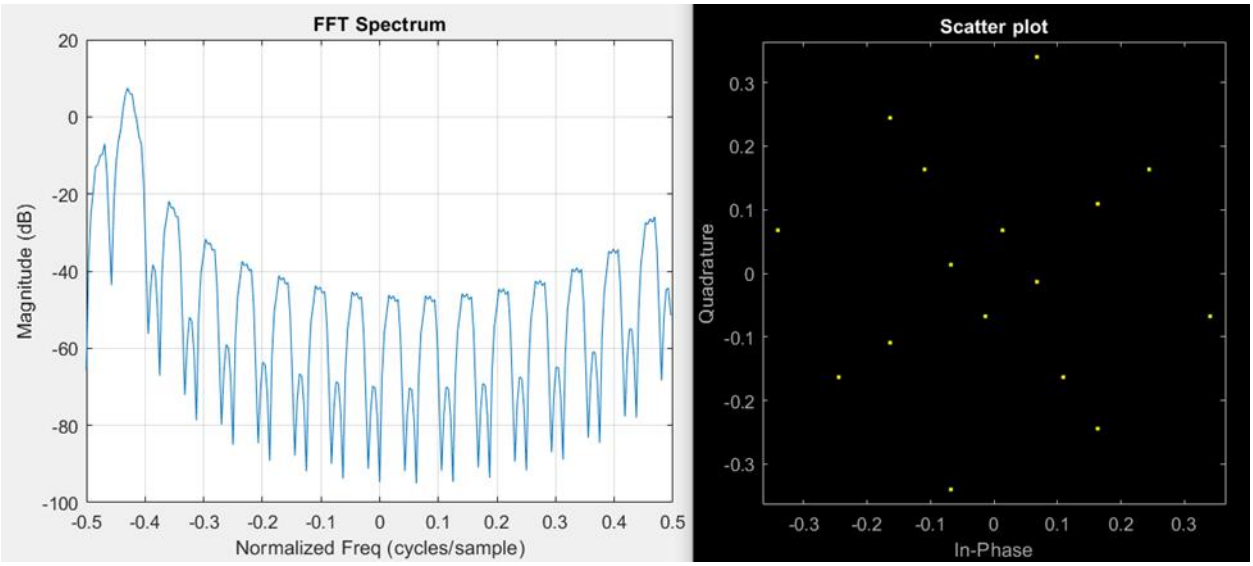
<f14 only enabled with CP=3>



<f16 only enabled with CP=3>



<f18 only enabled with CP=3>



2. *Channel Estimation and Equalization*
This Matlab problem will involve implementing an OFDM receiver. The file "*ofdm.mat*" contains an OFDM packet with a frequency offset of f_0 (which is assumed to be constant throughout the packet), that is corrupted by a frequency selective fade and additive white gaussian noise. The relevant information is as follows:

Parameter	Value
FFT Size	64
Active Subcarriers	$n = [1, 26]$ and $n = [38, 63]$
CP Length	16
Training Sequence Length	128
Training Sequence Format	OFDM BPSK
Number of OFDM Blocks	10

Table 1: Table of OFDM Parameters

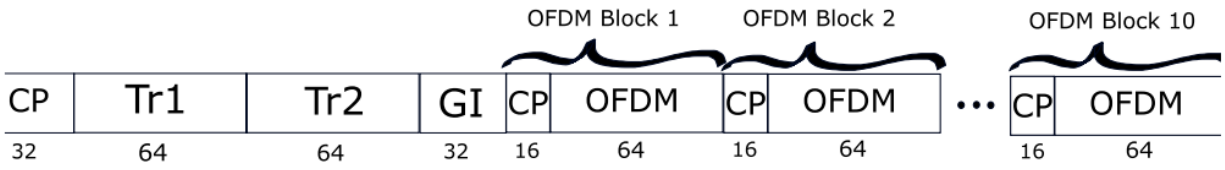


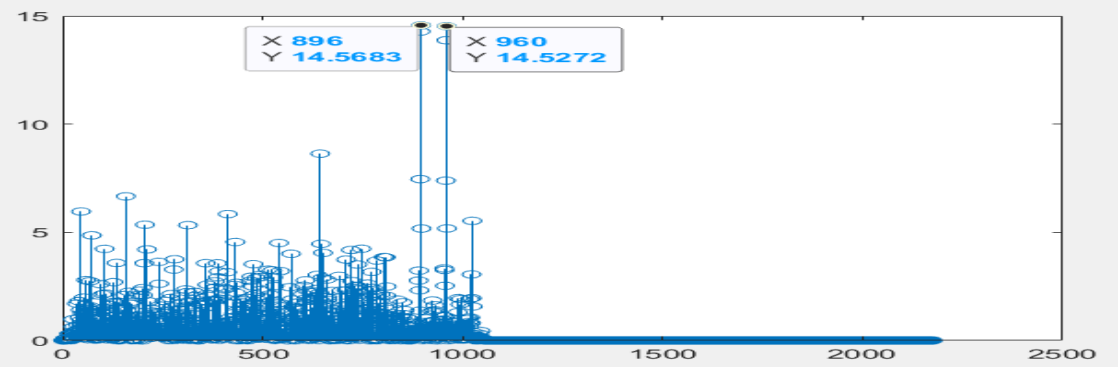
Figure 2: OFDM Packet Format

a) Find the start time of the packet using the cross-correlation, the BPSK training sequence is provided to you as the variable s. Remember, you will have to take s and modulate it into an OFDM symbol.

i. Using the procedure as described in Theory Problem 4, determine the frequency offset f_0. Note that Z_2

$$Z_2 = r^*[N_1]r[N_1 + L_{\text{long}}]$$

calculates only a per sample estimate and that you can significantly improve the performance by averaging over every sample contained within each training sequence Tr_i.



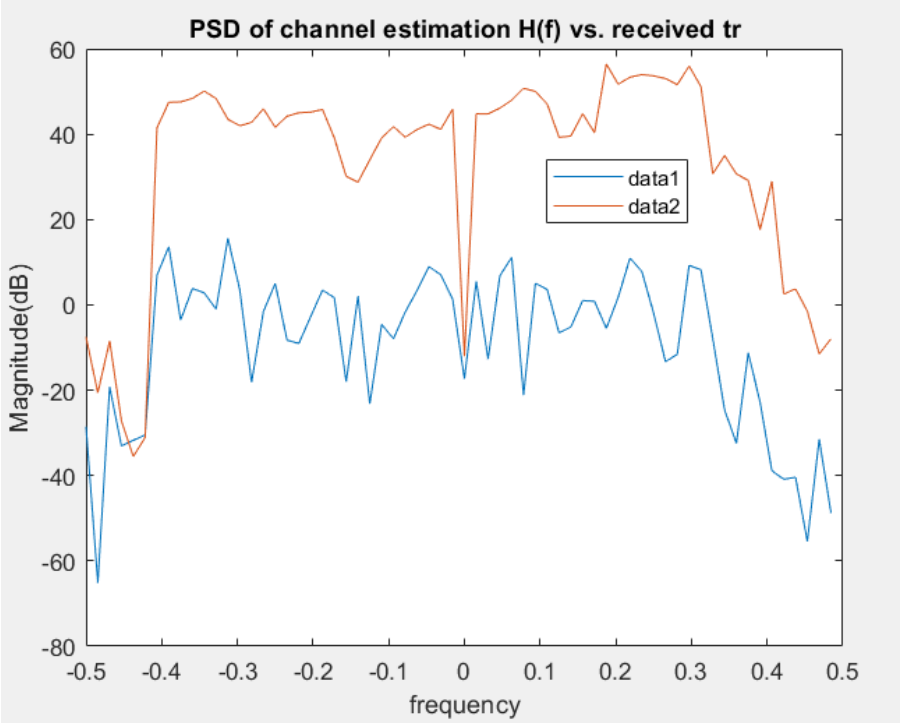
ii. Once you have determined f_0, correct your entire packet by applying a frequency offset of $-f_0$ (i.e. de-rotating your packet).

```
phi = angle(trn_msg_xcorr(960))-angle(trn_msg_xcorr(896))
fo = phi/(2*pi*64)
rxmsg_crr =  rxmsg.y.*exp(-1i*2*pi*(fo))
```

iii. After you have removed the frequency offset, estimate the frequency domain channel coefficients $\widehat{H}$.

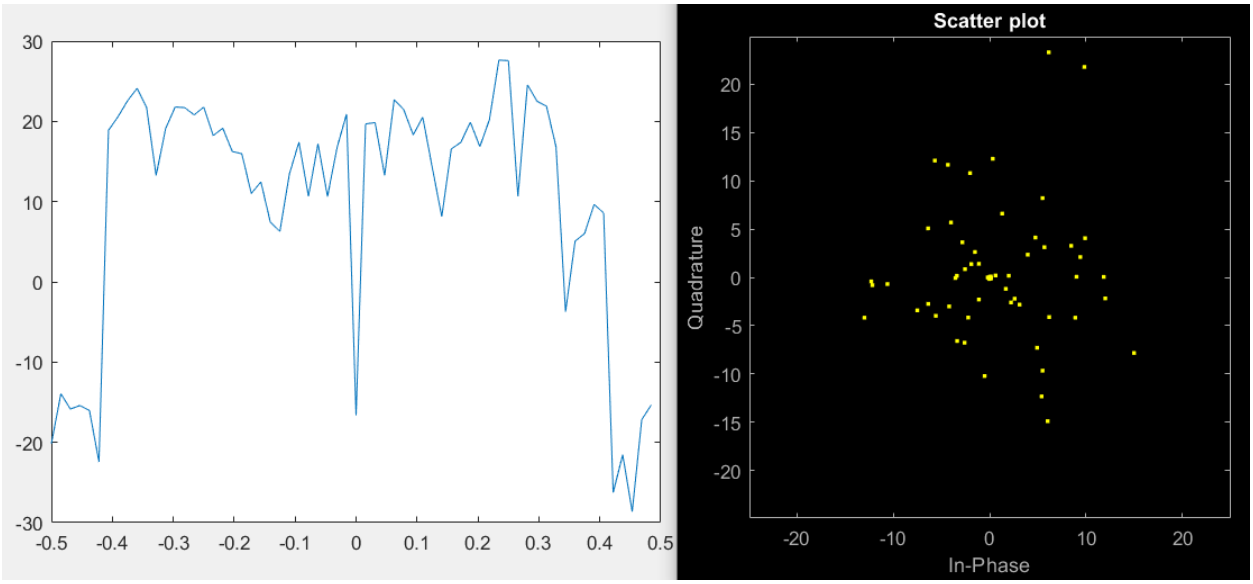
A. Compare the plot of your estimated channel coefficients vs. the actual received signal spectrum.

< red: received training sequence, blue: channel coefficient >

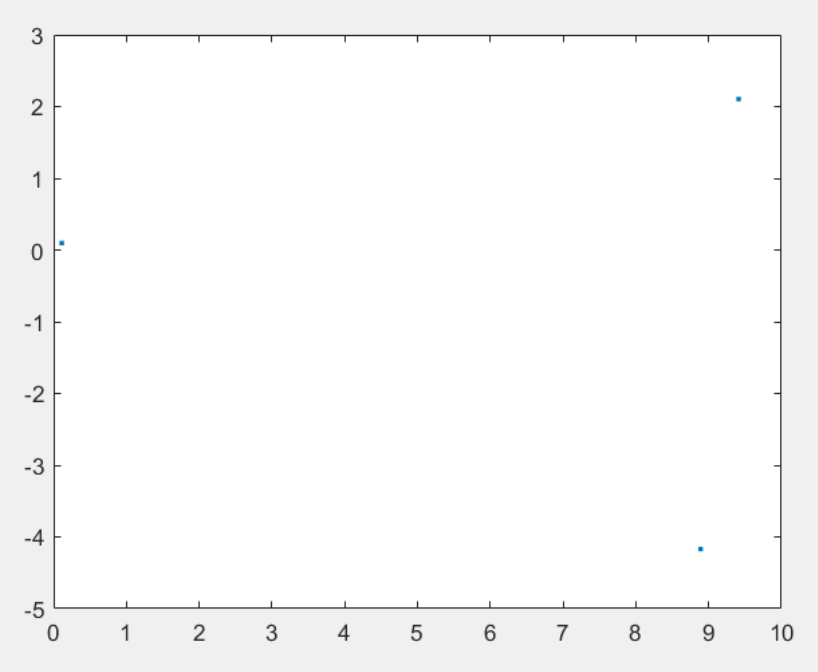


iv. For the first OFDM block find and remove the cyclic prefix (CP) and then demodulate the OFDM block using the FFT.

A. Plot the constellation of all subchannels overlapped on top of each other

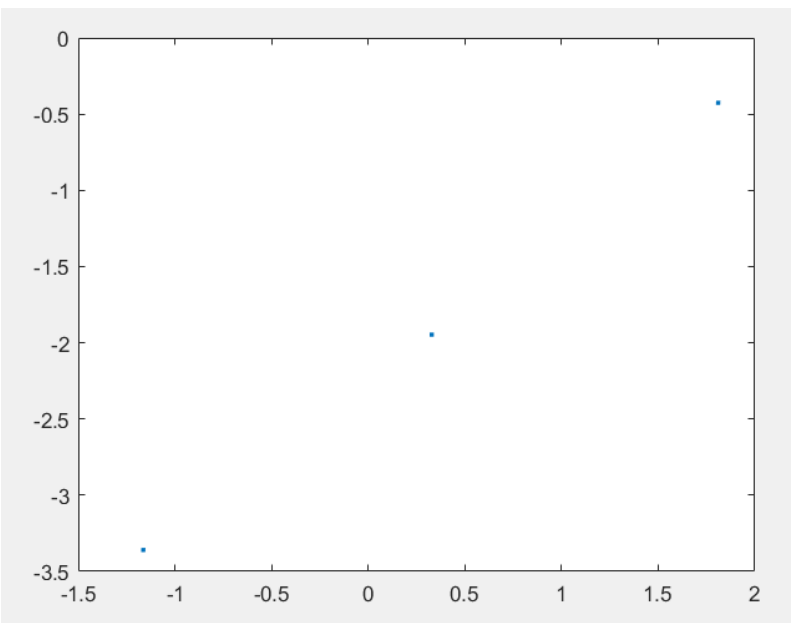


B. On separate figures, plot the constellation for subcarriers $n = 0, 1, 2$.



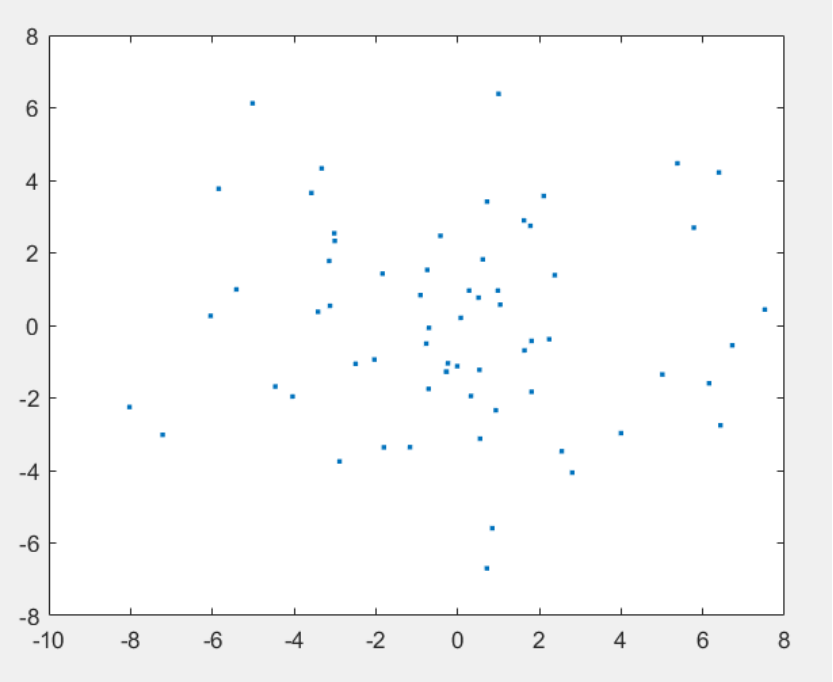
v. For the first OFDM block, apply the frequency domain equalizer $\widehat{G}$.

A. On separate figures, plot the constellation for subcarriers $n = 0, 1,$



vi. Apply the equalizer to the entire packet.

A. Plot the constellation of all subchannels for the entire packe



< MATLAB problem 1 code >

```
FFT = 256;
N =32;
L = 100;
cp_n =3;
n=19;

k_lim = floor( (N - N/4 + 4) / 2 );
%bitmask = [ 1:k_lim (-k_lim:-1) + N ] + 1;
%bitmask = [ 1:32];
bitmask = [ n ];
% Na: number of active data channels
Na = numel(bitmask);

S_mat =zeros(N,L);

rng(0,'twister');
S_mat(bitmask,:) = (randsrc(Na,L)+1i*randsrc(Na,L)) * sqrt(1/2);

s_mat = sqrt(N)*ifft(S_mat); %s_mat is time-domain samples
%s_mat_cp = sqrt(N)*ifft(S_mat_cp);
s_mat_cp = zeros(N+cp_n, L);
s_mat_cp(1:N+cp_n, 1:L ) = [s_mat(1:N, :) ; s_mat(1:cp_n,:)]

s = reshape(s_mat,1,[]);
s_cp = reshape(s_mat_cp,1,[]);

s_fltd = conv(s,[1 zeros(1,15) -1  ]);
s_fltd = s_fltd(1:N*L);
```

```
s_fltd_mat = reshape(s_fltd, [N, L]);
s_fltd_mat_1 = s_fltd_mat(1:N/2, :);

s_cp_fltd = conv(s_cp,[1 zeros(1,15) -1  ]);
s_cp_fltd = s_cp_fltd(1:(N+cp_n)*L);
s_cp_fltd_mat = reshape(s_cp_fltd, [N+cp_n, L]);
s_cp_fltd_mat_1 = s_cp_fltd_mat(1:N/2,:)

s_fltd_16 = reshape(s_fltd_mat_1, 1,[]);
s_cp_fltd_16 = reshape(s_cp_fltd_mat_1,1,[]);

S_fft = pwelch(s,nFFT,nFFT/2,[],1,'twosided');
S_fltd_fft = pwelch(s_fltd_16,nFFT,nFFT/2,[],1,'twosided');
S_cp_fltd_fft = pwelch(s_cp_fltd_16,nFFT,nFFT/2,[],1,'twosided');

F = linspace(-0.5,0.5-1/nFFT,nFFT);

figure(1);
plot(F,db(fftshift(S_fft)));
xlabel('Normalized Freq (cycles/sample)')
ylabel('Magnitude (dB)');
title('FFT Spectrum')
grid on;

figure(2)
plot(F,db(fftshift(S_fltd_fft)));
xlabel('Normalized Freq (cycles/sample)')
ylabel('Magnitude (dB)');
title('FFT Spectrum')
grid on;

figure(3)
plot(F,db(fftshift(S_cp_fltd_fft)));
xlabel('Normalized Freq (cycles/sample)')
ylabel('Magnitude (dB)');
title('FFT Spectrum')
grid on;

scatterplot(s(n:32:end))
scatterplot(s_cp(n:N+cp_n:end))
scatterplot(s_fltd_16(n:N/2:end))
scatterplot(s_cp_fltd_16(n:N/2:end))
```

< code for MALAB problem2>

```
nFFT = 64;
nFFT1 = 256;
rxmsg = load("C:\Users\epae\OneDrive -
Qualcomm\Documents\personal\UCSD\WES268B\ofdm_pkt.mat");

rxmsg_fft = pwelch(rxmsg.y,nFFT,nFFT/2,[],1,'twosided');
F1 = linspace(-0.5,0.5-1/nFFT1,nFFT1);
F = linspace(-0.5,0.5-1/nFFT,nFFT);
```

```
figure(1)
plot(F1,db(fftshift(rxmsg_fft)))

trn = [0 rxmsg.s(1:26) zeros(1,10) rxmsg.s(27:52) 0];
trn_ifft = ifft(trn');
trn_fft = fft(trn_ifft.',nFFT)

figure(2)
plot(F,db(fftshift(trn_fft)))


[ trn_msg_xcorr ] = c_corr( rxmsg.y, trn_ifft.');

figure(3)
stem(abs(trn_msg_xcorr).^2)

phi = angle(trn_msg_xcorr(193))-angle(trn_msg_xcorr(129))
fo = phi/(2*pi*64)
fo_vec = exp(-1i*2*pi*(fo*(1:1088-96)));
rxmsg_crr =  rxmsg.y(97:1088).*fo_vec;

figure(4)
rxmsg_crr_fft = pwelch(rxmsg_crr,nFFT,nFFT/2,[],1,'twosided');
%rxmsg_crr_fft = fft(rxmsg_crr,nFFT)/sum(rxmsg_crr);
plot(F1,db(fftshift(rxmsg_crr_fft)))
%plot(F,db(fftshift(rxmsg_crr_fft)))
hold on
plot(F1,db(fftshift(rxmsg_fft)))
%plot(F,db(fftshift(rxmsg_fft)))

figure(5)
%rxmsg_crr_tr_fft = pwelch(rxmsg_crr(129-32:129+32-1),nFFT,nFFT/2,[],1,'twosided');
rxmsg_crr_tr_fft = fft(rxmsg_crr(33:96),nFFT);
%plot(F1,db(fftshift(trn_fft)))
plot(F,db(fftshift(trn_fft)))
hold on
%plot(F1, db(fftshift(rxmsg_crr_tr_fft)))
plot(F, db(fftshift(rxmsg_crr_tr_fft)))

%%=============================================================
%%< T matrix >
T = zeros(96,32);
%trn = [0 rxmsg.s(1:26) zeros(1,10) rxmsg.s(27:52) 0];
trn_96 = [trn(33:1:64) trn(1:1:64)];
T= toeplitz(trn_96,[trn_96(1) zeros(1,31)]);
%%=================================================================
%%=================================================================
% <h_ex>
tr_crr= rxmsg_crr(1:96);
stem(tr_crr)
% stem(trn_96)
% stem(rxmsg_crr)
h_ex = pinv(T)*tr_crr.'
H_ex = fft(h_ex,nFFT);
figure(3)
```

```
subplot(211)
stem(h_ex);
title('impulse response of h_ex(t)')
ylabel('Magnitude(dB)')
xlabel('frequency')
subplot(212)
plot(-0.5:1/nFFT:0.5-1/nFFT,fftshift(20*log(abs(H_ex))))
title('PSD of channel estimation H(f)')
ylabel('Magnitude(dB)')
xlabel('frequency')
%%=====================================================================
% <check h_ex>
trn_96_check = T*h_ex;
figure(4)
subplot(211)
stem(trn_96)
title('original message preamble')
subplot(212)
stem(trn_96_check)
title('preambgle recovered by H(f)')
%%============================================================
%%============================================================
%%< R matrix >
tr_crr= rxmsg_crr(1:96);
R= toeplitz(tr_crr,[tr_crr(1) zeros(1,31)]);
%%=============================================================
% <g_ex>
%trn_96 = [trn(33:1:64) trn(1:1:64)];
g_ex = pinv(R)*trn_96.';
G_ex = fft(g_ex,nFFT);
figure(5)
subplot(211)
stem(g_ex)
title('Impuse response of equalizer g_ex(t)')
ylabel('Magnitude(linear)')
xlabel('time')
subplot(212)
plot(-0.5:1/nFFT:0.5-1/nFFT,fftshift(20*log(abs(G_ex))))
title('PSD of equalizer G(f)')
ylabel('Magnitude(dB)')
xlabel('frequency')
%%====================q=========================================
% < h_ex*g_ex >
eq_check = conv(h_ex,g_ex);
EQ_check = fft(eq_check,nFFT);
figure(6)
subplot(211)
stem(eq_check)
title('g_ex(t)(*)h_ex(t)')
ylabel('Magnitude(dB)')
xlabel('time')
subplot(212)
plot(-0.5:1/nFFT:0.5-1/nFFT,fftshift(10*log(abs(EQ_check))))
title('G_ex(f)*H_ex(f)')
ylabel('Magnitude(dB)')
```

```
xlabel('frequency')
%%====================q=========================================
% < g_ex(*)msg.y >
rxmsg_eql = conv(g_ex,rxmsg_crr);
p_rxmsg_eql = rxmsg_eql.^2
p_rxmsg_eql_fft = fft(p_rxmsg_eql,nFFT);
figure(7)
subplot(211)
plot(-0.5:1/nFFT:0.5-1/nFFT,fftshift(10*log(abs(p_rxmsg_eql_fft))))
title('PSD of equalized message by g_ex(t)(*)msg(t)')
ylabel('Magnitude(dB)')
xlabel('normalized frequency(fc/fs)')
subplot(212)
plot(-0.5:1/nFFT:0.5-1/nFFT,fftshift(10*log(abs(EQ_check))))
title('Eaulizer check by G_ex(f)*H_ex(f)')
ylabel('Magnitude(dB)')
xlabel('frequency')

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


figure(8)
ofdm1 = rxmsg_crr(32+64+64+32+16+1:32+64+64+32+16+64);
rxmsg_crr_ofdm1_fft = fft(ofdm1,nFFT);
plot(F, db(fftshift(rxmsg_crr_ofdm1_fft)))

figure(9)
plot(real(rxmsg_crr_ofdm1_fft(1:3)), imag(rxmsg_crr_ofdm1_fft(1:3)), '.')

figure(10)
rxmsg_eql_fft = fft(rxmsg_eql, nFFT);
plot(real(rxmsg_eql_fft(1:3)), imag(rxmsg_eql_fft(1:3)), '.')

figure(11)
plot(real(rxmsg_eql_fft(1:64)), imag(rxmsg_eql_fft(1:64)), '.')
```