A picture containing text

Description automatically generated

= \*Fs = 0.8us\*20MHz = 16samples/short sequence

= \*Fs = 3.2us\*20MHz = 64samples/long sequence

Text

Description automatically generated

Graphical user interface, text

Description automatically generated

r[n1] = x[n1], r[n1+] = x[n1+]

Z1 = [n1]r[n1 + ]=

r[N1] = x[N1], r[N1+] = x[N1+]

Z2 = [N1]r[N1 + ]= x[N1+]

x[N1+] =

Text

Description automatically generated with low confidence

=

=

Text

Description automatically generated with low confidence

=

=

Text

Description automatically generated

Max phase can accurately estimate is

So, max < 1/

max < 1/ for long sequence.

Text

Description automatically generated

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:

=

The CBER = 1.657e-10 when plug in pb=10^-4



In Case II, the probability of k subchannels having errors

=

Where Pk = C(7,k) \* \* (1-\* (1-

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

p1 = 0.0004205

p1' = 0.4783

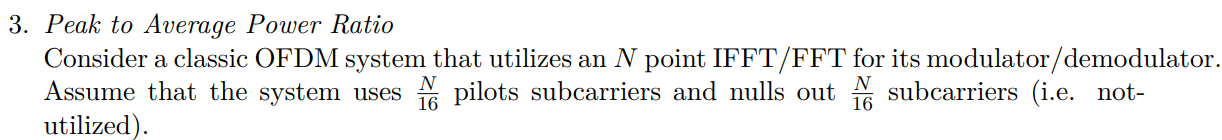
Pk for k=2 to 7 are: [3.027e-10, 2.019e-8, 5.332e-7, 5.892e-6, 3.499e-5, 0.001278]

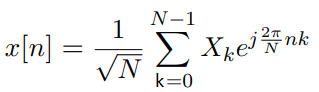
The CBER = 0.4798



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.





Graphical user interface, application, Word

Description automatically generated

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]| = A), A: the amplitude of each QPSK symbol.

max(|x[n) = 2\*

E(|x[n) = ) \* SUM(|X[n]),

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

the average power of each symbol(QPSK) =

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

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

E(|x[n]) = (1/N) \* SUM(|X[k]) =

PAPR = |x[n]/ E(|x[n]) = (2\*= 15N/8

Text

Description automatically generated

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

i.e. Pr{PAPR(x[n]) 10 } = (1 - e0.99

(1 - e0.99

If take the natural logarithm of both sides:

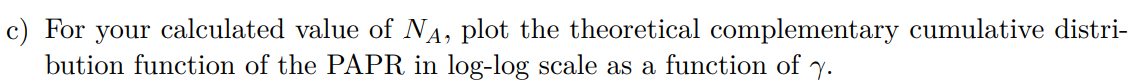
ln (1 - e ln(0.99)

If solve for NA:

ln(0.99)/ ln (1 – e as ln (1 – e

< 221.368354

Max(



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');

Chart

Description automatically generated

Text

Description automatically generated

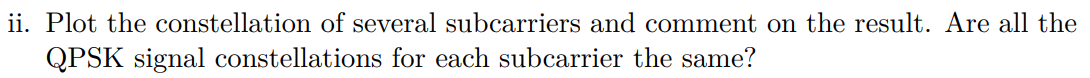
< without fading > < with fading >

Chart, histogram

Description automatically generated

Chart, histogram

Description automatically generated



< subcarrier f0 only enabled without CP> Chart

Description automatically generated

< subcarrier f2 only enabled without CP >

Chart

Description automatically generated

< subcarrier f14 only enabled without CP >

Chart

Description automatically generated

< subcarrier f16 only enabled without CP >

Chart, scatter chart

Description automatically generated

< subcarrier f18 only enabled without CP >

Chart

Description automatically generated

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

Text

Description automatically generated

<f0 only enabled with CP=16>

Chart, scatter chart

Description automatically generated

<f2 only enabled with CP=16>

Chart

Description automatically generated

<f14 only enabled with CP=16>

Chart, scatter chart

Description automatically generated

<f16 only enabled with CP=16>

Chart, scatter chart

Description automatically generated

<f18 only enabled with CP=16>

Chart, scatter chart

Description automatically generatedWith 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>

Chart

Description automatically generated

<f2 only enabled with CP=3>

Graphical user interface, chart

Description automatically generated

<f14 only enabled with CP=3>

Chart

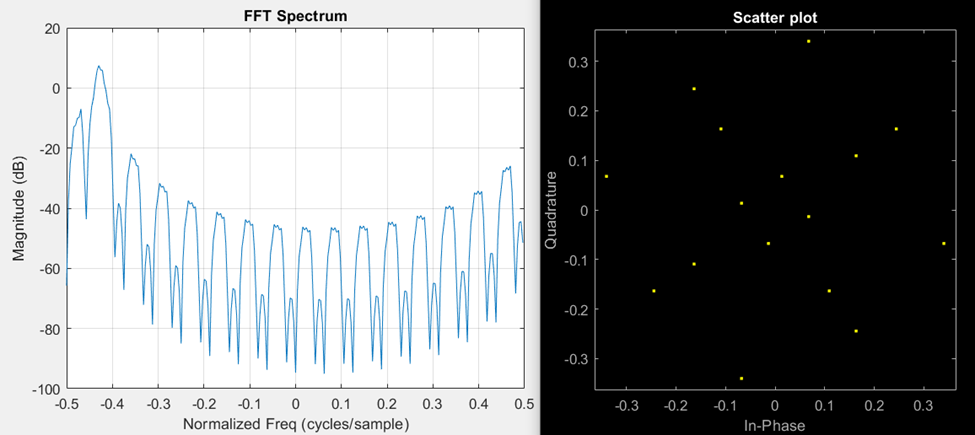
Description automatically generated

<f16 only enabled with CP=3>

Graphical user interface, chart

Description automatically generated

<f18 only enabled with CP=3>



Table

Description automatically generated

Table

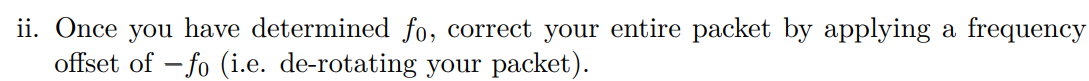
Description automatically generated with medium confidence

Text

Description automatically generated

Chart

Description automatically generated with medium confidence



phi = angle(trn\_msg\_xcorr(193))-angle(trn\_msg\_xcorr(129))

fo = phi/(2\*pi\*64)

rxmsg\_crr = rxmsg.y.\*exp(-1i\*2\*pi\*(fo))

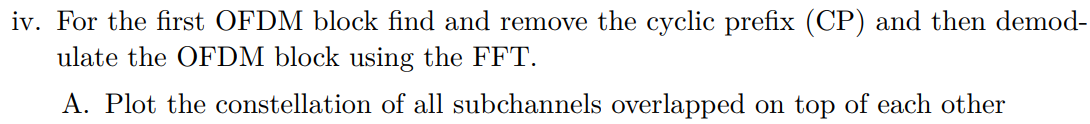
Text

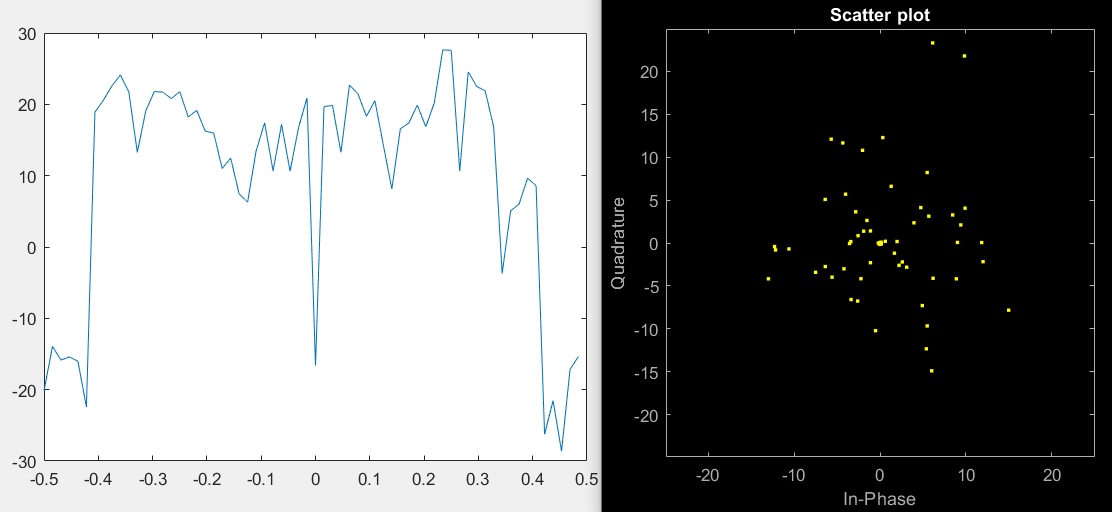
Description automatically generated with medium confidence

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

Chart

Description automatically generated

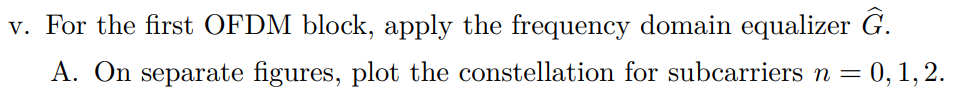
****





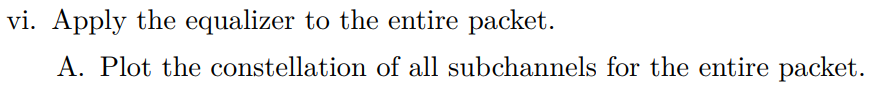
Chart

Description automatically generated



Chart, scatter chart

Description automatically generated



Chart, scatter chart

Description automatically generated

< 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)), '.')