SSY-135 Project part 2

Group E: Oskar Claeson Erik Tilly Yuling Zhang Haitham Babbili

I. Introduction

The objective of this report is to increase the understanding of wireless transmission systems and in particular orthogonal frequency division multiplexing (OFDM) systems. This is done through designing and simulating an OFDM communication system in Matlab. A study of how the system behaves over two different channels is made. The two channels that were simulated are an additive white Gaussian noise (AWGN) channel and a time-varying frequency selective channel known as Rayleigh fading. This paper illustrates the method of designing such an OFDM system and evaluates the performance of said system. The evaluation of the system includes different plots and scatter-plots to provide a ground to discuss topics such as Symbol Error Rate (SER), Signal to Noise Ratio (SNR), the effects of cyclic prefix and more.

II. OFDM communication design and evaluation

OFDM technology has been introduced in most of the wireless network systems due to efficiency in the frequency spectrum. The technology is based on dividing the mainstream transmitted signals into many sub-stream signals carried over multi-subchannels. This achieves a higher data rate and introduces higher importance in eliminating the inter symbol interference(ISI), which is why cyclic prefix is needed in the system. As modulation QPSK modulation was implemented before creating OFDM symbols. Which is done by Creating vectors of length N containing QPSK symbols and the implement cyclic prefix to them and after that convert it to the time domain by using Inverse fast Fourier transform (IFFT) which is the computational version of inverse discrete Fourier transform (IDFT) that you can see in equation 1.

$$x(n) = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} X(k) \exp \frac{j2\pi kn}{N} \tag{1}$$

where x(n) is the n-time signal value, X(k) is the spectral value, n is the time index, k is frequencies index and N length of the sequence. The value of $N = 2^p$ where $p = \log_2(N)$ and N is even number.

A. Determining the parameters

The amount of taps the channel used is set such that it covers all paths. In this project, the delay spread is $5.4\mu s$ and sample time $T_s = \frac{1}{bandwidth} = 1\mu s$. Therefore the amount of channel taps L is calculated according to equation 2 to be 6:

$$L \geq \frac{\tau_{DS}}{T_s} \tag{2}$$

In order to trick the receiver, the data is periodic a set of the last symbols which are added to the start of the data sequence. This set of symbols are called cyclic prefix. To avoid Inter-Symbol Interference (ISI), the length of the cyclic prefix, denoted N_{cp}, need to be long enough to cover the total delay spread of the channel. If it is too short, then samples from a previous OFDM symbol may interfere with the newly transmitted samples. The relation between the number of channel taps and cyclic prefix length is shown below in equation 3.

$$N_{cp} \geq L - 1 \tag{3}$$

For a lower complexity of the DFT, N should be power of two. During an OFDM symbol transmission over the channel, which should be close to constant to make the equalization at the receiver more accurate. Therefore, this sets a constraint on the sample length N. Theoretically, the constraints are set according to equation 4, which means that the total length of the OFDM symbol is much shorter than the coherence time of the channel. If assuming that a factor of 10 corresponds to much smaller, then the limiting value of $N + N_{cp}$ can be calculated according to equation 5 which in this case equals to approximately 1000. This means that the theoretical maximum N as a power of 2 is $N = 512$. However, in the simulations, we noticed that the system could handle larger N without error occurring but to avoid long simulation times and not underestimated the channel, the length N was set to 128.

$$(N + Ncp) f_D T_s \ll 1 \tag{4}$$

$$N + Ncp = \frac{0.1}{f_D T_s} \tag{5}$$

The noise power spectrum density N_0 given for the project had the unit $[W/Hz]$. To make sure it has the correct unit for power, W, it is calculated according equation 6. Where the bandwidth is given as two-sided.

$$N_0 = 2 \times 2.07 \times 10^{-20} \times \frac{bandwidth}{2} \quad [W] \tag{6}$$

In the transmitter part of the simulation, the symbol modulation was chosen to be a QPSK modulation as mentioned before. These symbols have an energy constraint $\mathbf{E}\{|s_k^m|^2\} = E$. Where the value of E can be related to the average transmit power times the symbol time according to equation 7. As an example: if $P = 0.1W$ and $T_s = 10^-6s$, then E is calculated as $E = 10^{-7}J$

$$E = P \cdot T_s \tag{7}$$

B. OFDM system design

In the transmitter a OFDM symbol are created by building a vector with QPSK symbols with the length N and apply the IFFT operation to the vector to get it in the time domain. Then they are multiplied with $\sqrt{N}$ to normalize it according to equation 8.

$$x[i] = \sqrt{N} \cdot \mathrm{IFFT(s)} \tag{8}$$

After that, the cyclic prefix is implemented by copying the N_{cp} first QPSK symbols and place a copy of them at the end of the vector. This makes it possible to manage it as a periodic signal, however this also increases the length of the vector to $N + N_{cp}$. Then this vector is sent through the channel and this process is repeated M times.

When the signal arrives at the receiver, the first step is to remove the cyclic prefix. This is achieved by removing the N_{cp} last QPSK symbols that has been received. The channel affects the signal both with rotation and amplitude. This is taken into consideration by taking an estimation of the channel and then remove it in the frequency domain. After that the signal is mapped with maximum likelihood. Since the QPSK symbols are random generated, they are equiprobable, hence, they are mapped to the symbol with the minimum distance.

III. Simulation over different channels

A. AWGN channel

Firstly, the transmitter is created as a simple QPSK modulated sample sequence of N samples. These are then inversely discrete Fourier transformed to time domain and the cyclic prefix is added. The OFDM symbol is then put through the channel. The AWGN channel is described as $c_0(nT_s) = 1$ and $c_{l \neq 0}(nT_s) = 0$, $\forall n$. To simulate this in Matlab, the received signal is not equalized as this channel, c, does not introduce rotation nor scaling. However, the effects of the path loss is multiplied and random samples from a complex Gaussian distribution is added to the transmitted signal. This do introduce some scaling and can be seen in the scatter plots corresponding to the AWGN channel simulation shown in Appendix A. The receiver then simply checks which quadrant the received symbols are placed in the IQ-plane and chooses the QPSK symbol corresponding to that quadrant.

B. Time-varying frequency selective channel

The transmitter works the same way as for the AWGN channel simulation. In this project, the provided function Fading_channel.p is used to generate the time-varying frequency selective channel $c_l(nT_s)$. This channel distorts the amplitude and phase of the signal. When running the signal through the channel function, an estimation of the channel is also provided. When the OFDM symbol time is much shorter than the coherence time of the channel, then the first row of this channel estimation can be seen as the time domain response of the fading channel. After the signal has gone through the channel, in the same way as for AWGN channel simulation, the path loss effect and noise is added. The received signal is discrete Fourier transformed back to frequency domain to obtain the samples of the form $y_n = C_n \cdot s_n + w_n$, where C is related to the FFT of the time domain response of the channel. In order to remove the effects of the channel, the received samples are equalized, which is done by dividing y with C. These equalized samples are shown in scatter plots located in Appendix B. To recover the symbols correctly, the ML estimate boils down to minimizing the distance between the received symbols and the QPSK constellation points. The effective data rate of the system over the OFDM system is calculated to be approximately 3.85 Gbit/s according to following equation, where $bits\_per\_symbol = 2$ for QPSK:

$$r = (N/(N + N_{cp})) \cdot f_c \cdot bits\_per\_symbol \tag{9}$$

C. Energy vs SER

To demonstrate the relation between the SNR and SER of the system, a SER vs SNR plot for the two channels were created. These are shown in figures 1 and 2. The simulation is a verification of how the SER corresponds to specific SNRs when using the Fading channel and AWGN channel. The simulated values are also compared to theoretical values of AWGN SER, which is created according to equation 10.

$$SER = 2 \cdot Q\left(\sqrt{\frac{E(i) \cdot pathloss}{N_0}}\right) \tag{10}$$

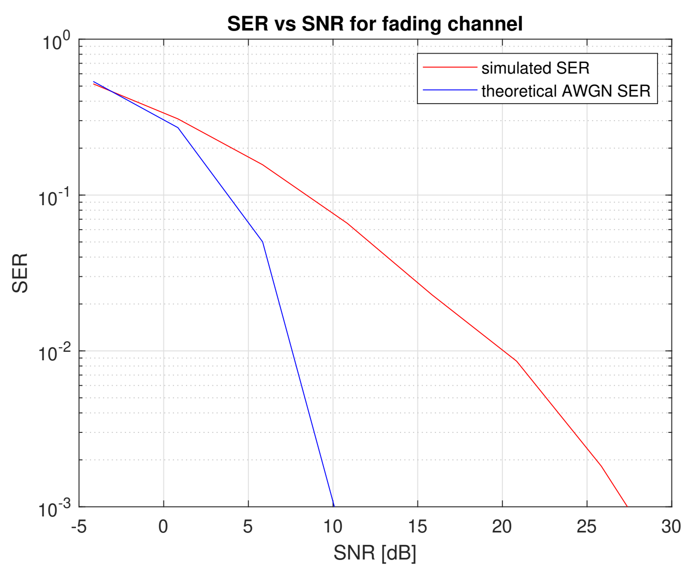

Fig. 1. Plot of simulated SER vs SNR over the Rayleigh fading channel compared with theoretical values for AWGN channel

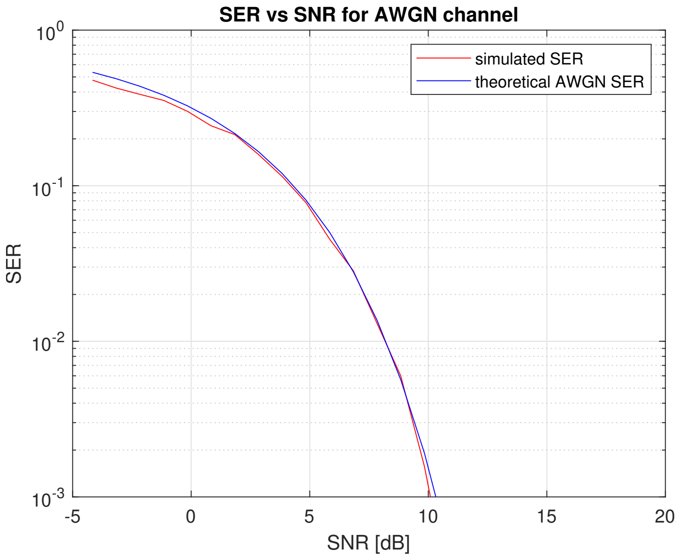

Fig. 2. Plot of simulated and theoretical SER vs SNR over the AWGN channel

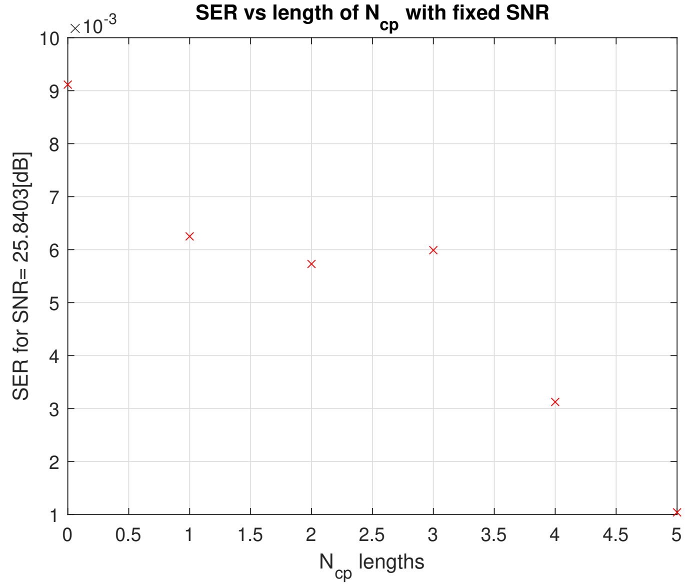

Fig. 3. SER with different value of N_{cp}

As can be seen in figures 1 and 2 the worst value of the SER is about 0,5, which corresponds to the same probability that a random guess would be for the QPSK. Something else that can be noticed in figure 2 and 1 is that the transmission over the AWGN channel performs better than the Fading channel and for some reason even better than the theoretical one. This is expected since the theoretical values corresponds to an ideal case, and the AWGN channel is the same as the fading channel but without the effects of shadowing and multipath.

D. Analysis of cyclic prefix

The cyclic prefix is generated to prevent ISI and Inter Carrier Interference (ICI). At the same time, keep the orthogonality between sub-carriers. The length of cyclic prefix (N_{cp}) will effect the transmission rate and its performance.

The effect of the cyclic prefix can be studied in Appendix C where a comparison between figure 14, that is a scatter plot when no cyclic prefix is used, compared to the figure 19, were the it is long enough. Then it can be observed that the cyclic prefix delivers a tighter grouping round the constellation point compared to figure 14.

In figure 3 the correlation between SER and N_{cp} for a specific SNR is illustrated. Here it is even more clear that a to short prefix will in the end generate errors.

A longer cyclic prefix will make the communication system more robust against larger delay spreads of the channel. However, the transmission rate will suffer for it, since the length of the OFDM symbol is fixed.

If the value of N_{cp} is too small, the previous OFDM symbol could leak into the current OFDM symbol which means that it does not eliminate interference anymore. This in turn is only waste of space, since it does not improve the transmission and still takes place from the actual information that is transmitted.

IV. Members contribution

The way this part of the project was done was once again have all members try to figure out on their own how the MATLAB simulation part is to be implemented and then together discuss the solutions. In the end, the MATLAB code was mainly done by Oskar and Yuling. The text were mainly written by Erik, Yuling and Haitham. Oskar fixed most of the languages and all figures and some erroneous texts.

APPENDIX A

Here are the scatter plots of received symbols over AWGN channel with specific subcarriers. They are plotted with fixed length of cyclic prefix $N_{cp} = 5$ and a fixed $SNR = 25.84$ dB.

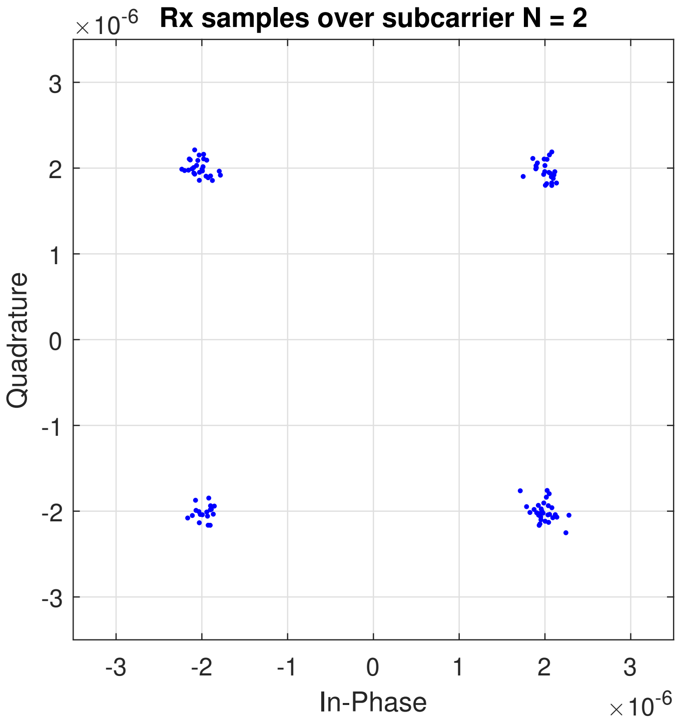

Fig. 4. Scatterplot of samples transmitted over subcarrier #2

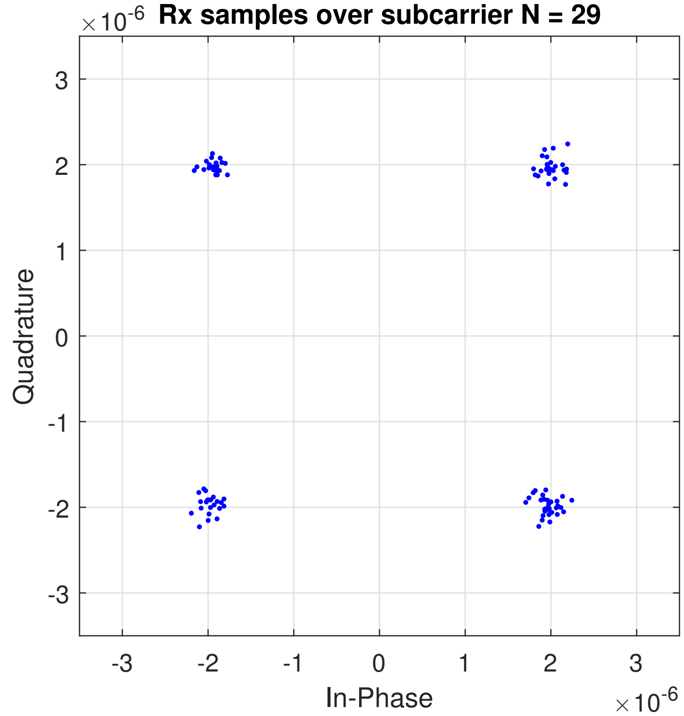

Fig. 6. Scatterplot of samples transmitted over subcarrier #29

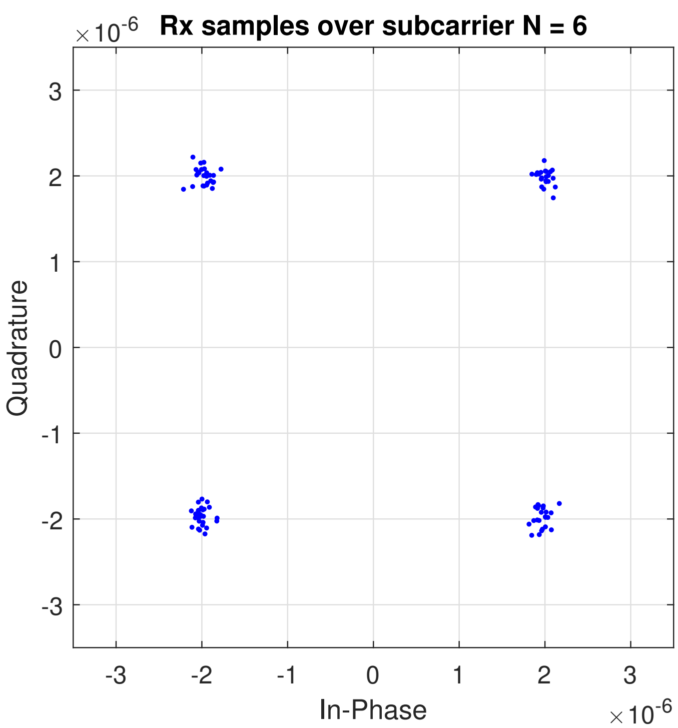

Fig. 5. Scatterplot of samples transmitted over subcarrier #6

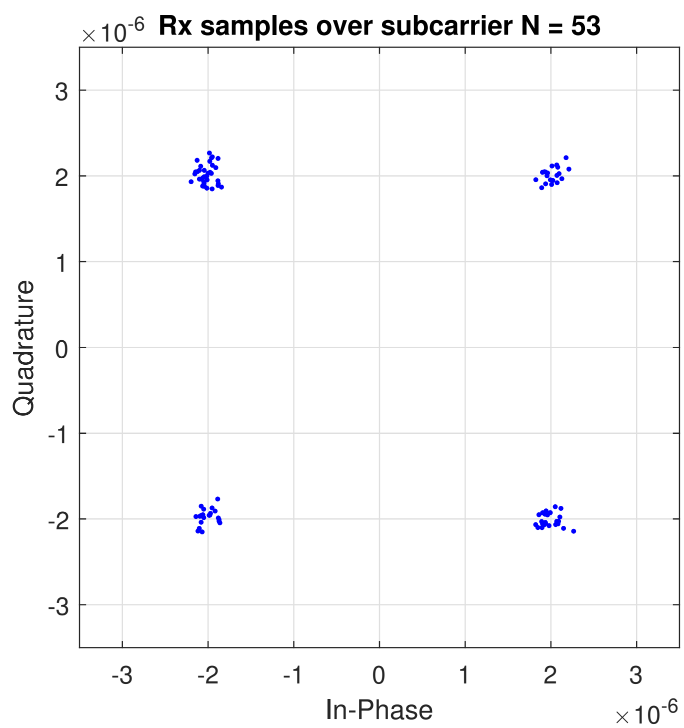

Fig. 7. Scatterplot of samples transmitted over subcarrier #53

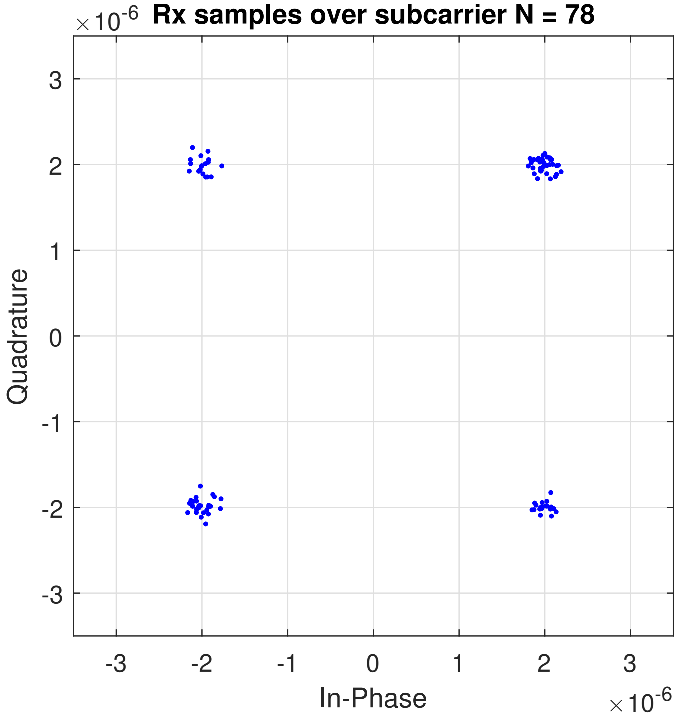

Fig. 8. Scatterplot of samples transmitted over subcarrier #78

APPENDIX B

These scatter plots shows received QPSK symbols over fading channel with specific subcarriers. They are plotted with $N_{cp} = 5$ and a fixed $SNR = 25.84$ dB.

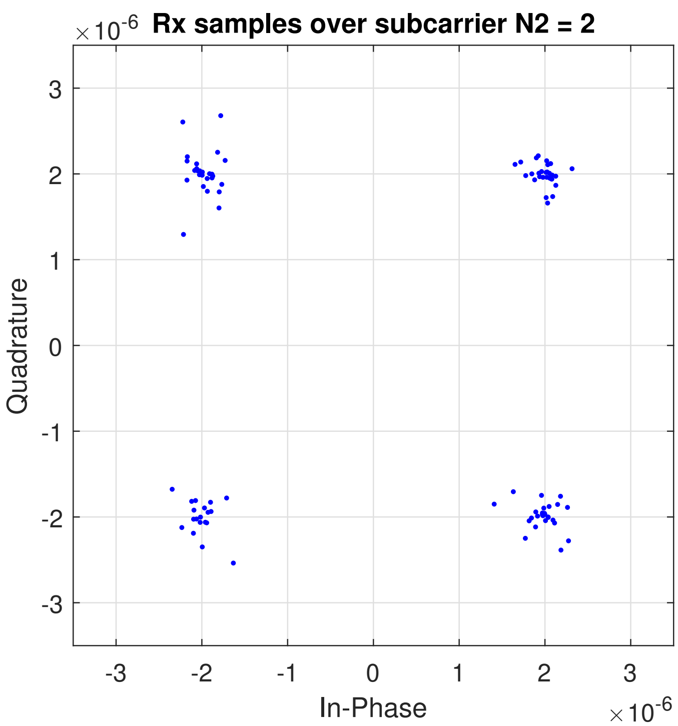

Fig. 9. Scatterplot of samples transmitted over subcarrier #2

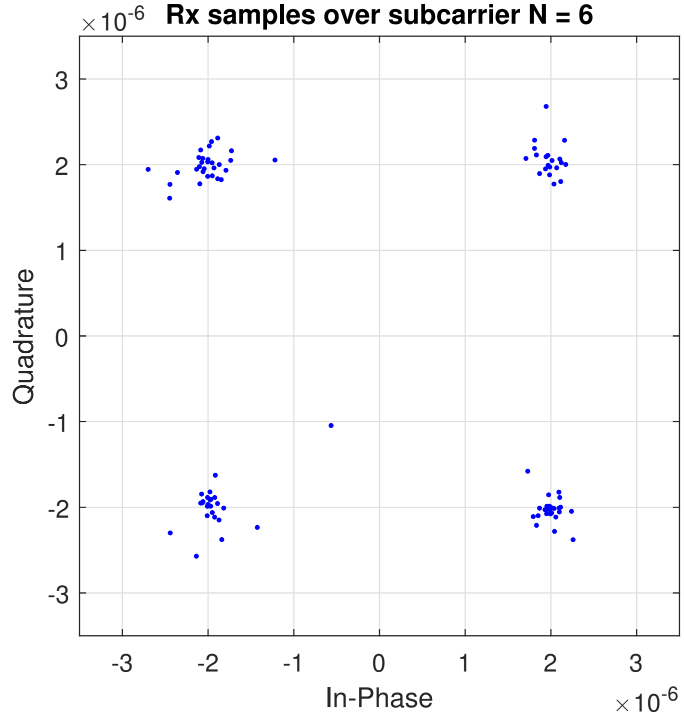

Fig. 10. Scatterplot of samples transmitted over subcarrier #6

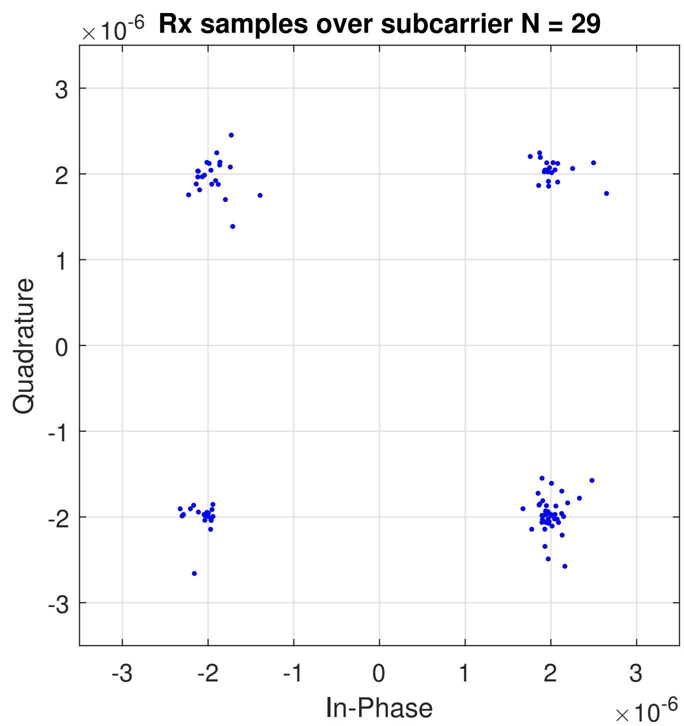

Fig. 11. Scatterplot of samples transmitted over subcarrier #29

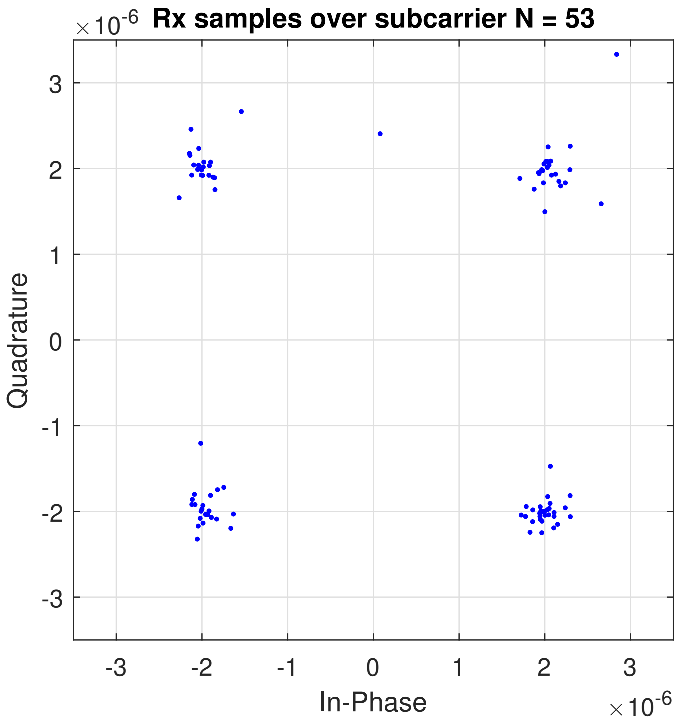

Fig. 12. Scatterplot of samples transmitted over subcarrier #53

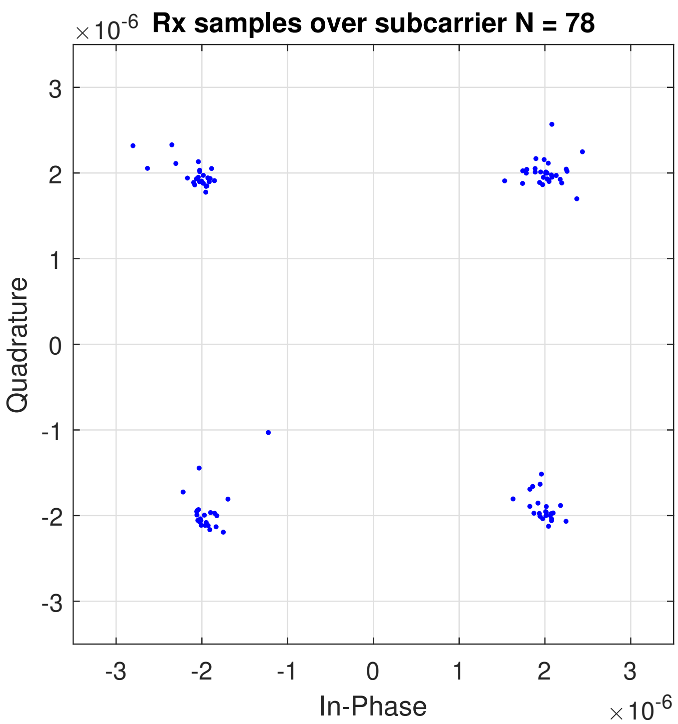

Fig. 13. Scatterplot of samples transmitted over subcarrier #78

APPENDIX C

Here are scatter plots with varying length of cyclic prefix.

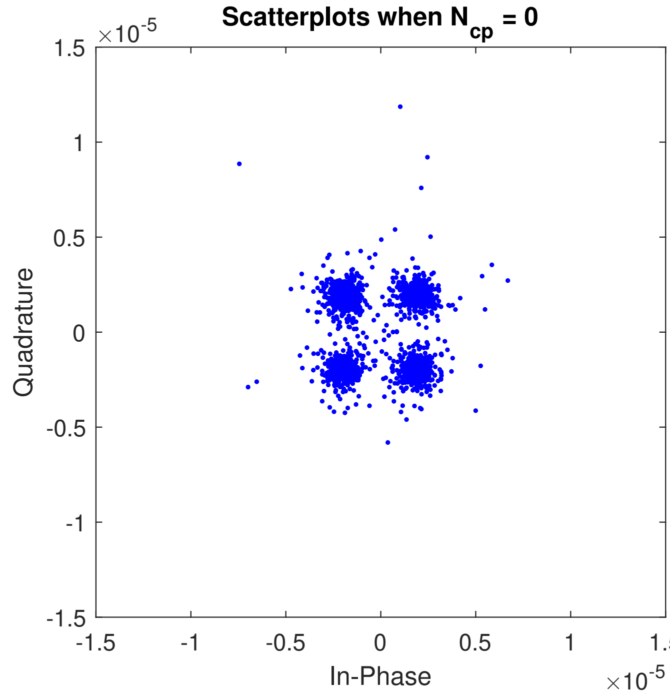

Fig. 14. Scatter plot with $N_{cp} = 0$

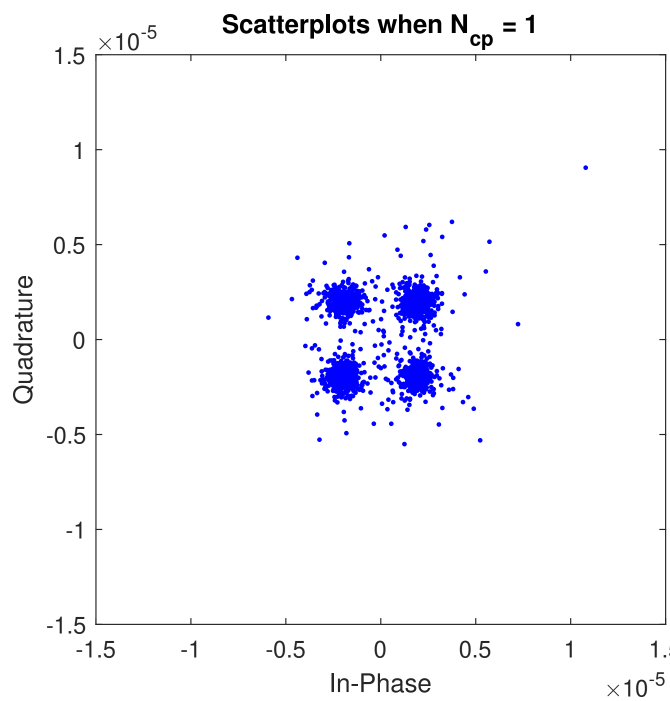

Fig. 15. Scatter plot with $N_{cp} = 1$

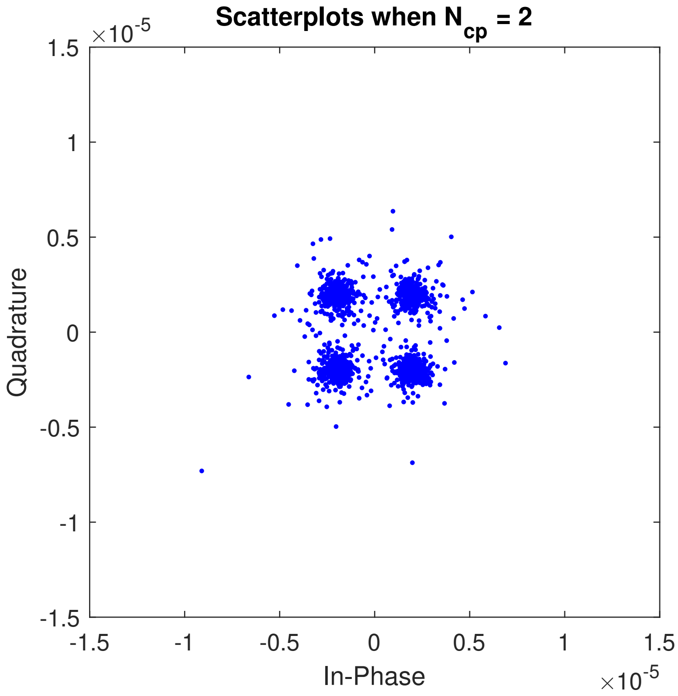

Fig. 16. Scatter plot with $N_{cp} = 2$

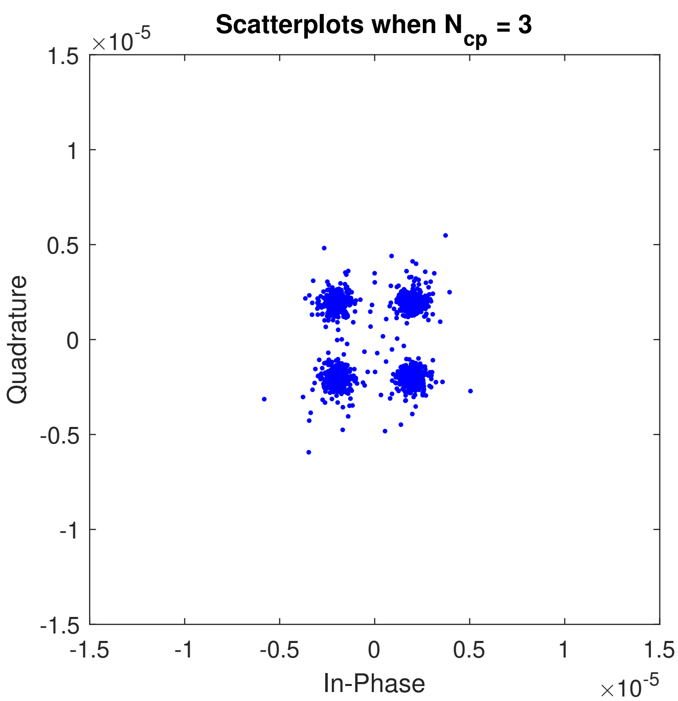

Fig. 17. Scatter plot with $N_{cp} = 3$

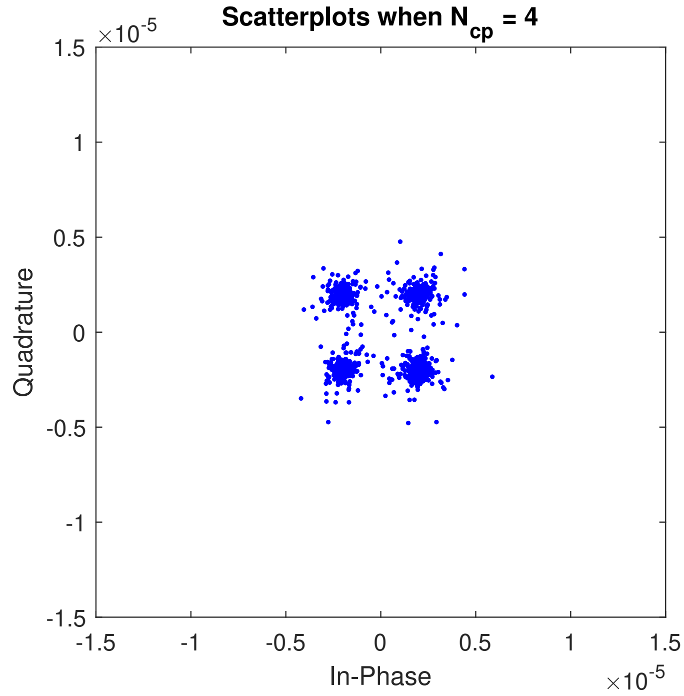

Fig. 18. Scatter plot with $N_{cp} = 4$

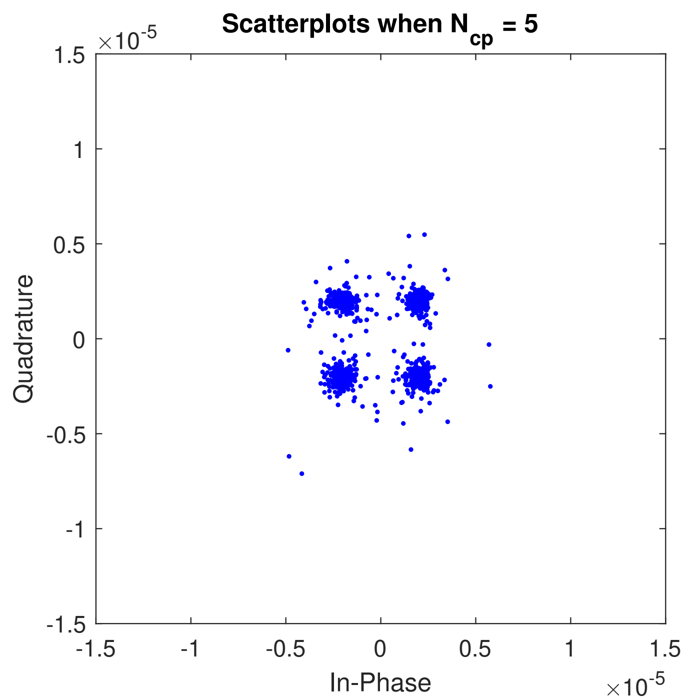

Fig. 19. Scatter plot with $N_{cp} = 5$

Appendix D

Here the MATLAB code written by the authors are given.

```matlab
%% Part 2
% settings
clc, clear all
addpath('./functions')

% initiate parameters
f_c = 2e9; % 2GHz frequency carrier
BW = 1e6; % 1MHz bandwidth
Ts = 1/BW; % symbol time
% f_s = 1/Ts; % sampling frequency
N0 = 2.07e-20*BW; % noise power (times the doublesided-bandwidth due to the unit)
v = 15; % velocity in m/s
f_D = v/physconst('LightSpeed')*f_c; % calculate the doppler frequency
fdTs = f_D*Ts; % Normalized Doppler frequency
t_ds = 5.4e-6; % delay spread
pathLoss = 10^(-101/10); % -101dB = 10*log10(Prx/Ptx)
L = ceil(t_ds/Ts); % L*T must be greater or equal to the delay spread, also an integer.
tau = [0 1 2 3 4 5]; % The path delay in samples
Ncp = L-1; % minimum number of cyclic prefix L-1, you can have more but is unnecessary.
N = 2^7; % number of samples, have it so that it is in a power of 2 (easy fft)
% we want the case when (N + Ncp)fDTs << 1 (basically much lower than coherence time)
if (N + Ncp)*fdTs > 0.1
    error('N is too large')
end

%% Simualation
mode = 0; % 0 is for no fading channel, 1 is for using fading channel
M = 100; % number of OFDM symbols we want to transmit
Ptx = 20:5:90; % create a vector of different transmission power and space
% it evenly in dB scale to get a smooth curve (more points make it edgy due to small variances)
E = 10.^(Ptx/10)*Ts; % E = Ptx*Ts [Joule = Watt*Seconds]
SNR = nan(1,length(E));
SNR_subcarrier = nan(N,length(E));
SER = nan(1,length(E));
SERtheo = nan(1,length(E));
OFDMsymbols = zeros(N,M,length(E));
for i = 1:length(E) % (loop with varying E values to plot SER over SNR)
    P = 1/L*ones(1,L); % PDP, divide equally on all taps
    QPSKconst = [1+1i 1-1i -1+1i -1-1i]*(sqrt(E(i))/sqrt(2)); % create the qpsk constellation (doesn't matter if gray)
    QPSKsymbols = zeros(N,M);
    msg = zeros(N,M);
    for m = 1:M
        % transmitter part
        msg(:,m) = randi([1 4],N,1);
        QPSKsymbols(:,m) = QPSKconst(msg(:,m)); % the channel input of L channels
        z = sqrt(N)*ifft(QPSKsymbols(:,m),N); % apply the ifft to get signal in time domain
        signal = [z(end-Ncp+1:end);z]; % add cyclic prefix

        % Channel part
        switch mode
            case 0 % here we only have awgn
                RXsignal = signal.';
                C = [1 zeros(1,length(RXsignal)-Ncp-1)]; % this is how it looks but we don't need it
                awgn = sqrt(N0/2)*(randn(size(RXsignal))+1i*randn(size(RXsignal)));
                RXsignal = RXsignal*sqrt(pathLoss) + awgn; % add pathloss and complex AWGN samples on recieved signal
                RXsignal = RXsignal(Ncp+1:end); % remove cyclic prefix
                y = fft(RXsignal)/sqrt(N);
                s = y; % for AWGN channel we dont need to equalize
            case 1
                % generate Rayleigh fading channel (needed for second parts)
                [r, h] = Fading_Channel(signal, tau, fdTs, P);
```

```
                % simulate channel with premade function
                r = r(1:end-L+1); % remove the delayed symbols
                RXsignal = r;
                hm=h(1,:); % time domain response since (N + Ncp)fDTs << 1.
                C = fft(hm,N);
                awgn = sqrt(N0/2)*(randn(size(RXsignal))+1i*randn(size(RXsignal)));
                RXsignal = RXsignal*sqrt(pathLoss) + awgn; % add pathloss and complex AWGN samples on recieved signal
                RXsignal = RXsignal(Ncp+1:end); % remove cyclic prefix
                y = fft(RXsignal)/sqrt(N);
                s = y./C.'; % compute the equalization (remove effects of the channel such as rotation)
            otherwise
                error('set mode to correct mode')
        end
        OFDMsymbols(:,m,i) = s; % store the symbols in this matrix.
    end
    % Reciever part
    recievedOFDM = reshape(OFDMsymbols(:,:,i),[N*M 1]);
    % check minimum distance
    distance = abs(repmat(recievedOFDM,1,4) - repmat(QPSKconst, length(recievedOFDM), 1)).^2; %compute the distance to each possible symbol
    [~, idx] = min(distance, [], 2); % find the constellation index for symbol alternative at minimum distance for every recieved symbol
    % calculate SER (alternative way is via symerr function)
    originalMSG = reshape(msg,[N*M 1]);
    SER(i) = length(find(originalMSG ~= idx))/length(idx);
    SNR(i) = 10*log10(E(i)*pathLoss/(N0)); % theoretical SNR in dB
    SERtheo(i) = 2*qfunc(sqrt(E(i)*pathLoss/(N0))); % theoretical SER
    SNR_subcarrier(:,i) = 10*log10(mean(abs(OFDMsymbols(:,:,i)).^2,2)/(N0)); % average subcarrier SNR in dB
end
avgSNR = mean(SNR_subcarrier,1); % average SNR in dB
%% Plot some scatterplots
if mode == 0
    scatterplot(OFDMsymbols(2,:,7)), grid on
    axis([-3.5e-6 3.5e-6 -3.5e-6 3.5e-6])
    title('Rx samples over subcarrier N = 2')
    saveas(gcf,'AWGNN2','epsc')
    scatterplot(OFDMsymbols(6,:,7)), grid on
    axis([-3.5e-6 3.5e-6 -3.5e-6 3.5e-6])
    title('Rx samples over subcarrier N = 6')
    saveas(gcf,'AWGNN6','epsc')
    scatterplot(OFDMsymbols(29,:,7)), grid on
    axis([-3.5e-6 3.5e-6 -3.5e-6 3.5e-6])
    title('Rx samples over subcarrier N = 29')
    saveas(gcf,'AWGNN29','epsc')
    scatterplot(OFDMsymbols(53,:,7)), grid on
    axis([-3.5e-6 3.5e-6 -3.5e-6 3.5e-6])
    title('Rx samples over subcarrier N = 53')
    saveas(gcf,'AWGNN53','epsc')
    scatterplot(OFDMsymbols(78,:,7)), grid on
    axis([-3.5e-6 3.5e-6 -3.5e-6 3.5e-6])
    title('Rx samples over subcarrier N = 78')
    saveas(gcf,'AWGNN78','epsc')
end
if mode == 1
    scatterplot(OFDMsymbols(2,:,7)), grid on
    axis([-3.5e-6 3.5e-6 -3.5e-6 3.5e-6])
    title('Rx samples over subcarrier N2 = 2')
    saveas(gcf,'ScatterN2','epsc')
    scatterplot(OFDMsymbols(6,:,7)), grid on
    axis([-3.5e-6 3.5e-6 -3.5e-6 3.5e-6])
    title('Rx samples over subcarrier N = 6')
    saveas(gcf,'ScatterN6','epsc')
    scatterplot(OFDMsymbols(29,:,7)), grid on
```

```
        axis([-3.5e-6 3.5e-6 -3.5e-6 3.5e-6])
        title('Rx samples over subcarrier N = 29')
        saveas(gcf,'ScatterN29','epsc')
        scatterplot(OFDMsymbols(53,:,7)), grid on
        axis([-3.5e-6 3.5e-6 -3.5e-6 3.5e-6])
        title('Rx samples over subcarrier N = 53')
        saveas(gcf,'ScatterN53','epsc')
        scatterplot(OFDMsymbols(78,:,7)), grid on
        axis([-3.5e-6 3.5e-6 -3.5e-6 3.5e-6])
        title('Rx samples over subcarrier N = 78')
        saveas(gcf,'ScatterN78','epsc')
end
%% Plot SER vs SNR
switch mode
    case 0
        figure()
        semilogy(SNR,SER,'r'), grid on, hold on
        semilogy(SNR,SERtheo,'b')
        ylim([0.001 1]), xlim([-5 20])
        xlabel('SNR [dB]'), ylabel('SER')
        legend('simulated SER','theoretical AWGN SER')
        title('SER vs SNR for AWGN channel')
        saveas(gcf,'SERvsSNR_awgn','epsc')
    case 1
        figure()
        semilogy(SNR,SER,'r'), grid on, hold on
        semilogy(SNR,SERtheo,'b')
        ylim([0.001 1]), xlim([-5 30])
        xlabel('SNR [dB]'), ylabel('SER')
        title('SER vs SNR for fading channel')
        legend('simulated SER','theoretical AWGN SER')
        saveas(gcf,'SERvsSNR_fading','epsc')
end
%% plot SNR vs E
figure()
semilogx(E,avgSNR,'r--','LineWidth',2), grid on, hold on
semilogx(E,SNR,'k--','LineWidth',2)
title('SNR vs symbol energy E')
xlabel('log(E)'), ylabel('SNR [dB]')
legend('Average simulation SNR','Theoretical SNR','Location','NorthWest')
saveas(gcf,'SNRvsE','epsc')

%% Simulate with reducing Ncp lengths
E = 1e-1;
snr = 10*log10(E*pathLoss/N0);
P = 1/L*ones(1,L); % PDP, divide equally on all taps
QPSKconst = [1+1i 1-1i -1+1i -1-1i]*(sqrt(E)/sqrt(2)); % create the qpsk constellation
OFDMsymbols = zeros(N,M);
QPSKsymbols = zeros(N,M);
msg = zeros(N,M);
Ncp = L-1:-1:0;
SER = nan(1,length(Ncp));
for i = 1:length(Ncp)
    for m = 1:M
        % transmitter part
        msg(:,m) = randi([1 4],N,1);
        QPSKsymbols(:,m) = QPSKconst(msg(:,m)); % the channel input of L channels
        z = sqrt(N)*ifft(QPSKsymbols(:,m),N); % apply the ifft to get signal in time domain
        signal = [z(end-Ncp(i)+1:end);z]; % add cyclic prefix
        % channel part
        [r, h] = Fading_Channel(signal, tau, fdTs, P); % simulate channel with premade function
        r = r(1:end-L+1); % remove the delayed symbols
        RXsignal = r;
        hm=h(1,:); % time domain response since (N + Ncp)fDTs << 1.
        C = fft(hm,N);
        awgn = sqrt(N0/2)*(randn(size(RXsignal))+1i*randn(size(RXsignal)));
        RXsignal = RXsignal*sqrt(pathLoss) + awgn; % add complex AWGN samples on recieved signal
        RXsignal = RXsignal(Ncp(i)+1:end); % remove cyclic prefix
        y = fft(RXsignal)/sqrt(N);
        s = y./C.'; % compute the equalization (remove effects of the channel such as rotation)
        OFDMsymbols(:,m) = s; % store the symbols in this matrix.
    end
    % Reciever part
    recievedOFDM = reshape(OFDMsymbols,[N*M 1]);
```

```
    % check minimum distance
    distance = abs(repmat(recievedOFDM,1,4) - repmat(QPSKconst, length(recievedOFDM), 1)).^2; %compute the distance to each possible symbol
    [~, idx] = min(distance, [], 2); % find the constellation index for symbol alternative at minimum distance for every recieved symbol
    % calculate SER
    originalMSG = reshape(msg,[N*M 1]);
    SER(i) = length(find(abs(originalMSG-idx)))/length(idx);
    scatterplot(recievedOFDM)
    axis([-1.5e-5 1.5e-5 -1.5e-5 1.5e-5])
    title(['Scatterplots when N_{cp} = ' num2str(Ncp(i))])
    saveas(gcf,['scatter_Ncp' num2str(Ncp(i))],'epsc')
end
figure()
plot(Ncp,SER,'rx'), grid on
title('SER vs length of N_{cp} with fixed SNR')
xlabel('N_{cp} lengths'), ylabel(['SER for SNR= ' num2str(snr) '[dB]'])
saveas(gcf,'SER_Ncp','epsc')
```