LBP Mid-Term Evaluation Report

Aman Sinha
19116007

Project Name- Crest Factor Reduction Technique for 4G/5G Waveforms

My work until Mid Term Semester- Used Clipping Reduction Method as PAPR Reduction Technique in OFDM Signal.

MATLAB Simulation Code-

```
% channel noise on the ofdm signals and then introduces clipping as
% a PAPR reduction method
clear all
clc
close
%   ---------------
%   A: Setting Parameters
%   ---------------
M = 4;                      %   QPSK signal constellation
no_of_data_points = 128;        %   have 128 data points
block_size = 8;                 %   size of each ofdm block
cp_len = ceil(0.1*block_size);  %   length of cyclic prefix
no_of_ifft_points = block_size; %   128 points for the FFT/IFFT
no_of_fft_points = block_size;
%   -------------------------------------------
%   B:  %   +++++   TRANSMITTER    +++++
%   -------------------------------------------
%   Generate 1 x 128 vector of random data points
data_source = randsrc(1, no_of_data_points, 0:M-1);
figure(1)
stem(data_source); grid on; xlabel('Data Points'); ylabel('transmitted data phase representation')
title('Transmitted Data "O"')


%   Perform QPSK modulation
qpsk_modulated_data = pskmod(data_source, M);
scatterplot(qpsk_modulated_data);title('MODULATED TRANSMITTED DATA');
```

```
%   Do IFFT on each block
%   Make the serial stream a matrix where each column represents a pre-OFDM
num_cols=length(qpsk_modulated_data)/block_size;
data_matrix = reshape(qpsk_modulated_data, block_size, num_cols);

%   Create empty matix to put the IFFT'd data
cp_start = block_size-cp_len;
cp_end = block_size;

%   Operate columnwise & do CP
for i=1:num_cols,
   ifft_data_matrix(:,i) = ifft((data_matrix(:,i)),no_of_ifft_points);
   %   Compute and append Cyclic Prefix
   for j=1:cp_len,
      actual_cp(j,i) = ifft_data_matrix(j+cp_start,i);
   end
   %   Append the CP to the existing block to create the actual OFDM block
   ifft_data(:,i) = vertcat(actual_cp(:,i),ifft_data_matrix(:,i));
end

%  Convert to serial stream for transmission
[rows_ifft_data cols_ifft_data]=size(ifft_data);
len_ofdm_data = rows_ifft_data*cols_ifft_data;

%   Actual OFDM signal to be transmitted
ofdm_signal = reshape(ifft_data, 1, len_ofdm_data);
figure(3)
plot(real(ofdm_signal)); xlabel('Time'); ylabel('Amplitude');
title('OFDM Signal');grid on;

%   ------------------------------------------------------------
%   C:  %   +++++   clipping as a PAPR reduction method    +++++
%   ------------------------------------------------------------
avg=0.4;
clipped=ofdm_signal;
for i=1:length(clipped)
  if clipped(i) > avg
    clipped(i) = avg;
    end
    if clipped(i) < -avg
```

```
    clipped(i) = -avg;
    end
end
figure(4)
plot(real(clipped)); xlabel('Time'); ylabel('Amplitude');
title('clipped Signal');grid on;

%   ----------------------------------------
%   D:  %   +++++   HPA    +++++
%   ----------------------------------------
%To show the effect of the PA simply we will add random complex noise
%when the power exceeds the avg. value, otherwise it add nothing.

% Generate random complex noise
noise = randn(1,len_ofdm_data) +  sqrt(-1)*randn(1,len_ofdm_data);

% Transmitted OFDM signal after passing through HPA

%without clipping
for i=1:length(ofdm_signal)
  if ofdm_signal(i) > avg
    ofdm_signal(i) = ofdm_signal(i)+noise(i);
    end
    if ofdm_signal(i) < -avg
    ofdm_signal(i) = ofdm_signal(i)+noise(i);
    end
end
figure(5)
plot(real(ofdm_signal)); xlabel('Time'); ylabel('Amplitude');
title('OFDM Signal after HPA');grid on;

%with clipping
avg=0.4;
for i=1:length(clipped)
  if clipped(i) > avg
    clipped(i) = clipped(i)+noise(i);
    end
    if clipped(i) < -avg
    clipped(i) = clipped(i)+noise(i);
    end
```

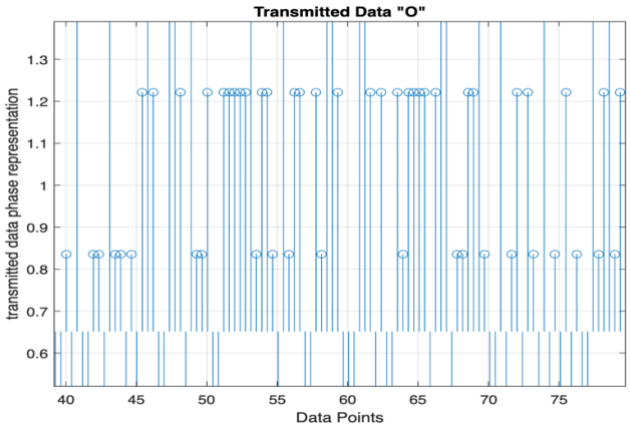
```
end
figure(6)
plot(real(clipped)); xlabel('Time'); ylabel('Amplitude');
title('clipped Signal after HPA');grid on;



%   -------------------------------
%   E:  %   +++++   CHANNEL    +++++
%   -------------------------------
%   Create a complex multipath channel
channel = randn(1,block_size) + sqrt(-1)*randn(1,block_size);

%   ----------------------------------------
%   F:  %   +++++   RECEIVER    +++++
%   ----------------------------------------

%   1.  Pass the ofdm signal through the channel
after_channel = filter(channel, 1, ofdm_signal);

%   2.   Add Noise
awgn_noise = awgn(zeros(1,length(after_channel)),0);

%   3.  Add noise to signal...

recvd_signal = awgn_noise+after_channel;

%   4.  Convert Data back to "parallel" form to perform FFT
recvd_signal_matrix = reshape(recvd_signal,rows_ifft_data, cols_ifft_data);

%   5.  Remove CP
recvd_signal_matrix(1:cp_len,:)=[];

%   6.  Perform FFT
for i=1:cols_ifft_data,
    %   FFT
    fft_data_matrix(:,i) = fft(recvd_signal_matrix(:,i),no_of_fft_points);
end

%   7.  Convert to serial stream
```

```
recvd_serial_data = reshape(fft_data_matrix, 1,(block_size*num_cols));

%   8.  Demodulate the data
qpsk_demodulated_data = pskdemod(recvd_serial_data,M);

figure(7)
stem(qpsk_demodulated_data,'rx');
grid on;xlabel('Data Points');ylabel('received data phase representation');title('Received Data "X"')
%   -----------------------------------------------------
%   F:  %   +++++   RECEIVER of clipped signal    +++++
%   -----------------------------------------------------

%   1.  Pass the ofdm signal through the channel
after_channel = filter(channel, 1, clipped);

%   2.   Add Noise
awgn_noise = awgn(zeros(1,length(after_channel)),0);

%   3.  Add noise to signal...

recvd_signal = awgn_noise+after_channel;

%   4.  Convert Data back to "parallel" form to perform FFT
recvd_signal_matrix = reshape(recvd_signal,rows_ifft_data, cols_ifft_data);

%   5.  Remove CP
recvd_signal_matrix(1:cp_len,:)=[];

%   6.  Perform FFT
for i=1:cols_ifft_data,
    %   FFT
    fft_data_matrix(:,i) = fft(recvd_signal_matrix(:,i),no_of_fft_points);
end

%   7.  Convert to serial stream
recvd_serial_data = reshape(fft_data_matrix, 1,(block_size*num_cols));

%   8.  Demodulate the data
qpsk_demodulated_data = pskdemod(recvd_serial_data,M);
figure(8)
```

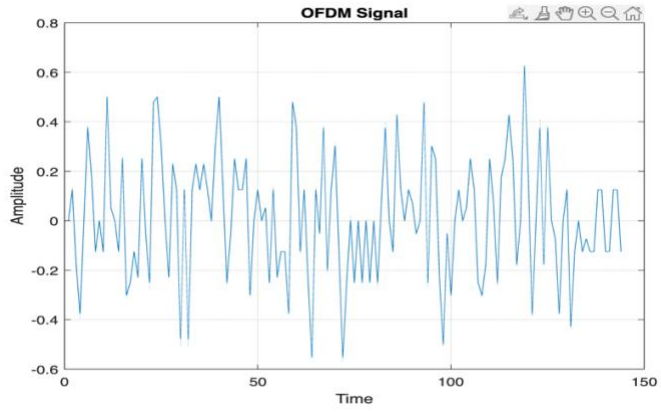
```
stem(qpsk_demodulated_data,'rx');
grid on;xlabel('Data Points');ylabel('received data phase representation');title('Received Data clipped "X"')
```

Simulation Work and Output:

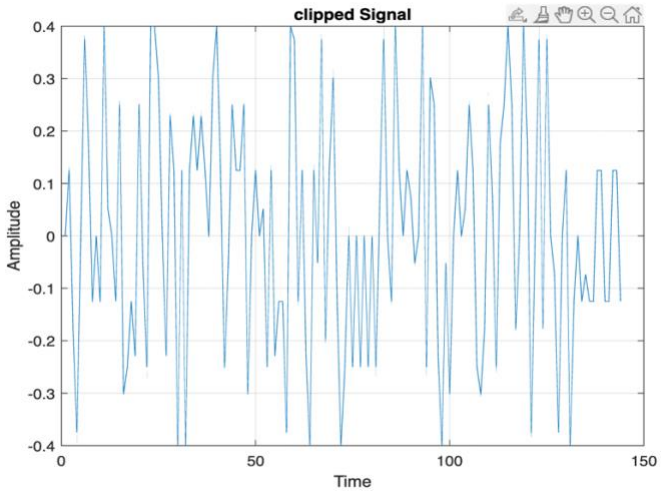
1. Transmitted Data Phase Representation

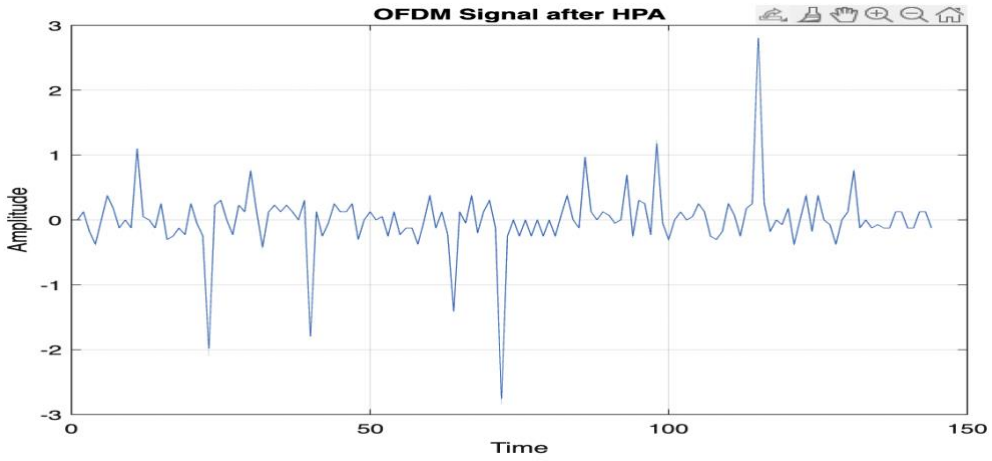


2. Clipping as PAPR Reduction Technique

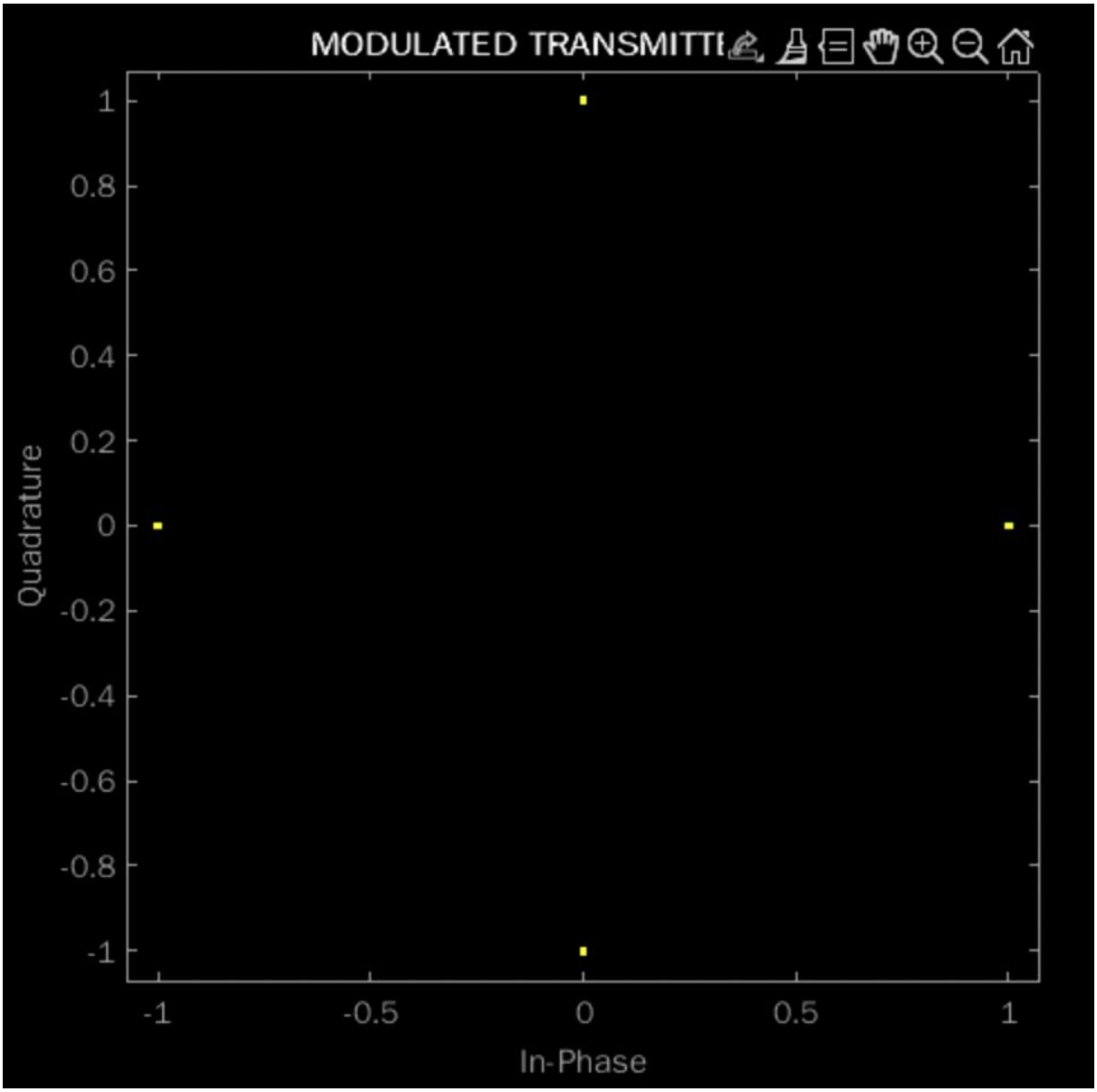


3. After Reducution





Demodulated Data Representation :


MODULATED TRANSMITTI
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
Quadrature
-1
-0.5
0
0.5
1
In-Phase