Table of Contents


```
%Vishnu Kaimal
%DongKyu Kim
%MIMO/OFDM
%MIMO and OFDM combination
%ECE408 Wireless Communications
%Professor Hoerning
clc; clear all; close all;
```

parameters

```
numIter = 3;% The number of iterations of the simulation
nSym = 5e3; % The number of symbols per packet
M = 16;     % Modulation Order
Mt = 2;     %number of transmitters
Mr = 2;     %number of receivers
EbNo = -10:1:30; %EbNo range to iterate over for plot

SNR_Vec = EbNo + 10*log10(log2(M))+10*log10(64/80);%SNR conversion
 from EbNo
lenSNR = length(SNR_Vec);
ber = zeros(2,3,numIter,lenSNR); %ber store
H = sqrt(1/2)*(randn(Mr,Mt,nSym*48/2,numIter)+1j*randn(Mr,Mt,nSym*...
    48/2,numIter));%channel for each iteration

index = [1:5 7:19 21:26 28:33 35:47 49:53]+5; %frame parameters
index_pilot = [6 20 34 48]+5; % frame parameters

%params for rayleigh frequency selective channel
Ts = 1e-3;
Fd = 0;
tau = [0 1e-5 3.5e-5 12e-5];
pdb = [0 -1 -1 -3];
```

OFDM zero forcing

```
U = zeros(Mr,Mt,nSym*48/2);
V = U;
tx_chan = zeros(Mr,1,nSym*48/2);
tx_chan2 = tx_chan;
tx_process = tx_chan;
```

```
for i = 1:numIter
    bits = randi([0,M-1],1, nSym*48);     % Generate random bits
    bits_c = reshape(bits,2,[]);
    mod_data = qammod(bits,M);
    ofdm_data = reshape(mod_data,48,[]);
    ofdm_frame = zeros(64,nSym);
    ofdm_frame(index,:) = ofdm_data;
    ofdm_frame(index_pilot,:) = 1;

    ifft_ofdm = ifft(ofdm_frame,64);
    ofdm_trans = [ifft_ofdm(49:64,:); ifft_ofdm]; %guard

    %construction of frequency selective channel
    h = rayleighchan(Ts, Fd, tau, pdb);
    chan = zeros(80,nSym);
    ofdm_chan = zeros(80,nSym);
    for k=1:nSym
        chan(:,k) = filter(h,ones(80,1));
        ofdm_chan(:,k) = chan(:,k).*ofdm_trans(:,k); %apply channel to
 signal
    end

    for j = 1:lenSNR
        noise = sqrt(1/2)*(randn(80,nSym)+1j*randn(80,nSym));
        %split half of noise into ofdm and half into channel
        ofdm_noisy = ofdm_chan + 10^(-1*(SNR_Vec(j)-3)/20)*noise;

        ofdm_no_guard = ofdm_noisy(17:end,:);
        ofdm_orig_frame = fft(ofdm_no_guard,64);

        ofdm_zf = ofdm_orig_frame./chan(17:end,:);
        ofdm_rcv_data = ofdm_zf(index,:);
        mod_rcv_data = reshape(ofdm_rcv_data,1,[]); % this is the OFDM
 symbol
        tx_mimo = reshape(mod_rcv_data,2,[]);
%         tx_mimo = repmat(mod_rcv_data,2,1);
        tx_mimo = permute(tx_mimo,[1,3,2]);

        % we decided not to penalize MIMO channel with a noise
% with precoding MIMO
        for k=1:nSym*48/2 % divide by 2 because 2 tx antenna
            [U(:,:,k),S(:,:,k),V(:,:,k)] = svd(H(:,:,k,i));
            tx_chan(:,:,k) = H(:,:,k,i)*V(:,:,k)*tx_mimo(:,:,k);
            tx_chan2(:,:,k) = H(:,:,k,i)*tx_mimo(:,:,k);
        end
        chan_n =
 sqrt(1/2)*(randn(Mr,1,nSym*48/2)+1j*randn(Mr,1,nSym*48/2));
        txNoisy = tx_chan + 10^(-1*(SNR_Vec(j)-3)/20)*chan_n;
        for k=1:nSym*48/2
            tx_process(:,:,k) = S(:,:,k)^-1*U(:,:,k)'*txNoisy(:,:,k);
        end

        rx = qamdemod(tx_process,M);
```

```
        % Compute and store the BER for this iteration
        [~, ber(1,1,i,j)] = biterr(bits_c, squeeze(rx));

% with zero forcing MIMO
        txNoisy = tx_chan2 + 10^(-1*(SNR_Vec(j)-3)/20)*chan_n;
        for k=1:nSym*48/2
            W(:,:,k) = (H(:,:,k,i)'*H(:,:,k,i))^-1*H(:,:,k,i)';
            tx_process(:,:,k) = W(:,:,k)*txNoisy(:,:,k);
        end
        rx = qamdemod(tx_process,M);
        % Compute and store the BER for this iteration
        [~, ber(1,2,i,j)] = biterr(bits_c, squeeze(rx));

% with MMSE MIMO

        for k=1:nSym*48/2
            W(:,:,k) =
 (H(:,:,k,i)'*H(:,:,k,i)+eye(Mt)*10^(-1*(SNR_Vec(j)-3)/20))^-1*H(:,:,k,i)';
            tx_process(:,:,k) = W(:,:,k)*txNoisy(:,:,k);
        end

        rx = qamdemod(tx_process,M);

        % Compute and store the BER for this iteration
        [~, ber(1,3,i,j)] = biterr(bits_c, squeeze(rx));

    end
end
```

OFDM MMSE

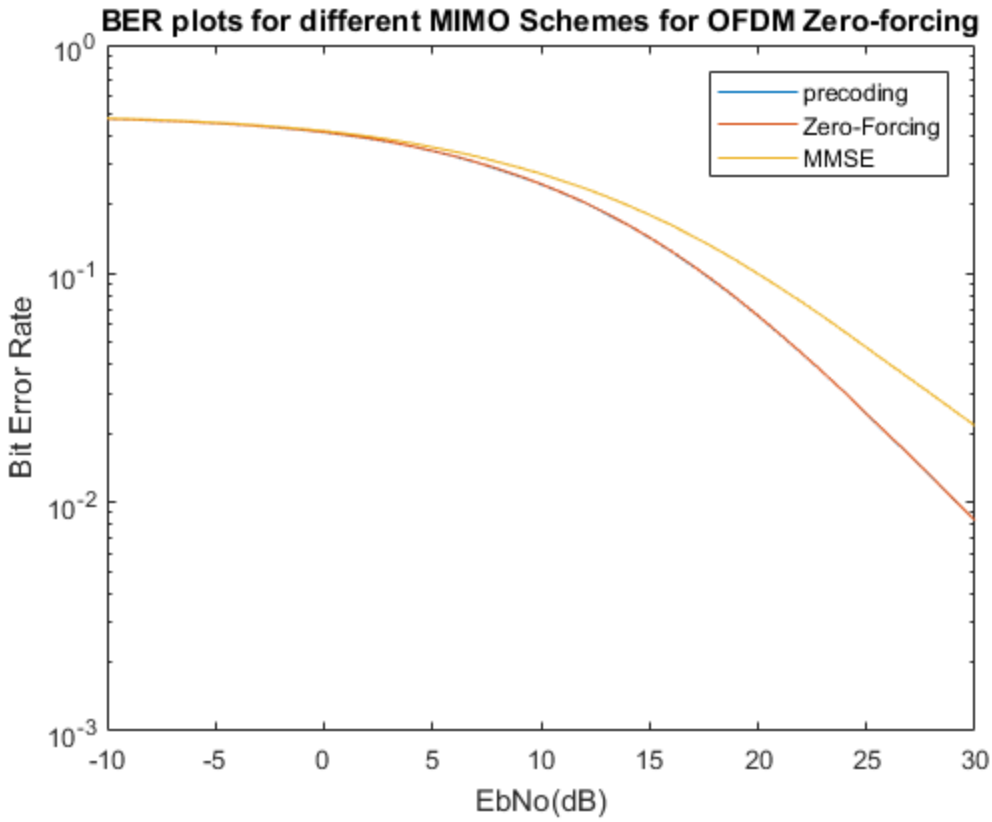
```
U = zeros(Mr,Mt,nSym*48/2);
V = U;
tx_chan = zeros(Mr,1,nSym*48/2);
tx_chan2 = tx_chan;
tx_process = tx_chan;

for i = 1:numIter
    bits = randi([0,M-1],1, nSym*48);     % Generate random bits
    bits_c = reshape(bits,2,[]);
    mod_data = qammod(bits,M);
    ofdm_data = reshape(mod_data,48,[]);
    ofdm_frame = zeros(64,nSym);
    ofdm_frame(index,:) = ofdm_data;
    ofdm_frame(index_pilot,:) = 1;

    ifft_ofdm = ifft(ofdm_frame,64);
    ofdm_trans = [ifft_ofdm(49:64,:); ifft_ofdm]; %guard

    %construction of frequency selective channel
    h = rayleighchan(Ts, Fd, tau, pdb);
    chan = zeros(80,nSym);
    ofdm_chan = zeros(80,nSym);
```

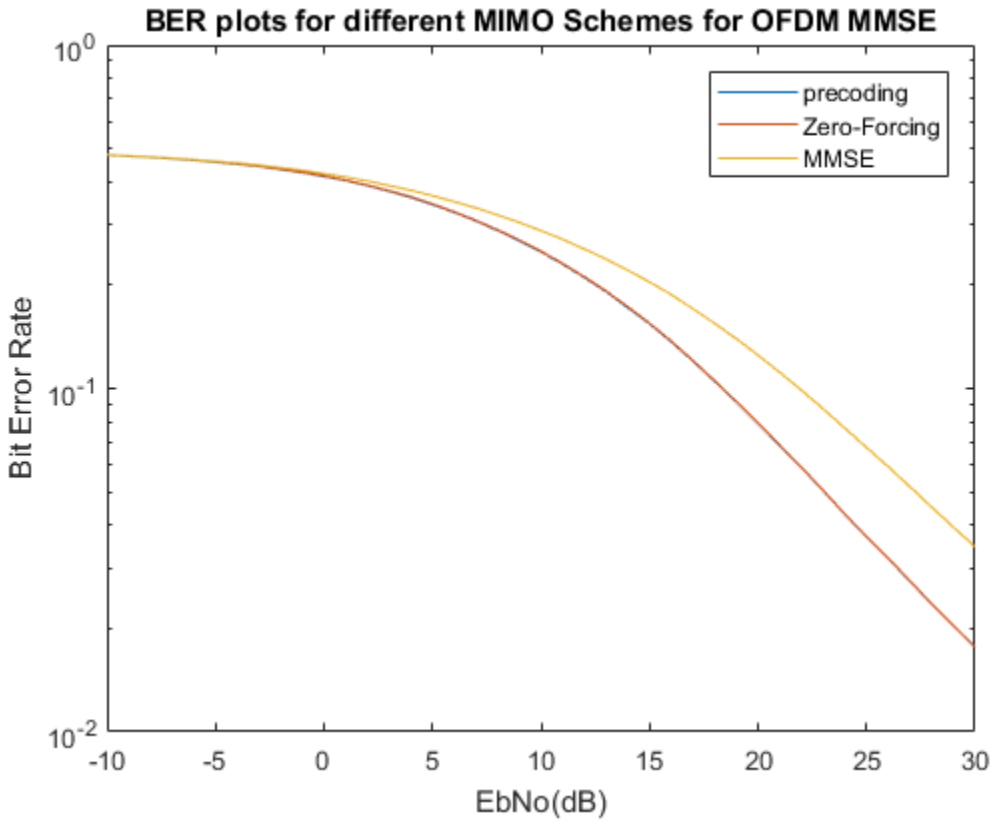
```
    for k=1:nSym
        chan(:,k) = filter(h,ones(80,1));
        ofdm_chan(:,k) = chan(:,k).*ofdm_trans(:,k); %apply channel to
 signal
    end

    for j = 1:lenSNR
        noise = sqrt(1/2)*(randn(80,nSym)+1j*randn(80,nSym));
        %split half of noise into ofdm and half into channel
        ofdm_noisy = ofdm_chan + 10^(-1*(SNR_Vec(j)-3)/20)*noise;

        ofdm_no_guard = ofdm_noisy(17:end,:);
        ofdm_orig_frame = fft(ofdm_no_guard,64);

        norm = conj(chan(17:end,:)).*chan(17:end,:) +
 10^(-1*(SNR_Vec(j)-3)/20);
        ofdm_zf = ofdm_orig_frame.*conj(chan(17:end,:))./norm;
        ofdm_rcv_data = ofdm_zf(index,:);
        mod_rcv_data = reshape(ofdm_rcv_data,1,[]); % this is the OFDM
 symbol
        tx_mimo = reshape(mod_rcv_data,2,[]);
%         tx_mimo = repmat(mod_rcv_data,2,1);
        tx_mimo = permute(tx_mimo,[1,3,2]);

        % we decided not to penalize MIMO channel with a noise
% with precoding MIMO
        for k=1:nSym*48/2 % divide by 2 because 2 tx antenna
            [U(:,:,k),S(:,:,k),V(:,:,k)] = svd(H(:,:,k,i));
            tx_chan(:,:,k) = H(:,:,k,i)*V(:,:,k)*tx_mimo(:,:,k);
            tx_chan2(:,:,k) = H(:,:,k,i)*tx_mimo(:,:,k);
        end
        chan_n =
 sqrt(1/2)*(randn(Mr,1,nSym*48/2)+1j*randn(Mr,1,nSym*48/2));
        txNoisy = tx_chan + 10^(-1*(SNR_Vec(j)-3)/20)*chan_n;
        for k=1:nSym*48/2
            tx_process(:,:,k) = S(:,:,k)^-1*U(:,:,k)'*txNoisy(:,:,k);
        end
        rx = qamdemod(tx_process,M);
        % Compute and store the BER for this iteration
        [~, ber(2,1,i,j)] = biterr(bits_c, squeeze(rx));

% with zero forcing MIMO
        txNoisy = tx_chan2 + 10^(-1*(SNR_Vec(j)-3)/20)*chan_n;
        for k=1:nSym*48/2
            W(:,:,k) = (H(:,:,k,i)'*H(:,:,k,i))^-1*H(:,:,k,i)';
            tx_process(:,:,k) = W(:,:,k)*txNoisy(:,:,k);
        end
        rx = qamdemod(tx_process,M);
        % Compute and store the BER for this iteration
        [~, ber(2,2,i,j)] = biterr(bits_c, squeeze(rx));

% with MMSE MIMO

        for k=1:nSym*48/2
```

```
            W(:,:,k) =
 (H(:,:,k,i)'*H(:,:,k,i)+eye(Mt)*10^(-1*(SNR_Vec(j)-3)/20))^-1*H(:,:,k,i)';
            tx_process(:,:,k) = W(:,:,k)*txNoisy(:,:,k);
        end

        rx = qamdemod(tx_process,M);

        % Compute and store the BER for this iteration
        [~, ber(2,3,i,j)] = biterr(bits_c, squeeze(rx));

    end
end
```

plot

```
ber = mean(ber,3); %take mean across all iterations

%plotting all
figure;
semilogy(EbNo,squeeze(ber(1,1,1,:)),'DisplayName','precoding');
hold on;
semilogy(EbNo,squeeze(ber(1,2,1,:)),'DisplayName','Zero-Forcing');
semilogy(EbNo,squeeze(ber(1,3,1,:)),'DisplayName','MMSE');

title('BER plots for different MIMO Schemes for OFDM Zero-forcing');
xlabel('EbNo(dB)');
ylabel('Bit Error Rate');
legend('show');
hold off;

figure;
semilogy(EbNo,squeeze(ber(2,1,1,:)),'DisplayName','precoding');
hold on;
semilogy(EbNo,squeeze(ber(2,2,1,:)),'DisplayName','Zero-Forcing');
semilogy(EbNo,squeeze(ber(2,3,1,:)),'DisplayName','MMSE');

title('BER plots for different MIMO Schemes for OFDM MMSE');
xlabel('EbNo(dB)');
ylabel('Bit Error Rate');
legend('show');
hold off;
```


BER plots for different MIMO Schemes for OFDM Zero-forcing
precoding
Zero-Forcing
MMSE
Bit Error Rate
EbNo(dB)


BER plots for different MIMO Schemes for OFDM MMSE
precoding
Zero-Forcing
MMSE
Bit Error Rate
EbNo(dB)

Published with MATLAB® R2017a