Contents


```
% ECE408 - Wireless Communications
% Jongoh (Andy) Jeong
% MRRC, Alamouti Space-Time Block Coding (STBC) Simulations
% Date: March 11, 2020
clear all; close all; clc;
```

(0) Flat-fading Rayleigh Channel Setup

```
nIter = 1e2;
M = 2;                  % modulation type
k = log2(M);            % bits per symbol
mod = comm.PSKModulator(M,0);
demod = comm.PSKDemodulator(M,0);
EbNo = 0:2.5:50;                      % bit to noise power ratio (dB)
EsNo = EbNo + 10*log10(k);            % symbol to noise power ratio(dB)
samprate = 1;                         % sampling rate (Tsymbol : Tsample)
snr = EsNo - 10*log10(samprate);      % adjusted SNR (dB)

% parameters for rayleigh channels
Ts = 1e-5;                            % sample time
N = int64(1/Ts);                      % message word length
fd = 10;                              % Maximum Doppler Shift frequency

r_chan1 = newRayleighChan(N, fd);
r_chan2 = newRayleighChan(N, fd);
r_chan3 = newRayleighChan(N, fd);
r_chan4 = newRayleighChan(N, fd);

disp('[INFO] Rayleigh channels created.')
```

```
[INFO] Rayleigh channels created.
```

(1) BPSK Tx (flat-fading rayleigh channel) no diversity

```
fprintf('[INFO] Uncoded (no diversity) in simulation\t')
bers = zeros([length(snr),nIter]);
% iterate nIter times
for ii = 1:nIter
    fprintf('.')
    x = randi([0 M-1],N,1);              % random message bits
    modulated = step(mod,x);             % modulate message to PSK symbols
```

```
    filtered = r_chan1.*modulated;      % transmit symbols through Rayleigh flat fading ch
annel

    % allocate memory space
    transmitted = zeros(length(modulated),length(snr));
    demodulated = zeros(length(modulated),length(snr));
    for i=1:length(snr)
        % pass through an AWGN channel
        transmitted(:,i) = awgn(filtered,snr(i),'measured')./r_chan1;
        % demodulate PSK symbols
        demodulated(:,i) = step(demod,transmitted(:,i));
    end
    % compute Bit Error Rate
    [~,bers(:,ii)] = biterr(demodulated,x);
end
ber1 = mean(bers,2);

disp('[INFO] Uncoded (no diversity) complete')
```

```
[INFO] Uncoded (no diversity) in simulation     .......................................
..........................................................[INFO] Uncoded (no diversity) co
mplete
```

(2) BPSK Tx (flat-fading rayleigh channel), MRRC (2 Rx)

```
fprintf('[INFO] MRRC 2 Rx in simulation\t')
bers = zeros([length(snr),nIter]);
% iterate nIter times
for ii = 1:nIter
    fprintf('.')
    x = randi([0 M-1],N,1);             % random message bits
    modulated = step(mod,x);            % modulate message to PSK symbols
    filtered1 = r_chan1 .* modulated;   % transmit symbols through Rayleigh flat fading ch
annel
    filtered2 = r_chan2 .* modulated;

    % antenna path gains = sum channel powers of both Rx antennas
    h = [r_chan1, r_chan2];

    % signals from Rayleigh channel for both Rx antennas
    filtered = [filtered1,filtered2];

    % allocate memory space
    combined = zeros(length(modulated),length(snr));
    demodulated = zeros(length(modulated),length(snr));
    for i=1:length(snr)
        % pass through an AWGN channel
        n_filtered = awgn(filtered,snr(i),'measured');
        % combine symbols at 2 Rx antennas
        combined(:,i) = sum(conj(h).*n_filtered,2)./sum(h.*conj(h),2);
        % demodulate PSK symbols
        demodulated(:,i) = step(demod,combined(:,i));
    end
        % compute Bit Error Rate
    [~,bers(:,ii)] = biterr(demodulated,x);
end
ber2 = mean(bers,2);
```

```
disp('[INFO] MRRC 2 Rx complete')
```

```
[INFO] MRRC 2 Rx in simulation ............................................................
...........................................[INFO] MRRC 2 Rx complete
```

(3) BPSK Tx (flat-fading rayleigh channel), MRRC (4 Rx)

```
fprintf('[INFO] MRRC 4 Rx in simulation\t')
bers = zeros([length(snr),nIter]);
% iterate nIter times
for ii = 1:nIter
    fprintf('.')
    x = randi([0 M-1],N,1);             % random message bits
    modulated = step(mod,x);            % modulate message to PSK symbols
    filtered1 = r_chan1.*modulated;     % transmit symbols through Rayleigh flat fading ch
annel
    filtered2 = r_chan2.*modulated;
    filtered3 = r_chan3.*modulated;
    filtered4 = r_chan4.*modulated;

    % antenna path gains = sum channel powers of all Rx antennas
    h = [r_chan1,r_chan2,r_chan3,r_chan4];

    % signals from Rayleigh channel for both Tx antennas
    filtered = [filtered1,filtered2,filtered3,filtered4];

    % allocate memory space
    combined = zeros(length(modulated),length(snr));
    demodulated = zeros(length(modulated),length(snr));
    for i=1:length(snr)
        % pass through an AWGN channel
        n_filtered = awgn(filtered,snr(i),'measured');
        % combine symbols at Rx antenna
        combined(:,i) = sum(conj(h).*n_filtered,2)./sum(h.*conj(h),2);
        % demodulate PSK symbols
        demodulated(:,i) = step(demod,combined(:,i));
    end

        % compute Bit Error Rate
    [~,bers(:,ii)] = biterr(demodulated,x);
end
ber3 = mean(bers,2);
disp('[INFO] MRRC 4 Rx complete')
```

```
[INFO] MRRC 4 Rx in simulation ............................................................
...........................................[INFO] MRRC 4 Rx complete
```

(4) BPSK Tx (flat-fading rayleigh channel), Alamouti (2 Tx, 1 Rx)

```
fprintf('[INFO] Alamouti 2 Tx, 1 Rx in simulation\t')
bers = zeros([length(snr),nIter]);
% iterate nIter times
for ii = 1:nIter
    fprintf('.')
    x = randi([0 M-1],N,1);             % random message bits
```

```
modulated = step(mod,x);            % modulate message to PSK symbols

% Alamouti space-time block coding (STBC)
% s0: symbols from both Tx antenna 0 and 1 at time t
% s1: symbols from both Tx antenna 0 and 1 at time t+T
% ... where T = symbol duration
% ======================================
% (example) code s0 and s1:
%           antenna 0       antenna 1
% time t       s0              s1
% time t+T    -s1*             s0*
% (next iter)  ...             ...

oddIdx = 1:2:N;  % odd indices (MATLAB odd indexing; otherwise even)
evenIdx = 2:2:N; % even indices (MATLAB odd indexing; otherwise even)
s0 = modulated(oddIdx);     % Tx symbols from antenna 0
s1 = modulated(evenIdx);    % Tx symbols from antenna 1

coded_syms = zeros(N,2);    % coded_syms: [N x 2] dimension
coded_syms(oddIdx, 1:2) = sqrt(1/2) * [s0,s1];
coded_syms(evenIdx, 1:2) = sqrt(1/2) * [-conj(s1) , conj(s0)];

% Constant Gaussian random channel characteristics for both Tx
h = sqrt(1/2) * kron((randn(N/2,2) + 1j*randn(N/2,2)),[1,1]');

% signals from Rayleigh channel for both Tx antennas
filtered = h.*coded_syms;

% allocate memory space
combined = zeros(length(modulated),length(snr));
decoded = zeros(length(modulated),length(snr));
demodulated = zeros(length(modulated),length(snr));
for i=1:length(snr)
    % pass through an AWGN channel
    n_filtered = awgn(filtered,snr(i),'measured');
    % combine symbols at Rx antenna
    combined(:,i) = sum(n_filtered,2);
    % separate into two r symbols by each Tx antenna
    r0 = combined(oddIdx,i);
    r1 = combined(evenIdx,i);

    % Rx has perfect knowledge of channel characteristics
    h0_rx = h(oddIdx,1);
    h1_rx = h(oddIdx,2);

    % maximum ratio combining technique (max likelihood)
    h_rx = zeros(N,2);
    h_rx(oddIdx,:) = [conj(h0_rx), h1_rx];
    h_rx(evenIdx,:) = [conj(h1_rx), -h0_rx];
    hHh = h_rx.*conj(h_rx);

    % s_hat: Equation (12) from Alamouti paper
    s_hat = zeros([N,1]);
    s_hat(oddIdx) = (conj(h0_rx) .* r0) + (h1_rx .* conj(r1));
    s_hat(evenIdx) = (conj(h1_rx) .* r0) - (h0_rx .* conj(r1));

    decoded(:,i) = sum(s_hat,2)./sum(hHh,2);

    % demodulate PSK symbols
    demodulated(:,i) = step(demod,decoded(:,i));
end
```

```
    % compute Bit Error Rate
    [~,bers(:,ii)] = biterr(demodulated,x);
end
ber4 = mean(bers,2);
disp('[INFO] Alamouti 2 Tx, 1 Rx complete')
```

```
[INFO] Alamouti 2 Tx, 1 Rx in simulation        .....................................
..........................................................[INFO] Alamouti 2 Tx, 1 Rx compl
ete
```

(5) BPSK Tx (flat-fading rayleigh channel), Alamouti (2 Tx, 2 Rx)

```
fprintf('[INFO] Alamouti 2 Tx, 2 Rx in simulation\t')
numRxAntennas = 2;
bers = zeros([length(snr),nIter]);
% iterate nIter times
for ii = 1:nIter
    fprintf('.')
    x = randi([0 M-1],N,1);             % random message bits
    modulated = step(mod,x);            % modulate message to PSK symbols

    % Alamouti space-time block coding (STBC)
    % s0: symbols from both Tx antenna 0 and 1 at time t
    % s1: symbols from both Tx antenna 0 and 1 at time t+T
    % ... where T = symbol duration
    % ======================================
    % (example) code s0 and s1:
    %           antenna 0       antenna 1
    % time t       s0              s1
    % time t+T    -s1*             s0*
    % (next iter)  ...             ...

    oddIdx = 1:2:N;  % odd indices (MATLAB odd indexing; otherwise even)
    evenIdx = 2:2:N; % even indices (MATLAB odd indexing; otherwise even)
    s0 = modulated(oddIdx);     % Tx symbols from antenna 0
    s1 = modulated(evenIdx);    % Tx symbols from antenna 1

    coded_syms = zeros(N,2*numRxAntennas); % coded_syms: [N x (2*numRxAntennas)] dimension
    coded_syms(oddIdx, 1:4) = sqrt(1/2) * [s0,s1,s0,s1];
    coded_syms(evenIdx, 1:4) = sqrt(1/2) * [-conj(s1),conj(s0),-conj(s1),conj(s0)];

    % Constant Gaussian random channel characteristics for both Tx
    h = sqrt(1/2)*kron( randn(N/2,2*numRxAntennas) + 1j*randn(N/2,2*numRxAntennas),...
                    [1,1]' );
    % signals from Rayleigh channel for both Tx antennas
    filtered = h.*coded_syms;

    % allocate memory space
    combined = zeros(length(modulated),numRxAntennas,length(snr));
    decoded = zeros(length(modulated),length(snr));
    demodulated = zeros(length(modulated),length(snr));
    for i=1:length(snr)
       % pass through an AWGN channel
       n_filtered = awgn(filtered,snr(i),'measured');
       % combine symbols at Rx antenna
       combined(:,:,i) = [sum(n_filtered(:,1:2),numRxAntennas), ...
                          sum(n_filtered(:,3:4),numRxAntennas)];
```

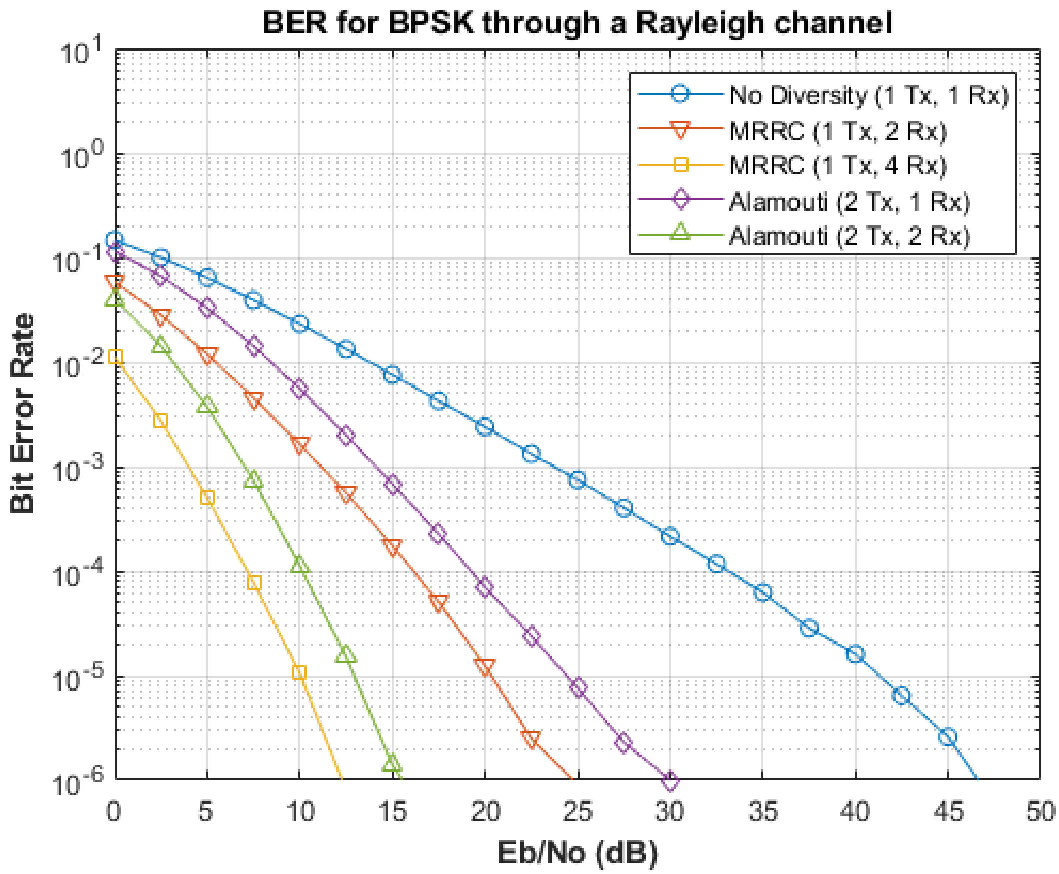
```
        % separate into two r symbols by each Tx antenna
        r0_1 = combined(oddIdx,1,i);
        r1_1 = combined(evenIdx, 1, i);
        r0_2 = combined(oddIdx, 2, i);
        r1_2 = combined(evenIdx, 2, i);

        % constant received symbols at each Rx antenna, so replicate for
        % each Rx antenna pair usin 'kron' function
        r0_1 = kron(r0_1,[1,1]');
        r1_1 = kron(conj(r1_1),[1,1]');
        r0_2 = kron(r0_2, [1,1]');
        r1_2 = kron(conj(r1_2),[1,1]');
        r = [r0_1, r1_1, r0_2, r1_2]; % dimension: [N x (2*numRxAntennas)]

        % Rx has perfect knowledge of channel characteristics
        h_rx = zeros(N,2*numRxAntennas);
        h0_1 = h(oddIdx,1);
        h1_1 = h(oddIdx,2);
        h0_2 = h(oddIdx,3);
        h1_2 = h(oddIdx,4);

        % maximum ratio combining technique (max likelihood)
        h_rx(oddIdx,:) = [conj(h0_1), h1_1, conj(h0_2), h1_2];
        h_rx(evenIdx,:) = [conj(h1_1), -h0_1, conj(h1_2), -h0_2];
        hHh = h_rx.*conj(h_rx);

        % s_hat: Equation (12) from Alamouti paper
        s_hat = h_rx .* r;

        decoded(:,i) = sum(s_hat,2)./sum(hHh,2);

        % demodulate PSK symbols
        demodulated(:,i) = step(demod,decoded(:,i));
    end

    % compute Bit Error Rate
    [~,bers(:,ii)] = biterr(demodulated, x);
end
ber5 = mean(bers,2);
disp('[INFO] Alamouti 2 Tx, 2 Rx complete')
```

```
[INFO] Alamouti 2 Tx, 2 Rx in simulation        ..........................................
..........................................................[INFO] Alamouti 2 Tx, 2 Rx compl
ete
```

Simulation Results (BER curve)

```
figure('Name','BER for BPSK through a Rayleigh channel');
semilogy(EbNo,ber1,'-o',EbNo,ber2,'-v',               ...
            EbNo,ber3,'-s',EbNo,ber4,'-d',            ...
            EbNo,ber5,'-^'                            ...
        );
title('BER for BPSK through a Rayleigh channel', ...
        'FontWeight','bold','FontSize',11);
grid on;
xlabel('Eb/No (dB)','FontWeight','bold');
ylabel('Bit Error Rate','FontWeight','bold');
```

```
xlim([0 50]); ylim([10e-7 10e0]);
legend('No Diversity (1 Tx, 1 Rx)','MRRC (1 Tx, 2 Rx)',...
       'MRRC (1 Tx, 4 Rx)', 'Alamouti (2 Tx, 1 Rx)',...
       'Alamouti (2 Tx, 2 Rx)');
```



Appendix: Rayleigh channel generation

```
function [channel_statistics] = newRayleighChan(N, fm)

    % [channel_statistics] = newRayleighChan(N, fm)
    % [Usage]
    %   newRayleighChan creates channel coefficients that mimicks a
    %   flat-flading Rayleigh channel from inputs 'N' (number of sample
    %   points) and 'fm' (maximum Doppler shift frequency).
    %   N should typically be a power of two.
    % >> procedure reference: Rappaport textbook Ch 5: pg 222

    N = N/2;                % compensate for IFFT for 2*N sample points
    % df = 2*fm / (N-1); % frequency spacing between adjacent spectral lines
    % T = 1/df;             % time duration of a fading waveform

    % column vectors
    randnoise1_pos = randn([N/2,1]) + 1j*randn([N/2,1]);
    randnoise1_neg = flipud(conj(randnoise1_pos));
    randnoise1 = [randnoise1_neg; randnoise1_pos];

    randnoise2_pos = randn([N/2,1]) + 1j*randn([N/2,1]);
    randdnoise2_neg = flipud(conj(randnoise2_pos));
    randnoise2 = [randdnoise2_neg; randnoise2_pos];

    f = linspace(-fm, fm, N);    % freq span vector
    fc = 0;                      % carrier frequency at 0 Hz
    SEz = 1.5 ./ ( pi*fm*sqrt( 1-( (f-fc)/fm ).^2 ) );
```

```
    % adjustment for 'Inf' at boundary of this Doppler Spectrum
    SEz(1) = 0;
    SEz(end) = 0;

    sqrtSEz = sqrt(SEz);
    randnoise1 = sqrtSEz .* randnoise1.';
    randnoise2 = sqrtSEz .* randnoise2.';

    time_randnoise1 = ifft(randnoise1, 2*N);
    time_randnoise2 = ifft(randnoise2, 2*N);

    % add the squares of each signal point in time to create an N-point time series
    sum = (time_randnoise1).^2 + (time_randnoise2).^2;
    channel_statistics = sqrt(sum).';

    % uncomment for Doppler Spectrum plot
    % figure;
    % plot(f,sqrtSEz);
    % title(sprintf('Doppler Spectrum for f_m=%d Hz, f_c = %d Hz',fm,fc));
    % xlabel('Frequency (Hz)','FontWeight','bold','FontSize',12);
    % ylabel('S_E_Z (f)','FontWeight','bold','FontSize',12);
end
```

Published with MATLAB® R2019b