# Part 5: Channel Estimation & Equalization

## Table of Contents

# Parameters

```
FFT_size = 4096;
CP_length = 288;
CP_OFDM_length = FFT_size+CP_length;
num_sc = 240;
SNR = -10:2:10;
num_MC = 1000;

QAM_mod = 4;
c_init = 120897;

h = [1 0.5]';

pilot_subcarriers = 1 + 4 * (0:59) + 1;
data_subcarriers = 1:num_sc;
```

# PBCH Pilot

```
% OFDM Modulation
PBCH_pilot_stream = generate_PBCH_pilot(c_init);
% Map symbol to subcarrier
d_PBCH_pilot = zeros(FFT_size,1);
k = 0:59;
d_PBCH_pilot(1+4*k+1) = PBCH_pilot_stream;
% FFT
OFDM_PBCH_pilot_body = ifft(d_PBCH_pilot)*sqrt(FFT_size);
% Add CP
CP_OFDM_PBCH_pilot = [OFDM_PBCH_pilot_body(end-
CP_length+1:end);OFDM_PBCH_pilot_body];
```

# PBCH Data

```
% OFDM Modulation
PBCH_data_stream = generate_PBCH_data(num_sc, QAM_mod);
% Map symbol to subcarrier
d_PBCH_data = [PBCH_data_stream;zeros(FFT_size-num_sc,1)];
% FFT
OFDM_PBCH_data_body = ifft(d_PBCH_data)*sqrt(FFT_size);
% Add CP
CP_OFDM_PBCH_data = [OFDM_PBCH_data_body(end-
CP_length+1:end);OFDM_PBCH_data_body];
```

# Concatenation

```
CP_OFDM_chain = [CP_OFDM_PBCH_pilot; CP_OFDM_PBCH_data];
```

# Channel without Noise

```
received_signal = conv(CP_OFDM_chain,h);
% OFDM Demodulation
received_CP_OFDM_chain = received_signal(1:CP_OFDM_length*2);
% Remove CP
received_OFDM_pilot_body =
received_CP_OFDM_chain(CP_length+1:CP_OFDM_length);
% FFT
received_pilot = fft(received_OFDM_pilot_body);
% Actual H (actual channel)
H_actual = received_pilot(pilot_subcarriers) ./ PBCH_pilot_stream;

NMSE = zeros(num_MC, length(SNR));
SER = zeros(num_MC, length(SNR));
SER_all = zeros(num_MC, length(SNR));

for SNR_id = 1:length(SNR)
    for MC_id = 1:num_MC

        % Noise
        received_signal_noisy = awgn(received_signal,SNR(SNR_id),'measured');
        % OFDM Demodulation
        received_CP_OFDM_chain = received_signal_noisy(1:CP_OFDM_length*2);
        % Remove CP
        received_OFDM_pilot_body =
received_CP_OFDM_chain(CP_length+1:CP_OFDM_length);
        received_OFDM_data_body =
received_CP_OFDM_chain(CP_OFDM_length+CP_length+1:end);
        % FFT
        received_pilot = fft(received_OFDM_pilot_body);
        received_data = fft(received_OFDM_data_body);
```

# Pilot Subcarriers

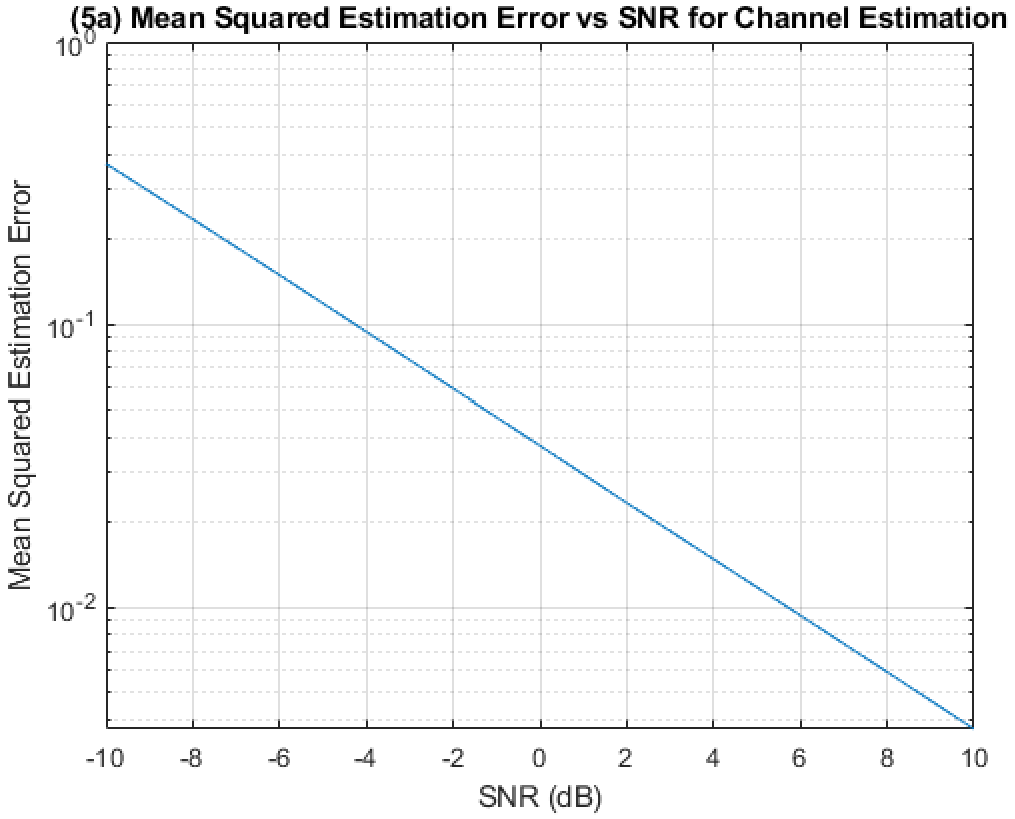
```
        % Channel Estimation
        ch_est = received_pilot(pilot_subcarriers) ./ PBCH_pilot_stream;

        % Normalized Squared Error -->
        NMSE(MC_id, SNR_id) = calculate_NMSE(H_actual, ch_est);

        % Equalization
        equalized_pilot_data = received_data(pilot_subcarriers) ./ ch_est;

        % Symbol Error Rate
        SER(MC_id, SNR_id) = calculate_SER(equalized_pilot_data,
PBCH_data_stream(pilot_subcarriers), QAM_mod);
```

# All Subcarriers

```
        % Channel Estimation
        ch_est_full = interp1(pilot_subcarriers, ch_est,
(data_subcarriers)', 'linear', 'extrap');

        % Equalization
        equalized_data = received_data(data_subcarriers) ./ ch_est_full;

        % Symbol Error Rate
        SER_all(MC_id, SNR_id) = calculate_SER(equalized_data,
PBCH_data_stream, QAM_mod);

    end
end
mean_NMSE = mean(NSE, 1);
mean_SER = mean(SER, 1);
mean_SER_all = mean(SER_all, 1);
```

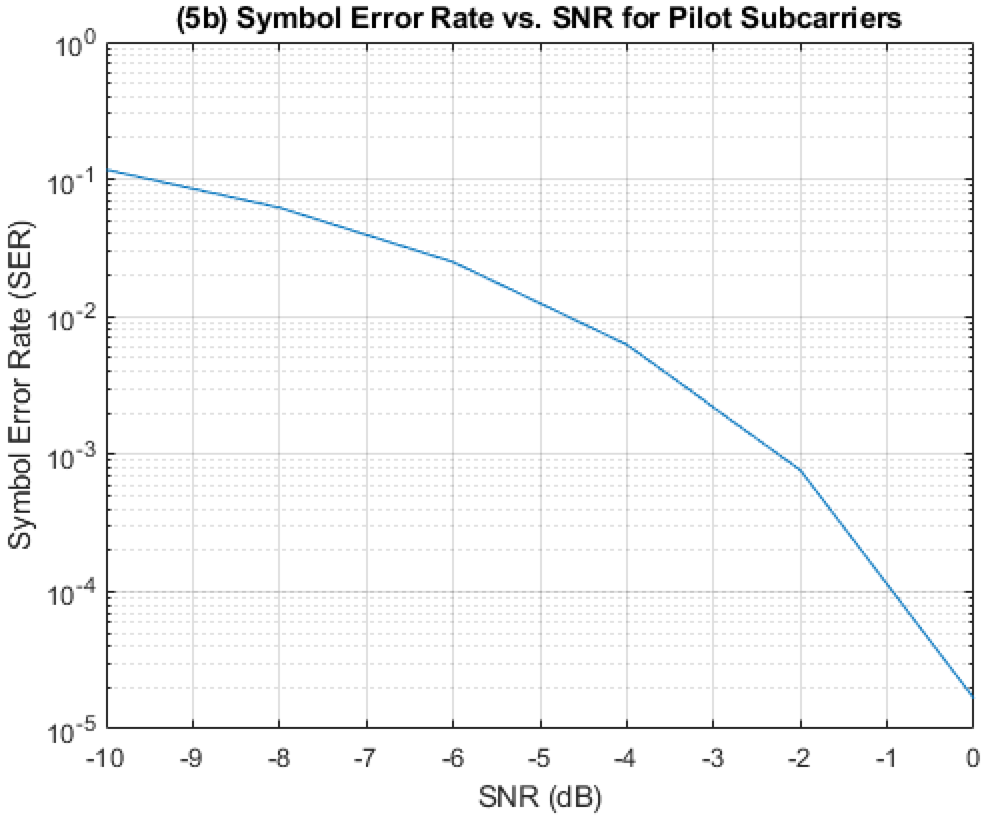
# (a) Plot Mean Squared Estimation Error

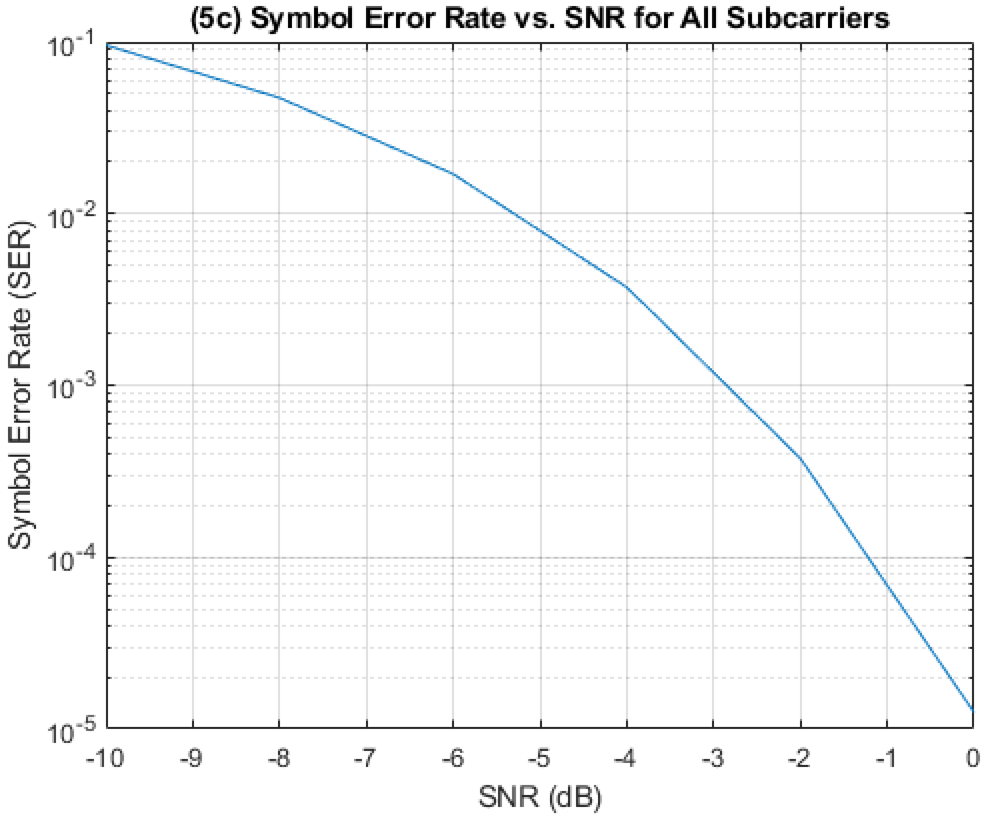
```
figure;
semilogy(SNR, mean_NMSE);
grid on;
xlabel("SNR (dB)");
ylabel("Mean Squared Estimation Error");
title("(5a) Mean Squared Estimation Error vs SNR for Channel Estimation")
```



# (b) Plot SER vs. SNR

```
figure;
semilogy(SNR, mean_SER);
grid on;
xlabel('SNR (dB)');
ylabel('Symbol Error Rate (SER)');
title('(5b) Symbol Error Rate vs. SNR for Pilot Subcarriers');

% (c) Plot SER vs. SNR
figure;
semilogy(SNR, mean_SER_all);
grid on;
xlabel('SNR (dB)');
ylabel('Symbol Error Rate (SER)');
title('(5c) Symbol Error Rate vs. SNR for All Subcarriers');

function QPSK_pilot_stream = generate_PBCH_pilot(c_init)
    c = zeros(120,1);
    QPSK_pilot_stream = zeros(60,1);
    x_1 = zeros(1800,1);
    x_2 = zeros(1800,1);
    x_1_init = [1; zeros(30,1)];
    x_1(1:31) = x_1_init;
    x_2_init = zeros(31,1);
    x_2_init_char = dec2bin(c_init);
    for i = 1:length(x_2_init_char)
        x_2_init(length(x_2_init_char)-i+1) = str2double(x_2_init_char(i));
    end
    x_2(1:31) = x_2_init;
    for n = 1:1800
        x_1(n+31) = mod(x_1(n+3)+x_1(n),2);
        x_2(n+31) = mod(x_2(n+3)+x_2(n+2)+x_2(n+1)+x_2(n),2);
    end
    for n = 0:119
        c(n+1) = mod(x_1(n+1600+1)+x_2(n+1600+1),2);
    end
    for n = 0:59
        QPSK_pilot_stream(n+1) = 1/sqrt(2)*(1-2*c(2*n+1)) + 1j/
sqrt(2)*(1-2*c(2*n+2));
    end
end

function QPSK_data_stream = generate_PBCH_data(num_sc, QAM_mod)
    data_bit_stream = randi([0 1],num_sc*log2(QAM_mod),1);
    QPSK_data_stream =
qammod(data_bit_stream,QAM_mod,InputType='bit',UnitAveragePower=true);
end

function nsme = calculate_NMSE(H_actual, H_estimated)
    % Calculate the squared error between the actual and estimated channel
responses
    error = H_estimated - H_actual;
    squared_error = abs(error).^2;

    % Calculate the true power of the actual channel response
    squared_actual = abs(H_actual).^2;

    % Normalize the squared error by the true power --> TODO: Don't need to
do this??
    nsme = mean(squared_error ./ squared_actual);
end

function ser = calculate_SER(equalized_data, true_data, QAM_mod)
    % QPSK demodulation for both equalized and true data
    estimated_symbols = qamdemod(equalized_data, QAM_mod, 'OutputType',
'bit', 'UnitAveragePower', true);
    true_symbols = qamdemod(true_data, QAM_mod, 'OutputType', 'bit',
'UnitAveragePower', true);

    % Calculate SER
    ser = sum(estimated_symbols ~= true_symbols) / length(true_symbols);
end
```


(5b) Symbol Error Rate vs. SNR for Pilot Subcarriers
Symbol Error Rate (SER)
$10^{0}$
$10^{-1}$
$10^{-2}$
$10^{-3}$
$10^{-4}$
$10^{-5}$
-10
-9
-8
-7
-6
-5
-4
-3
-2
-1
0
SNR (dB)


(5c) Symbol Error Rate vs. SNR for All Subcarriers
Symbol Error Rate (SER)
$10^{-1}$
$10^{-2}$
$10^{-3}$
$10^{-4}$
$10^{-5}$
-10
-9
-8
-7
-6
-5
-4
-3
-2
-1
0
SNR (dB)

*Published with MATLAB® R2024a*