Part 2: Frequency offset compensation & PSS search

Table of Contents

Parameters

```
FFT_size = 4096;
CP_length = 288;
SCS = 30e3;
Ts = 1/FFT_size/SCS;
CP_OFDM_length = FFT_size+CP_length;
num_sc = 240; % number of subcarriers
N_id_2 = 2
delta_f = 2.03e3; % kHz
```

```
N_id_2 =

     2
```

PSS

```
% OFDM Modulation
PSS_stream = PSS_BPSK(N_id_2);
% Map symbol to subcarrier
d_PSS = [zeros(56,1);PSS_stream;zeros(FFT_size-183,1)];
% FFT
OFDM_PSS_body = ifft(d_PSS)*sqrt(FFT_size);
% Add CP
CP_OFDM_PSS = [OFDM_PSS_body(end-CP_length+1:end);OFDM_PSS_body];
```

Channel and Noise

```
h = 1;
signal_after_channel = conv(CP_OFDM_PSS,h);

SNR = 20;
N_0 = 10^(-SNR/10) * (norm(signal_after_channel)^2/
length(signal_after_channel));
noise = sqrt(N_0/2)*(randn(length(signal_after_channel),1) +
1j*randn(length(signal_after_channel),1));
received_PSS_signal = signal_after_channel .*
exp(1j*2*pi*delta_f*(0:length(signal_after_channel)-1)'*Ts) + noise;
```

Coarse frequency offset estimation

```
% Searches from -15kHz to 15kHz with 100 Hz incremenets
freq_offset = -SCS/2:100:SCS/2;

% Correlation results between received signal and reference signals for each
value of NiD2: 3 row matrix (0-2) for NID2 corresponding frequency offsets
corr = zeros(3,length(freq_offset));

for i = 0:2 % iterates through each NID2 value
    PSS_ref_stream = PSS_BPSK(i); % reference waveform for NID2 value
    d_PSS_ref = [zeros(56,1);PSS_ref_stream;zeros(FFT_size-183,1)]; %
prepares for modulation
    OFDM_PSS_ref_body = ifft(d_PSS_ref); % converts to frequency domain
    CP_OFDM_PSS_ref = [OFDM_PSS_ref_body(end-
CP_length+1:end);OFDM_PSS_ref_body]; % adds CP to OFDM signal

    for j = 1:numel(freq_offset) % itereates thorugh each frequency offset
value to compute correlation for different frequency offsets
        % Calculates correlation between received signal and reference
signal with frequency offset
        corr(i+1,j) = abs(received_PSS_signal' * (CP_OFDM_PSS_ref .*
exp(1j*2*pi*freq_offset(j)*(0:length(CP_OFDM_PSS_ref)-1)'*Ts)));
    end
end
% Finds position of maximum correlation value
[~,N_id_2_est_pos] = max(max(abs(corr),[],2));
% converts index position to actual NID2 value
N_id_2_est = N_id_2_est_pos - 1;
```

Fine Frequency Offset Estimation

```
freq_offset_est = -angle(received_PSS_signal(FFT_size+1:FFT_size+CP_length)'
* received_PSS_signal(1:CP_length))/2/pi*SCS;
freq_offset_est_kHz = freq_offset_est / 1e3;
```

(a) Determine (NID2, Frequency offset estimation)

```
disp('(2a) Determine N(2)_ID and Estimated Frequency Offset: ');
disp(['Estimated N(2)_ID: ', num2str(N_id_2_est)]); % from course frequency
offset estimation
disp(['Estimated Frequency Offset (Hz): ', num2str(freq_offset_est)]); %
from fine frequency offset estimation
disp(['Estimated Frequency Offset (kHz): ', num2str(freq_offset_est_kHz)]);
```

```
(2a) Determine N(2)_ID and Estimated Frequency Offset:
Estimated N(2)_ID: 2
Estimated Frequency Offset (Hz): 2043.1866
Estimated Frequency Offset (kHz): 2.0432
```

Calculate root mean square

```
FFT_size = 4096;
CP_length = 288;
SCS = 30e3;
Ts = 1/FFT_size/SCS;
CP_OFDM_length = FFT_size + CP_length;
num_sc = 240; % number of subcarriers
N_id_2 = 2;
delta_f = 2.03e3; % kHz
num_MC = 1000;

SNR_range = 0:2:20; % SNR values in dB
num_SNR = length(SNR_range);
RMSE = zeros(num_MC, num_SNR); % Root Mean Square Error

for SNR_id = 1:num_SNR
    for MC_id = 1:num_MC

        SNR = SNR_range(SNR_id);
        N_0 = 10^(-SNR/10) * (norm(signal_after_channel)^2/
length(signal_after_channel));
        noise = sqrt(N_0/2)*(randn(length(signal_after_channel),1) +
1j*randn(length(signal_after_channel),1));
        received_PSS_signal = signal_after_channel .*
exp(1j*2*pi*delta_f*(0:length(signal_after_channel)-1)'*Ts) + noise;
```

Fine Frequency Offset Estimation

```
        freq_offset_est =
-angle(received_PSS_signal(FFT_size+1:FFT_size+CP_length)' *
received_PSS_signal(1:CP_length))/2/pi*SCS;
        freq_offset_est_kHz = freq_offset_est / 1e3;
```

Calculate RMSE

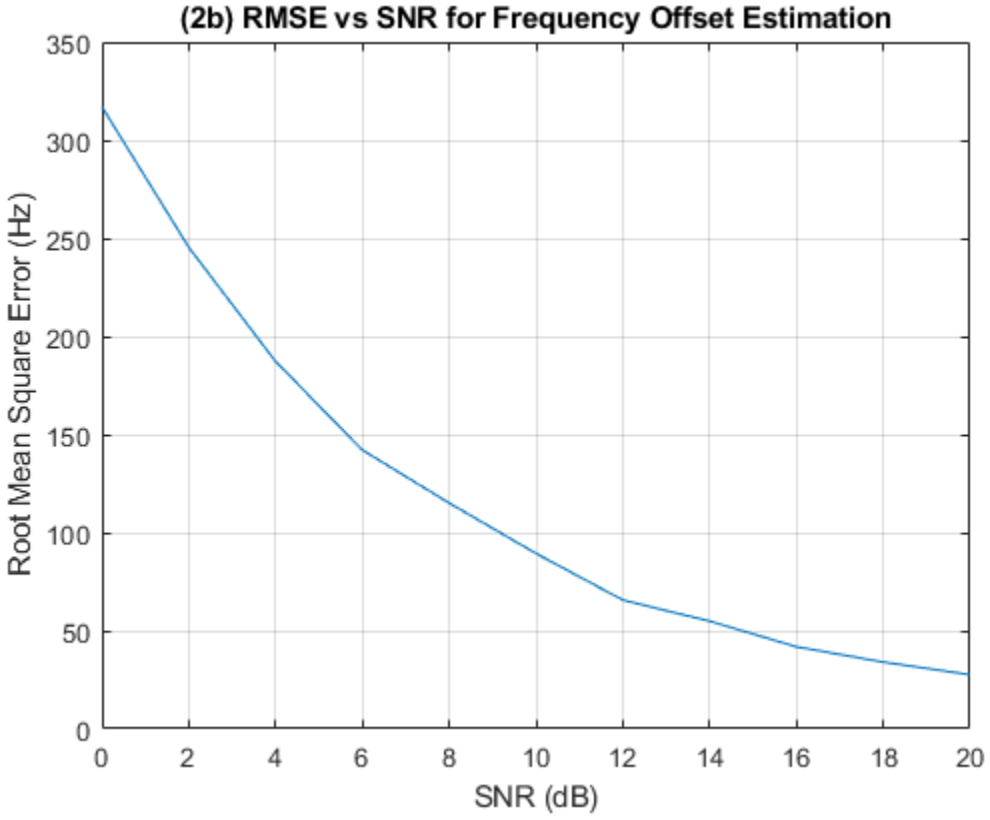
```
        RMSE(MC_id, SNR_id) = (freq_offset_est - delta_f)^2;

    end
end

mean_RMSE = sqrt(mean(RMSE, 1));
```

(b) Plot RMSE vs SNR

```
figure;
plot(SNR_range, mean_RMSE);
xlabel('SNR (dB)');
ylabel('Root Mean Square Error (Hz)');
title('(2b) RMSE vs SNR for Frequency Offset Estimation');
grid on;
```



Function Definitions

```
function BPSK_stream = PSS_BPSK(N_id_2)
    x = zeros(127,1);
    BPSK_stream = zeros(127,1);
    x_init = [0 1 1 0 1 1 1];
```

```
    x(1:7) = x_init;
    for i = 1:120
        x(i+7) = mod(x(i+4)+x(i),2);
    end
    for n = 0:126
        m = mod(n + 43*N_id_2,127);
        BPSK_stream(n+1) = 1-2*x(m+1);
    end
end
```

Published with MATLAB® R2024a