Experiment 9: Linear Phase F I R Filter Design using Frequency sampling method

Name	Prathamesh Mane
UID no. & Branch	**2022200078 (B1)**
Experiment No.	9

Code :

```
clear;
 clc;

 filter_type = input(['Select Filter Type: \n1: Low Pass Filter (LPF)' ...
 '\n2: High Pass Filter (HPF) \n3: Band Pass Filter (BPF)' ...
 '\n4: Band Stop Filter (BSF) \nEnter choice (1-4): '])
 Fs = input('Enter Sampling Frequency (Hz): ')
 if Fs <= 0
 error('Sampling Frequency (Fs) must be greater than 0.')
 end

 Ap = input('Enter Pass Band Attenuation (Ap) in dB: ')
 As = input('Enter Stop Band Attenuation (As) in dB (> 40): ')
 if As <= 40
 error('Stop Band Attenuation must be greater than 40 dB.')
 end

 switch filter_type

 case 1
 disp('Designing Low Pass Filter (LPF)');
 Fp = input('Enter Pass Band Frequency (Fp) in Hz: ')
 Fs_stop = input('Enter Stop Band Frequency (Fs) in Hz: ')
 if Fp <= 0 || Fp >= Fs/2 || Fs_stop <= 0 || Fs_stop >= Fs/2 || Fp >= Fs_stop
 error('Frequencies must satisfy: 0 < Fp < Fs_stop < Fs/2.');
 end

 case 2
 disp('Designing High Pass Filter (HPF)');
 Fp = input('Enter Pass Band Frequency (Fp) in Hz: ')
 Fs_stop = input('Enter Stop Band Frequency (Fs) in Hz: ')
 if Fp <= 0 || Fp >= Fs/2 || Fs_stop <= 0 || Fs_stop >= Fs/2 || Fs_stop >= Fp
 error('Frequencies must satisfy: 0 < Fs_stop < Fp < Fs/2.');
 end

 case 3
 disp('Designing Band Pass Filter (BPF)');
 Fp1 = input('Enter Lower Pass Band Frequency (Fp1) in Hz: ')
```

```
Fp2 = input('Enter Upper Pass Band Frequency (Fp2) in Hz: ')
Fs_stop1 = input('Enter Lower Stop Band Frequency (Fs_stop1) in Hz: ')
Fs_stop2 = input('Enter Upper Stop Band Frequency (Fs_stop2) in Hz: ')
if Fp1 <= 0 || Fp2 <= 0 || Fs_stop1 <= 0 || Fs_stop2 <= 0 || ...
Fp1 >= Fp2 || Fs_stop1 >= Fp1 || Fs_stop2 <= Fp2 || ...
Fs_stop2 >= Fs/2 || Fs_stop1 >= Fs/2
error('Frequencies must satisfy: Fs_stop1 < Fp1 < Fp2 < Fs_stop2 < Fs/2.');
end

case 4
disp('Designing Band Stop Filter (BSF)');
Fp1 = input('Enter Lower Pass Band Frequency (Fp1) in Hz: ')
Fp2 = input('Enter Upper Pass Band Frequency (Fp2) in Hz: ')
Fs_stop1 = input('Enter Lower Stop Band Frequency (Fs_stop1) in Hz: ')
Fs_stop2 = input('Enter Upper Stop Band Frequency (Fs_stop2) in Hz: ')
if Fp1 <= 0 || Fp2 <= 0 || Fs_stop1 <= 0 || Fs_stop2 <= 0 || ...
Fp1 >= Fp2 || Fs_stop1 >= Fp1 || Fs_stop2 <= Fp2 || ...
Fs_stop2 >= Fs/2 || Fs_stop1 >= Fs/2
error('Frequencies must satisfy: Fs_stop1 < Fp1 < Fp2 < Fs_stop2 < Fs/2.');
end

otherwise
error('Invalid choice.');
end

N = input('Enter filter Order (N): ')
if mod(N, 2) ~= 0
warning('Filter order N is recommended to be even for symmetric FIR filters.');
end

f = (0:N) / N;
H = zeros(size(f));

switch filter_type

   case 1
Wp = Fp / (Fs/2);
H(f <= Wp) = 10^(-Ap/20);
H(f > Wp & f < 1) = 10^(-As/20);

   case 2
Wp = Fp / (Fs/2);
H(f >= Wp) = 10^(-Ap/20);
H(f < Wp) = 10^(-As/20);

   case 3
Wp1 = Fp1 / (Fs/2);
Wp2 = Fp2 / (Fs/2);
H(f >= Wp1 & f <= Wp2) = 10^(-Ap/20);
H(f < Wp1 | f > Wp2) = 10^(-As/20);

   case 4
Wp1 = Fp1 / (Fs/2);
```

```
Wp2 = Fp2 / (Fs/2);
H(f < Wp1 | f > Wp2) = 10^(-Ap/20);
H(f >= Wp1 & f <= Wp2) = 10^(-As/20);
end

h_linear = real(ifft(H, 'symmetric'));
h_linear = h_linear(1:N+1);
theta = rand(1, N+1) * 2 * pi;

H_non_linear = H .* exp(1j * theta);
h_non_linear = real(ifft(H_non_linear, 'symmetric'));
[H_resp_linear, f_resp] = freqz(h_linear, 1, 1024, Fs);
[H_resp_non_linear, ~] = freqz(h_non_linear, 1, 1024, Fs);

subplot(2, 1, 1);
plot(f_resp, 20 * log10(abs(H_resp_linear)), 'b', 'LineWidth', 1.5);
hold on;
plot(f_resp, 20 * log10(abs(H_resp_non_linear)), 'r--', 'LineWidth', 1.5);
grid on;
title('Magnitude Spectrum');
xlabel('Frequency (Hz)');
ylabel('Magnitude (dB)');
legend('Linear Phase', 'Non-linear Phase');
ylim([-100 5]);

subplot(2, 1, 2);
plot(f_resp, angle(H_resp_linear) * 180/pi, 'b', 'LineWidth', 1.5);
hold on;
plot(f_resp, angle(H_resp_non_linear) * 180/pi, 'r--', 'LineWidth', 1.5);
grid on;
title('Phase Spectrum');
xlabel('Frequency (Hz)');
ylabel('Phase (Degrees)');
legend('Linear Phase', 'Non-linear Phase');
```