

# BRI509: Introduction to Brain Signal Processing

## Assignment No. 3

**CANOY RAYMART JAY**

Student ID #: **2020021376**

June 4, 2021

- (a) Draw the Bode diagrams of Do(C4), Mi(E4), Sol(G4) and their chord(C4+E4+G4).

### Source Code

```
1 % BRI509 (Introduction to Brain Signal Processing)
2 % Assignment # 3
3 % Author: Raymart Jay E. Canoy
4 % Student ID #: 2020021376
5
6 % Notes:
7 % "The frequencies 440 Hz and 880 Hz both correspond to the
8 % musical note A,
9 % but one octave apart. In the western musical scale, there
10 % are 12 notes in
11 % every octave. These notes are evenly distributed (
12 % geometrically), so the
13 % next note above A, which is B flat, has frequency (440 x
14 % beta), where
15 % beta is the twelfth root of two."
16 % Source: https://ptolemy.berkeley.edu/eecs20/week8/scale.html
17
18 %% Draw the Bode diagrams of:
19 % (a) Do(C4)
20 % (b) Mi(E4)
21 % (c) Sol(G4)
22 % (d) Chord(C4+E4+G4)
23
24 % (00) Initialization
25 clear; clc;
26
27 fs = 1200;
28 T = 1/fs;
29 dur = 1;
30 L = dur*fs;
31 t = (0:L-1)*T;
32
33 % (01) Sinusoid signal
34 beta = 2^(1/12);
35 beta_exponents = [-9, -7, -5, -4, -2, 0, 2, 3];
36 note = @(index) sin(2*pi*(440*(beta^(index)))*t);
37
```

```

34 % C major
35 musical_scale_keys = {'C4', 'D4', 'E4', 'F4', 'G4', 'A4', 'B4',
    , 'C5'};
36 musical_scale_values = cell(length(musical_scale_keys), 1);
37
38 for ind = 1 : length(musical_scale_keys)
39     musical_scale_values{ind} = note(beta_exponents(ind));
40 end
41
42 % (02) Sofege for C major
43 music_octave = containers.Map(musical_scale_keys,
    musical_scale_values);
44
45 %% (03) Bode Diagram
46 assignment = {'C4', 'E4', 'G4'};
47 beta_exponents_assignment = [-9, -5, -2];
48 chord = music_octave('C4') + music_octave('E4') + music_octave
    ('G4');
49
50 for i = 1 : 4
51     % Fourier Transform
52     n = 256*2^(nextpow2(L));
53     if i < 4
54         x = music_octave(assignment{i});
55     else
56         x = chord;
57     end
58     X = fftshift(fft(x, n, 2));
59
60     f = [0: fs/n : fs/2 - fs/n];
61     PdB = 20*log10(2*abs(X(end/2+1:end))./L));
62     phase = rad2deg(angle(X(end/2+1:end)));
63
64     A = [ones(size(f, 2), 1), [1:1:size(f, 2)]];
65     theta = pinv(A'*A)*(A'*f');
66
67     if i < 4
68         f_center_index = round((440*(beta^(
beta_exponents_assignment(i))) - theta(1))/theta(2));

```

```

69     else
70         f_center_index = 144017;
71     end
72     figure,
73     p1 = plot([0: fs/n : fs/2 - fs/n], 2*abs(X(end/2+1:end))./L
74 ), 'k')
75     grid on
76     xlim([0 fs/2])
77     ylim([0 1])
78     xlabel('Frequency, $f$ ($Hz$)', 'Interpreter', 'Latex', '
FontSize', 16)
79     ylabel('$|X(f)|$', 'Interpreter', 'Latex', 'FontSize', 16)
80     if i < 4
81         title(sprintf('Single-sided spectrum of %s (f = %2f Hz
)', assignment{i}, 440*(beta^(beta_exponents_assignment(i)))
82 ))
83     else
84         title('Single-sided spectrum of chord (C4+E4+G4)')
85     end
86     saveas(gcf, sprintf('FFT_Prob%02d.png', i))
87
88     figure;
89     ax1 = axes();
90     plot(ax1, f, PdB, 'k');
91     hold on
92     if i < 4
93         plot(ax1, f(f_center_index - 5000)*ones(1200, 1),
94 linspace(min(PdB), max(PdB), 1200), 'r', 'LineWidth', 1);
95         plot(ax1, f(f_center_index + 5000)*ones(1200, 1),
96 linspace(min(PdB), max(PdB), 1200), 'r', 'LineWidth', 1);
97     else
98         plot(ax1, f(f_center_index - 35000)*ones(1200, 1),
99 linspace(min(PdB), max(PdB), 1200), 'r', 'LineWidth', 1);
100        plot(ax1, f(f_center_index + 35000)*ones(1200, 1),
101 linspace(min(PdB), max(PdB), 1200), 'r', 'LineWidth', 1);
102    end
103    set(gca, 'xscale', 'log')
104    grid on
105    xlim([min(f) max(f)])

```

```

100     ylim([min(PdB) max(PdB)])
101     xlabel('\omega / 2\pi$', 'Interpreter', 'Latex', 'FontSize', 16)
102     ylabel('$|X(f)|_{dB}$', 'Interpreter', 'Latex', 'FontSize', 16)
103     if i < 4
104         title(sprintf('Bode Diagram of %s (f = %2f Hz) [Amplitude]', assignment{i}, 440*(beta^(beta_exponents_assignment(i)))))
105     else
106         title('Bode Diagram of chord (C4+E4+G4) [Amplitude]')
107     end
108     ax2 = axes('Position', [0.2 0.5 0.3 0.3]);
109     ax2.XColor = 'red';
110     ax2.YColor = 'red';
111     if i < 4
112         plot(ax2, f(f_center_index - 5000:f_center_index+5000), PdB(f_center_index - 5000:f_center_index+5000), 'k')
113     else
114         plot(ax2, f(f_center_index - 35000:f_center_index+35000), PdB(f_center_index - 35000:f_center_index+35000), 'k')
115     end
116     set(gca, 'xscale', 'log', 'XColor', 'red', 'YColor', 'red')
117     grid on
118     ylim([min(PdB) max(PdB)])
119     saveas(gcf, sprintf('Bode_diagram%02d.png', i))
120
121     figure;
122     ax1 = axes();
123     plot(ax1, f, phase, 'k');
124     hold on
125     if i < 4
126         plot(ax1, f(f_center_index - 5000)*ones(1200, 1), linspace(min(phase), max(phase), 1200), 'r', 'LineWidth', 1);
127         plot(ax1, f(f_center_index + 5000)*ones(1200, 1), linspace(min(phase), max(phase), 1200), 'r', 'LineWidth', 1);

```

```

1);
28     else
29         plot(ax1, f(f_center_index - 35000)*ones(1200, 1),
linspace(min(phase), max(phase), 1200), 'r', 'LineWidth',
1);
30         plot(ax1, f(f_center_index + 35000)*ones(1200, 1),
linspace(min(phase), max(phase), 1200), 'r', 'LineWidth',
1);
31     end
32     set(gca, 'xscale', 'log')
33     grid on
34     xlim([min(f) max(f)])
35     ylim([min(phase) max(phase)])
36     xlabel('$\omega / 2\pi$', 'Interpreter', 'Latex', '
FontSize', 16)
37     ylabel('$\angle X(f)$ (deg)', 'Interpreter', 'Latex', '
FontSize', 16)
38     if i < 4
39         title(sprintf('Bode Diagram of %s (f = %2f Hz) [Phase]
', assignment{i}, 440*(beta^(beta_exponents_assignment(i))))
)
40     else
41         title('Bode Diagram of chord (C4+E4+G4) [Phase]')
42     end
43     ax2 = axes('Position', [0.177 0.16 0.3 0.3]);
44     ax2.XColor = 'red';
45     ax2.YColor = 'red';
46     if i < 4
47         plot(ax2, f(f_center_index - 5000:f_center_index+5000)
, phase(f_center_index - 5000:f_center_index+5000), 'k')
48     else
49         plot(ax2, f(f_center_index - 35000:f_center_index
+35000), phase(f_center_index - 35000:f_center_index+35000)
, 'k')
50     end
51     set(gca, 'xscale', 'log', 'XColor', 'red', 'YColor', 'red'
)
52     grid on
53     ylim([min(phase) max(phase)])

```

```

54     saveas(gcf, sprintf('Bode_diagram_phase%02d.png', i))
55 end
56
57
58 %% (04) Audio Files
59 % (00) Initialization
60 clear; clc;
61
62 fs = 48000;
63 T = 1/fs;
64 dur = 1;
65 L = dur*fs;
66 t = (0:L-1)*T;
67
68 % (01) Sinusoid signal
69 beta = 2^(1/12);
70 beta_exponents = [-9, -7, -5, -4, -2, 0, 2, 3];
71 note = @(index) sin(2*pi*(440*(beta^(index)))*t);
72
73 % C major
74 musical_scale_keys = {'C4', 'D4', 'E4', 'F4', 'G4', 'A4', 'B4',
75                       'C5'};
76 musical_scale_values = cell(length(musical_scale_keys), 1);
77
78 for ind = 1 : length(musical_scale_keys)
79     musical_scale_values{ind} = note(beta_exponents(ind));
80 end
81
82 % (02) Sofege for C major
83 music_octave = containers.Map(musical_scale_keys,
84                               musical_scale_values);
85
86 assignment = {'C4', 'E4', 'G4'};
87 chord = music_octave('C4') + music_octave('E4') + music_octave('G4');
88
89 for i = 1 : 4
90     if i < 4
91         mp3_file = [zeros(1, 100) music_octave(assignment{i})

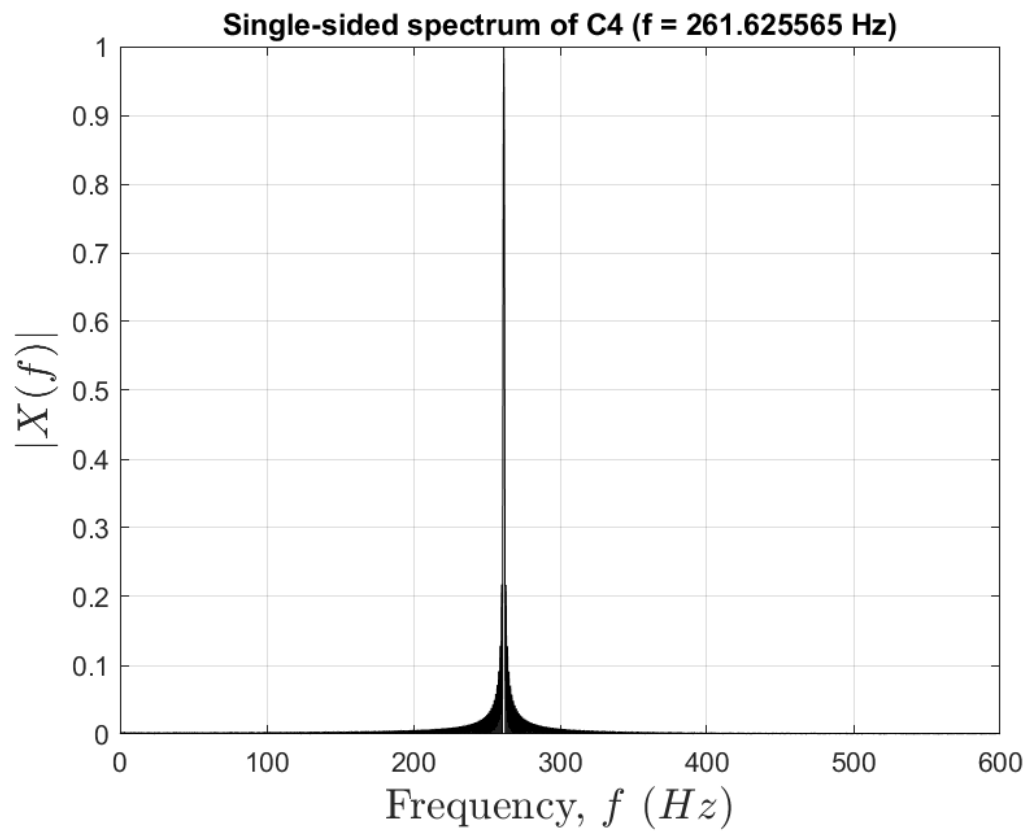
```

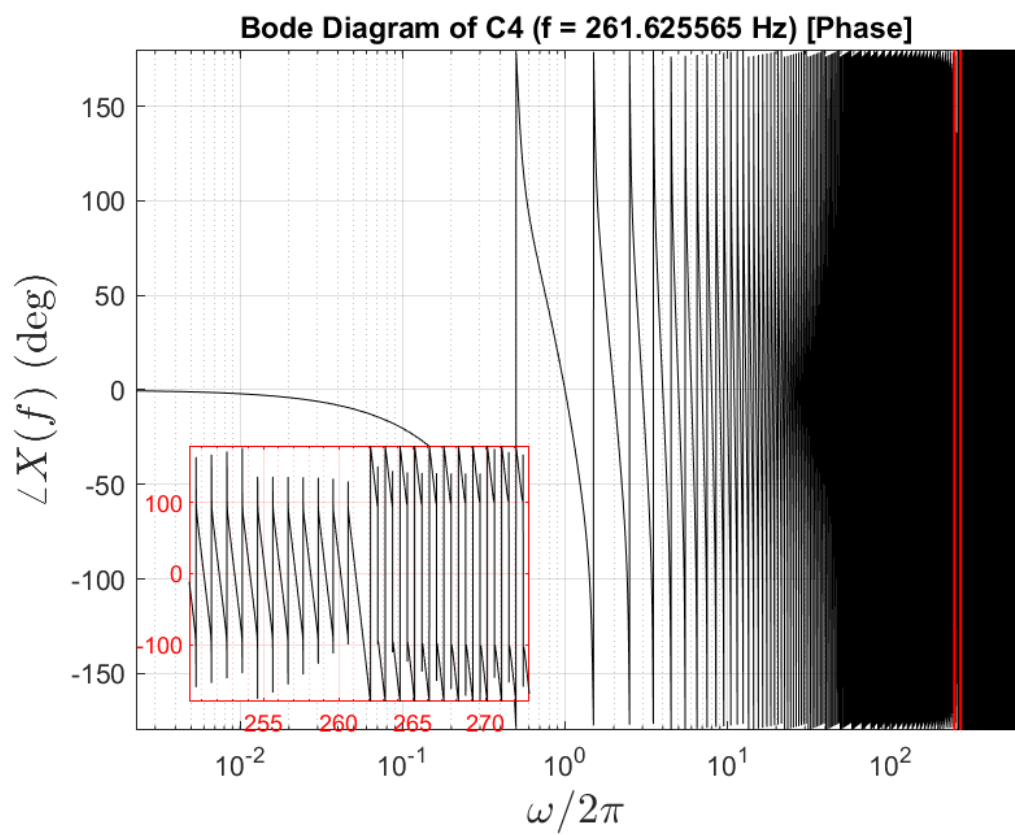
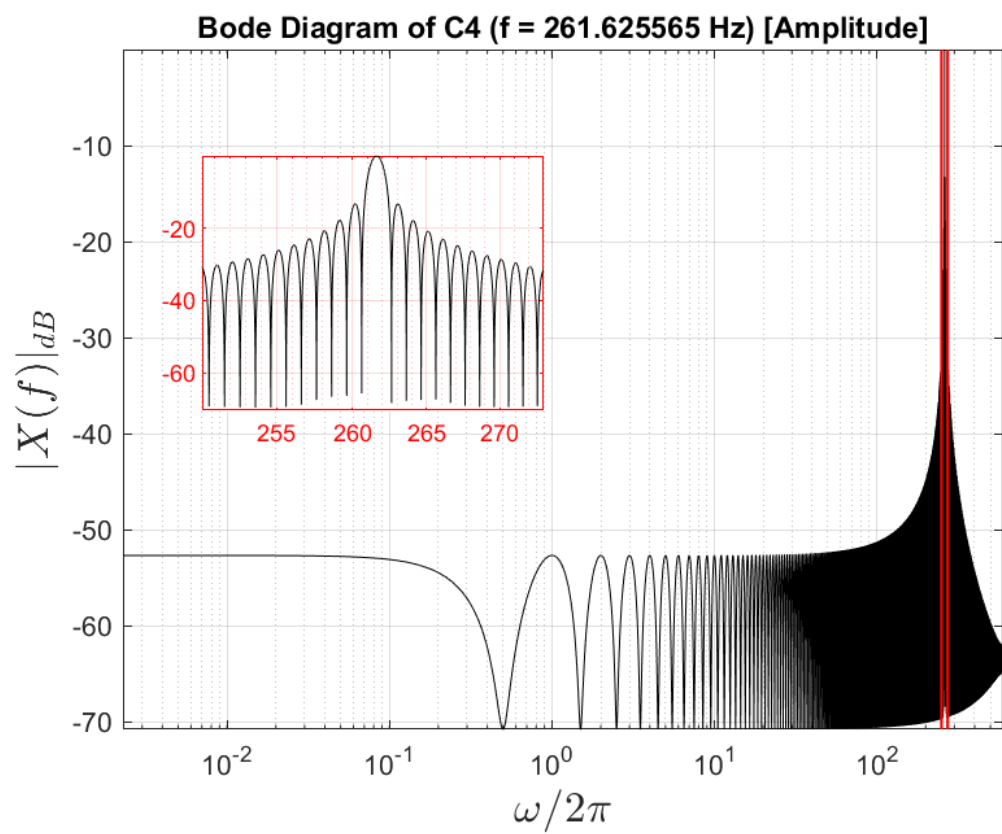
```

    zeros(1, 100)]';
    audiowrite(sprintf('
Assignment3_2020021376_CanoyRaymartJay_%s.mp4', assignment{i
}), mp3_file, fs)
    else
    mp3_file = [zeros(1, 100) chord zeros(1, 100) zeros(1,
100)]';
    audiowrite(sprintf('
Assignment3_2020021376_CanoyRaymartJay_%s.mp4', 'chord'),
mp3_file, fs)
    end
end
end

```

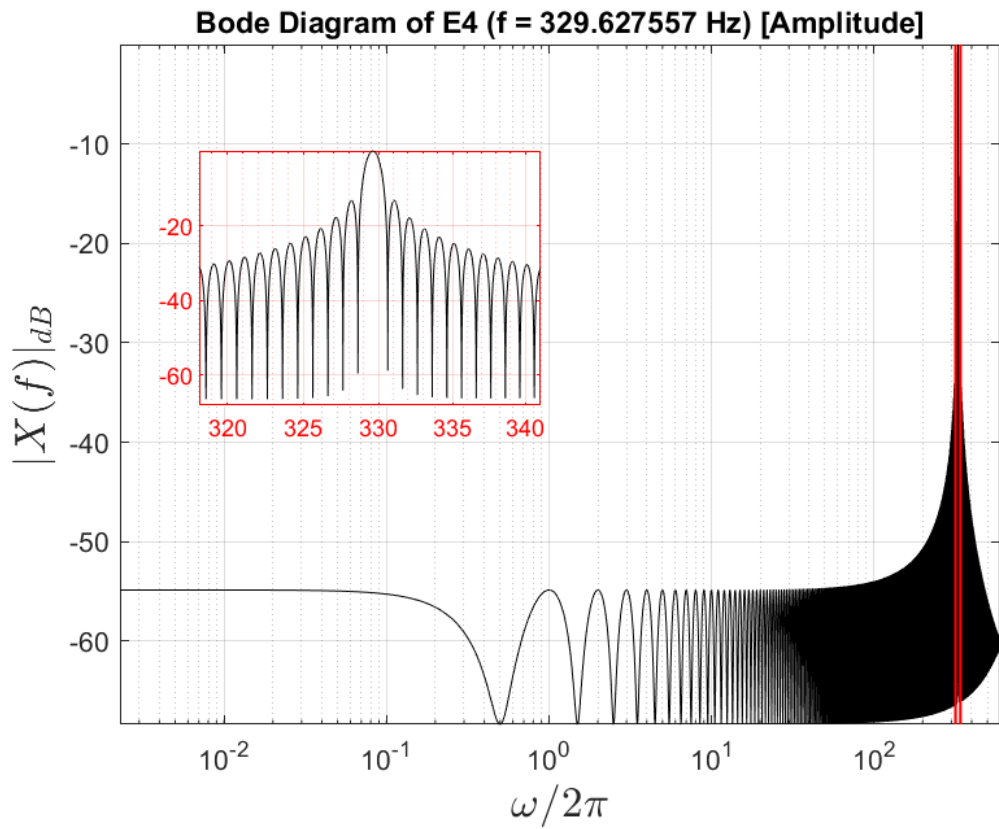
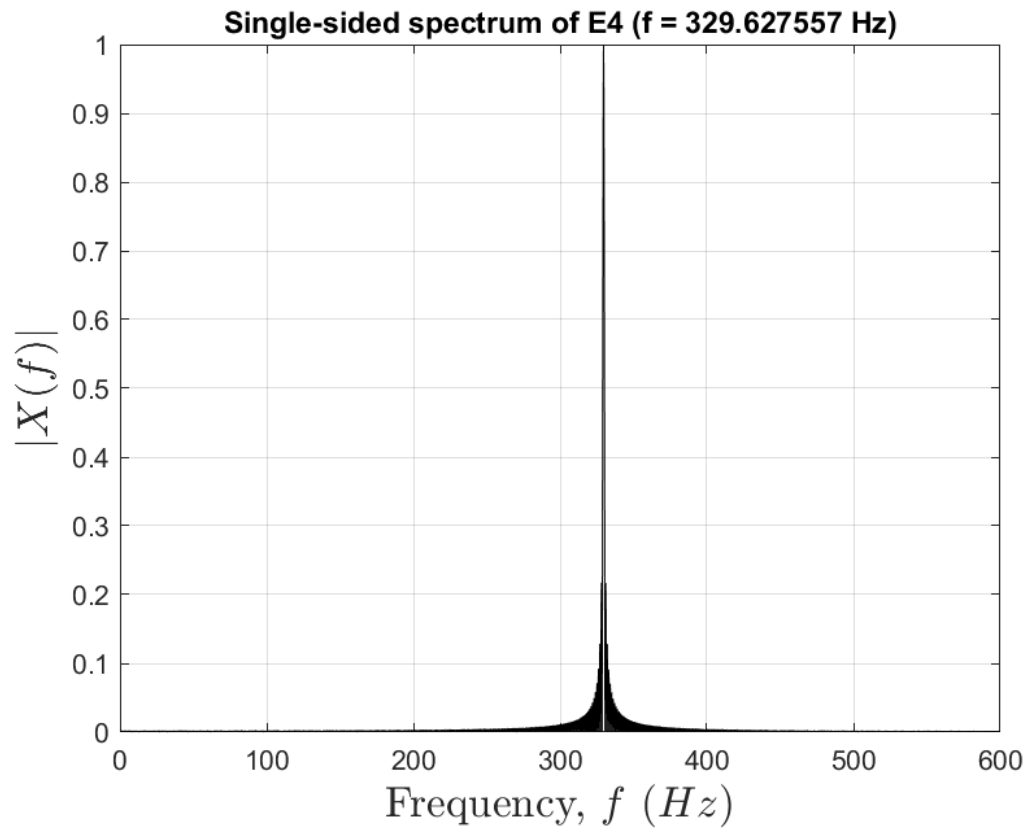
(b) Bode diagram of C4

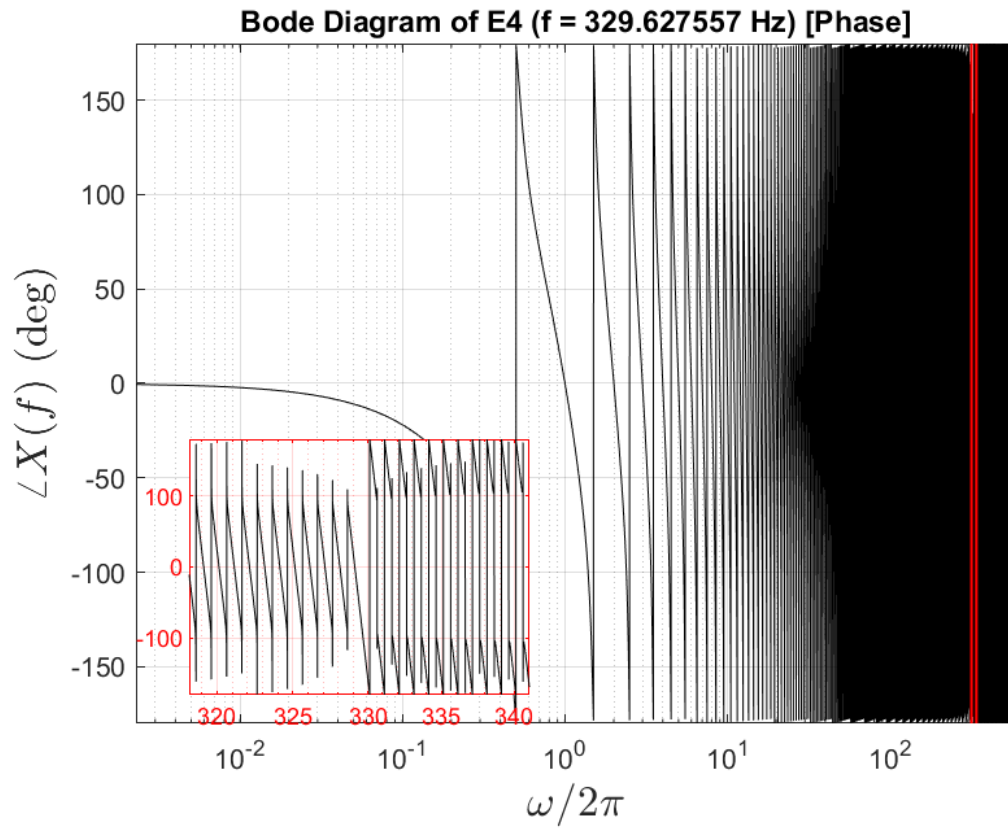




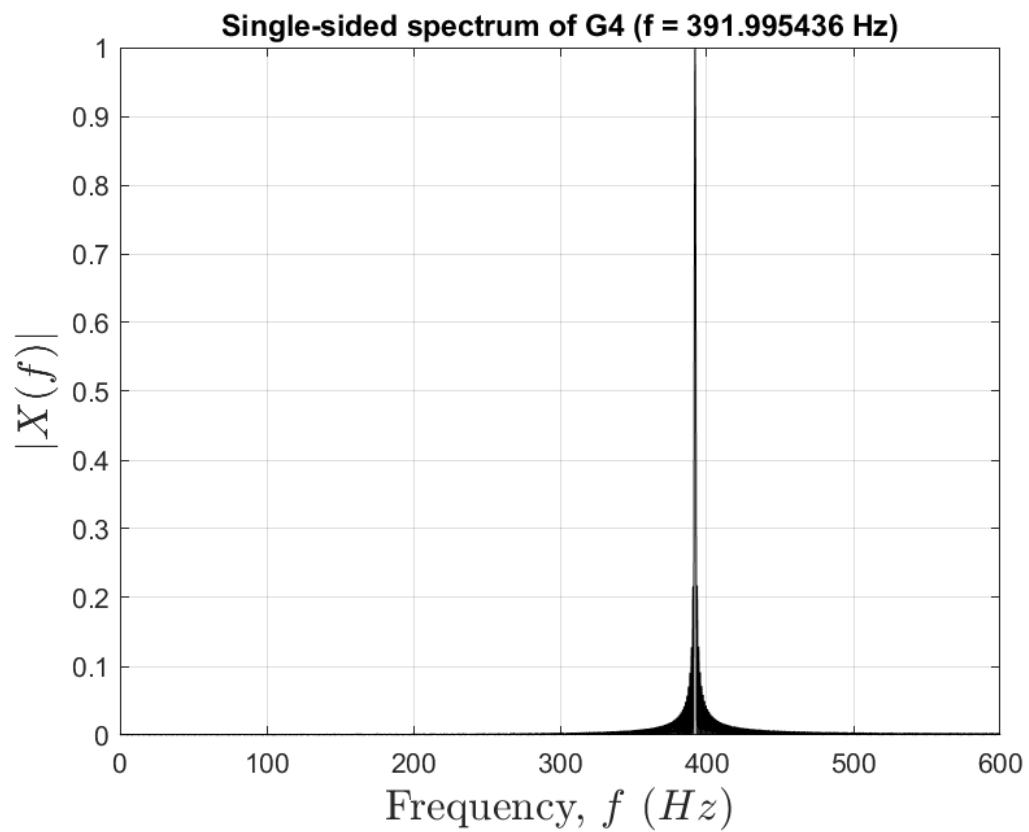


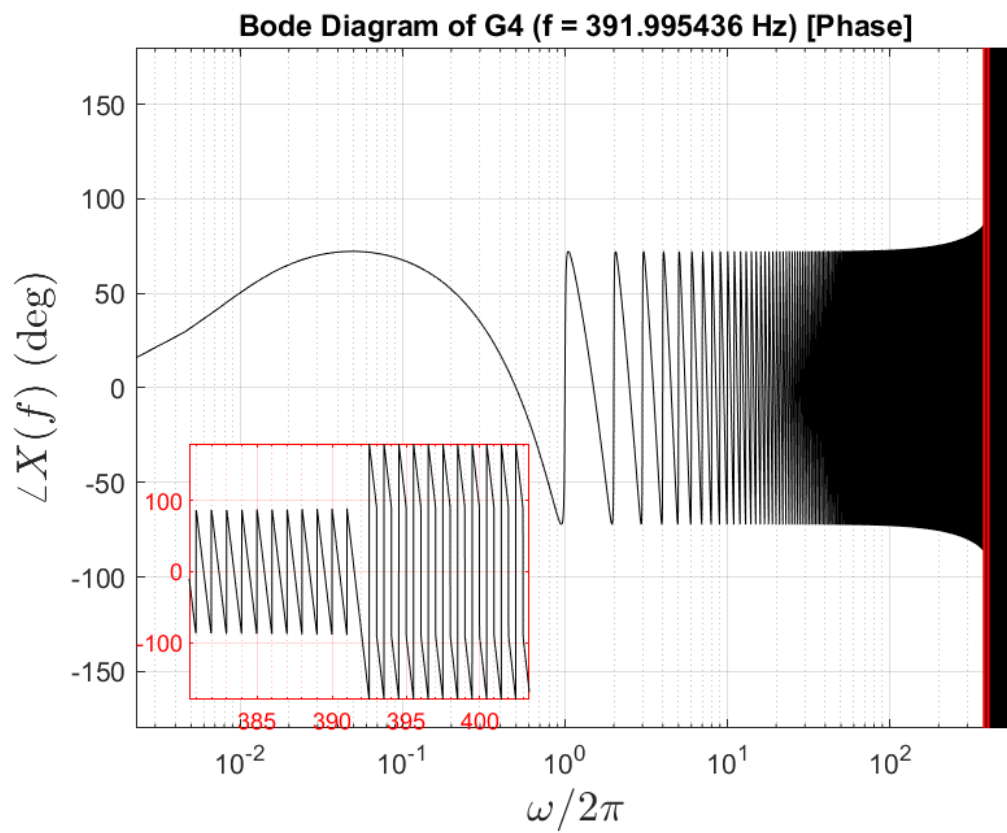
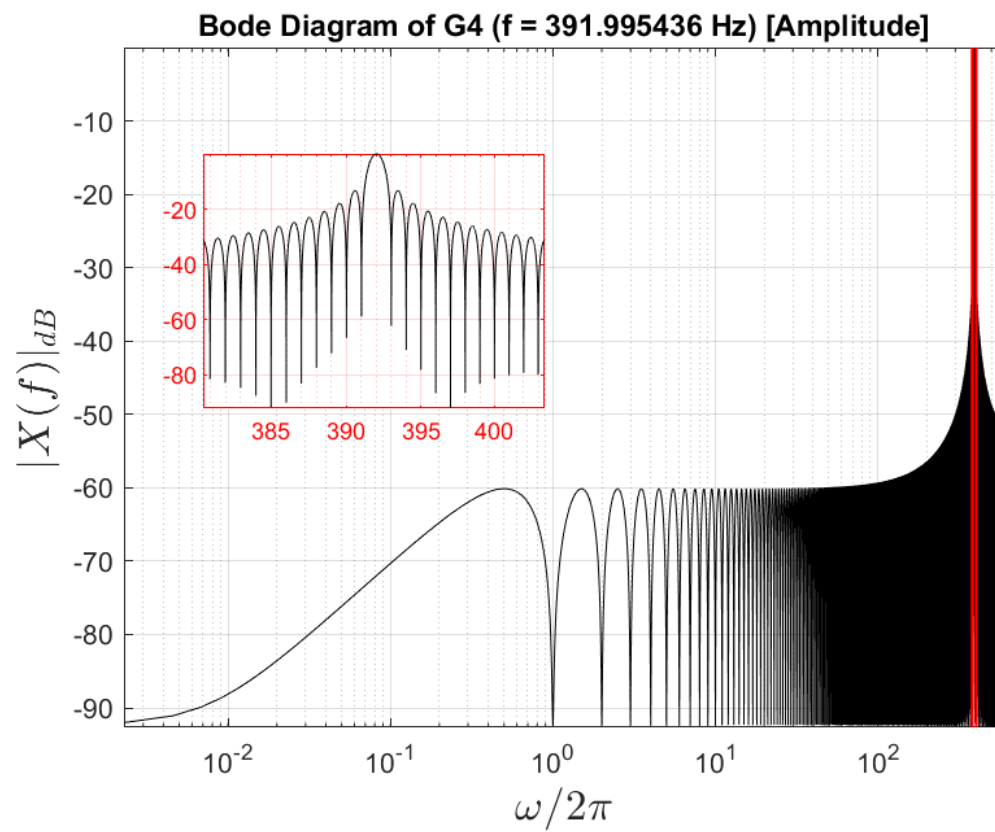
(c) Bode diagram of E4



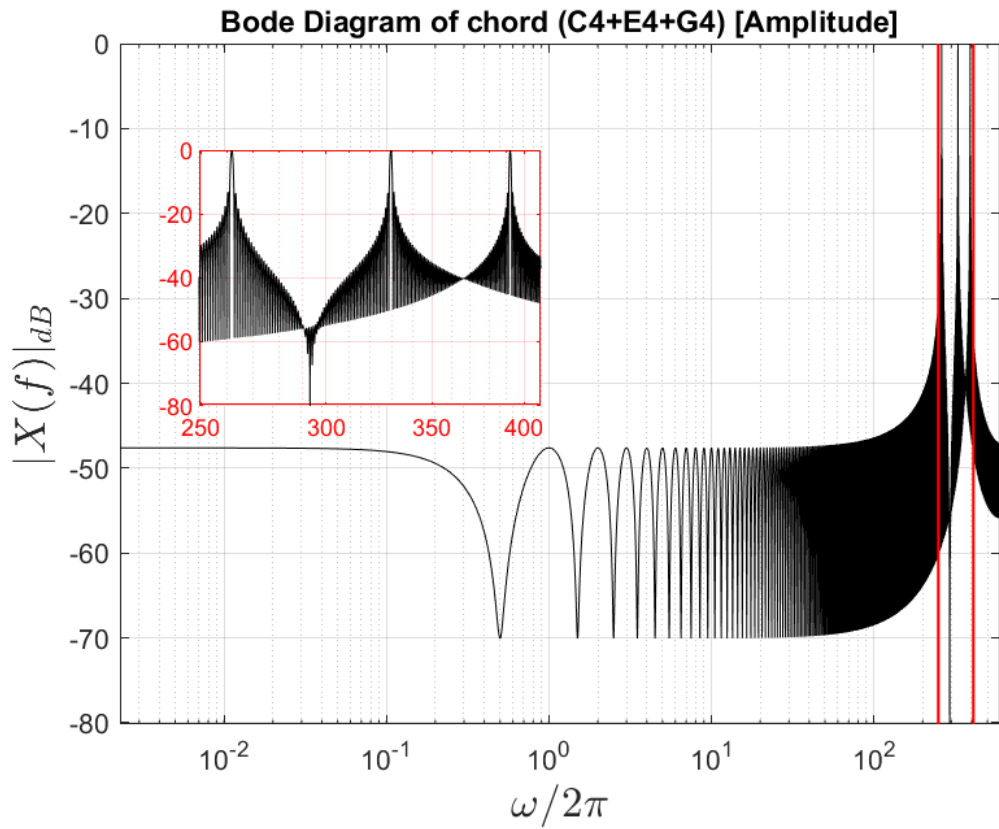
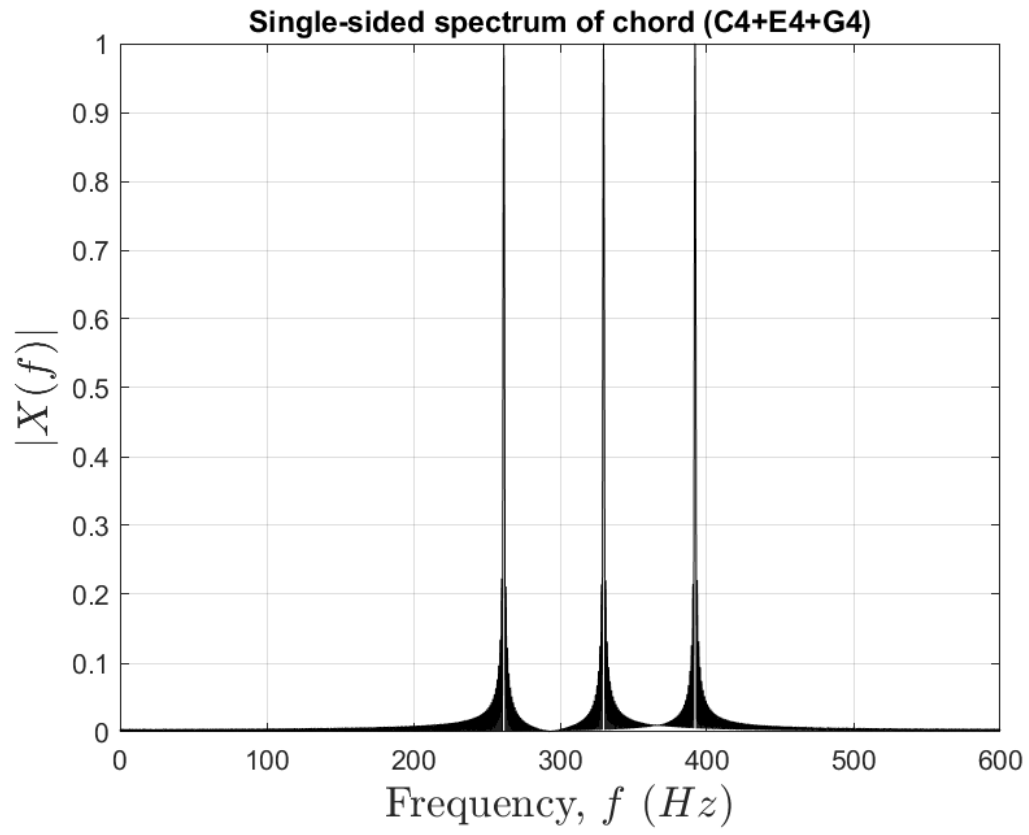


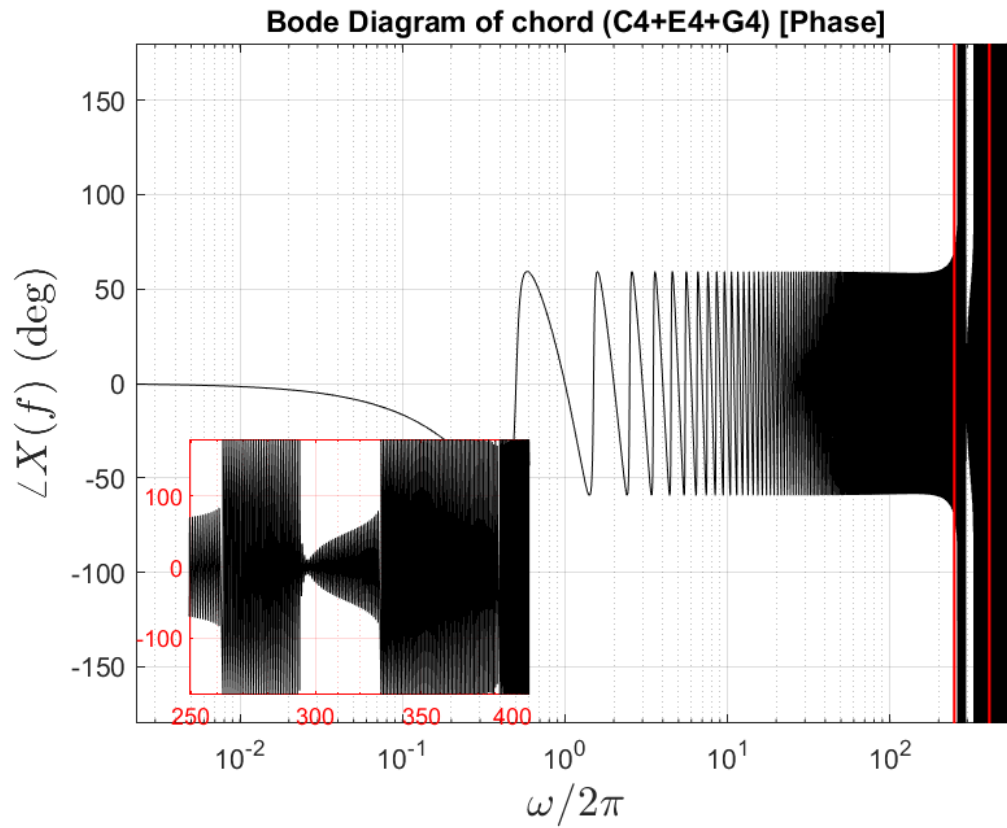
(d) Bode diagram of G4





(e) Bode diagram of C4+E4+G4





**Note: All the files were uploaded on GitHub**

All the files in this document were uploaded on Github, and can be accessed at:

<https://github.com/rjcanoy03/BRI509/tree/Assignment%233>

If there are errors in the solution or codes kindly email,  
recanoy@korea.ac.kr.