

Computer Assignment 2
EEE391- Basics of Signals and Systems

Efe Acer
21602217



Bilkent University, CS

Contents

Part a	3
Part b	7
Part c	11
Appendix	15
Complete Matlab Code	15

The *convolution sum* is defined for discrete signals $x[n]$ and $y[n]$ as follows:

$$z[n] = x[n] * y[n] = \sum_{\ell=-\infty}^{\infty} x[\ell] \cdot y[n - \ell] \quad (1)$$

If both $x[n]$ and $y[n]$ are finite sequences, meaning that they have a finite number of nonzero values, i.e. their supports are finite; we can reduce the summation in (1) by restricting the summation interval to the support of the sequences. Recall that a sequence's support refers to the subset of the sequence's domain containing the elements that are all mapped to nonzero values. Let the set of integers $\{-M_1, -M_1 + 1, \dots, 0, \dots, M_2 - 1, M_2\}$ contain the support of $x[n]$ where M_1 and M_2 are both positive integers. Similarly, let the set $\{-N_1, -N_1 + 1, \dots, 0, \dots, N_2 - 1, N_2\}$ contain the support of $y[n]$, again N_1 and N_2 are positive integers. Then, the *convolution sum* can be rewritten as:

$$z[n] = x[n] * y[n] = \sum_{\ell=-M_1}^{M_2} x[\ell] \cdot y[n - \ell] \quad (2)$$

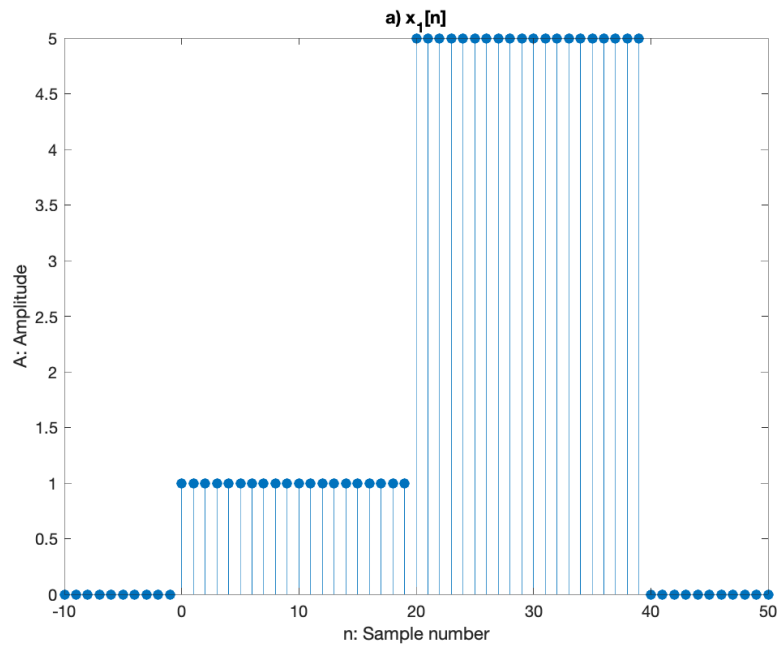
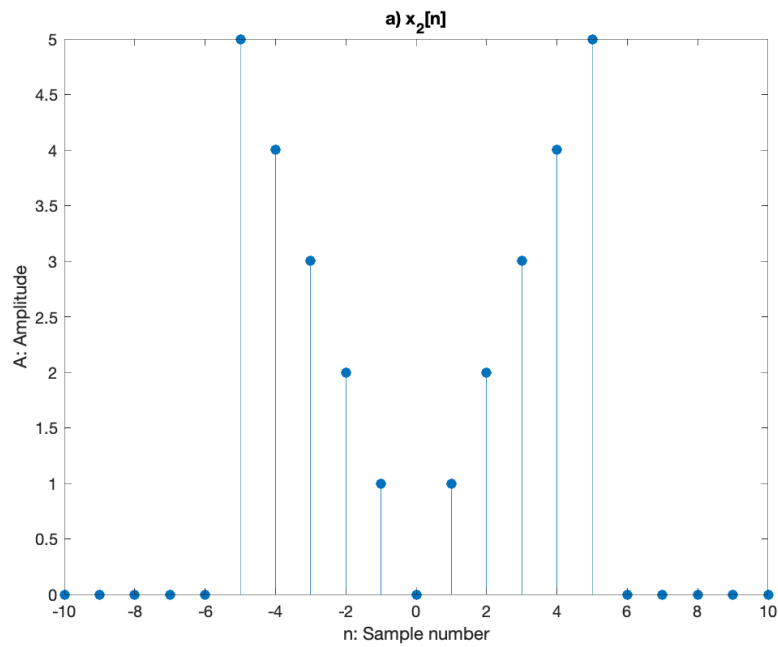
or equivalently:

$$z[n] = x[n] * y[n] = \sum_{k=-N_1}^{N_2} x[n - k] \cdot y[k] \quad (3)$$

The equivalence of (2) and (3) is a direct implication of the transitivity property of the *convolution sum*:

$$\sum_{\ell=-\infty}^{\infty} x[n] \cdot y[n - \ell] \xleftrightarrow{k=n-\ell} \sum_{k=-\infty}^{-\infty} x[n - k] \cdot y[k] = \sum_{k=-\infty}^{\infty} x[n - k] \cdot y[k] \quad (4)$$

The derivation above is translated into a **Matlab** function, namely **convolve**, as specified in the assignment sheet. The **convolve** function together with the complete **Matlab** code for the assignment are located in the Appendix section of this report. The digital signals given in the assignment sheet, $x_1[n]$ and $x_2[n]$, are stored in **Matlab** arrays after implementing their defining functions, then the requested convolutions are computed using the **convolve** function. The results are plotted after the time axis is properly adjusted, and their correctness are checked using **Matlab**'s built-in **conv** function. The following pages display the given digital signals $x_1[n]$ and $x_2[n]$ together with the convolutions, for parts a (1), b (2) and c (3).

Figure 1: $x_1[n]$ for part aFigure 2: $x_2[n]$ for part a

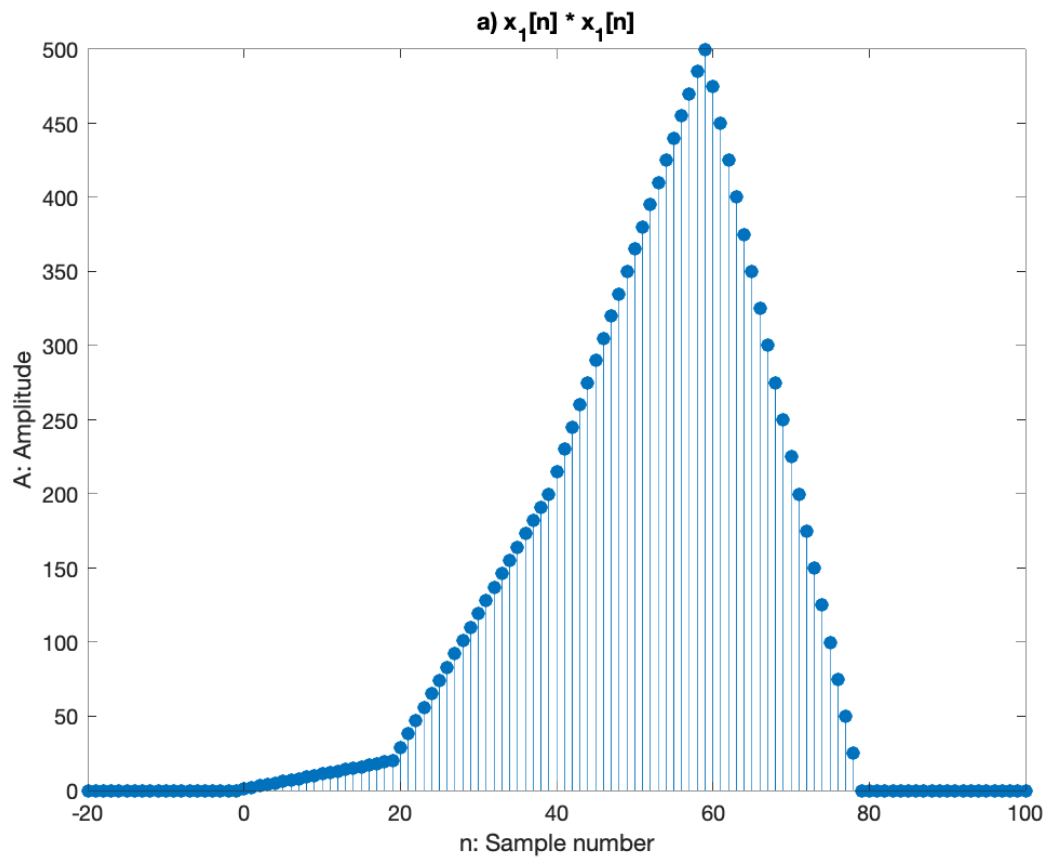
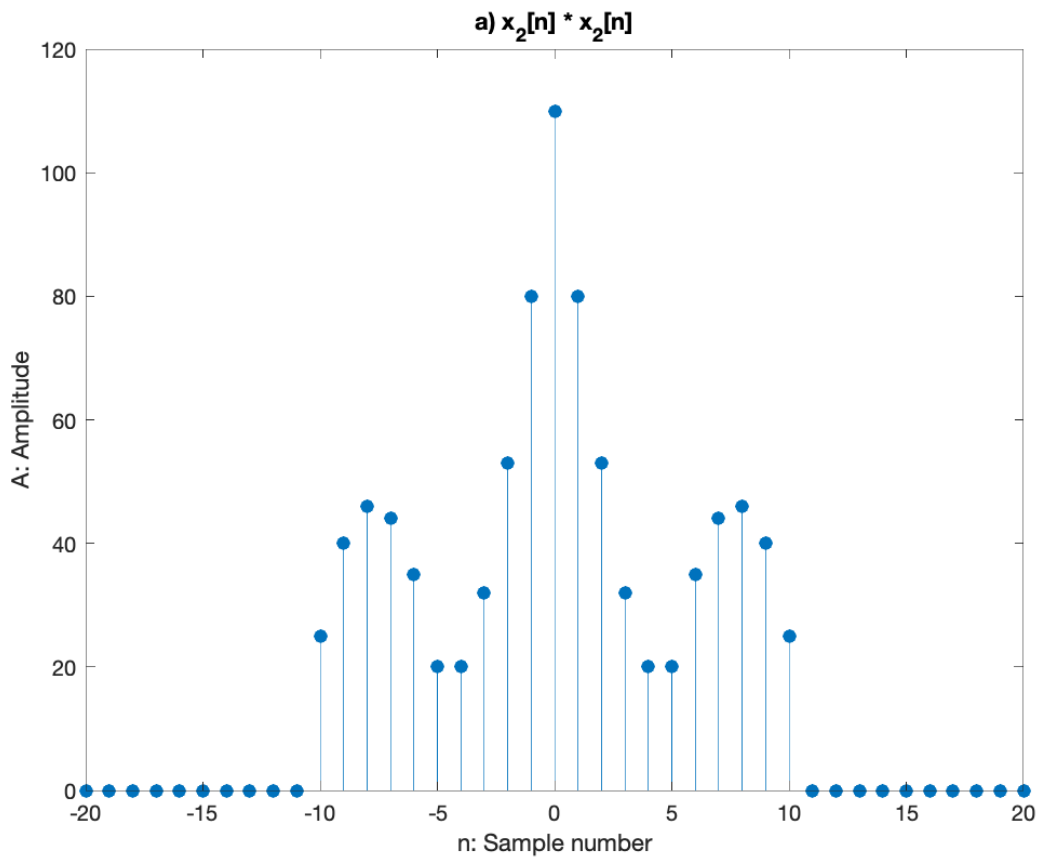
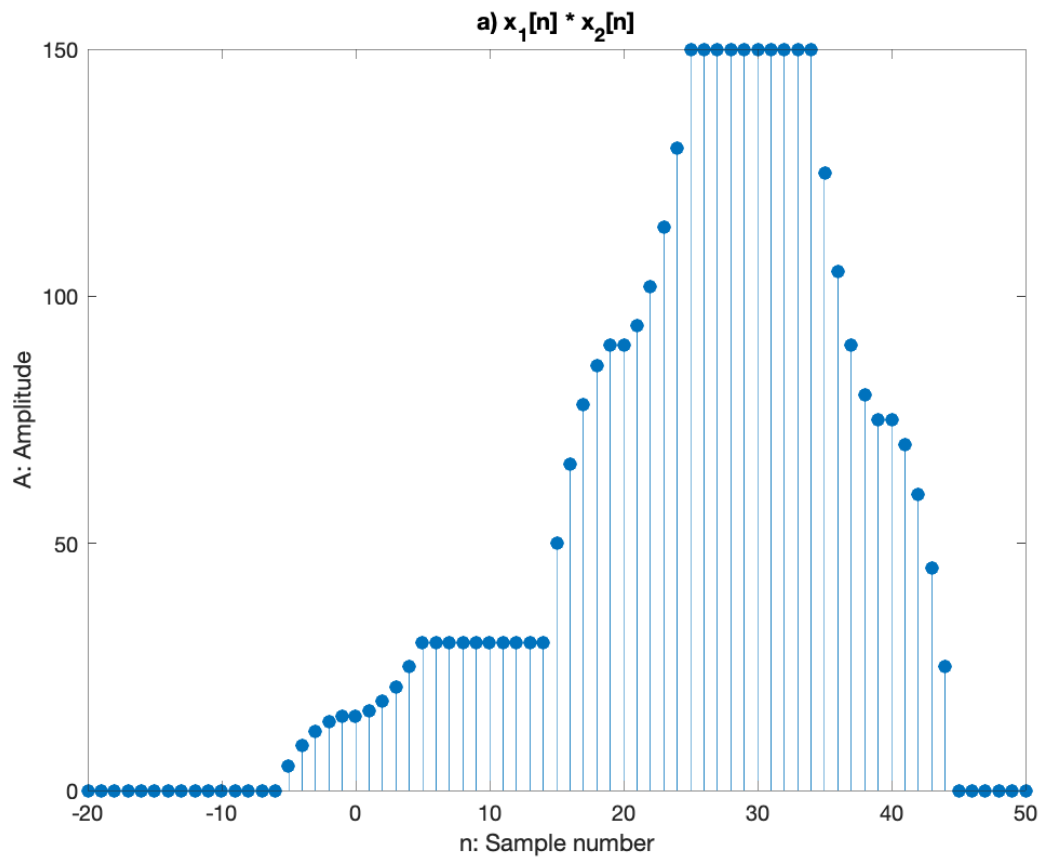


Figure 3: $x_1[n] * x_1[n]$ for part a

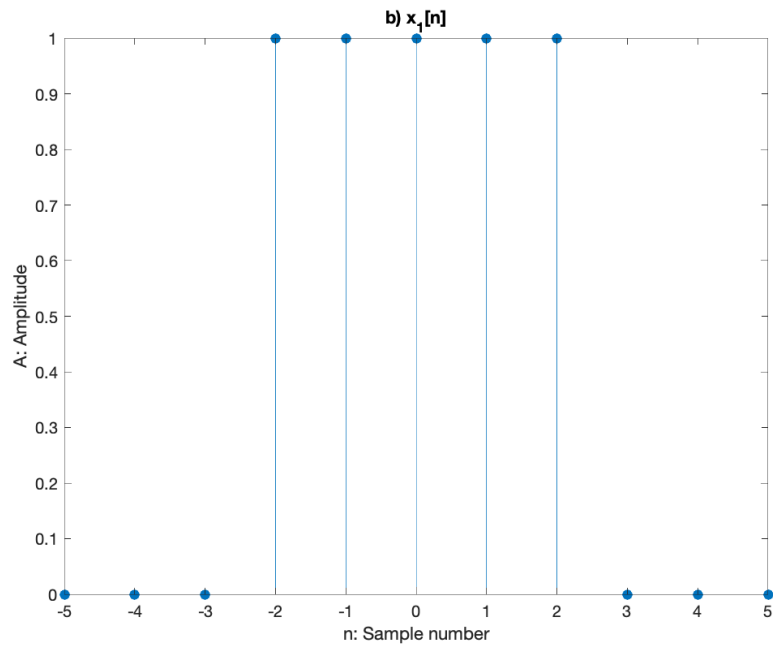
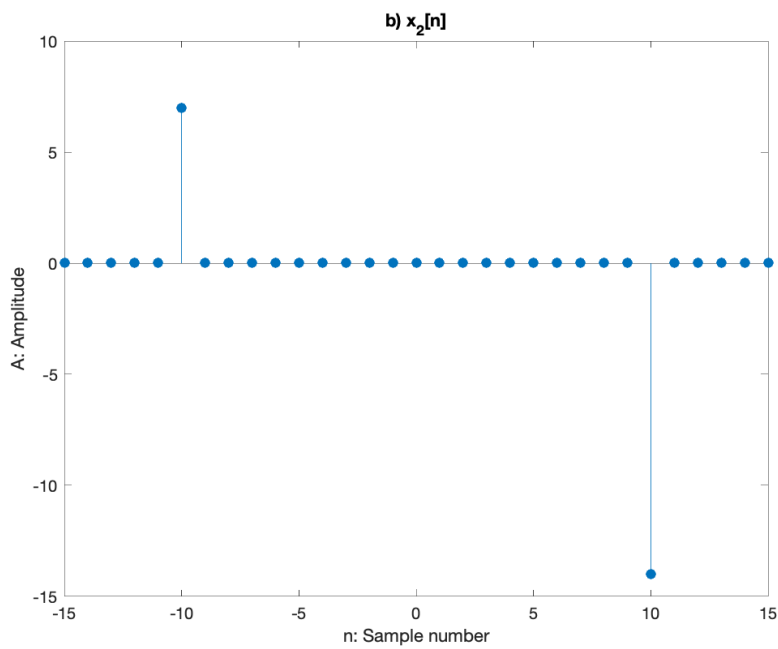
First nonzero time index of $x_1[n] * x_1[n]$ is 0.

Figure 4: $x_2[n] * x_2[n]$ for part a

First nonzero time index of $x_2[n] * x_2[n]$ is -10.

Figure 5: $x_1[n] * x_2[n]$ for part a

First nonzero time index of $x_1[n] * x_2[n]$ is -5.

Figure 6: $x_1[n]$ for part bFigure 7: $x_2[n]$ for part b

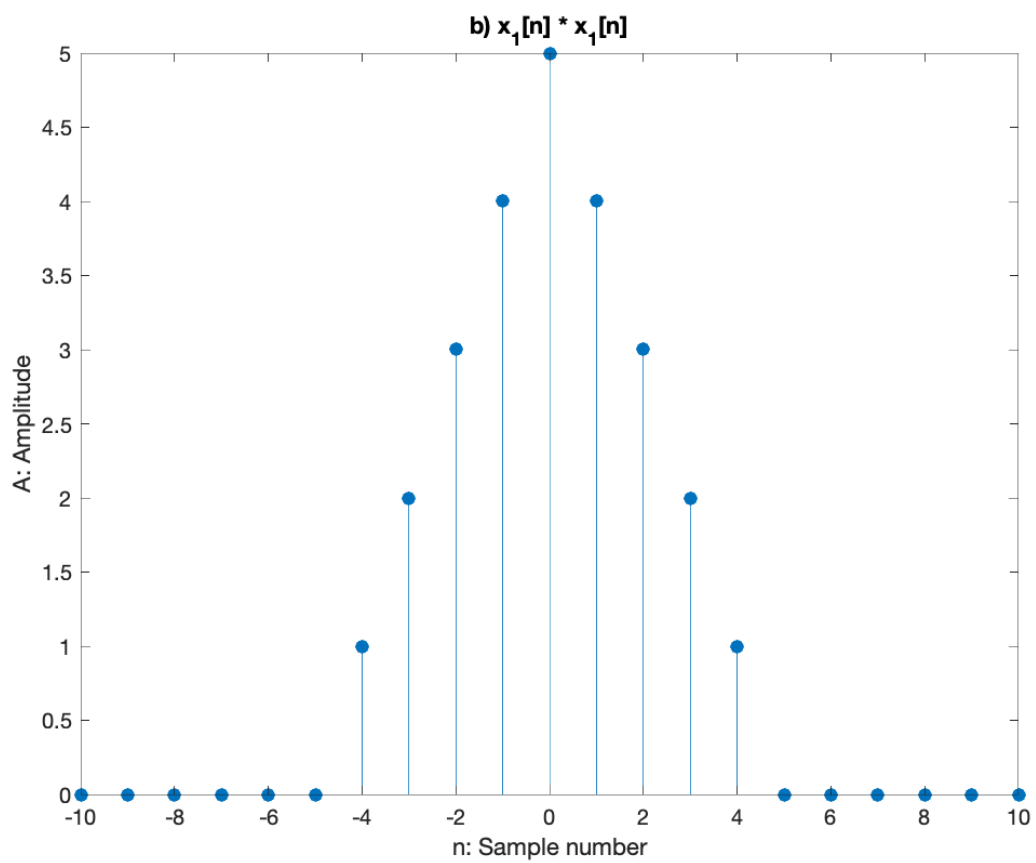


Figure 8: $x_1[n] * x_1[n]$ for part b

First nonzero time index of $x_1[n] * x_1[n]$ is -4.

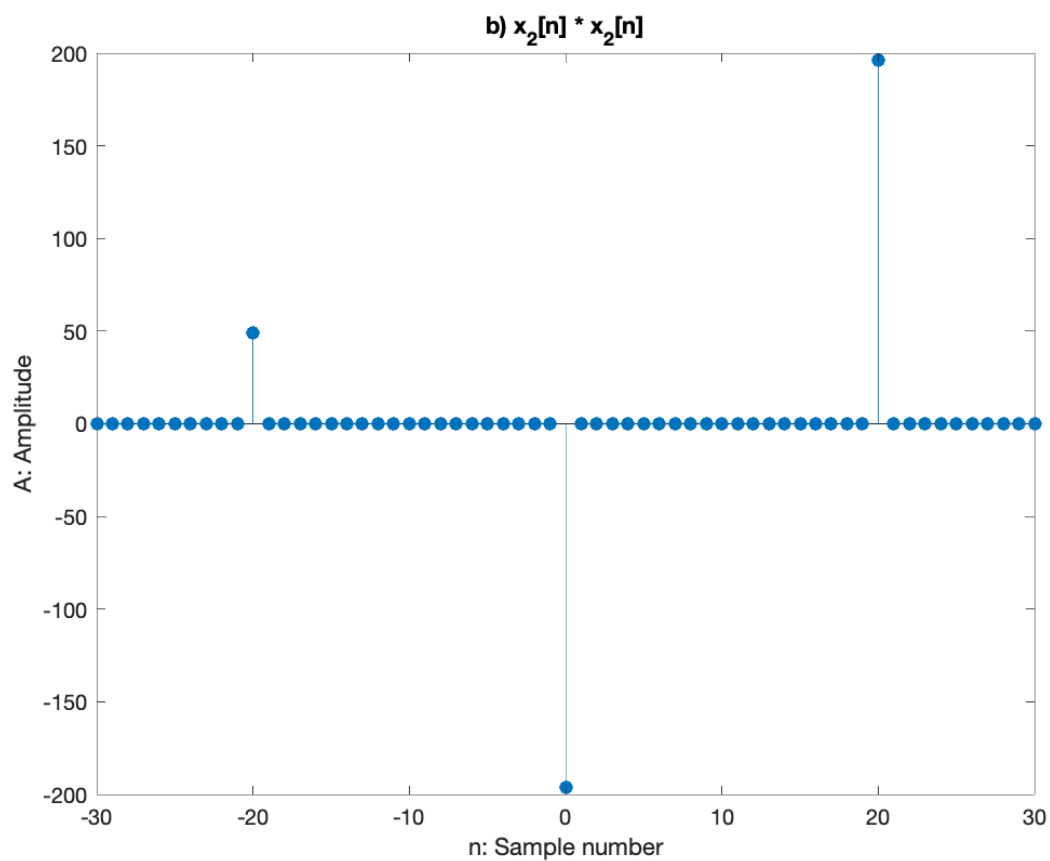
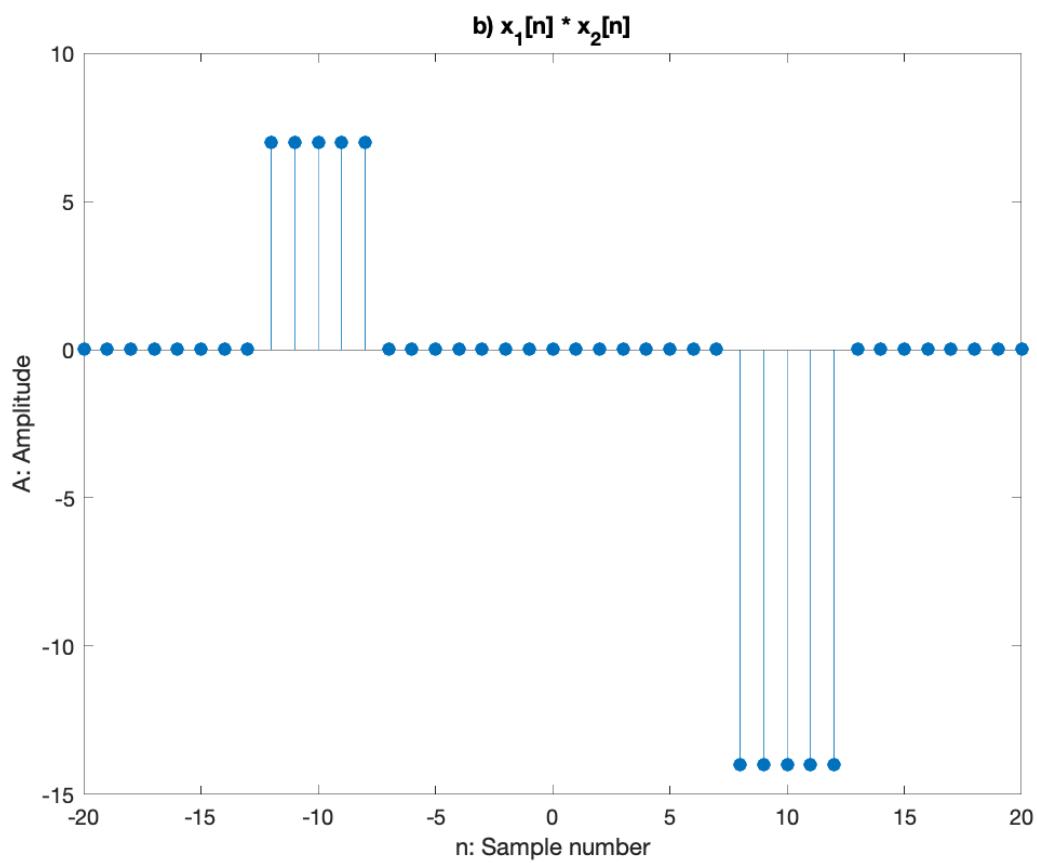
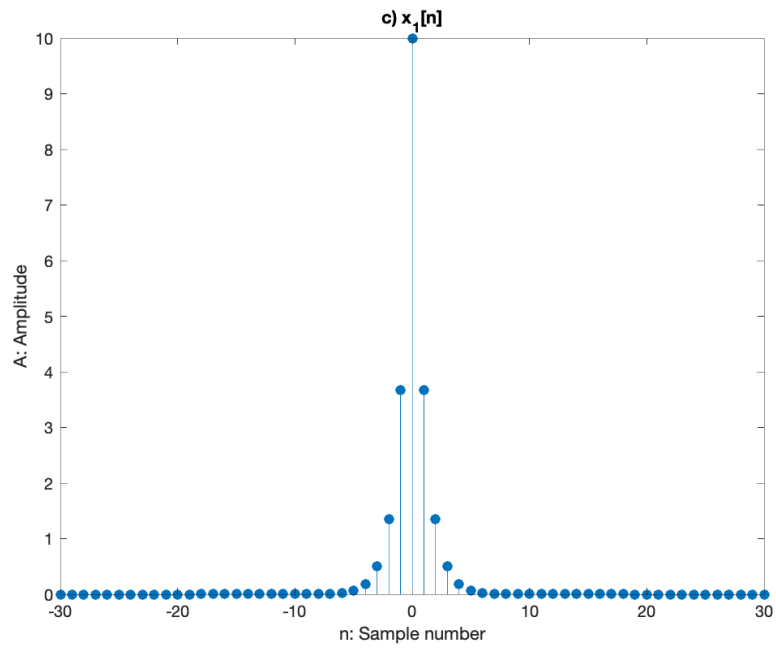
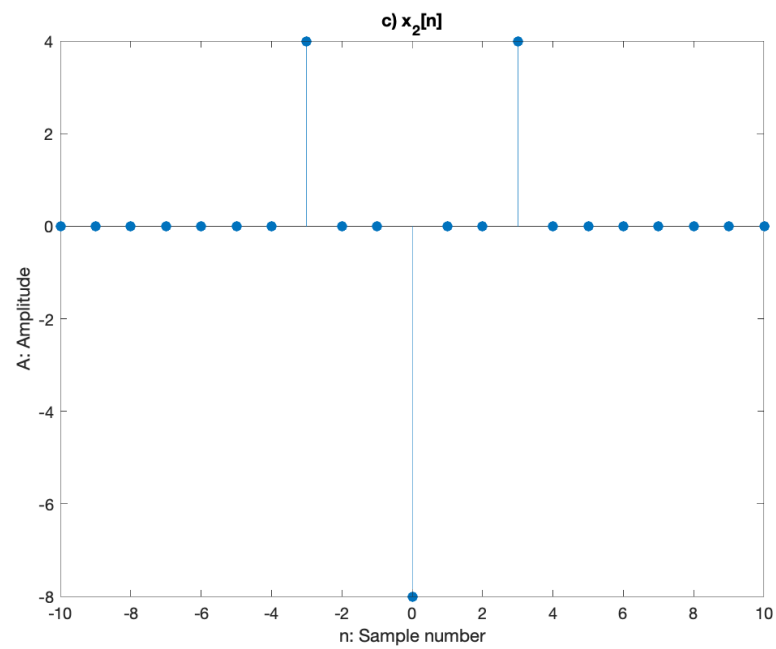


Figure 9: $x_2[n] * x_2[n]$ for part b

First nonzero time index of $x_2[n] * x_2[n]$ is -20.

Figure 10: $x_1[n] * x_2[n]$ for part b

First nonzero time index of $x_1[n] * x_2[n]$ is -12.

Figure 11: $x_1[n]$ for part cFigure 12: $x_2[n]$ for part c

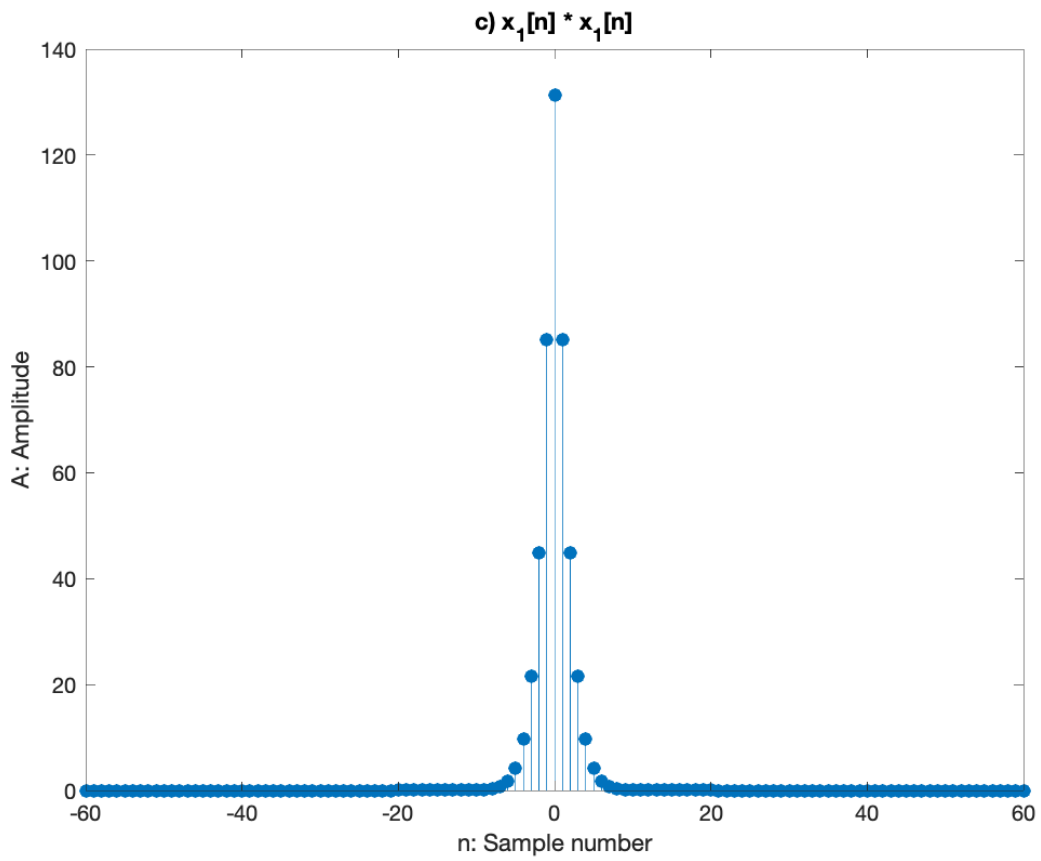
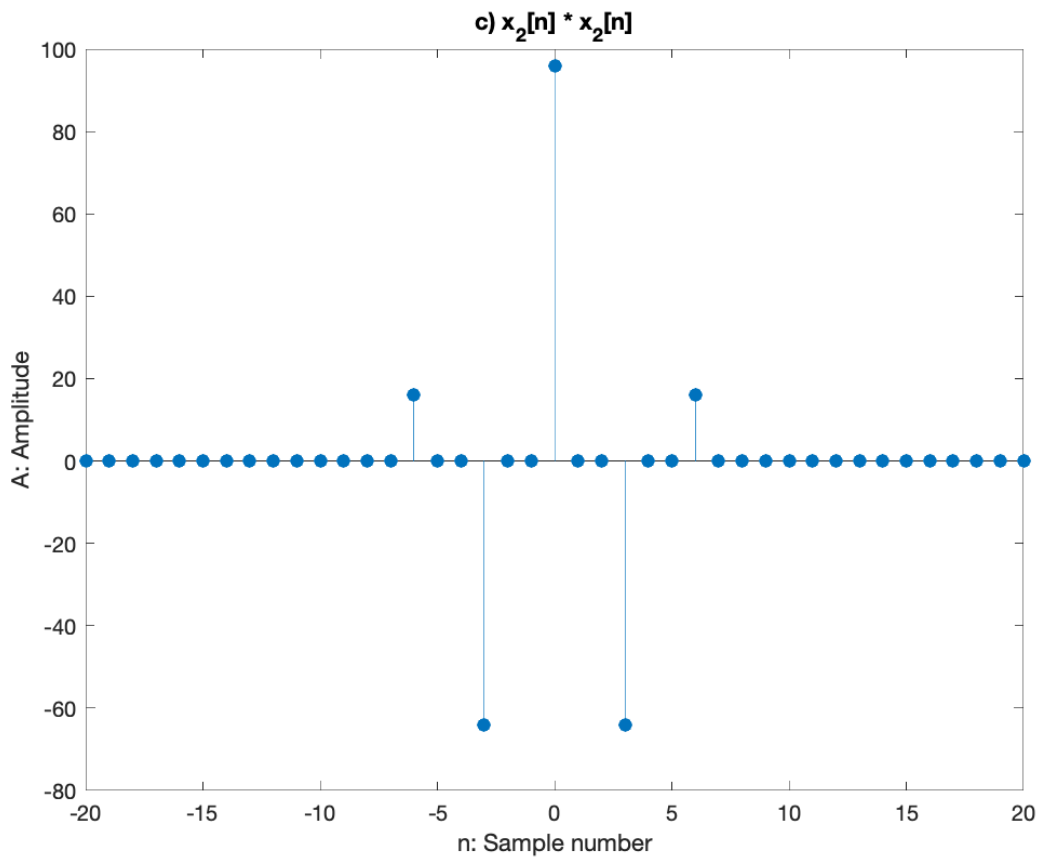


Figure 13: $x_1[n] * x_1[n]$ for part c

First nonzero time index of $x_1[n] * x_1[n]$ is -50, however the corresponding value is very close to zero.

Figure 14: $x_2[n] * x_2[n]$ for part c

First nonzero time index of $x_2[n] * x_2[n]$ is -6.

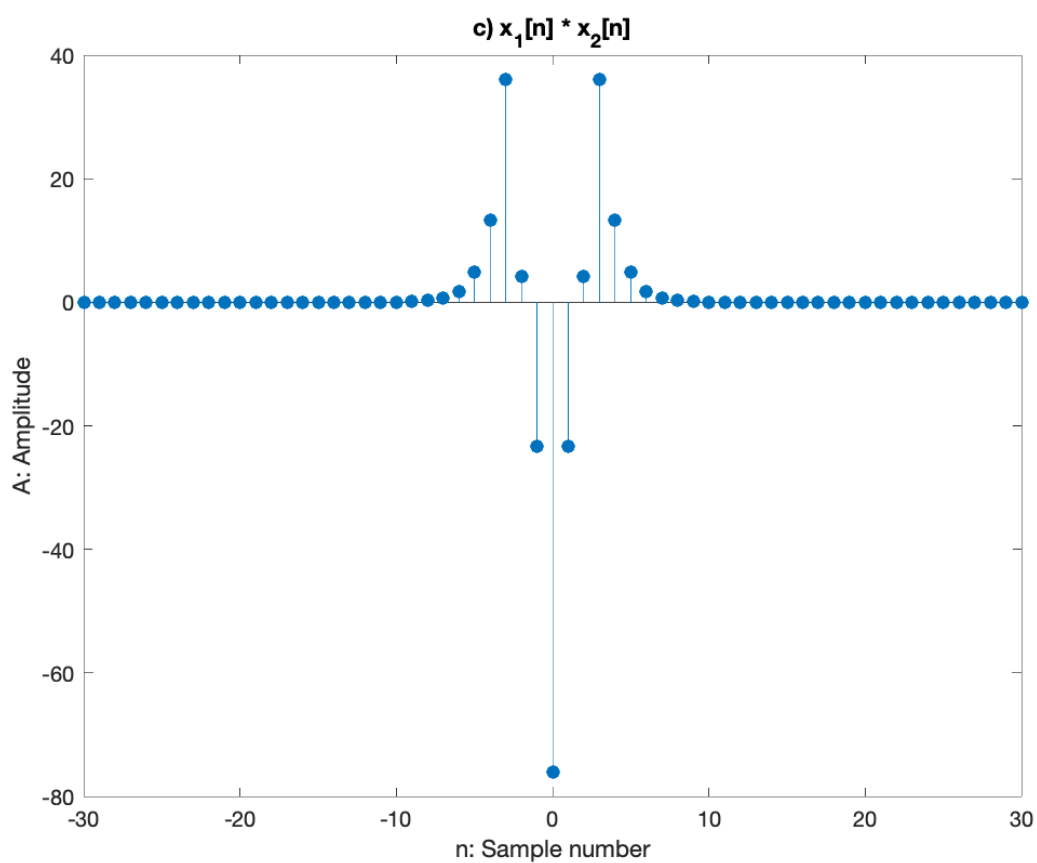


Figure 15: $x_1[n] * x_2[n]$ for part c

First nonzero time index of $x_1[n] * x_2[n]$ is -28, however the corresponding value is very close to zero.

Complete Matlab Code

```
1 % EEE391 Computer Assignment 2
2 % Author: EFE ACER
3
4 %% a)
5
6 offset = 101;
7 x1 = zeros(1, 201);
8 for n = -100:100
9     if n >= 0 && n <= 19
10         x1(n + offset) = 1;
11     elseif n >= 20 && n <= 39
12         x1(n + offset) = 5;
13     end
14 end
15 n_x1 = 0;
16
17 figure(1);
18 stem(-100:100, x1, 'filled');
19 xlim([-10, 50]);
20 title('a) x_1[n]');
21 xlabel('n: Sample number');
22 ylabel('A: Amplitude');
23
24 x2 = zeros(1, 201);
25 for n = -100:100
26     if abs(n) <= 5
27         x2(n + offset) = abs(n);
28     end
29 end
30 n_x2 = -5;
31
32 figure(2);
33 stem(-100:100, x2, 'filled');
34 xlim([-10, 10]);
35 title('a) x_2[n]');
36 xlabel('n: Sample number');
37 ylabel('A: Amplitude');
38
39 %z = conv(x1, x1);
40 [z, n_z] = convolve(x1, x1, n_x1, n_x1);
41 figure(3);
42 idx = find(z ~= 0, 1, 'first');
```



```
43 time_vals = (1:length(z)) - (idx - n_z);
44 stem(time_vals, z, 'filled');
45 xlim([-20, 100])
46 title('a)  $x_1[n] * x_1[n]$ ');
47 xlabel('n: Sample number');
48 ylabel('A: Amplitude');
49
50 %z = conv(x2, x2);
51 [z, n_z] = convolve(x2, x2, n_x2, n_x2);
52 figure(4);
53 idx = find(z ~= 0, 1, 'first');
54 time_vals = (1:length(z)) - (idx - n_z);
55 stem(time_vals, z, 'filled');
56 xlim([-20, 20])
57 title('a)  $x_2[n] * x_2[n]$ ');
58 xlabel('n: Sample number');
59 ylabel('A: Amplitude');
60
61 %z = conv(x1, x2);
62 [z, n_z] = convolve(x1, x2, n_x1, n_x2);
63 figure(5);
64 idx = find(z ~= 0, 1, 'first');
65 time_vals = (1:length(z)) - (idx - n_z);
66 stem(time_vals, z, 'filled');
67 xlim([-20, 50])
68 title('a)  $x_1[n] * x_2[n]$ ');
69 xlabel('n: Sample number');
70 ylabel('A: Amplitude');
71
72 %% b)
73
74 x1 = zeros(1, 201);
75 for n = -100:100
76     if -2 * n + 4 >= 0
77         x1(n + offset) = x1(n + offset) + 1;
78     end
79     if -n - 3 >= 0
80         x1(n + offset) = x1(n + offset) - 1;
81     end
82 end
83 n_x1 = -2;
84
85 figure(6);
86 stem(-100:100, x1, 'filled');
```

```
87 xlim([-5, 5])
88 title('b) x_1[n]');
89 xlabel('n: Sample number');
90 ylabel('A: Amplitude');
91
92 x2 = zeros(1, 201);
93 for n = -100:100
94     if -n - 10 == 0
95         x2(n + offset) = 7;
96     elseif -n + 10 == 0
97         x2(n + offset) = -14;
98     end
99 end
100 n_x2 = -10;
101
102 figure(7);
103 stem(-100:100, x2, 'filled');
104 xlim([-15, 15])
105 title('b) x_2[n]');
106 xlabel('n: Sample number');
107 ylabel('A: Amplitude');
108
109 %z = conv(x1, x1);
110 [z, n_z] = convolve(x1, x1, n_x1, n_x1);
111 figure(8);
112 idx = find(z ~= 0, 1, 'first');
113 time_vals = (1:length(z)) - (idx - n_z);
114 stem(time_vals, z, 'filled');
115 xlim([-10, 10])
116 title('b) x_1[n] * x_1[n]');
117 xlabel('n: Sample number');
118 ylabel('A: Amplitude');
119
120 %z = conv(x2, x2);
121 [z, n_z] = convolve(x2, x2, n_x2, n_x2);
122 figure(9);
123 idx = find(z ~= 0, 1, 'first');
124 time_vals = (1:length(z)) - (idx - n_z);
125 stem(time_vals, z, 'filled');
126 xlim([-30, 30])
127 title('b) x_2[n] * x_2[n]');
128 xlabel('n: Sample number');
129 ylabel('A: Amplitude');
130
```

```
131 %z = conv(x1, x2);
132 [z, n_z] = convolve(x1, x2, n_x1, n_x2);
133 figure(10);
134 idx = find(z ~= 0, 1, 'first');
135 time_vals = (1:length(z)) - (idx - n_z);
136 stem(time_vals, z, 'filled');
137 xlim([-20, 20])
138 title('b) x_1[n] * x_2[n]');
139 xlabel('n: Sample number');
140 ylabel('A: Amplitude');
141
142 %% c)
143
144 x1 = zeros(1, 201);
145 for n = -100:100
146     if abs(n) <= 25
147         x1(n + offset) = 10 * exp(-abs(n));
148     end
149 end
150 n_x1 = -25;
151
152 figure(11);
153 stem(-100:100, x1, 'filled');
154 xlim([-30, 30])
155 title('c) x_1[n]');
156 xlabel('n: Sample number');
157 ylabel('A: Amplitude');
158
159 x2 = zeros(1, 201);
160 for n = -100:100
161     if n + 3 == 0
162         x2(n + offset) = 4;
163     elseif n == 0
164         x2(n + offset) = -8;
165     elseif n - 3 == 0
166         x2(n + offset) = 4;
167     end
168 end
169 n_x2 = -3;
170
171 figure(12);
172 stem(-100:100, x2, 'filled');
173 xlim([-10, 10])
174 title('c) x_2[n]');
```

```

175 xlabel('n: Sample number');
176 ylabel('A: Amplitude');
177
178 %z = conv(x1, x1);
179 [z, n_z] = convolve(x1, x1, n_x1, n_x1);
180 figure(13);
181 idx = find(z ~= 0, 1, 'first');
182 time_vals = (1:length(z)) - (idx - n_z);
183 stem(time_vals, z, 'filled');
184 xlim([-60, 60])
185 title('c) x_1[n] * x_1[n]');
186 xlabel('n: Sample number');
187 ylabel('A: Amplitude');
188
189 %z = conv(x2, x2);
190 [z, n_z] = convolve(x2, x2, n_x2, n_x2);
191 figure(14);
192 idx = find(z ~= 0, 1, 'first');
193 time_vals = (1:length(z)) - (idx - n_z);
194 stem(time_vals, z, 'filled');
195 xlim([-20, 20])
196 title('c) x_2[n] * x_2[n]');
197 xlabel('n: Sample number');
198 ylabel('A: Amplitude');
199
200 %z = conv(x1, x2);
201 [z, n_z] = convolve(x1, x2, n_x1, n_x2);
202 figure(15);
203 idx = find(z ~= 0, 1, 'first');
204 time_vals = (1:length(z)) - (idx - n_z);
205 stem(time_vals, z, 'filled');
206 xlim([-30, 30])
207 title('c) x_1[n] * x_2[n]');
208 xlabel('n: Sample number');
209 ylabel('A: Amplitude');
210
211 %% Discrete Convolution Sum Implementation:
212
213 function [z, n_z] = convolve(x, y, n_x, n_y)
214 % CONVOLVE Computes the discrete convolution sum of two finite supported
215 % sequences x and y. n_x and n_y denotes the first time indices of the
216 % sequences which correspond to nonzero values. z returns the convolution
217 % result and n_z denotes the first time index of z's support.
218     z = zeros(1, length(x) + length(y) - 1);

```

```
219     for i = 1:length(x)
220         for j = 1:length(y)
221             z(i + j - 1) = z(i + j - 1) + x(i) * y(j);
222         end
223     end
224     n_z = n_x + n_y;
225 end
```