Digital Signal Processing HW2 MATLAB Part

Name: Jingyang Zhang
Net ID: jz2807

1) 1.3.13:

Smoothing data by N-point convolution.

Save the data file `DataEOG.txt` from the course website. Load the data into Matlab using the command `load DataEOG.txt` Type `whos` to see your variables. One of the variables will be `DataEOG`. For convenience, rename it to `x` by typing: `x = DataEOG;` This signal comes from measuring electrical signals from the brain of a human subject.

Make a stem plot of the signal $x(n)$. You will see it doesn't look good because there are so many points. Make a plot of $x(n)$ using the `plot` command. As you can see, for long signals we get a better plot using the `plot` command. Although discrete-time signals are most appropriately displayed with the `stem` command, for *long* discrete-time signals (like this one) we use the plot command for better appearance.

Create a simple impulse response for an LTI system:

`h = ones(1,11)/11;`

Compute the convolution of h and x:

`y = conv(x, h);`

Make a MATLAB plot of the output y.

(a) How does convolution change x? (Compare x and y.)

(b) How is the length of y related to the length of x and h?

(c) Plot x and y on the same graph. What problem do you see? Can you get y to "line up" with x?

(d) Use the following commands:
```
y2 = y;
y2(1:5) = [];
y2(end-4:end) = [];
```
What is the effect of these commands? What is the length of `y2`? Plot `x` and `y2` on the same graph. What do you notice now?

(e) Repeat the problem, but use a different impulse response:
`h = ones(1,31)/31;`
What should the parameters in part (d) be now?

(f) Repeat the problem, but use
`h = ones(1,67)/67;`
What should the parameters in part (d) be now?

Comment on your observations.

To turn in: The plots, your Matlab commands to create the signals and plots, and discussion.

Solution:

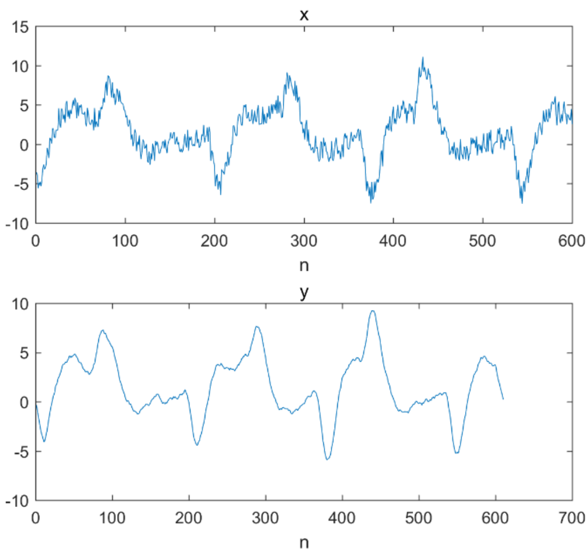
.m file(s): jyz_1_3_13.m

Code:

```
load DataEOG.txt
x = DataEOG;
s1 = subplot(2, 1, 1);
s2 = subplot(2, 1, 2);
h = ones(1, 11) / 11;
y = conv(x, h);
plot(s1, x);
xlabel(s1, 'n')
title(s1, 'x')
```

```
plot(s2, y);
xlabel(s2, 'n')
title(s2, 'y')
fprintf('the length of x is %g\n', length(x))
fprintf('the length of y is %g', length(y))
```

Result(plots):




```
>> jyz_1_3_13
the length of x is 600
fx the length of y is 610>>
```

(a) How does convolution change x? (Compare x and y.)
Comparing the data of y to x, it is obvious that the y can be seen as the 'smoothed x' data, which is, the peaks of the data value becomes less in number, the smoothness in plot is apparent.

(b) How is the length of y related to the length of x and h?
According to the property of convolution, length(y) = length(x) + length(h) -1. In this case, the length of x is 600, the length of h is 11, and the length of y is 610.

(c) Plot x and y on the same graph. What problem do you see? Can you get y to "line up" with x?

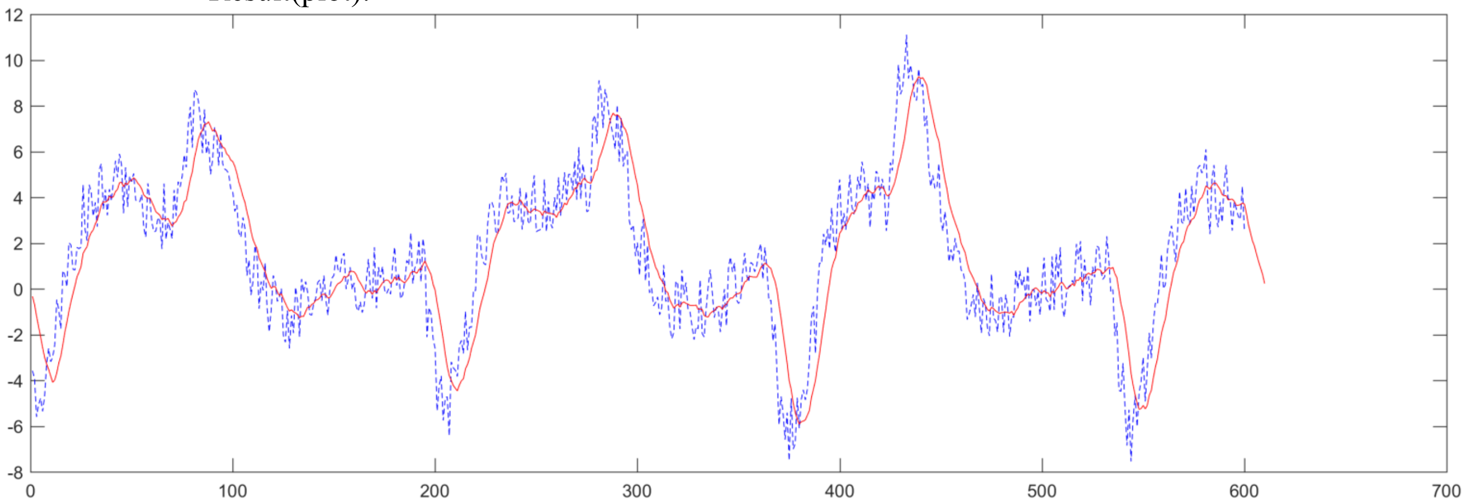
.m files(s): jyz_1_3_13_c.m

Code:

```
load DataEOG.txt
x = DataEOG;
n1 = (1: length(x));
h = ones(1, 11) / 11;
```

```
y = conv(x, h);
n2 = (1: 610);
plot(n1, x, 'b--',    n2, y, 'r');
```

Result(plot):



Result(discussion):

The plot of x and the plot of y cannot overlap each other completely, there exists a delay of time between *x* and *y*, and *y* cannot 'line up' with *x*.

(d) Use the following commands:

```
y2 = y;
y2(1 : 5) = [];
y2(end - 4 : end) = [];
```

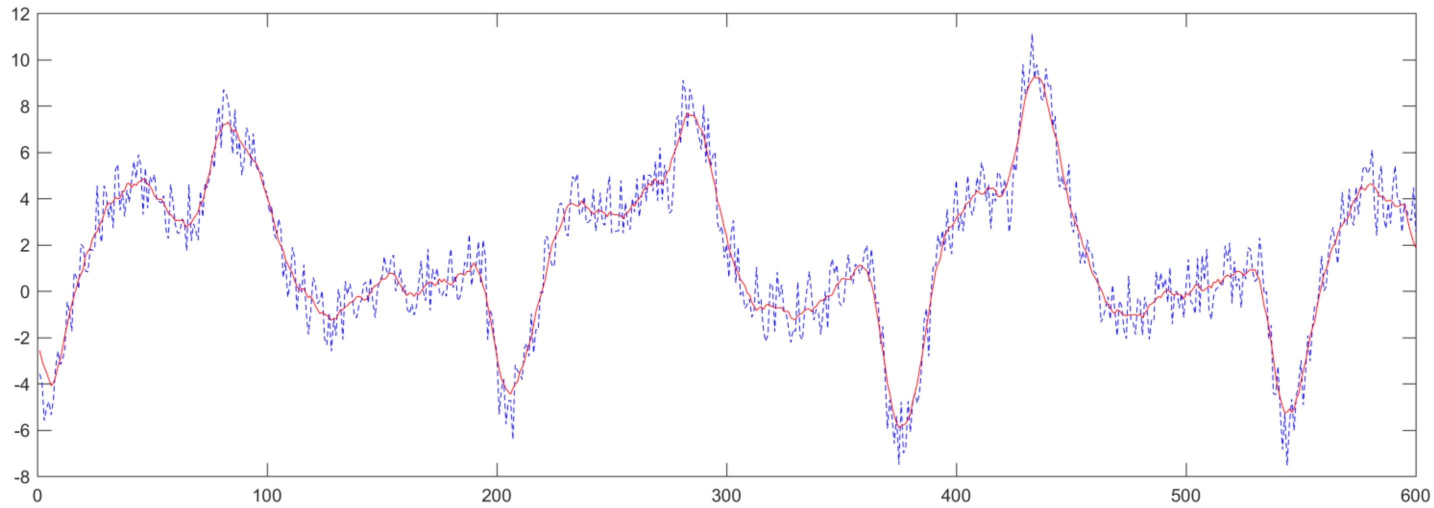
What is the effect of these commands? What is the length of y2? Plot x and y2 on the same graph. What do you notice now?

.m files(s): jyz_1_3_13_d.m

Code:

```
load DataEOG.txt
x = DataEOG;
h = ones(1, 11) / 11;
y = conv(x, h);
y2 = y;
y2(1 : 5) = [];
y2(end - 4 : end) = [];
n = (1: length(x));
plot(n, x, 'b--',    n, y2, 'r');
fprintf('the length of x is %g\n', length(x))
fprintf('the length of y2 is %g', length(y2))
```

Result(plot):




```
>> jyz1_3_13_d
the length of x is 600
the length of y2 is 600>>
```

Result(discussion):
These commands cut the first and last 5 data of *y*, and let *x* and *y2* shares the same length. As a result, there is no time delay between *x* and *y2* any more.

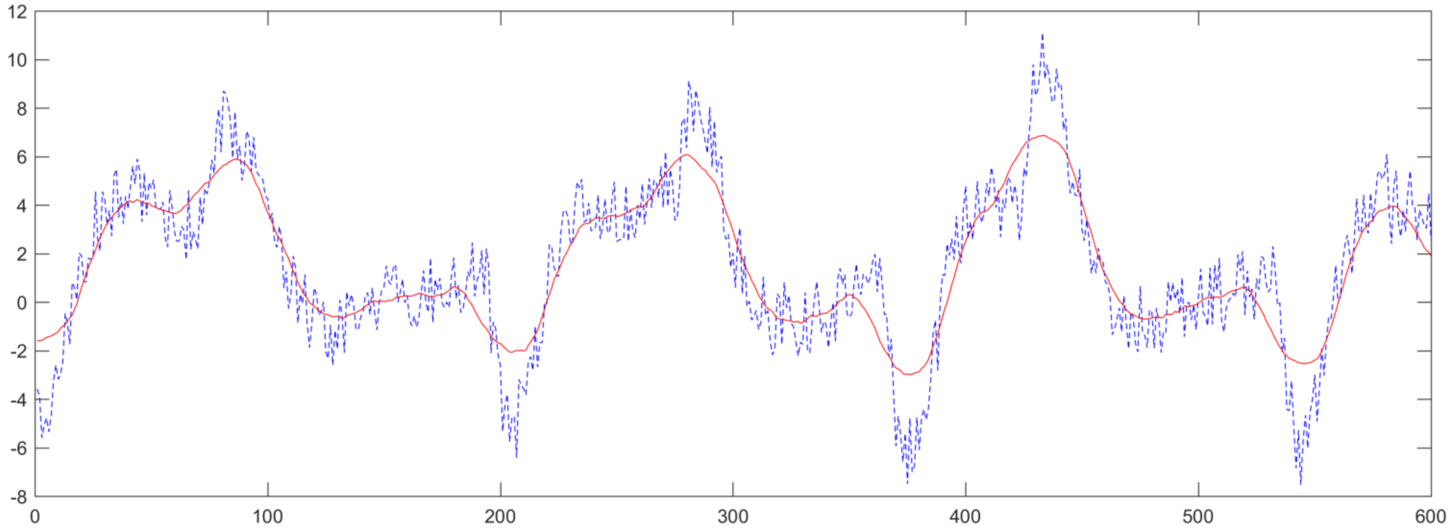
(e) Repeat the problem, but use a different impulse response:
h = ones(1, 31) / 31;
What should the parameters in part (d) be now?

.m files(s): jyz_1_3_13_e.m

Code:

```
load DataEOG.txt
x = DataEOG;
h = ones(1, 31) / 31;
y = conv(x, h);
y3 = y;
y3(1 : 15) = [];
y3(end - 14 : end) = [];
n = (1: length(x));
plot(n, x, 'b--',    n, y3, 'r');
fprintf('the length of x is %g\n', length(x))
fprintf('the length of y2 is %g', length(y3))
```

Result(plot):




```
>> jyz1_3_13_e
the length of x is 600
the length of y3 is 600>>
```

Result(discussion):
The plot of *y3* is even smoother than that of *y2*. The command of cutting becomes:

```
y3 = y;
y3(1 : 15) = [];
y3(end - 14 : end) = [];
```

The first and last [length(*h*)-1] / 2 data should be cut from *y*.

(f) Repeat the problem, but use
h = ones(1 , 67) / 67;
What should the parameters in part (d) be now?

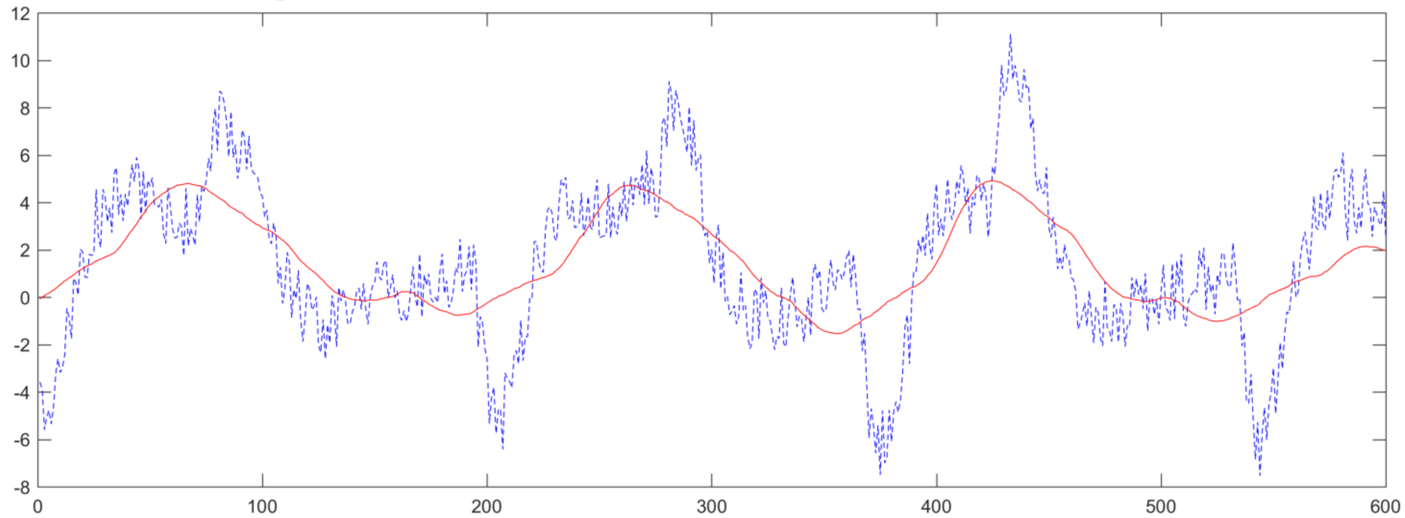
.m files(s): jyz_1_3_13_e.m

Code:

```
load DataEOG.txt
x = DataEOG;
h = ones(1, 67) / 67;
y = conv(x, h);
y4 = y;
y4(1 : 33) = [];
y4(end - 32 : end) = [];
n = (1: length(x));
plot(n, x, 'b--',    n, y4, 'r');
```

fprintf('the length of x is %g\n', length(x))
fprintf('the length of y4 is %g', length(y4))

Result(plot):




```
>> jyz1_3_13_f
the length of x is 600
the length of y4 is 600>>
```

Result(discussion):

The plot of *y4* is even smoother than that of *y3*. The command of cutting becomes:

y4 = y;
y4(1 : 33) = [];
y4(end - 32 : end) = [];

The first and last [length(*h*)-1] / 2 data should be cut from *y*.