MECH 306 Tutorial 1:

Analysis and Graphical Representation of Data

Jasper

January 11, 2019

Contents

1 Histogram Exercise

1.1 Importing the data

For the sake of brevity, I have exported the initial `BoltData` variable to a separate file.
Let's import it first.

```
load('boltdata');
```

We can check that the data loaded correctly by checking its size.

```
ans = size(BoltData)
```

```
| 91 | 3 |
```

1.2 Removing incorrect results

Now that the data has been imported into MATLAB, we can start removing invalid data from our analysis.

1.2.1 Invalid bolt numbers

There were only 12 types of bolts handed out during the experiment, and the only valid bolt ID numbers are integers from 1-12
Let's use a new variable to hold our cleaned data set.

```
cleanBoltData = BoltData;
```

Now we create a test for rows of that match our criteria

```
% Create a column vector for every row in cleanBoltData such that
% rows where the first item is an integer between 1 and 12 are
% equal to one, and zero otherwise
filter = cleanBoltData(:,1) >= 1 & cleanBoltData(:,1) <=12 ...
         & mod(cleanBoltData(:,1),1) == 0;
```

With this filter we can remove unwanted rows from our data.

```
% Select all rows in cleanBoltData where the corresponding value in filter is one
cleanBoltData = cleanBoltData(filter,:);
```

We can double check that rows have been removed by checking the size again

```
ans = size(cleanBoltData)
```

```
| 88 | 3 |
```

1.2.2 Removing unreasonable lengths

According to the tutorial instructions reasonable bolt length measurements should be between 72 and 115 millimetres, so we remove any values outside of that range.

```
filter = cleanBoltData(:,2) >= 72 & cleanBoltData(:,2) <= 115;
cleanBoltData = cleanBoltData(filter,:);
```

Again we can double check that rows have been removed by checking the size again

```
ans = size(cleanBoltData)
```

```
| 72 | 3 |
```

1.2.3 Removing unreasonable diameters

According to the tutorial instructions reasonable bolt diameter measurements should be between 12 and 16 millimetres, so we remove any values outside of that range.

```
filter = cleanBoltData(:,3) >= 12 & cleanBoltData(:,3) <= 16;
cleanBoltData = cleanBoltData(filter,:);
```

Again we can double check that rows have been removed by checking the size again

```
ans = size(cleanBoltData)
```

```
| 61 | 3 |
```

1.2.4 Further removing outliers

Now that the obvious outliers have been removed we can graph the histograms for the bolt length and diameter

```
name = 'initiallen.svg';
% need many bins because of bimodal distribution
histogram(cleanBoltData(:,2),40);
title('Bolt Length')
xlabel('Bolt Length, mm')
ylabel('Number of Bolts')

set(gcf, 'PaperUnits', 'inches', 'PaperPosition', [0 0 7 5])

saveas(gcf, name)
ans = name
```

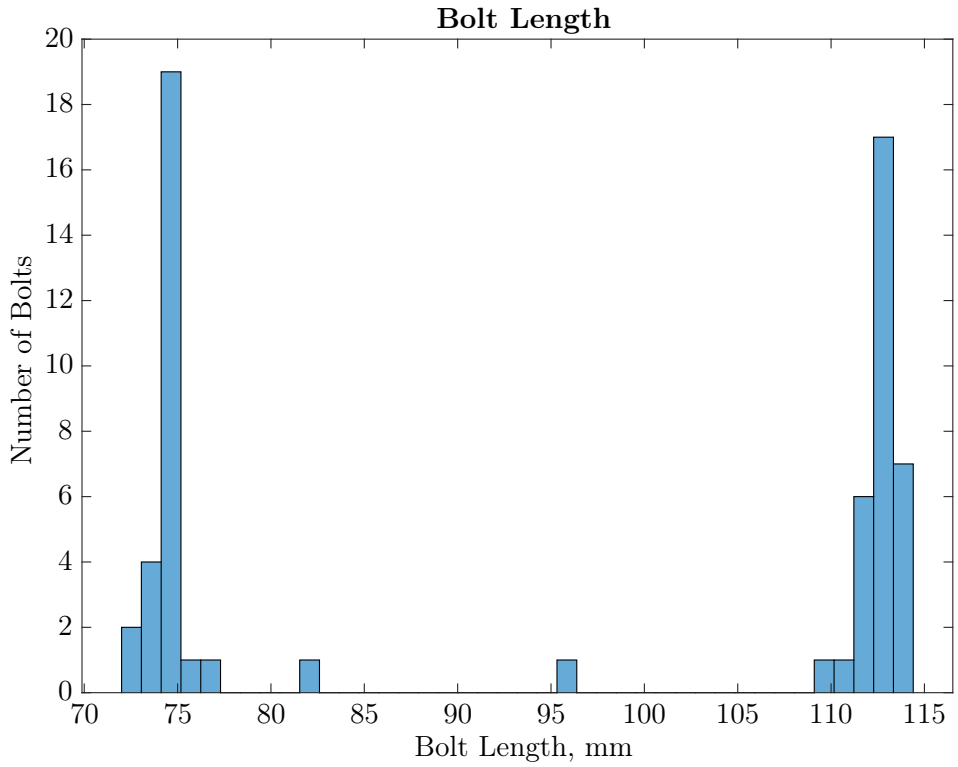

Figure 1: Histogram of bolt length after removing most of the invalid data

```
name = 'initialdia.svg';
histogram(cleanBoltData(:,3),8);
title('Bolt Diameter')
xlabel('Bolt Diameter, mm')
ylabel('Number of Bolts')

set(gcf, 'PaperUnits', 'inches', 'PaperPosition', [0 0 7 5])

saveas(gcf, name)
ans = name
```

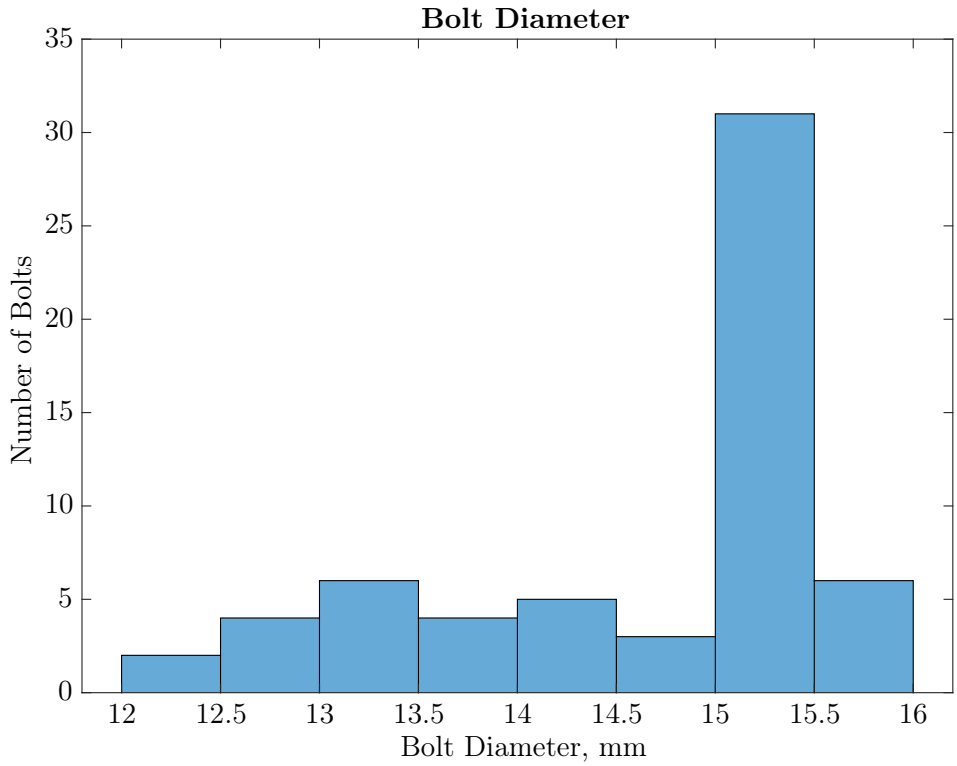

Figure 2: Histogram of bolt diameters after removing most of the invalid data

From Figure 1 it appears that there are still outliers in the middle of the two peaks.
Let's remove values between 70 and 80 millimetres as well as between 100 and 120 millimetres.

```
filter = (cleanBoltData(:,2) >= 70 & cleanBoltData(:,2) <= 80) | ...
         (cleanBoltData(:,2) >= 100 & cleanBoltData(:,2) <= 120);
cleanBoltData = cleanBoltData(filter,:);
```

1.3 Final Results

Let's plot the final set of data.

```
name = 'finallen.svg';
histogram(cleanBoltData(:,2),40);
title('Bolt Length')
xlabel('Bolt Length, mm')
ylabel('Number of Bolts')

set(gcf, 'PaperUnits', 'inches', 'PaperPosition', [0 0 7 5])

saveas(gcf, name)
ans = name
```

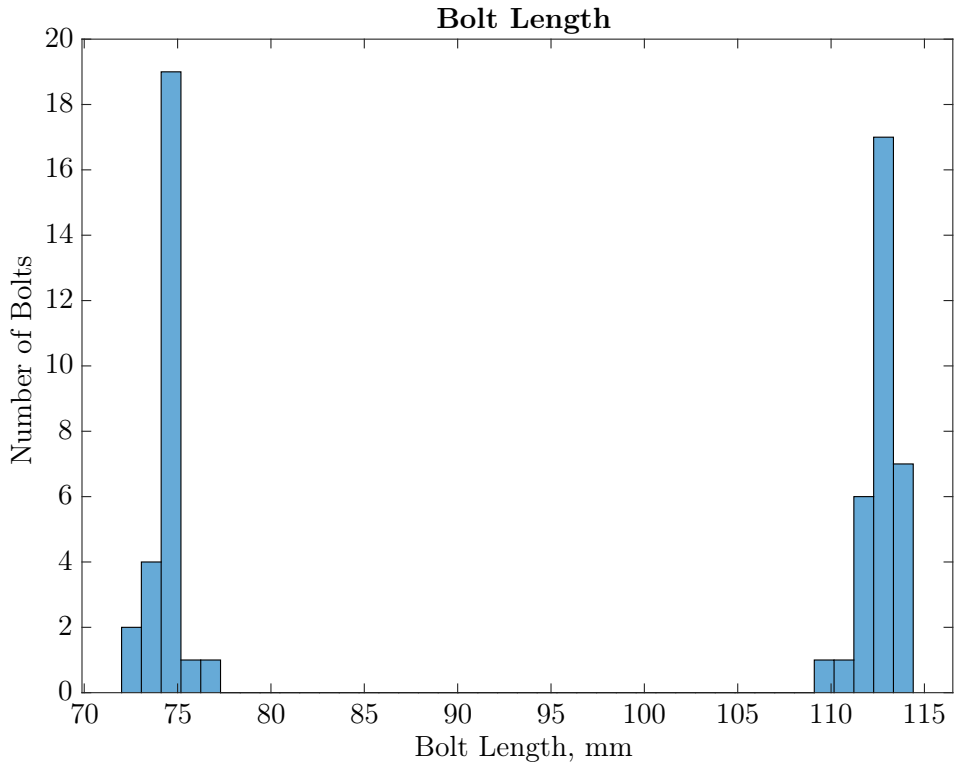

Figure 3: Histogram of bolt lengths after removing all of the invalid data

```
name = 'finaldia.svg';
histogram(cleanBoltData(:,3),8);
title('Bolt Diameter')
xlabel('Bolt Diameter, mm')
ylabel('Number of Bolts')

set(gcf, 'PaperUnits', 'inches', 'PaperPosition', [0 0 7 5])

saveas(gcf, name)
ans = name
```

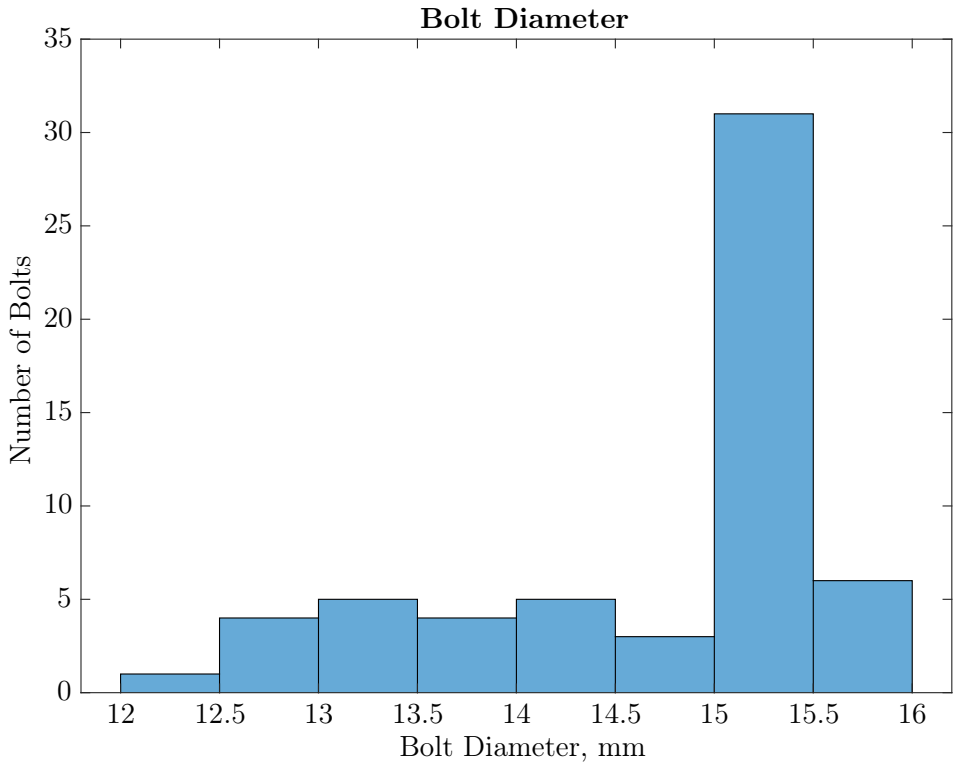

Figure 4: Histogram of bolt diameters after removing all of the invalid data

Let's also grab the averages and standard deviations for the bolt lengths and diameters.

```
% sort data into two length groups
iL1 = 0;
iL2 = 0;
for i = 1:1:size(cleanBoltData,1)
    if cleanBoltData(i,2)>=71 && cleanBoltData(i,2)<=78
        iL1 = iL1+1;
        LengthData1(iL1) = cleanBoltData(i,2);
    elseif cleanBoltData(i,2)>=108 && cleanBoltData(i,2)<=115
        iL2 = iL2+1;
        LengthData2(iL2) = cleanBoltData(i,2);
    end
end

% Find average and standard deviation of each bolt length
LengthAvg = [mean(LengthData1) mean(LengthData2)];
LengthStd = [ std(LengthData1)  std(LengthData2)];

% sort data into two diameter groups
iD1 = 0;
iD2 = 0;
for i = 1:1:size(cleanBoltData,1)
    if cleanBoltData(i,3)>=12 && cleanBoltData(i,3)<=14
        iD1 = iD1+1;
        DiamData1(iD1) = cleanBoltData(i,3);
    elseif cleanBoltData(i,3)>14 && cleanBoltData(i,3)<=16
        iD2 = iD2+1;
        DiamData2(iD2) = cleanBoltData(i,3);
    end
end

% Find average and standard deviation of each bolt diameter
DiamAvg = [mean(DiamData1) mean(DiamData2)];
DiamStd = [ std(DiamData1)  std(DiamData2)];

len_avg = sprintf('LengthAvg %f %f', LengthAvg(:));
len_std = sprintf('LengthStd %f %f', LengthStd(:));

dia_avg = sprintf('DiamAvg %f %f', DiamAvg(:));
dia_std = sprintf('DiamStd %f %f', DiamStd(:));

ans = sprintf('%s\n%s\n%s\n%s', len_avg, len_std, dia_avg, dia_std)
```

```
| LengthAvg 74.562963 112.631250 |
| LengthStd 0.878000 0.827915    |
| DiamAvg 13.166667 15.073171    |
| DiamStd 0.618347 0.384724      |
```

2 Box Plot exercise

2.1 Importing the data

```
A = [112.5 113 113 113.4 112 112.5 113.5]';
D = [112 112.5 113.5 112.5 113 113 112.5]';
E = [112 112 112.5 113 113.5 113.5 114.3]';
G = [113 111 112 112.5 112.5 110 112.5]';
H = [111.5 112 114 112 112.5 113 112.5]';

all = [A D E G H];
All = reshape(all, [], 1);
```

2.2 Plotting the bolts separately

```
name = 'boxsep.svg';

boxplot(all, ['A' 'D' 'E' 'G' 'H']')
title('Individual Bolt Lengths')
xlabel('Bolt ID')
ylabel('Bolt Length, mm')

set(gcf, 'PaperUnits', 'inches', 'PaperPosition', [0 0 7 5])

saveas(gcf, name)
ans = name
```

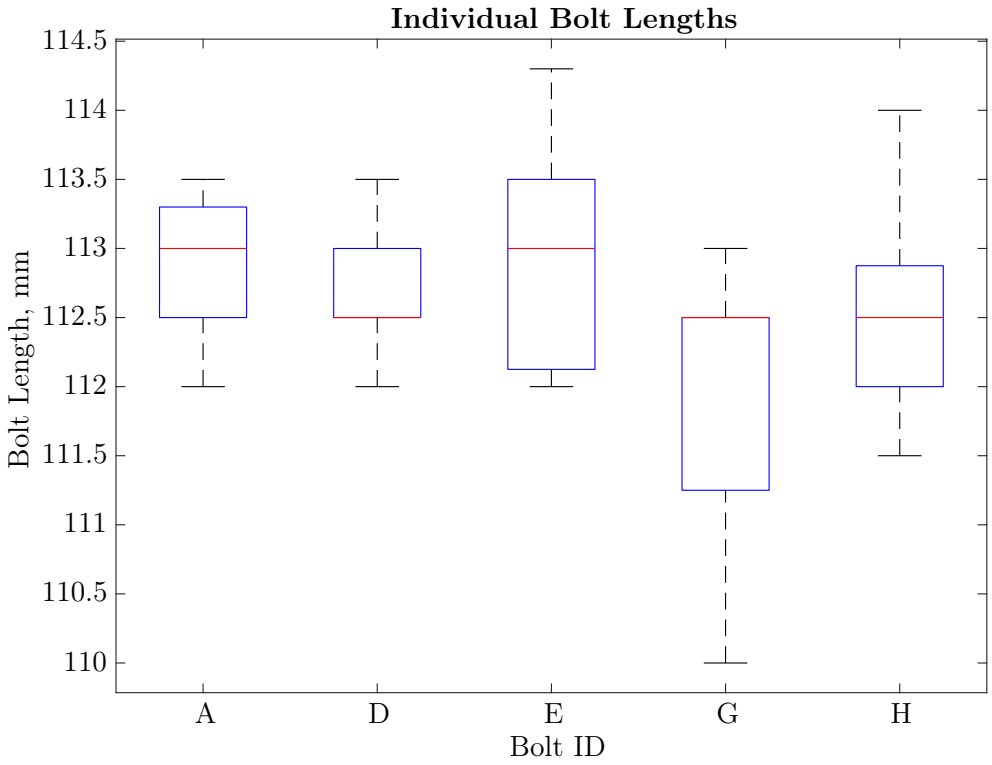

Figure 5: Box plot of individual bolt lengths using data supplied from tutorial

2.3 Plotting the bolts together

```
name = 'boxtog.svg';

boxplot(All)
title('All Bolt Lengths')
xlabel('All Bolts')
ylabel('Bolt Length, mm')

set(gcf, 'PaperUnits', 'inches', 'PaperPosition', [0 0 7 5])
set(gca, 'XTickLabel', {' '})

saveas(gcf, name)
ans = name
```

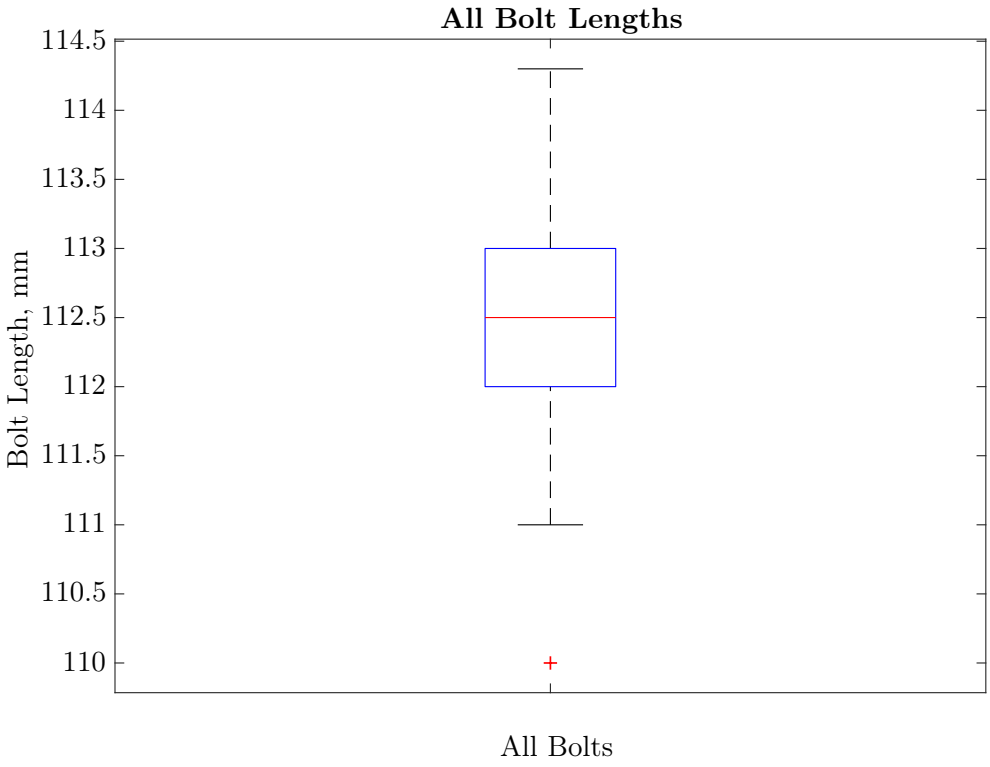

Figure 6: Box plot of all bolt lengths using data supplied from tutorial

2.4 Signifcant Statistics

2.4.1 Bolt A

```
mean_A = mean(A);
median_A = median(A);
quartiles_A = quantile(A, [0.25 0.50 0.75]);
stddev_A = std(A);
var_A = var(A);

mean_A_str = sprintf('mean_A: %f', mean_A);
median_A_str = sprintf('median_A: %f', median_A);
quartiles_A_str = sprintf('quartiles_A: %f %f %f', quartiles_A(:));
stddev_A_str = sprintf('stddev_A: %f', stddev_A);
var_A_str = sprintf('var_A: %f', var_A);

ans = sprintf('%s\n%s\n%s\n%s\n%s', ...
              mean_A_str, median_A_str, ...
              quartiles_A_str, stddev_A_str, ...
              var_A_str)
```

```
| mean_A: 112.842857                              |
| median_A: 113.000000                            |
| quartiles_A: 112.500000 113.000000 113.300000 |
| stddev_A: 0.538074                              |
| var_A: 0.289524                                 |
```

For the sake of brevity only the results will be included in the following secttions

2.4.2 Bolt D

```
| mean_D: 112.714286                              |
| median_D: 112.500000                            |
| quartiles_D: 112.500000 112.500000 113.000000 |
| stddev_D: 0.487950                              |
| var_D: 0.238095                                 |
```

2.4.3 Bolt E

```
| mean_D: 112.714286                              |
| median_D: 112.500000                            |
| quartiles_D: 112.500000 112.500000 113.000000 |
| stddev_D: 0.487950                              |
| var_D: 0.238095                                 |
```

2.4.4 Bolt G

```
| mean_G: 111.928571                              |
| median_G: 112.500000                            |
| quartiles_G: 111.250000 112.500000 112.500000 |
| stddev_G: 1.057850                              |
| var_G: 1.119048                                 |
```

2.4.5 Bolt H

```
| mean_H: 112.500000                              |
| median_H: 112.500000                            |
| quartiles_H: 112.000000 112.500000 112.875000 |
| stddev_H: 0.816497                              |
| var_H: 0.666667                                 |
```

2.4.6 All bolts

```
| mean_All: 112.591429                              |
| median_All: 112.500000                            |
| quartiles_All: 112.000000 112.500000 113.000000 |
| stddev_All: 0.822611                              |
| var_All: 0.676689                                 |
```

3 Conclusion

Raw data (especially crowd sourced data) is often not in the format you want. Because of this it is hard to be confident on the veracity of any specific data point, even ignoring ones that are obviously wrong.
Using software to do data analysis can be very convenient, but adds another layer to the process that can cause hard to find errors (i.e. deleting valid data or not deleting invalid data). It is very important to double check any code that is used in data analysis