Stats Project 2 Part 1:

For part 1 of the project, there were 3 programs that we needed to create. The first one is Plotter, the second one is a Salter, and the third one is a Smoother.

Plotter:  
The plotter generates the first data points using the function y = mx + b. The main part of the program generates the data points using the Random() from Java. First, it sets ‘m’ equal to random number between 1-3, and ‘b’ a random number between 1-5. A start point and an end point is required, the default values are 0 and 10 respectively but the user can change them. Using the start and end points, a range is generated by taking the difference between them. From there, it utilizes a simple for loops that assigns the x and y values to an array list.

Here are two scatter plots that were generated using the output from the Plotter:

A graph with blue dots

Description automatically generated

A graph with a line going up

Description automatically generated

Salter:

The salter is quite simple, it’s first task is to read a csv file with x and y values and assign each one to an appropriate array list and then salt the data. The salter method takes in a list and a salt value that is used to generate a random value between 1-salt value so that it can add or subtract the y value based on a random choice.

Here are two scatter plots created in excel based on the output of the Salter:

A graph with blue dots and numbers

Description automatically generated

A graph with a line

Description automatically generated

Unfortunately, because I did not change the salt value for the bigger data set, it looks very similar to the input. Had I used a bigger salt value, I am sure it would had looked somewhat similar to the salted output with the 20 plot points.

Smoother:

Similar to the salter, the smoother reads a csv file and saves the x and y data points into two array lists. This was a little more complicated than the salter but it wasn’t too bad. The smoother takes an array list and a window value. It loops through all the y values using a for loop. First it checks if i < window value and if so, takes the average and adds it to a new array. If the i happens to be bigger than the window value but lower than the size of the array minus the window value, it takes the average and adds it to the new array. Otherwise, it adds however many values are left and averages them and adds it to the new array.

Here are 6 scatter plots produced from the Smoother. There are two different Plot Points and the data was smoother three times:

A graph of a plot

Description automatically generated

A graph with blue dots and numbers

Description automatically generated

A screenshot of a graph

Description automatically generated

It is very interesting to see this, because it used the salted data points produced from the salter and smoothed it significantly after the first time. However, in the 2nd and 3rd time, it kept its shape, but the data points got more condensed. So, the first point went up the y-axis by a little and the last point went down the y-axis by a little as well.

MATLAB

The tutorial is followed to learn MATLAB is from: <https://www.mathworks.com/help/matlab/learn_matlab/desktop.html>

This was a very easy-to-follow tutorial to learn the basics. I found that this was very similar to python, so it was easy to get the hang of it. However, I will admit that I had already learned a little bit of MATLAB like 4 years ago when I was studying for engineering. This was the main reason I switched to Computer Science. First step was to create variables, addition, multiplication and more. The next step was matrices and array creating and indexing which was simple. After that it was string arrays and calling functions. After this, 2-D and 3-D plots came but it was easy as I had previously learned this before and it was very similar to python/pytorch and with that the tutorial was done.

Plotter, Salter, Smoother:

To do this, I simply followed the code that I had already written from Java and made some adjustments. However, I did not create methods as I found it hard to do, so at the moment, everything was hardcoded. I only used an array with 10 values and was able to generate 10 data points using the y = mx + b function. After that was done, I salted and smoothed it and plot it the points.

The following screenshot is the plot that was generated in MATLAB using my program:

A graph with red blue and yellow lines

Description automatically generated

This plot looked very similar to my results from java and the excel sheets. The y = mx + b is a straight line, meanwhile the salted is all over the place and the smoothed data is somewhat in-between the original and the salted data. I really liked MATLAB because it was much easier than Java which is what the next part is about.

JFreeChart and Math – The Commons Math User Guide – Statistics

For this part, rather than creating our code from scratch, the task was to user other functional libraries like JFreeChart. I followed the JFreeChart tutorial from tutorialspoints at: <https://www.tutorialspoint.com/jfreechart/jfreechart_xy_chart.htm>

Unfortunately, while I did installed the Apache Commons Math, I was unable to use it in my program. I instead imported my Plotter, Salter, and Smoother from the first part and used JFreeChart to plot the data points. This was something I have never done before, so even though I followed the tutorial, I was having a lot of trouble trying to plot three datapoints. I was successful in the end and here is the result:

A graph with lines and numbers

Description automatically generated

I really liked the looked of it and the results is very similar to the plot that was done in MATLAB.