JFreeCharts is a library that can be downloaded to java that allows the user to easily use a bunch of charts directly in your IDE. The types of charts include pie, bar, line, scatter, bubble, timeseries, gantt charts. You can also use XYSeries Graphs, which we used in our project. These are some examples of the code and what the graphs look like when you write code and what the outputs look like. You can either have the output saved as a file or output as a window. The following charts are the pie and bar charts.

Text

Description automatically generated

Chart, bar chart

Description automatically generated

Text

Description automatically generated

Chart, pie chart

Description automatically generated

The Apache Common Math library is a library full of math methods that you can call to solve given problems. This library has a wide range of methods and has a dedicated stats library inside of it. From finding the mean, median, and mode to probabilities and distributions. It is said to have a more random number generator that is better than Math.random() and that is normalDistribution.sample(). You could have it solve calculus problems such as integrals to Linear Algebra with its solving matrices. As you can see below, I wrote some examples of what the Apache Common Math library can do.

Text

Description automatically generated

Above is an example of solving for the mean, median, and standard deviation. All you need is the array and add all the values of the array to the stats library. Then, you can use the following methods to find the solutions.

A picture containing logo

Description automatically generated

Above is an example that you can solve probabilities and distributions by using the library. Normal Distribution is a better random number generator compared to the built-in java one.

Graphical user interface, text

Description automatically generated

Above is an example of solving calculus style problems with this library. You can solve integrals very easily just by creating objects inside of the library and calling the integrate method.

Graphical user interface

Description automatically generated with low confidence

Above is an example of solving matrices styled problems. You can solve all the operations of matrices. Just create the matrix object and call the method that you want the program to solve from the library.

Below are all of the outputs from all the examples above. Wanted to show what the outputs would look like if you wrote these programs.

Text

Description automatically generated

Now for the project, the goal is to create a function, salter, and smoother by using Apache Common Math and JFreeCharts. The coding of the function and salter are the same as the CSVTester program. The smoother is the only difference between them. I utilized the Apache library smoother by setting a windowSize, which is the range, and the method inside the library called addValue, which will add the value to the total value. At the end of the loop, I called the method named getMean, which will take the value that is stored in the variable and divide it by the windowSize. As you can see from the graphs of all the programs the smoother gets you close to the original graph.

Chart, line chart

Description automatically generated

The graph of is correct. I had the program plot 50 values. The graph is basic because there is nothing crazy to this part of the program nor the graph.

Chart, scatter chart

Description automatically generated

The salting of doesn’t make the graph look like the original anymore. So, the salter did its job since you cannot tell that the graph used to be . I had the program loop through every value in the array and randomized the y-value. To randomize the y-value I first used a random number generator to decide whether to add to or subtract from the original y-value. From there I used another random number from the range of 0 to 999. After the randomization of all the y-values, I call the method to graph the salter with the new values.

Chart, line chart

Description automatically generated

The smoother is the class that was changed the most since using addon libraries. Apache Common Math library made programming the smoother very easy. To start I looped through the length of the salter array. For every iteration, I first cleared the data in the stats library then checked where the position is in the array by using multiple if statements. Inside the correct if statement I looped through all the values in the range and called the addValue method from the stats library to store the total value. I also set the size of the WindowSize by the range of the loop. Once the loop is complete, I call the getMean method in the stats library and set that value to the y-value at the given position. Then, repeat till the first loop is complete. Finally, I called the JFreeCharts method to graph the smoother for me. From the graph above the smoothed function looks roughly the same as .

I would assume that if I kept smoothing the smoother over again, I would be closer to the original graph.

Work Cited:

<https://www.javatpoint.com/jfreechart-pie-chart>

<https://www.if.pw.edu.pl/~ertman/pojava/?download=jfreechart_tutorial.pdf>

<https://www.baeldung.com/apache-commons-math>