**Plotting, Salting, Smoothing 1 (Java)**

**Plotter:**

A graph with a line

Description automatically generatedMy plotting program has 4 total parameters that can be changed to affect the resulting graph. The calculateFunction method has 2 parameters, lowerRange and upperRange, which determine the range of x values to be calculated and graphed. For example, calling calculateFunction(-25, 26) will calculate for x values ranging from -25 to 25.

calculateFunction(-25, 26);

writeToCSV(51, 1);

A graph of a function

Description automatically generatedAnother parameter is the interval parameter in the writeToCSV method. This determines the interval between points when writing the data from the calculateFunction method to a CSV file.

A graph of a function

Description automatically generatedFor example, if you call calculateFunction(-100, 101), this will calculate all of the x and y values from -100 to 100. Then when calling writeToCSV, if the interval parameter is set to 1, it will plot every single point that was calculated.

writeToCSV(“GraphPoints.csv”, 201, 1);

However, if the parameter is set to 5, it will plot every 5th point instead of all of them.

writeToCSV(“GraphPoints.csv”, 201, 5);

A graph of a function

Description automatically generatedThe last parameter in the GraphPlotter program is numberOfPoints, also in the writeToCSV method. This parameter controls how many of the points that were calculated from the calculateFunction method to write to the CSV.

For example, taking the previous call of writeToCSV and changing numberOfPoints from 201 to 101, it will plot the first 101 points (from -100 to 0) instead of from -100 to 100.

**Salter:**

A graph with blue lines and white text

Description automatically generatedMy salter program has 1 parameter that can affect the data plotted, and that is the saltValueRange parameter in the saltData method. This determines the range of random values that the saltValue could be generated as, which is then added or subtracted from the current y value before being re-randomized.

For example, with a saltValueRange of 100, the saltValue can be generated as a value from 0 to 100, resulting in a salt that still somewhat resembles the original graph.

A graph with blue lines

Description automatically generated

saltData(100)

However, when setting the saltValueRange to 500, the saltValue can be generated from 0 to 500, resulting in a much more volatile salt that doesn’t resemble the original function at all.

saltData(500);

**Smoother:**

My smoothing program also has 1 parameter that can affect the data, and that is the windowValue parameter in the smoothData method. This parameter affects how many y values from the left and right of a point to take in order to calculate the average, which will then be set as the new value of the point. For example, with windowValue set to 3, 3 points to the left and right will be taken for the average. This results in a smooth that is not very clean the first time, and takes 2 more smooths in order to resemble the original parabola.

A graph of a graph

Description automatically generatedA graph with a line

Description automatically generatedA graph with a line

Description automatically generated

But when setting the windowValue to 5, 5 points are taken from the left and right to calculate the average, which results in cleaner smooth that only takes 2 runs to achieve the same result.

A graph with blue lines

Description automatically generatedA graph of a function

Description automatically generated