**Octave User Manual**

Octave version of FunctionPlotter

Cachary Tolentino

CSCI-3327

**Table of Contents**

**Software Description3**

Detailed Description3.1

System Requirements3.2

**Installation Guide4**

**Scripts Overview5**

**Result Analysis8**

**Software Description**

Octave version of FunctionPlotter.

**Detailed Description**

The program performs the same tasks as the FunctionPlotter program. It allows the user to output three separate csv files: an original data set of quadratic values, a salted version, and a smoothened version.

**System Requirements**

**System Requirements**

* A working device, primarily a desktop or laptop
* GNU Octave

**Installation Guide**

To begin using the Octave scripts, simply download the following files.

A screenshot of a computer

AI-generated content may be incorrect.

After downloading the files, simply move the files to the folder containing your project. Once done, you can open GNU Octave. Then change the directory to the folder containing all the files you have downloaded. Finally, you can start using the scripts, especially the “tester.m” script which will run the entire program.

**Scripts Overview**

**tester.m**

The tester.m script simply calls one the other scripts (plotter, salter, and smoother) to produce the original set of data of quadratic values, then its salted output, aswell as its smoothened output. It does not require any parameters nor does it return any result. It works as a tester script for the other scripts.

**plotter.m**

The plotter.m script handles the execution of creating the csv file containing the original data of quadratic values. It asks the user for inputs regarding the variable values for the quadratic function and the name of the file. It will then input all the quadratic values into a data structure. Finally, it will call for the exporter to export the data into a csv file. It does not have any parameters, but it does return the name of the file it created.

**quadraticFunction.m**

The quadraticFunction.m script is a helper function for the plotter.m script. It simply returns the quadratic value based on the given variables of the quadratic function. It requires four parameters: x, a, b, and c. It will return y as a result.

**salter.m**

The salter.m script follows the same procedures as the plotter.m script, but it does not ask the user for any input. The input for the salting range is pre-defined before the script is ran. This is customizable by the user in the script file of tester.m. This script requires three parameters: filename, lowerBoound, and upperBound. The fileName is simply the name of the file that wants to be salted. While the bounds define the range of values that could be used for salting the data. The script also returns the name of the newly salted file.

**salterFunction.m**

The salterFunction.m script is a helper function for the salter.m script. It is tasked to salt the y-values of the provided data. This is performed by iterating through all the y-values of the data and randomly creating a salting value based on the given range. It will then randomly decide to either add onto the existing data or subtract from it. It requires three parameters: data, lowerBound, and upperBound. Data is the data containing the unsalted data values. This function will return the new data containing the salted values.

**smoother.m**

The smoother.m script also follows the exact procedures of the salter.m script. The only difference are in its parameters. It requires two parameters: fileName and windowValue. Window value is a user defined value which represents how many numbers around a certain index value would be averaged. This function does not return any value.

**smootherFunction.m**

The smootherFunction.m script is the helper function for the smoother.m script. It is tasked to smoothen the y-values of the provided data. This is performed by iterating through all the y-values of the data and finding the average of values given the window value. This will replace the current value at the current index. This function requires two parameters: data and windowValue. This function also returns newData which contains the smoothend values.

**exporter.m**

The export.m script simply exports the data provided into a csv file using csvwriter.

**Result Analysis**

(NOTE: This is like the same results from FunctionPlotter)

The following graphs are example outputs (graphed in Excel from the data points of the program provided) of the program. The quadratic output was configured with its quadratic formula as . Here we can see a typical graph for this type of formula with no problems.

A graph with a line drawn on it

AI-generated content may be incorrect.

In the following graphs, we can see a more interesting difference mainly due to the salting and smoothing of the graph above.

In the graph below, we can see the salted version of the quadratic output. Here the salting range used was from 1000 to 5000. This causes the data points to either be increased or decreased from their original value, hence, the points starting to deviate from the straight line in comparison to the original graph.

A graph with blue dots

AI-generated content may be incorrect.

Finally, here we can see the smoothened graph of the salted graph. This used a window value of 10, as such any points on the graph were replaced with the average value of the values from that focus point’s left and right side. This “smoothens” the salted values thus allowing the graph to appear more like the unsalted version.

A graph showing a line

AI-generated content may be incorrect.