Program Salt & Smooth

Statslibrary2

Laiba S Khan

2023

Table of Contents

[Plot, Salt, Smooth 2](#_Toc153392534)

[The Function 2](#_Toc153392535)

[Program Plotter 2](#_Toc153392536)

[Salter 5](#_Toc153392538)

[Smoother 6](#_Toc153392539)

[Maven 6](#_Toc153392540)

[Matlab Octave 6](#_Toc153392541)

[RSI Calculator 11](#_Toc153392542)

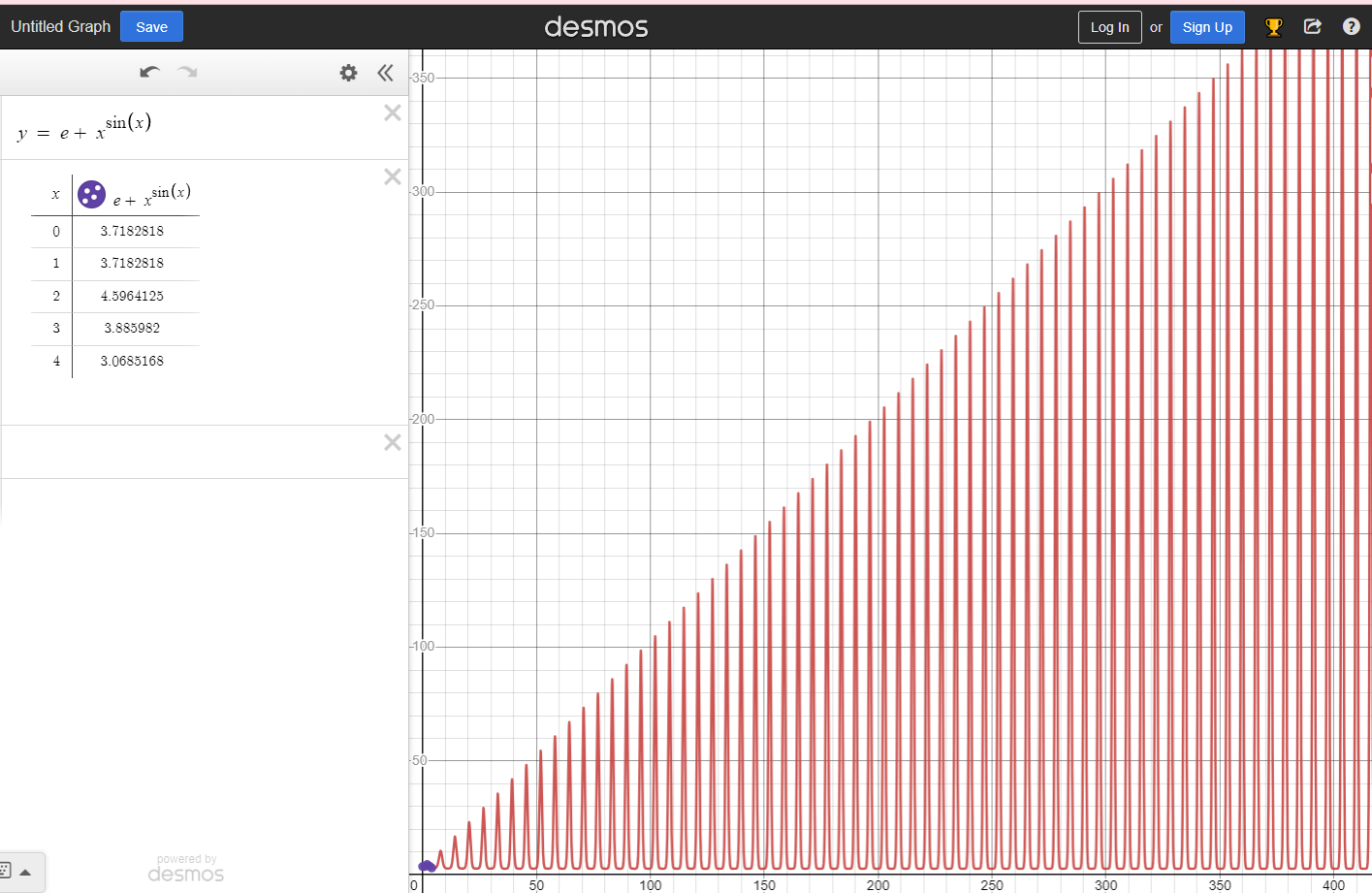
[Normal Distribution, Gamma, Beta Distribution Essay 12](#_Toc153392543)

[Formula Sheet 12](#_Toc153392544)

# Plot Salt Smooth

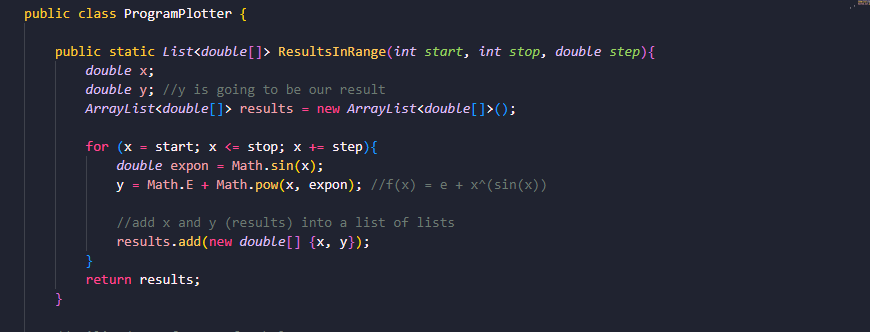
## The Function

The function that was chosen for this assignment was . This equation was chosen due to the way the graph appears, and its higher challenge to implement. The function includes both the eulers number, an exponent, and sin(x).



## Program Plotter

# ProgramPlotter.java was implemented with two methods, one for writing results into a CSV file, and one that calculated the function results. The method ResultsinRange method is as shown below.



The ResultsinRange method takes the parameters for when you want the x-value to start, stop, and the increments of it as well. It takes each input, calculates the result (which is the corresponding y-value), and adds it to a new ArrayList called results. It returns the list, allowing the CSVWriter to use the list and write a CSV file with your choice of file name.

A computer screen shot of text

Description automatically generated

The CSVWriter takes an arraylist (or list within a list), such as what is returned by ResultsinRange, and computes the y value of the equation. It then utilizes FileWriter and BufferedWriter to write the values of x and y as comma separated values. It also writes the headers, one describing the equation, and then input and output labels.

A computer screen shot of text

Description automatically generated

Each test shows different ranges and scenarios. All tests produce results with the correct ranges and steps. Notably, the last one produces a long column of NaN y-values. This is because y is undefined when x < 0.

## Salter

A computer screen shot of a program code

Description automatically generated

A reference was used for method ReadCSVFile. You can find the link within the code, and at the References page of this report. This method skips the first two lines within the CSV file, which are the headers (as written in the CSVWriter method). This reader is only compatible with CSV files that are formatted as such. It then iterates through the lines and stores them in a temporary array, then the returned list.

A computer screen shot of a code

Description automatically generated

Here is a helper function to generate a random salt value

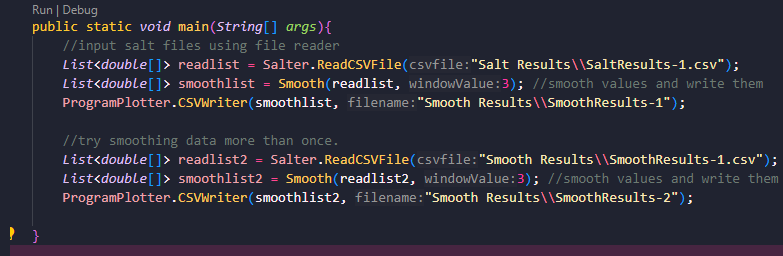
## Smoother

A computer screen shot of a program code

Description automatically generated

# 

The notes above describe the purpose of this code very sufficiently. It takes the input of a List within a list, and then uses temporary lists to iterate through and take the “moving mean”. This outputs a list that you can then use CSVWriter method in ProgramPlotter.java.



This outputs two tests, which both work sufficiently.

# Maven

# Matlab Octave

Continuing into using Matlab Octave, multiple resources were used as supplemental information. These are listed below.

|  |  |  |
| --- | --- | --- |
| Tutorials Used: | | |
| LINK: | | USES: |
| 1. | <https://www.mathworks.com/help/matlab/ref/plot.html> | Do original line plot. |
| 2. | <https://www.youtube.com/watch?v=aD8k4pYUBOk> | Graph design and plotting. |
| 3. | <https://www.mathworks.com/matlabcentral/answers/579033-how-to-add-a-noise-in-my-input-graph> | Noise/salt the graph. |
| 4. | <https://www.mathworks.com/help/matlab/ref/movmean.html> | Smoothing the graph. |

A screenshot of a computer

Description automatically generated

A screen shot of a graph

Description automatically generated

This is the original attempt to plot a line using the first tutorial.



A white background with black dots

Description automatically generated

A graph on a computer screen

Description automatically generated

Here is the graph of the original function plotted. This is read from second CSVfile generated by the first ProgramPlotter test (functionResults -config2). I copied and renamed the file in order to use it for this graph.

A screenshot of a computer code

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer screen

Description automatically generated

Here the noise is graphed on top of the original plot. In the second picture you can see exactly how salted the graph is. As you can see in the code above, I made the salting/noise factor 50, which you can clearly see in the graph as well. The salted is depicted as the purple line, while the unsalted is the yellow.



A close up of text

Description automatically generated

A screenshot of a computer screen

Description automatically generated

A screenshot of a graph

Description automatically generated

The code and graphs shown above are the final versions of the results. These results show the difference between the Programmed, Salted, and Smoothed plot points and graphs. It also shows how easy and dynamic of a program Matlab Octave is, and how it can be useful when evaluating data.

# RSI Calculator



Above is the screenshotted graph of Walt Disney RSI graph from the previous months. This is to show/estimate the potential results garned from the RSICalculator.java methods and whether or not they are correct.

A screen shot of a computer

Description automatically generated

This shows the results printed within the console when testing the calculateRSI method. This prints the last 10 days in the order of newest to oldest, matching their respective inputted days. It shows that the results are about right. They line up the graphs depiction.

# Normal Distribution, Gamma, Beta Distribution Essay

# Formula Sheet

JfreeChart and Apache:

<https://commons.apache.org/proper/commons-math/userguide/stat.html>

Using the StatUtils utility class:

1.2

// Compute statistics directly from the array

// assume values is a double[] array

double mean = StatUtils.mean(values);

double std = FastMath.sqrt(StatUtils.variance(values));

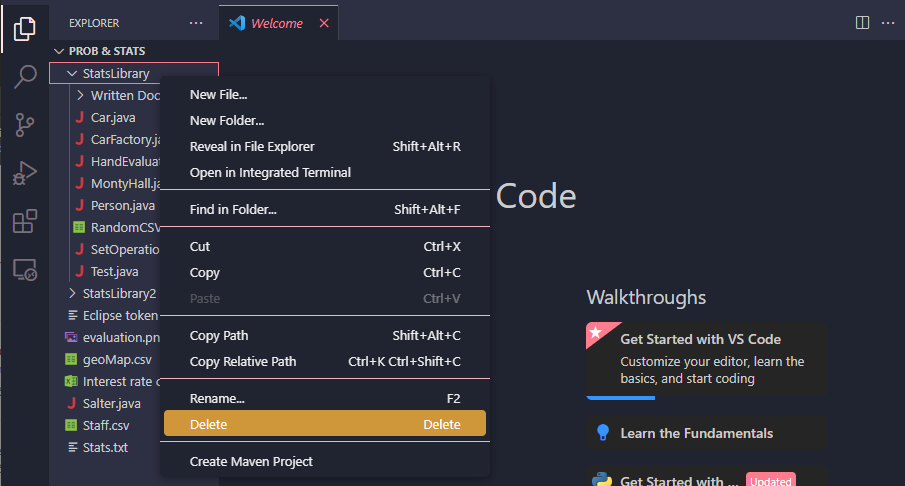
double median = StatUtils.percentile(values, 50);

// Compute the mean of the first three values in the array

mean = StatUtils.mean(values, 0, 3);

You can use this to do the smoothing

Or when using Mav select Create Maven Project after downloading maven extension (if you don’t already have it)



A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generatedA screenshot of a computer

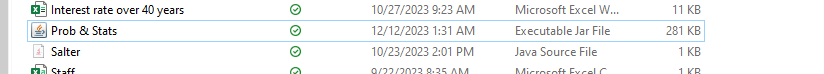
Description automatically generatedA screen shot of a computer program

Description automatically generated

After right clicking maven project and selecting custom command

A screen shot of a computer program

Description automatically generated



<https://finance.yahoo.com/quote/DIS/history/>