# Java API CSV : Report

By: Ryan Minneo

\*\*\*As with the other reports, all of the data will be on one google sheet. While there is a snip of the JFreeChart generated by my program, there will also be separated graphs and a combined graph in the google sheet.

[Java API : Plotter, Salter, Smoother](https://docs.google.com/spreadsheets/d/1GVmoYDPGN4QzYjyXxwy0yYKUKEZBV0_Tf7hp8KJJI1M/edit?usp=sharing)

In the other csv reports I separated the report into plotting, salting, and smoothing each with their own journal and result coverage. Here I designed my program in a way that all of the methods, which plot, salt, and smooth are all in the same file. I decided to go with an Overall Journal, since using the API tied in to all the various cogs in this mechanism of a project.

## Overall Journal

To begin the code, importing a few API’s into my IDE, eclipse, was necessary. These imports included JFreeCharts, Apache, and JFreeCommon. After importing those I wanted to search for a way to create a dataset, similar to a 2d array, for the plotter. This came in the form of an XYSeries which was implemented without a hitch. I also used a PolynomialFunction object in order to simplify the process of creating the data. After finishing the plot method by adding the series “data” to the class variable “dataset” . Since I wanted to be able to test if the plot method worked before moving onto the salter, I worked on implementing the actual chart.

The method “generateGraphs()” was a way to call the plot, salt, and smooth methods and then display them without a bunch of messy code. Implementing the chart with JFreeChart was particularly easy, and using the plot, salt, and smooth methods to add different “data” to the “dataset” allowed me to test the plot method and the displayability without actually implementing the salt and smooth methods yet.

The salt method was actually easier to implement than the plot method. A simple for loop going through the data set and +/- a value between [0, range] to every y value.

The smooth method took a little bit longer to implement correctly. At first I found a way to smooth it based on the Savitzky Golay Filter. However, it didn’t make much sense to me and I couldn’t get it to work properly. I eventually found the DescriptiveStatistics class in apache, which made smoothing much easier and there is a windowValue which can change.

After implementing all of these and making sure the data sets were being formed correctly, I made a generateCSVs method which takes each of the XYSeries’ and converts them into CSVs. I reused the code that was in my GraphPlotter from doing the base Java Plotter to convert them into CSVs, just had to modify it a tiny bit.

## Overall Results

Everything came out extremely well, though the smoother doesn’t seem as close to the original as the Octave and base Java Plotters. Either way, the CSVs worked out and the graph made from JFreeCharts came out incredibly well:

