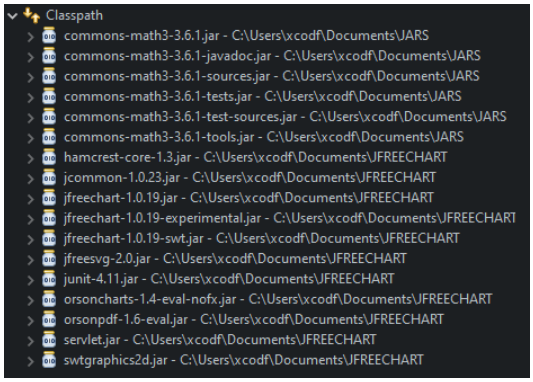
Nick Fichera

JFreeChart Report

So, initially I wasn't too sure how to start. I firstly imported all the jar files that I needed. This included all the following that was imported into my eclipse project. This included the JFreeChart libraries as well as Apache commons.



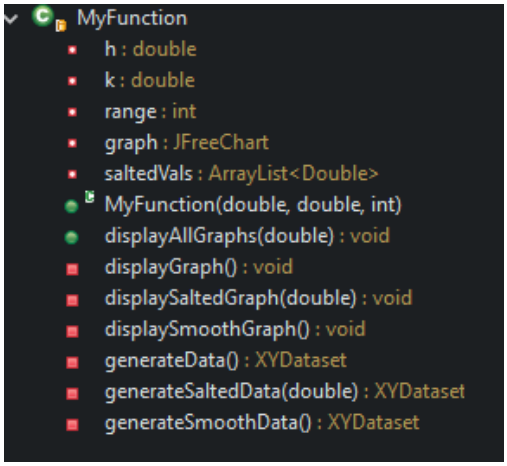
Moving forward I started to experiment with JFreeCharts. Initially, I tried doing JFreeChart j = new JFreeChart(), however I could not figure it out until I looked it up on the internet, where is where I learned about the ChartFactory class that could be used to create a JFreeChart object. After finding this out, I created a constructor for my class with constructor parameters h, k and range as this function is quadratic. I also created an ArrayList for the salted values that the smoothing function would use.

```
private double h;
private double k;
private int range;
private JFreeChart graph;
private ArrayList<Double> saltedVals;

public MyFunction(double h, double k, int range) {
    this.h = h;
    this.k = k;
    this.range = range;
    graph = ChartFactory.createXYLineChart("Function", "X", "Y", generateData(), PlotOrientation.VERTICAL, true,
            false, false);
    saltedVals = new ArrayList<>();
}
```

Afterword's, I created three functions that all return XYDataset. The function generateData() generates the regular graph; generateSaltedData() the salted graph; and generateSmoothData() the smoothed graph. generateSmoothData() accesses the saltedVals ArrayList, which is created by generateSaltedData() and relies on this method behind called first.

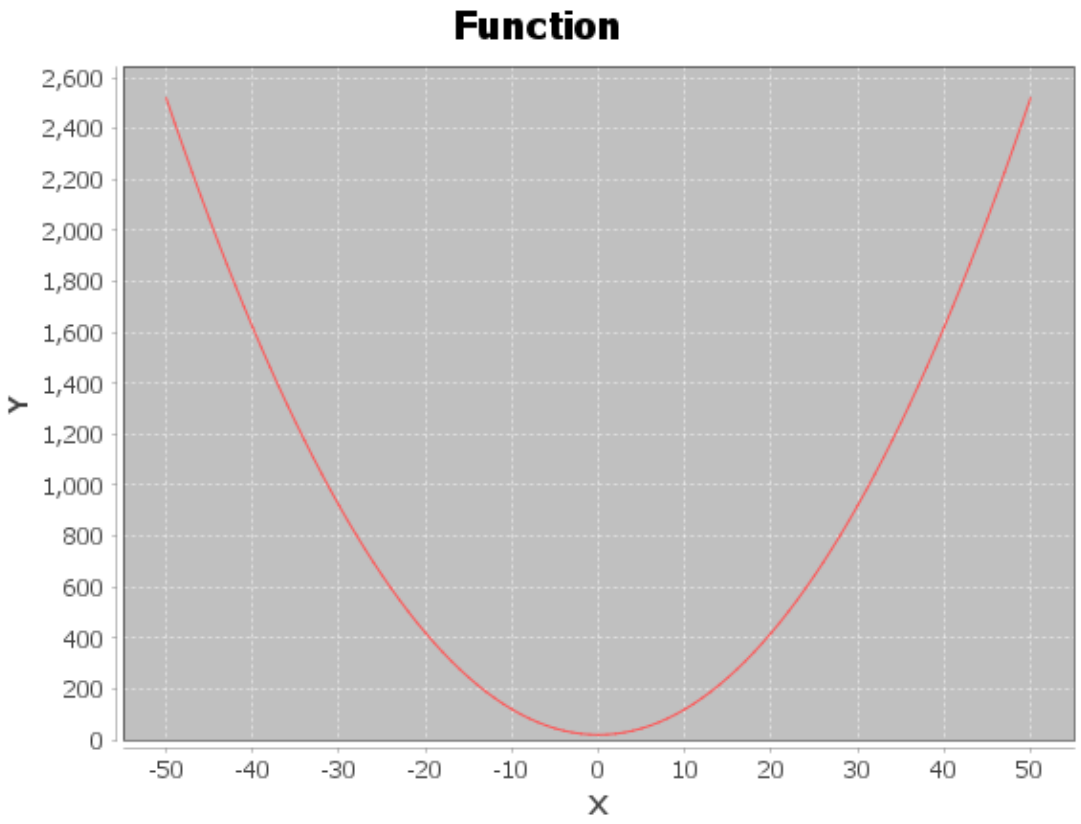
```
private XYDataset generateSmoothData() {
    XYSeries xy = new XYSeries("series");

    DescriptiveStatistics d = new DescriptiveStatistics();
    d.setWindowSize(2);

    for (int i = -range; i <= range; i++) {
        d.addValue(saltedVals.remove(0));
        xy.add(i, d.getMean());
    }

    XYSeriesCollection dataset = new XYSeriesCollection();
    dataset.addSeries(xy);

    return dataset;
}
```

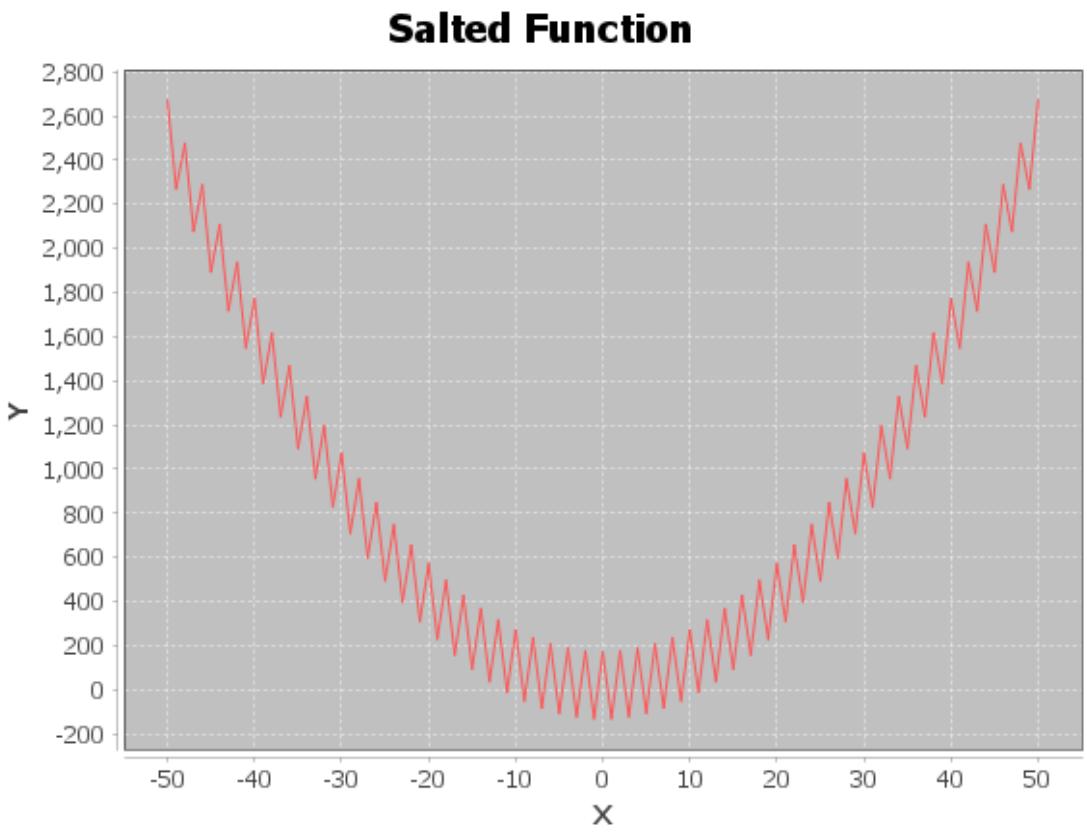
Lastly, I implemented four more functions. displayGraph(), displaySaltedGraph(), displaySmoothGraph(), and displayAllGraphs(). These functions take output from the three functions described above to create a ChartFrame to display the graph. displayAllGraphs() is the only public method (other than the constructor). The reason for this is that one, it makes the program more simple and easy to use, and two, it ensures that generateSaltedData() is called before generateSmoothData() because the salted data is generated in the generateSaltedData() method and generateSmoothData() needs to access the ArrayList that it has modified.

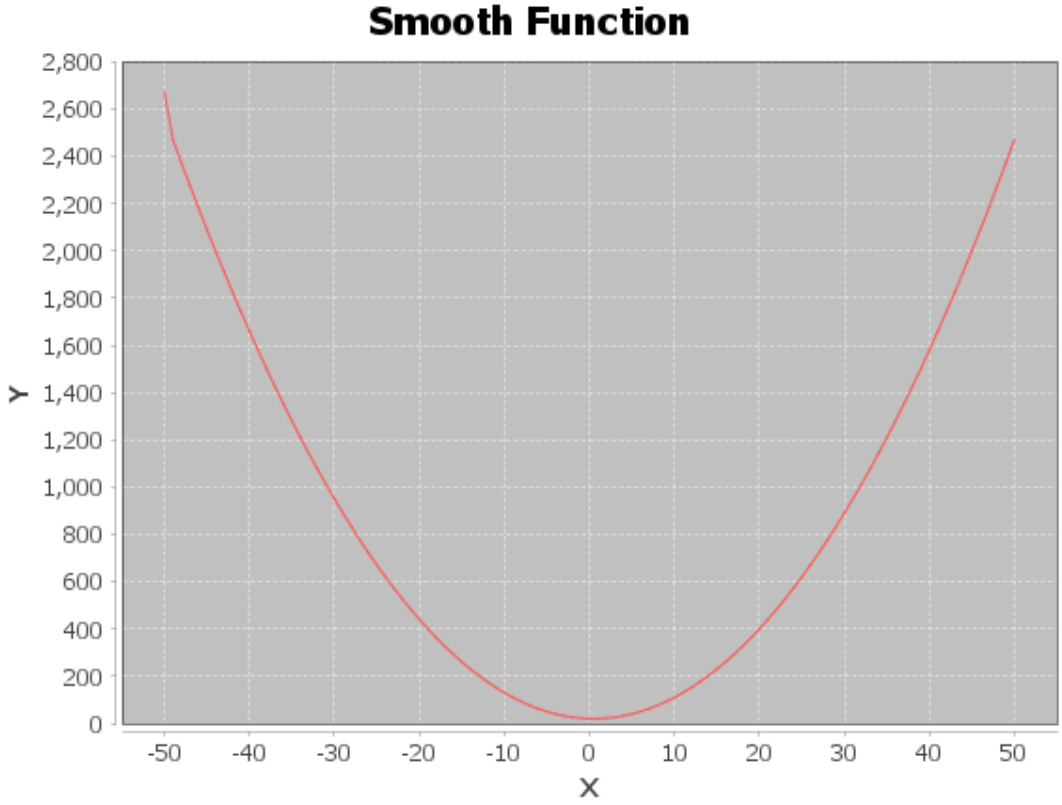



```
public static void main(String[] args) {
    new MyFunction(0, 20, 50).displayAllGraphs(150);
}
```

In a separate file, I have the main method which creates a MyFunction object and called displayAllGraphs() on it. The results of this are:




Salted Function
2,800
2,600
2,400
2,200
2,000
1,800
1,600
1,400
1,200
1,000
800
600
400
200
0
-200
Y
-50
-40
-30
-20
-10
0
10
20
30
40
50
X


Smooth Function
2,800
2,600
2,400
2,200
2,000
1,800
1,600
1,400
1,200
1,000
800
600
400
200
0
Y
-50
-40
-30
-20
-10
0
10
20
30
40
50
X