SimplePlotSaltSmooth Package Documentation

May 6, 2025

Introduction

This document provides comprehensive documentation for the `SimplePlotSaltSmooth` package, which includes six Java classes: `PlotterGraph`, `PlotterGraphTester`, `SaltData`, `Smoother`, `TestorSalt`, and `TestorSmooth`. The package is designed to generate, manipulate, and export data for a polynomial function ($y = x^3$), with functionality to add random noise (salting) and smooth the data using a moving average. Data is stored and exported as CSV files. The `PlotterGraph` class generates the base polynomial data, `SaltData` applies noise, `Smoother` smooths the data, and the tester classes (`PlotterGraphTester`, `TestorSalt`, `TestorSmooth`) demonstrate the workflow. This documentation covers the purpose, methods, fields, dependencies, and example usage of each class.

1 PlotterGraph Class

The `PlotterGraph` class represents a point in a polynomial graph ($y = x^3$) and provides methods to generate coordinates and export them to a CSV file.

Class Overview

- **Package**: `SimplePlotSaltSmooth`
- **Purpose**: Generate and export coordinates for the polynomial $y = x^3$.
- **Dependencies**: `java.io.*`, `java.util.*`, `java.lang.Math`
- **Constructor**: `PlotterGraph(int x, int y)`

Fields

- `private int x`: The x-coordinate of a point.
- `private int y`: The y-coordinate of a point.

Methods

- **`public PlotterGraph(int x, int y)`**
  - **Description**: Initializes a point with given x and y coordinates.
  - **Parameters**:
    - x: X-coordinate (integer).
    - y: Y-coordinate (integer).
- **`public int getX()`**
  - **Description**: Returns the x-coordinate.
  - **Returns**: X-coordinate as an integer.
- **`public void setX(int x)`**

    - **Description**: Sets the x-coordinate.
    - **Parameters**:
        - `x`: New x-coordinate.
- **`public int getY()`**
    - **Description**: Returns the y-coordinate.
    - **Returns**: Y-coordinate as an integer.
- **`public void setY(int y)`**
    - **Description**: Sets the y-coordinate.
    - **Parameters**:
        - `y`: New y-coordinate.
- **`public static List¡PlotterGraph¿ generateCoordinates(int startingX, int endX)`**
    - **Description**: Generates coordinates for the polynomial $y = x^3$ over the range [`startingX`, `endX`] with a step size of 1. Prints coordinates to the console.
    - **Parameters**:
        - `startingX`: Starting x-value.
        - `endX`: Ending x-value.
    - **Returns**: List of `PlotterGraph` objects representing points.
- **`public static void ExportToCSV(List¡PlotterGraph¿ PolynomialGraph)`**
    - **Description**: Exports the list of points to a CSV file named `PolynomialGraph.csv` with columns `x` and `y`.
    - **Parameters**:
        - `PolynomialGraph`: List of `PlotterGraph` objects.
    - **Exceptions**: Catches `IOException` and prints an error message.

Example Usage

```
List¡PlotterGraph¿ graph = PlotterGraph.generateCoordinates(-5, 5);
PlotterGraph.ExportToCSV(graph);
```

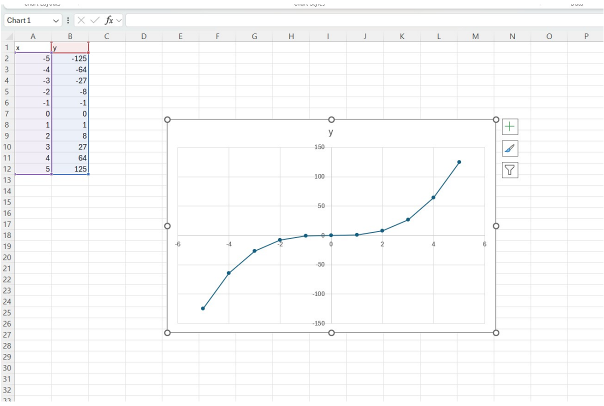
This generates coordinates for $y = x^3$ from $x = -5$ to $x = 5$ and exports them to `PolynomialGraph.csv`.

2 PlotterGraphTester Class

The `PlotterGraphTester` class serves as the entry point for generating and exporting the base polynomial data.

Class Overview

- **Package**: `SimplePlotSaltSmooth`
- **Purpose**: Test the `PlotterGraph` class by generating and exporting coordinates.
- **Dependencies**: `java.util.List`, `PlotterGraph`
- **Main Method**: `public static void main(String[] args)`



x	y
-5	-125
-4	-64
-3	-27
-2	-8
-1	-1
0	0
1	1
2	8
3	27
4	64
5	125

Figure 1: Polynomial Graph from CSV output

Methods

- **`public static void main(String[] args)`**
  - **Description**: Generates coordinates for $y = x^3$ from $x = -5$ to $x = 5$ and exports them to `PolynomialGraph.csv`.

Example Usage

```
java PlotterGraphTester
```

This runs the program, generating and exporting the polynomial data.

3 SaltData Class

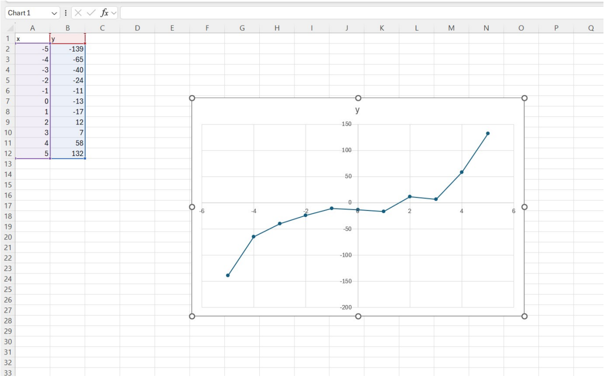
The `SaltData` class imports polynomial data from a CSV file, applies random noise (salting), and exports the salted data to a new CSV file.

Class Overview

- **Package**: `SimplePlotSaltSmooth`
- **Purpose**: Apply random noise to polynomial data and handle CSV import/export.
- **Dependencies**: `java.io.*`, `java.util.*`, `PlotterGraph`

Methods

- **`public static List¡PlotterGraph¿ importFromCSV(String fileName)`**
  - **Description**: Reads x and y coordinates from a CSV file, skipping the header row.
  - **Parameters**:
    - `fileName`: Path to the CSV file.
  - **Returns**: List of `PlotterGraph` objects.
  - **Exceptions**: Catches `IOException` and `NumberFormatException`, printing error messages.
- **`public static List¡PlotterGraph¿ applySalt(List¡PlotterGraph¿ data)`**



x	y
-5	-139
-4	-65
-3	-40
-2	-24
-1	-11
0	-13
1	-17
2	12
3	7
4	58
5	132

Figure 2: Salted Graph from CSV output

  - **Description**: Applies random noise (SALT) to the y-values of the input data. The noise is in the range [-20, 20] by default, with an option to scale based on y-value magnitude (disabled by default).
  - **Parameters**:
    * `data`: List of `PlotterGraph` objects.
  - **Returns**: List of `PlotterGraph` objects with salted y-values.

- **`public static void exportToCSV(List¡PlotterGraph¿ data, String fileName)`**
  - **Description**: Exports the list of points to a CSV file with columns `x` and `y`.
  - **Parameters**:
    * `data`: List of `PlotterGraph` objects.
    * `fileName`: Output CSV file name.
  - **Exceptions**: Catches `IOException` and prints an error message.

Example Usage

```
List¡PlotterGraph¿ data = SaltData.importFromCSV(”PolynomialGraph.csv”);
List¡PlotterGraph¿ saltedData = SaltData.applySalt(data);
SaltData.exportToCSV(saltedData, ”SaltedPolynomialGraph.csv”);
```

4 Smoother Class

The `Smoother` class imports salted polynomial data from a CSV file, applies a moving average to smooth the data, and exports the smoothed data to a new CSV file.

Class Overview

- **Package**: `SimplePlotSaltSmooth`
- **Purpose**: Smooth polynomial data using a moving average and handle CSV import/export.
- **Dependencies**: `java.io.*`, `java.util.*`, `PlotterGraph`

Methods

- **`public static List¡PlotterGraph¿ importFromCSV(String fileName)`**
  - **Description**: Reads x and y coordinates from a CSV file, skipping the header row.

  - **Parameters**:
    * `fileName`: Path to the CSV file.
  - **Returns**: List of `PlotterGraph` objects.
  - **Exceptions**: Catches `IOException` and `NumberFormatException`, printing error messages.

- **`public static List¡PlotterGraph¿ smoothData(List¡PlotterGraph¿ data)`**
  - **Description**: Smooths the y-values using a moving average with a window size of 11 (5 points before, current point, 5 points after). The average is rounded to the nearest integer.
  - **Parameters**:
    * `data`: List of `PlotterGraph` objects.
  - **Returns**: List of `PlotterGraph` objects with smoothed y-values.

- **`public static void exportToCSV(List¡PlotterGraph¿ data, String fileName)`**
  - **Description**: Exports the list of points to a CSV file with columns `x` and `y`.
  - **Parameters**:
    * `data`: List of `PlotterGraph` objects.
    * `fileName`: Output CSV file name.
  - **Exceptions**: Catches `IOException` and prints an error message.

Example Usage

```
List¡PlotterGraph¿ saltedData = Smoother.importFromCSV(”SaltedPolynomialGraph.csv”)
    ;
List¡PlotterGraph¿ smoothedData = Smoother.smoothData(saltedData);
Smoother.exportToCSV(smoothedData, ”SmoothedCSVFile.csv”);
```

5 TestorSalt Class

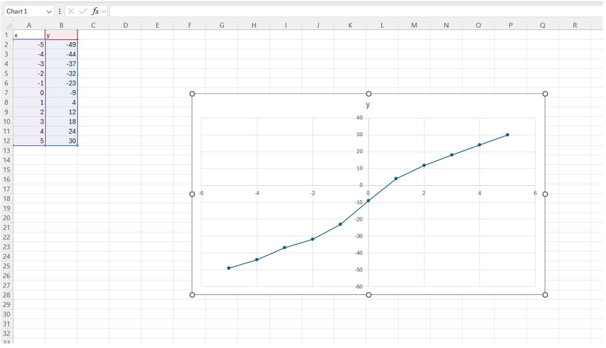
The `TestorSalt` class tests the salting functionality by importing polynomial data, applying noise, and exporting the salted data.

Class Overview

- **Package**: `SimplePlotSaltSmooth`
- **Purpose**: Test the `SaltData` class.
- **Dependencies**: `java.util.List`, `SaltData`, `PlotterGraph`
- **Main Method**: `public static void main(String[] args)`

Methods

- **`public static void main(String[] args)`**
  - **Description**: Imports data from `PolynomialGraph.csv`, applies the SALT algorithm, and exports the result to `SaltedPolynomialGraph.csv`.



x	y
-5	-49
-4	-44
-3	-37
-2	-32
-1	-23
0	-9
1	4
2	12
3	18
4	24
5	30

Figure 3: Smoothed Graph from CSV output

Example Usage

```
java TestorSalt
```

6 TestorSmooth Class

The `TestorSmooth` class tests the smoothing functionality by importing salted data, applying a moving average, and exporting the smoothed data.

Class Overview

- **Package**: `SimplePlotSaltSmooth`
- **Purpose**: Test the `Smoother` class.
- **Dependencies**: `java.util.List`, `Smoother`, `PlotterGraph`
- **Main Method**: `public static void main(String[] args)`

Methods

- **`public static void main(String[] args)`**
  - **Description**: Imports data from `SaltedPolynomialGraph.csv`, smooths it, and exports the result to `SmoothedCSVFile.csv`.

Example Usage

```
java TestorSmooth
```

7 Dependencies

- **Java Standard Library**: For file I/O and data structures (`java.io.*`, `java.util.*`, `java.lang.Math`).

8 Usage Notes

- The CSV files must follow the format with `x,y` headers and integer values.
- The `PolynomialGraph.csv` file must exist before running `TestorSalt`, and `SaltedPolynomialGraph.cs` must exist before running `TestorSmooth`.
- The salting range in `SaltData` is fixed at [-20, 20] unless `scaleSalt` is enabled.

- The smoothing window size in `Smoother` is fixed at 11; adjust `windowSize` for different smoothing effects.
- Console output can be removed for production use to improve performance.
- Ensure write permissions for the output CSV files.