Project 3

Joe DeLizza and Joshua Distefano

# Stats Library: Distributions

**Binomial Distribution**

public double biDist(int n, int y, double p, double q)

Binomial distribution models the number of successes in a sample size n drawn with replacement from population Size N. The biDist function returns the calculated binomial distribution given the four parameters. The function calls the combinations function from SetOperations on the parameters x and y and multiplies it by

**Hyper Geometric Distribution**

public double geoDist(int y, double p, double q)

Geometric distribution models the probability distribution of the number of fails before the first success. The geoDist returns the calculated geometric distribution given the three parameters. The function utilizes Java’s Math.pow function to calculate numbers raised to a power.

**Hyper Geometric Distribution**

public double hyperGeoDist(int r, int y, int N, int n)

Hyper geometric distribution is a discrete probability distribution that describes the probability of success in draws, without replacement. The hyperGeoDist function calculates the hyper geometric distribution by utilizing the combinations function from SetOperations on the given parameters.

**Poisson Distribution**

public double poisDist(int y, double l)

public double poisDist(int y, int n, double p)

The Poisson distribution is a statistical distribution showing the likely number of times that an event will occur within a specified period of time. The poisDist function is an overloaded function one for calculating the distribution given lambda and one for calculating the distribution with n and p.

**Probability of Continuous Random variables**

public double probDensFunc(double a, double b, double... c )

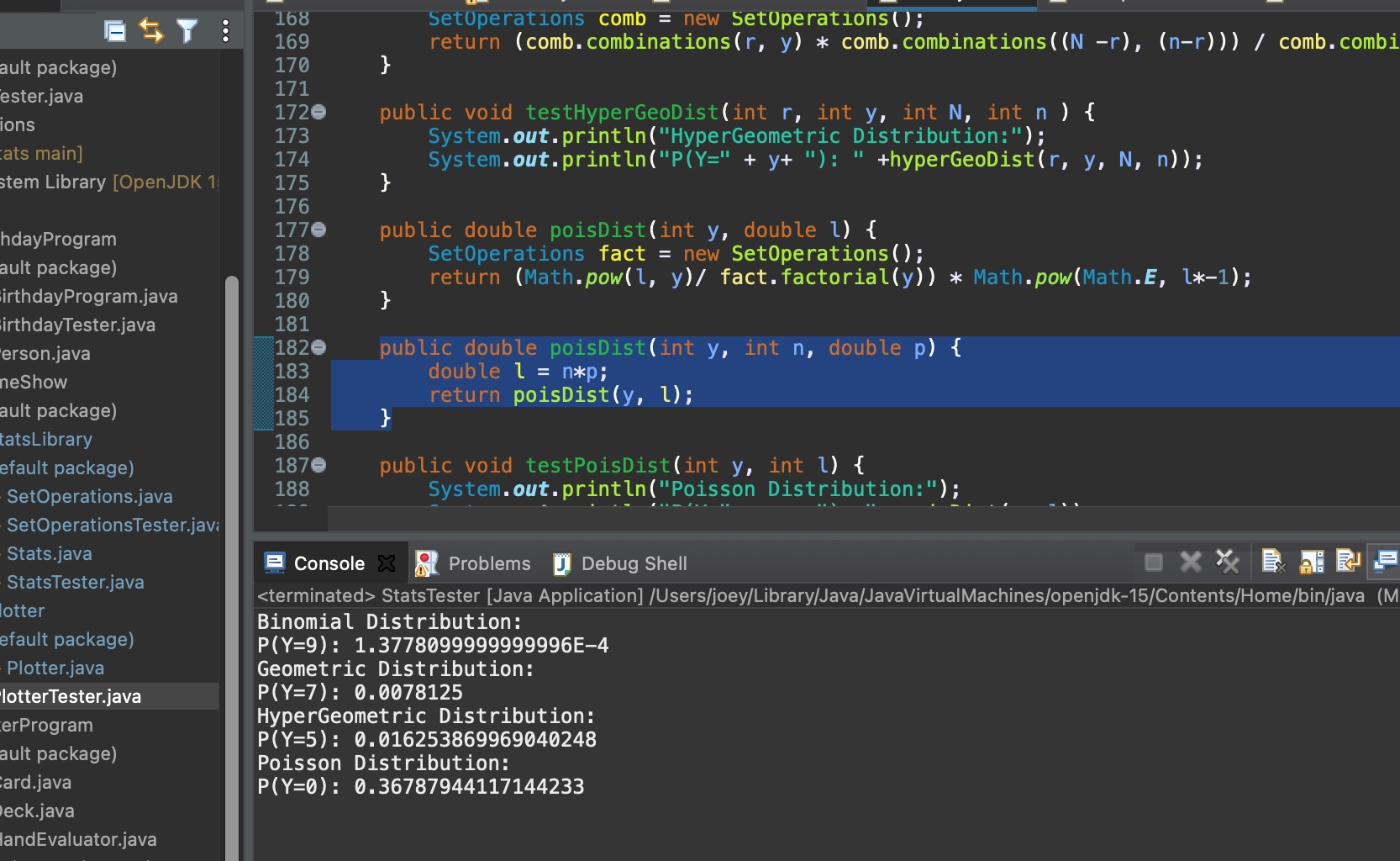
Not all data points can be described discretely. An example of something continuous would be anything dealing with time, as it can be infinitely split between an interval. The probDensFunc() returns the probability between an interval [a,b] given a continuous function. This function constructs a polynomial function from a user in the form of an array given by the user in the form c[0] = constant, c[1] = x, c[2] = x^2 where the user input is the coefficient. The function utilizes the Apache Math library to construct the function and integrate it from [a,b] to return the probability. The implementation of integration used is Simpson’s Rule, There are also functions that return the expected value and variance.

**Uniform Probability Distribution**

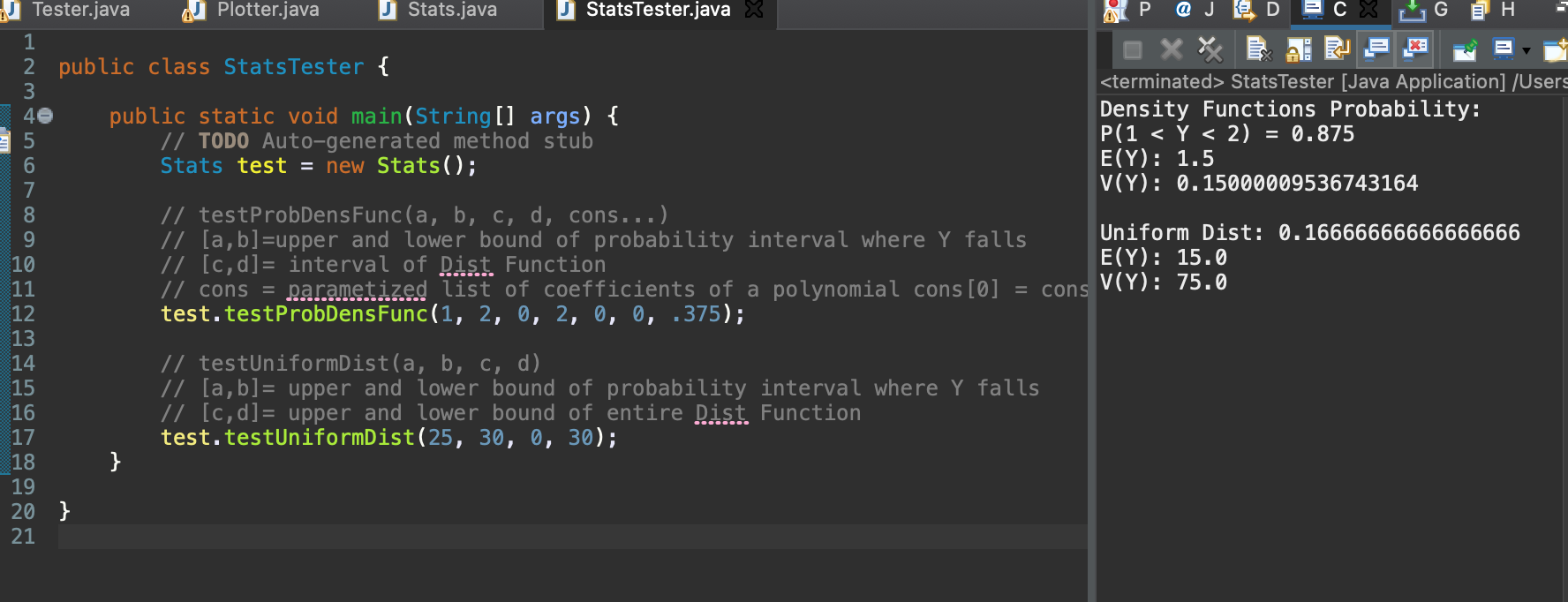
public double UniformDist(double c, double d, double a, double b) {

Given a random variable has a uniform distribution, this function returns the probability that the random variable falls between the interval [c,d]. The function creates a constant polynomial function of and integrates the function from [c,d]. This function also uses Simpson’s rule for integration.

**Discrete Distribution Outputs**



**Continuous Outputs**

****

# Apache Smoother

The Apache Smoother utilizes the Apache Math and JFreeCharts libraries to smooth data points read from a csv file and plot them on a graph. From the Apache Math library the program uses the libraries Stat.Util.mean(dataPoints, i, windowSize) to smooth the points.

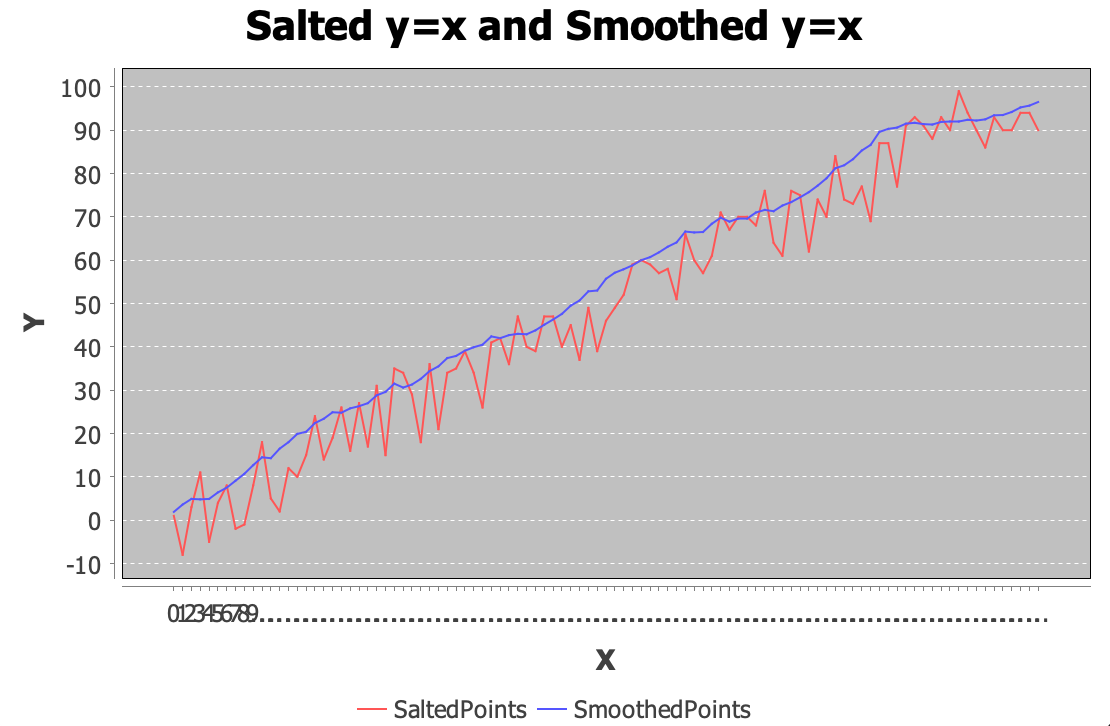
public ArrayList<Double> grabYPointsFromFile() throws IOException {

This function grabs and stores the y-axis points from a csv file into an ArrayList.

public double[] smoothYPoints(int windowSize) throws IOException

This function takes the average from the interval [ i, windowSize ] for every point in the passed array. To avoid getting an IndexOutOfBounds error the function starts decrementing the windowSize before the window can go out of bounds.

**JFreeCharts Output**



# Octave Smoother

The octave smoother utilizes the scientific programming language Octave to smooth and plot points from a given csv file. The script reads in a given csv file with x and y coordinates, and extracts the y-coordinates. Then uses Octaves movmean function to smooth the y-points around a given window size and then plots the results.

