Calbay, Franz Arvin 4CSB

Cunanan, Patrick Bryan GE-ELEC Project

Francisco, Angelica Jane

Santos, Angel Luis

Lagrange Polynomial Interpolation Method

I. Introduction

The Lagrange Polynomial Interpolation method, published and named after Joseph Louis Lagrange (1975) and discovered by Edward Warring (1779), is used mostly for polynomial interpolation. This method is used when we want to describe the ups and downs in a data set and hit every point. The method can be a tool to create a polynomial that goes through any desired set of points.

II. Formula

The formula of the Lagrange Polynomial Interpolation Method is:

$$f(x) = \frac{(x-x_1)\dots(x-x_n)}{(x_0-x_1)\dots(x_0-x_n)}f_0 + \cdots \frac{(x-x_0)\dots(x-x_{n-1})}{(x_n-x_0)\dots(x_n-x_{n-1})}f_n$$

III. Program

The program is a developed using Java Programming Language. The program will ask first how many pairs will be inputted for the computation. After you have inputted the pairs, it will ask what is the value of X. After inputting the value of x, the program will solve the value of P(x) and the resulting equation based on the formula of the Lagrange Polynomial Interpolation.

The following is the solving function of the program:

```
public double solve() {
    double answer = 0;

    Polynomial prod;
    ArrayList<double[]> terms = new ArrayList<>();
    double[] coeff;

    // Iterate each pair
    for (Pair p : pairs) {
        double numerator = 1;
        double denominator = 1;
        prod = new Polynomial(new double[]{1.0});

        // Compute for each term
        for (Pair q : pairs) {
            if (!p.same(q)) {
                numerator *= (x - q.getX());
                denominator *= (p.getX() - q.getX());
                Polynomial poly = new Polynomial(new double[]{1.0, -q.getX()});
                prod = prod.multiply(poly);
            }
        }
        coeff = prod.getCoefficient();
        for(int i=0; i<coeff.length; i++){
            coeff[i] *= p.getY()/denominator;
        }
        terms.add(coeff);

        // Multiply the term with Y value
        System.out.print(numerator + " / " + denominator + " * " + p.getY() + " = ");
        System.out.println((numerator / denominator) * p.getY());
        answer += (numerator * p.getY() / denominator);
    }

    // Print polynomial equation
    System.out.println("Equation = " +strEquation(addCoefficients(terms)));
    return answer;
}
```

IV. Sample Outputs

Sample Outputs of the Program:

Given: (12, 23), (23, 34), (34, 45), (45, 56), (56, 67); x = 10

```
How many pairs?: 5
Enter pair (x,y): 12,23
Enter pair (x,y): 23,34
Enter pair (x,y): 34,45
Enter pair (x,y): 45,56
Enter pair (x,y): 56,67
Value of x in P(x)?: 10
502320.0 / 351384.0 * 23.0 = 32.879584727819136
77280.0 / -87846.0 * 34.0 = -29.910525237347176
41860.0 / 58564.0 * 45.0 = 32.16481114677959
28704.0 / -87846.0 * 56.0 = -18.29820367461239
21840.0 / 351384.0 * 67.0 = 4.164333037360836
Equation =  1.000000000000020x^1 + 10.999999999999886
P(10) = 20.999999999999996
```

Given: (1, 3), (2, 4), (3, 5), (4, 6), (5, 7), (6, 8), (7, 9); x = 100

```
How many pairs?: 6
Enter pair (x,y): 1,3
Enter pair (x,y): 2,4
Enter pair (x,y): 3,5
Enter pair (x,y): 4,6
Enter pair (x,y): 5,7
Enter pair (x,y): 6,8
Value of x in P(x)?: 100
8.14930368E9 / -120.0 * 3.0 = -2.03732592E8
8.23245984E9 / 24.0 * 4.0 = 1.37207664E9
8.31733056E9 / -12.0 * 5.0 = -3.4655544E9
8.40396942E9 / 12.0 * 6.0 = 4.20198471E9
8.492432256E9 / -24.0 * 7.0 = -2.476959408E9
8.58277728E9 / 120.0 * 8.0 = 5.72185152E8
Equation =  0.999999999999986x^1 + 2.000000000000000
P(100) = 102.0
```