Университет ИТМО

Факультет программной инженерии и компьютерной техники

Лабораторная работа №4
по «Вычислительной математике»

Вариант 9

Выполнил: Кривошейкин Сергей

Группа P3214

Преподаватель: Малышева Т. А.

Санкт-Петербург

2020

Цель работы: найти функцию, являющуюся наилучшим приближением заданной табличной функции по методу наименьших квадратов.

Ход работы:

Линейная аппроксимация:

Рассмотрим в качестве эмпирической формулы линейную функцию:

$$\varphi(x, a, b) = ax + b$$

Сумма квадратов отклонений запишется следующим образом:

$$S = S(a, b) = \sum_{i=1}^{n} \varepsilon_i^2 = \sum_{i=1}^{n} [\varphi(x_i) - y_i]^2 = \sum_{i=1}^{n} (ax_i + b - y_i)^2 \to min$$

Для нахождения *a* и *b* необходимо найти минимум функции *S(a,b).* Необходимое условие существования минимума для функции *S*:

$$\begin{cases} \frac{\partial S}{\partial a} = 0 \\ \frac{\partial S}{\partial b} = 0 \end{cases} \quad \text{или} \quad \begin{cases} 2\sum_{i=1}^{n} (ax_i + b - y_i)x_i = 0 \\ 2\sum_{i=1}^{n} (ax_i + b - y_i) = 0 \end{cases}$$

Упростим полученную систему:

$$\begin{cases} a\sum_{i=1}^{n} x_i^2 + b\sum_{i=1}^{n} x_i = \sum_{i=1}^{n} x_i y_i \\ a\sum_{i=1}^{n} x_i + bn = \sum_{i=1}^{n} y_i \end{cases}$$

Введем обозначения:

$$SX = \sum_{i=1}^{n} x_i, \qquad SXX = \sum_{i=1}^{n} {x_i}^2, \qquad SY = \sum_{i=1}^{n} y_i, \qquad SXY = \sum_{i=1}^{n} x_i y_i$$

Получим систему уравнений для нахождения параметров a и b:

$$\begin{cases} aSXX + bSX = SXY \\ aSX + bn = SY \end{cases},$$

из которой находим:

$$a = \frac{SXY * n - SX * SY}{SXX * n - SX * SX}, \qquad b = \frac{SXX * SY - SX * SXY}{SXX * n - SX * SX}$$

Квадратичная аппроксимация:

Рассмотрим в качестве эмпирической формулы квадратичную функцию:

$$\varphi(x, a_0, a_1, a_2) = a_0 + a_1 x + a_2 x^2$$

Сумма квадратов отклонений запишется следующим образом:

$$S = S(a_0, a_1, a_2) = \sum_{i=1}^{n} (a_0 + a_1 x_i + a_2 x_i^2 - y_i)^2 \to min$$

Приравниваем к нулю частные производные *S* по неизвестным параметрам, получаем систему линейных уравнений:

$$\begin{cases} \frac{\partial S}{\partial a_0} = 2\sum_{i=1}^{n} (a_0 + a_1 x_i + a_2 x_i^2 - y_i) = 0 \\ \frac{\partial S}{\partial a_1} = 2\sum_{i=1}^{n} (a_0 + a_1 x_i + a_2 x_i^2 - y_i) x_i = 0 \\ \frac{\partial S}{\partial a_2} = 2\sum_{i=1}^{n} (a_0 + a_1 x_i + a_2 x_i^2 - y_i) x_i^2 = 0 \end{cases} \quad \begin{cases} a_0 n + a_1 \sum_{i=1}^{n} x_i + a_2 \sum_{i=1}^{n} x_i^2 = \sum_{i=1}^{n} y_i \\ a_0 \sum_{i=1}^{n} x_i + a_1 \sum_{i=1}^{n} x_i^2 + a_2 \sum_{i=1}^{n} x_i^3 = \sum_{i=1}^{n} x_i y_i \\ a_0 \sum_{i=1}^{n} x_i^2 + a_1 \sum_{i=1}^{n} x_i^3 + a_2 \sum_{i=1}^{n} x_i^4 = \sum_{i=1}^{n} x_i^2 y_i \end{cases}$$

Введем обозначения:

$$SX = \sum_{i=1}^{n} x_i, \qquad SXX = \sum_{i=1}^{n} {x_i}^2, \qquad SXXX = \sum_{i=1}^{n} {x_i}^3, \qquad SXXXX = \sum_{i=1}^{n} {x_i}^4$$

$$SY = \sum_{i=1}^{n} y_i, \qquad SXY = \sum_{i=1}^{n} x_i y_i, \qquad SXXY = \sum_{i=1}^{n} {x_i}^2 y_i$$

Получим систему уравнений для нахождения параметров a_0, a_1 и a_2, решая которую получаем необходимые параметры.

Аппроксимация с помощью других функций:

Помимо линейных зависимостей для описания результатов эксперимента используют также показательные, степенные, логарифмические функции. Эти функции легко могут быть приведены к линейному виду, после чего для определения коэффициентов аппроксимирующей функции можно использовать описанный выше алгоритм.

Аппроксимирующая функция задана степенной функцией вида: $\varphi(x) = ax^b$

Для применения метода наименьших квадратов степенная функция линеаризуется:

$$\ln\big(\varphi(x)\big) = \ln(ax^b) = \ln(a) + bln(x)$$

Введем обозначения: $Y = \ln\big(\varphi(x)\big)$; $A = \ln(a)$; $B = b$; $X = \ln(x)$

Получаем линейную зависимость: $Y = A + B\text{X}$. После определения коэффициентов A и B вернемся к принятым ранее обозначениям: $a = e^A$ и $b = B$.

Аппроксимирующая функция задана экспоненциальной функцией вида:

$$\varphi(x) = ae^{bx}$$

Для применения метода наименьших квадратов экспоненциальная функция линеаризуется:

$$\ln\big(\varphi(x)\big) = \ln(ae^{bx}) = \ln(a) + bx$$

Введем обозначения: $Y = \ln\big(\varphi(x)\big)$; $A = \ln(a)$; $B = b$

Получаем линейную зависимость: $Y = A + B\text{X}$. После определения коэффициентов A и B вернемся к принятым ранее обозначениям: $a = e^A$ и $b = B$.

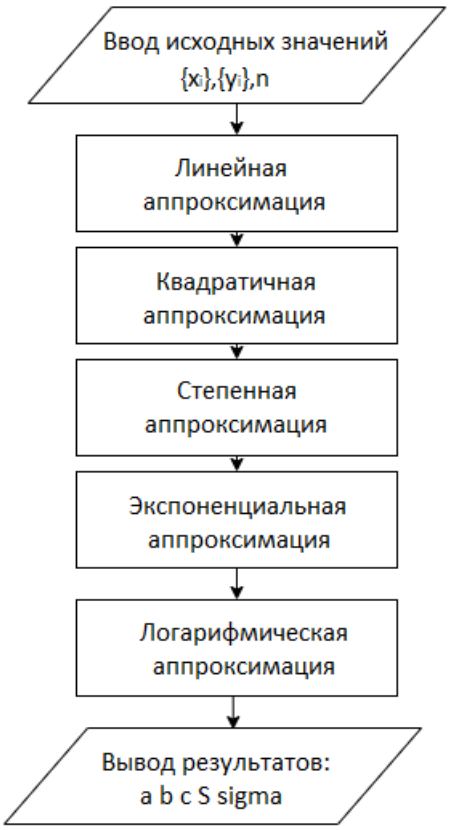
Аппроксимирующая функция задана логарифмической функцией вида:

$$\varphi(x) = aln(x) + b$$

Введем обозначения: $X = \ln(x)$

Получаем линейную зависимость: $\varphi(x) = aX + b$. Откуда находим коэффициенты *a, b*.

Блок-схема:



Листинг программы:

Calculation.java

```
public class Calculation {
    double a;
    double b;
    double c;
    double S;
    double sigma;
    double SX, SXX, SXXX, SXXXX, SXY, SXXY, SY;

    public void calculate(double[] x, double[] y, int n) {
        SX = 0;
        SXX = 0;
        SXXX = 0;
        SXXXX = 0;
        SXY = 0;
        SXXY = 0;
        SY = 0;
        for (int i = 0; i < n; i++) {
            SX += x[i];
            SXX += Math.pow(x[i], 2);
            SXXX += Math.pow(x[i], 3);
            SXXXX += Math.pow(x[i], 4);
            SXXY += Math.pow(x[i], 2) * y[i];
            SXY += x[i] * y[i];
            SY += y[i];
        }
    }

    public void Linear(double[] x, double[] y, int n) {
        calculate(x, y, n);
        a = (SXY * n - SX * SY) / (SXX * n - SX * SX);
        b = (SXX * SY - SX * SXY) / (SXX * n - SX * SX);
```

```
        S = 0;
        for (int i = 0; i < n; i++) {
            S += Math.pow((a * x[i] + b - y[i]), 2);
        }
        sigma = Math.sqrt(S / n);
    }

    public void pow(double[] x, double[] y, int n) {
        for (int i = 0; i < n; i++) {
            x[i] = Math.log(x[i]);
            y[i] = Math.log(y[i]);
        }
        calculate(x, y, n);
        double A = (SXY * n - SX * SY) / (SXX * n - SX * SX);
        double B = (SXX * SY - SX * SXY) / (SXX * n - SX * SX);
        a = Math.exp(B);
        b = A;
        S = 0;
        for (int i = 0; i < n; i++) {
            x[i] = Math.pow(Math.E, x[i]);
            y[i] = Math.pow(Math.E, y[i]);
            S += Math.pow((a * Math.pow(x[i], b) - y[i]), 2);
        }
        sigma = Math.sqrt(S / n);
    }

    public void exp(double[] x, double[] y, int n) {
        for (int i = 0; i < n; i++) {
            y[i] = Math.log(y[i]);
        }
        calculate(x, y, n);
        double A = (SXY * n - SX * SY) / (SXX * n - SX * SX);
        double B = (SXX * SY - SX * SXY) / (SXX * n - SX * SX);
        a = Math.exp(B);
        b = A;
        S = 0;
        for (int i = 0; i < n; i++) {
            y[i] = Math.exp(y[i]);
            S += Math.pow((a * Math.exp(b * x[i]) - y[i]), 2);
        }
        sigma = Math.sqrt(S / n);
    }

    public void log(double[] x, double[] y, int n) {
        for (int i = 0; i < n; i++) {
            x[i] = Math.log(x[i]);
        }
        calculate(x, y, n);
        a = (SXY * n - SX * SY) / (SXX * n - SX * SX);
        b = (SXX * SY - SX * SXY) / (SXX * n - SX * SX);
        S = 0;
        for (int i = 0; i < n; i++) {
            x[i] = Math.exp(x[i]);
            S += Math.pow((a * Math.log(x[i]) + b - y[i]), 2);
        }
        sigma = Math.sqrt(S / n);
    }

    public void quadratic(double[] x, double[] y, int n) {
        calculate(x, y, n);
        double[][] matrix = {
                {n, SX, SXX, SY},
                {SX, SXX, SXXX, SXY},
                {SXX, SXXX, SXXXX, SXXY},
        };
```

```
        double[][] m1 = {
                {matrix[0][0], matrix[0][1], matrix[0][2]},
                {matrix[1][0], matrix[1][1], matrix[1][2]},
                {matrix[2][0], matrix[2][1], matrix[2][2]},
        };
        double[][] m2 = {
                {matrix[0][3], matrix[0][1], matrix[0][2]},
                {matrix[1][3], matrix[1][1], matrix[1][2]},
                {matrix[2][3], matrix[2][1], matrix[2][2]},
        };
        double[][] m3 = {
                {matrix[0][0], matrix[0][3], matrix[0][2]},
                {matrix[1][0], matrix[1][3], matrix[1][2]},
                {matrix[2][0], matrix[2][3], matrix[2][2]},
        };
        double[][] m4 = {
                {matrix[0][0], matrix[0][1], matrix[0][3]},
                {matrix[1][0], matrix[1][1], matrix[1][3]},
                {matrix[2][0], matrix[2][1], matrix[2][3]},
        };

        double D = det(m1);
        double D1 = det(m2);
        double D2 = det(m3);
        double D3 = det(m4);

        if (D != 0) {
            a = D3 / D;
            b = D2 / D;
            c = D1 / D;
        } else {
            if (D1 == 0 && D2 == 0 && D3 == 0)
                System.out.println("Бесконечное кол-во решений!");
            else if (D1 != 0 || D2 != 0 || D3 != 0)
                System.out.println("Нет решений!");
        }
        S = 0;
        for (int i = 0; i < n; i++) {
            S += Math.pow((a * x[i] * x[i] + b * x[i] + c - y[i]), 2);
        }
        sigma = Math.sqrt(S / n);
    }

    public double det(double[][] matrix) {
        double ans = 0;
        ans = matrix[0][0] * (matrix[1][1] * matrix[2][2] - matrix[2][1] * matrix[1][2])
                - matrix[0][1] * (matrix[1][0] * matrix[2][2] - matrix[1][2] * matrix[2][0])
                + matrix[0][2] * (matrix[1][0] * matrix[2][1] - matrix[1][1] * matrix[2][0]);
        return ans;
    }

}
```

Main.java

```
public class Main {

    public static void main(String[] args) throws FileNotFoundException {
        System.out.println("Данные в файле должны быть разделены пробелом!");
        System.out.print("Введите имя файла для ввода данных:  ");
        Scanner in = new Scanner(System.in);
        String filename = in.nextLine();
        System.out.print("Введите количество пар x-y:  ");
        int n = in.nextInt();
        double[] x = new double[n];
        double[] y = new double[n];
```

```
        Scanner scanner = new Scanner(new File(filename));
        for (int i = 0; i < n; i++) {
            x[i] = scanner.nextDouble();
        }
        for (int i = 0; i < n; i++) {
            y[i] = scanner.nextDouble();
        }
        Calculation calc = new Calculation();
        System.out.print("Введите имя файла для вывода ответа:  ");
        String filename2 = in.next();
        FileOutputStream fos = new FileOutputStream(filename2);
        PrintStream fPrint = new PrintStream(fos);

        double [] AB = new double [11];

        String text = "Вид функции         |     a     |     b     |     c     |     S     |
Sigma   |";
        fPrint.println(text);

        calc.Linear(x, y, n);
        AB[0] = calc.a;
        AB[1] = calc.b;
        double min = calc.S;
        String minName = "Линейная";
        text = "f = a*x + b         |   " + String.format("%.3f", calc.a) + "      " +
String.format("%.3f", calc.b) + "         -         " + String.format("%.3f", calc.S) + "
" + String.format("%.3f", calc.sigma);
        fPrint.println(text);

        calc.pow(x, y, n);
        AB[2] = calc.a;
        AB[3] = calc.b;
        if (calc.S < min) {
            min = calc.S;
            minName = "Степенная";
        }
        text = "f = a*x^b           |   " + String.format("%.3f", calc.a) + "       " +
String.format("%.3f", calc.b) + "         -         " + String.format("%.3f", calc.S) + "
" + String.format("%.3f", calc.sigma);
        fPrint.println(text);

        calc.exp(x, y, n);
        AB[4] = calc.a;
        AB[5] = calc.b;
        if (calc.S < min) {
            min = calc.S;
            minName = "Экспоненциальная";
        }
        text = "f = a*e^(bx)        |   " + String.format("%.3f", calc.a) + "       " +
String.format("%.3f", calc.b) + "         -         " + String.format("%.3f", calc.S) + "
" + String.format("%.3f", calc.sigma);
        fPrint.println(text);

        calc.log(x, y, n);
        AB[6] = calc.a;
        AB[7] = calc.b;
        if (calc.S < min) {
            min = calc.S;
            minName = "Логарифмическая";
        }
        text = "f = a*lnx + b       |   " + String.format("%.3f", calc.a) + "     " +
String.format("%.3f", calc.b) + "         -         " + String.format("%.3f", calc.S) + "
" + String.format("%.3f", calc.sigma);
        fPrint.println(text);
```

```
        calc.quadratic(x, y, n);
        AB[8] = calc.a;
        AB[9] = calc.b;
        AB[10] = calc.c;
        if (calc.S < min) {
            min = calc.S;
            minName = "Квадратичная";
        }
        text = "f = a*x^2 + b*x + c |   " + String.format("%.3f", calc.a) + "       " +
String.format("%.3f", calc.b) + "       " + String.format("%.3f", calc.c) + "      " +
String.format("%.3f", calc.S) + "       " + String.format("%.3f", calc.sigma);
        fPrint.println(text);

        fPrint.println("Наиучшая аппроксимирующая функция - " + minName);
        Graph graph = new Graph();
        graph.build(AB);
    }
}
```

Graph.java

```
public class Graph {

    public double linear(double a, double b, double x) {
        return a * x + b;
    }

    public double pow(double a, double b, double x) {
        return a * Math.pow(x, b);
    }

    public double exp(double a, double b, double x) {
        return a * Math.exp(b * x);
    }

    public double log(double a, double b, double x) {
        return a * Math.log(x) + b;
    }

    public double quadratic(double a, double b, double c, double x) {
        return a * Math.pow(x, 2) + b * x + c;
    }

    public void build(double[] AB) {
        double a1 = AB[0];
        double b1 = AB[1];
        double a2 = AB[2];
        double b2 = AB[3];
        double a3 = AB[4];
        double b3 = AB[5];
        double a4 = AB[6];
        double b4 = AB[7];
        double a5 = AB[8];
        double b5 = AB[9];
        double c5 = AB[10];

        String linear = "f = a*x + b";
        String pow = "f = a*x^b";
        String exp = "f = a*e^(bx)";
        String log = "f = a*lnx + b";
        String quadratic = "f = a*x^2 + b*x + c";

        XYSeries LinSeries = new XYSeries(linear);
        XYSeries PowSeries = new XYSeries(pow);
        XYSeries ExpSeries = new XYSeries(exp);
        XYSeries LogSeries = new XYSeries(log);
```

```
        XYSeries QSeries = new XYSeries(quadratic);

        for (double i = 1; i < 4; i += 0.01) {
            LinSeries.add(i, linear(a1, b1, i));
            PowSeries.add(i, pow(a2, b2, i));
            ExpSeries.add(i, exp(a3, b3, i));
            LogSeries.add(i, log(a4, b4, i));
            QSeries.add(i, quadratic(a5, b5, c5, i));
        }

        XYSeriesCollection dataset = new XYSeriesCollection();
        dataset.addSeries(LinSeries);
        dataset.addSeries(PowSeries);
        dataset.addSeries(ExpSeries);
        dataset.addSeries(LogSeries);
        dataset.addSeries(QSeries);

        JFreeChart chart = ChartFactory.createXYLineChart("", "X", "Y", dataset,
PlotOrientation.VERTICAL, true, true, true);

        JFrame frame = new JFrame("График");
        frame.getContentPane().add(new ChartPanel(chart));
        frame.setSize(1300, 600);
        frame.show();
    }
}
```

Результат работы программы:

Исходные данные:

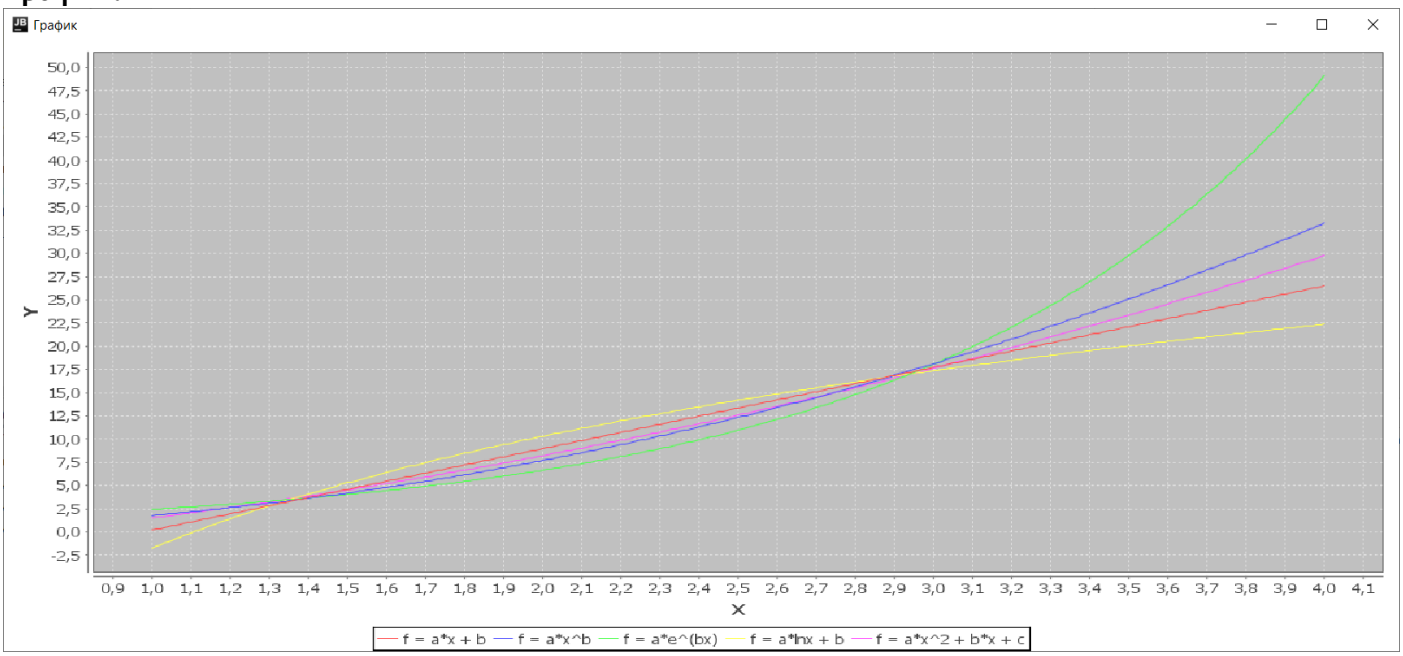
```
1 1,21 1,42 1,63 1,84 2,05 2,26 2,47 2,68 2,89 3,1 3,31 3,52
1,5 2,67 3,83 5,45 6,89 9,4 10,28 12,35 14,13 16,15 18,19 21,77 23,56
```

Ответ:

Вид функции	a	b	c	S	Sigma
f = a*x + b	8,761	-8,555	-	8,637	0,815
f = a*x^b	1,782	2,111	-	7,902	0,780
f = a*e^(bx)	0,900	1,000	-	74,075	2,387
f = a*lnx + b	17,389	-1,746	-	47,649	1,915
f = a*x^2 + b*x + c	1,352	2,651	-2,486	1,524	0,342

Наилучшая аппроксимирующая функция - Квадратичная

График:



Вывод: в ходе выполнения данной лабораторной работы была реализована программа на языке Java по поиску аппроксимирующей функции методом наименьших квадратов.