Университет ИТМО
Кафедра ВТ

Вычислительная математика

Лабораторная работа №4

Многошаговые методы Милна

Группа P3210
Нгу Фыонг Ань
ПРОВЕРИЛ:
Калёнова Ольга Вячеславовна

2018 год

Задание:

Задается ОбДифУр вида y’ + f(x,y) = 0 , пользователь задает начальные условия (x0, y0), конец отрезка и точность.

Программа сама вычисляет шаг в зависимости от точности для нахождения массива значений x и y.

Используя интерполирование 3-й работы строим график.

У кого 3-я работа была аппроксимация, строит график по полученным данным, задав очень маленькую точность.

//Не забудьте вставить примеры в отчет!
+ пару примеров ОДУ - с заранее известным решением вида y = f(x), где изначально y’ зависело не только от x.

Описание метода

9.2.5 Решение дифференциальных уравнений методом Милна

Отличие *метода Милна* от метода Адамса состоит в использовании в качестве интерполяционного полинома Ньютона.

Подставив в (9.23) вместо функции $f(t, x(t))$ интерполяционный полином Ньютона, построенный по точкам (t_{k-3}, x_{k-3}), (t_{k-2}, x_{k-2}), (t_{k-1}, x_{k-1}), (t_k, x_k) получаем первое приближение – прогноз Милна $\tilde{x}_{k+1}$ для значения функции в точке t_{k+1} [2]

$$\tilde{x}_{k+1} = x_{k-3} + \frac{4h}{3}(2f(t_{k-2}, x_{k-2}) - f(t_{k-1}, x_{k-1}) + 2f(t_k, x_k)) \quad (9.26)$$

Следующий полином Ньютона для функции $f(t, x(t))$ построим по точкам (t_{k-2}, x_{k-2}), (t_{k-1}, x_{k-1}), (t_k, x_k) и новой точке $(t_{k+1}, \tilde{x}_{k+1})$, после чего подставляем его в (9.23) и получаем второе приближение – корректор Милна [2]

$$x_{k+1} = x_{k-1} + \frac{h}{3}(f(t_{k-1}, x_{k-1}) + 4f(t_k, x_k) + f(t_{k+1}, \tilde{x}_{k+1})) \quad (9.27)$$

В методе Милна для вычисления значения $x(t_{k+1})$ необходимо последовательно применять формулы (9.26), (9.27), а первые четыре точки можно получить методом Рунге-Кутта.

Существует *модифицированный метод Милна*. В нем сначала вычисляется первое приближение по формуле (9.26), затем вычисляется управляющий параметр [2]

$$m_{k+1} = \tilde{x}_{k+1} + \frac{28}{29}(x_k - \tilde{x}_k) \quad (9.28)$$

после чего вычисляется значение второго приближения – корректор Милна по формуле

$$x_{k+1} = x_{k-1} + \frac{h}{3}(f(t_{k-1}, x_{k-1}) + 4f(t_k, x_k) + f(t_{k+1}, m_{k+1})) \quad (9.29)$$

В модифицированном методе Милна первые четыре точки можно получить методом Рунге-Кутта, а для вычисления значения $x(t_{k+1})$ необходимо последовательно применять формулы (9.26), (9.28), (9.29).

Код программы:

#Lab4_Milne

```
public class Lab4_Milne extends Application {
        double[] x = new double[1200];
        double[] t = new double[1200];
        double[] xp = new double[1200];
        double[] y = new double[1200];
        double[] f = new double[1200];
        double[] result = new double[1200];
        double a,b,h,x0;
        int n;
        int code;

    public static void main(String[] args) {
        launch(args);
    }

    public double cacul(double z){
        double sum = 0;
        for (int i = 0; i < n; i++) {
            sum = sum + result[i] * Math.pow(z, i);
        }
        return sum;
    }

    public void Milne(double a, double b, double h, double x0){

        n = (int) Math.round((b-a)/h);
        if ((b-a)/h>n) n = n+1;

        x[0] = x0;
        xp[0] = x[0];
        for (int i = 1; i<=n+1; i++) t[i-1] = a + (i - 1) * h;

        double K1,K2,K3,K4,delt;

        for (int i=1; i<4; i++){
            K1 = g(t[i - 1], x[i - 1]);
            K2 = g(t[i - 1] + h / 2, x[i - 1] + h / 2 * K1);
            K3 = g(t[i - 1] + h / 2, x[i - 1] + h / 2 * K2);
            K4 = g(t[i - 1] + h, x[i - 1] + h * K3);
            delt = h / 6 * (K1 + 2 * K2 + 2 * K3 + K4);
            x[i] = x[i - 1] + delt;
            xp[i] = x[i];
            //System.out.println(i + " " + K1 + " " + K2+ " " + K3 + " " + K4 + t[i] + " " + x[i]);
        }
        double m;
        for (int i = 3; i <n; i++){
            xp[i + 1] = x[i - 3]
```

```
            + 4 * h / 3 * (2 * g(t[i - 2], x[i - 2]) - g(t[i - 1],x[i - 1]))
            + 2*g(t[i], x[i]);
        m = xp[i + 1] + 28 / 29 * (x[i] - xp[i]);
        x[i + 1] = x[i - 1] + h / 3 * (g(t[i - 1], x[i - 1]) + 4 * g(t[i], x[i]) +g(t[i + 1], m));
    }
}

public double g(double x, double y){
    if (code ==1) return 6*x+2-3*x*x;
    if (code ==2) return y*(1/(2*x) -1);
    if (code ==3) return y + 10*Math.sin(x);
    return 0;
}

public double value(double x){
    if (code ==1)  return 3*x*x + 2*x - x*x*x;
    if (code ==2) return 10*Math.pow(Math.E, -x)*Math.sqrt(x);
    if (code ==3) return Math.pow(Math.E, x) - 5*Math.sin(x) - 5*Math.cos(x);
    return 0;
}

@Override
public void start(final Stage stage) {
    System.out.println("(1) y' = 6x + 2 - 3x^2  (y = 3x^2 + 2x - x^3)");
    System.out.println("(2) y' = y( 1/2x -1 )   (y = 10*e^(-x)*x^0.5)");
    System.out.println("(3) y' = y + 10sin(x)   (y = e^x - 5*sinx(x) - 5*cos(x))");
    System.out.println("Please choose your equation (1/2/3)");
    Scanner sc = new Scanner(System.in);
    String input;

    while (true){
        input = sc.nextLine();
        if (input.equals("1")) { code = 1;  break;
       } else if (input.equals("2")) { code = 2;  break;
        } else if (input.equals("3")) { code = 3; break;
        } else System.out.println("Invalid code. Please choose your equation (1/2)");
    }

    System.out.println("Please enter a:");
    while (true){
        try{
            input = sc.nextLine();
            a = Double.parseDouble(input);
            break;
        } catch (NumberFormatException e) {
            System.out.println("Invalid code. Please enter a:");
        }
    }

    System.out.println("Please enter b:");
    while (true){
        try{
```

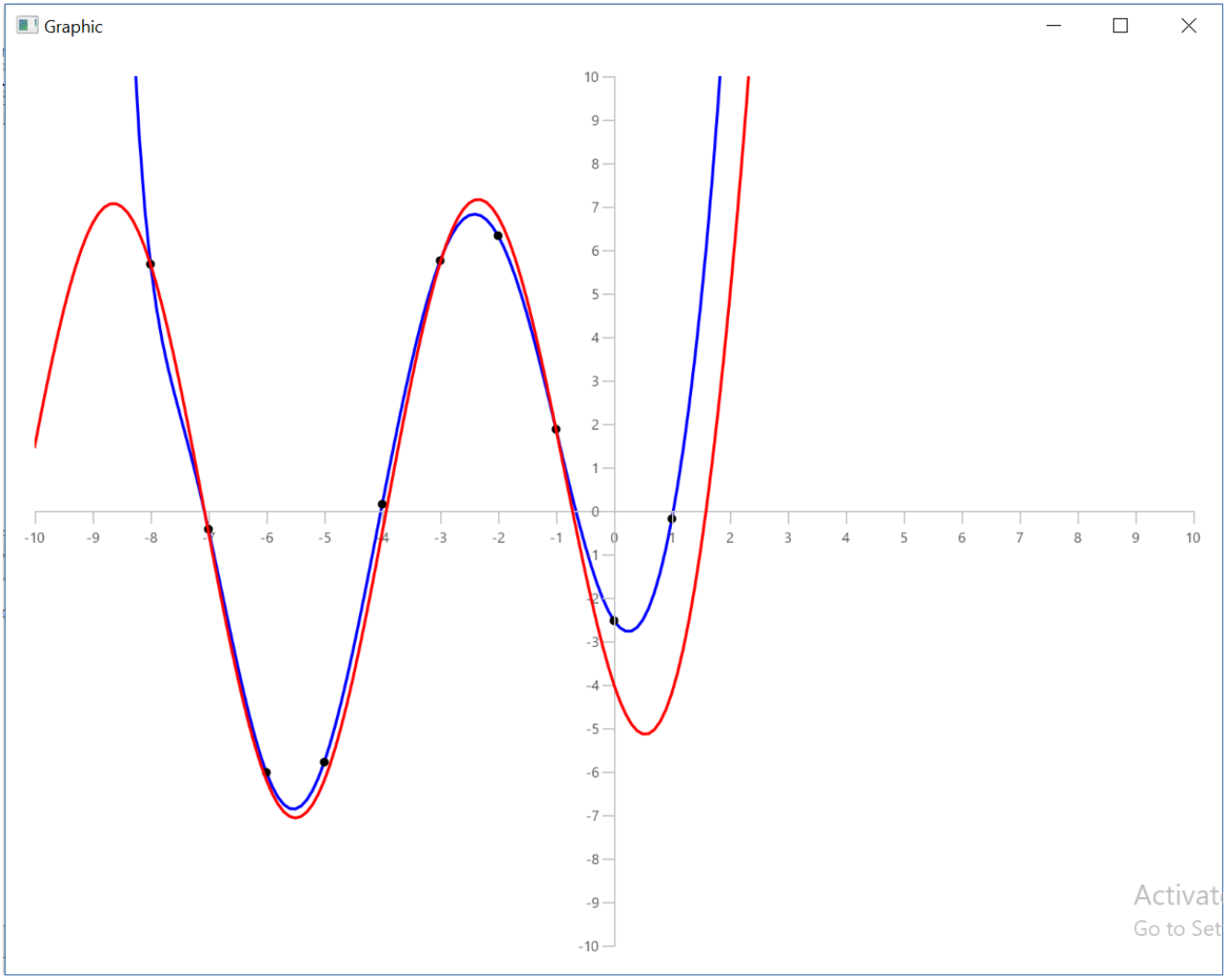
```
        input = sc.nextLine();
        b = Double.parseDouble(input);
        break;
    } catch (NumberFormatException e) {
        System.out.println("Invalid code. Please enter b:");
    }
}

System.out.println("Please enter accuracy:");
while (true){
    try{
        input = sc.nextLine();
        h = Double.parseDouble(input);
        break;
    } catch (NumberFormatException e) {
        System.out.println("Invalid code. Please enter accuracy:");
    }
}

System.out.println("Please enter f[a]:");
while (true){
    try{
        input = sc.nextLine();
        x0 = Double.parseDouble(input);
        break;
    } catch (NumberFormatException e) {
        System.out.println("Invalid code. Please enter f[a]:");
    }
}

Milne(a,b,h,x0);
n = n+1;
y = x;
x = t;

//for (int i = 0; i<n; i++) System.out.println(x[i] + " " + y[i] + " " + value(x[i]));

double [][] s = new double [1200][1200];
for (int i = 0; i < n; i++) {
    s[i][0] = x[i];
    s[i][1] = y[i];
}
int dis = 0;
for (int j = 2; j<= n+1; j++){
    dis++;
    int start = 0;
        for (int i = 0; i<n-dis; i++){
      s[i][j] = (s[i+1][j-1]-s[i][j-1])/(s[start+dis][0]-s[start][0]);
      start++;
  }
}
for (int i=0; i<n; i++) f[i] = s[0][i+1];
```

```
        result = new PolynomProduct(x,f).getresult();

        Axes axes = new Axes(
                800, 600,
                -10, 10, 1,
                -10, 10, 1
        );

        Plot plot1 = null;
        Plot plot = new Plot(
                z -> cacul(z),
                -10, 10, 0.1,
                axes, Color.BLUE);


        if (code == 1) {
           plot1 = new Plot(
                   z -> 3*z*z +2*z - z*z*z,
                   -10, 10, 0.1,
                   axes, Color.RED
           );
        }

        if (code == 2) {
           plot1 = new Plot(
                   z -> 10*Math.pow(Math.E, -z)*Math.sqrt(z),
                   0.0001, 10, 0.1,
                   axes, Color.RED
           );
        }

        if (code == 3) {
                   plot1 = new Plot(
                   z -> Math.pow(Math.E, z) - 5*Math.sin(z) - 5*Math.cos(z),
                   -10, 10, 0.1,
                   axes, Color.RED
           );
        }
        for (int i = 0; i < n; i++) {
              plot.getChildren().addAll(new Circle(400 + 40*x[i], 300 - 30 * y[i], 3, Color.BLACK));
           }
           StackPane layout = new StackPane(plot, plot1);
           layout.setPadding(new Insets(20));
           layout.setStyle("-fx-background-color: rgb(255,255,255);");

           stage.setTitle("Graphic");
           stage.setScene(new Scene(layout, Color.WHITE));
           stage.show();
    }

}
```

Тест:

(1) y' = 6x + 2 - 3x^2 (y = 3x^2 + 2x - x^3)
(2) y' = y(1/2x -1) (y = 10*e^(-x)*x^0.5)
(3) y' = y + 10sin(x) (y = e^x - 5*sinx(x) - 5*cos(x))
Please choose your equation (1/2/3)
3
Please enter a:
-8
Please enter b:
2
Please enter accuracy:
1
Please enter f[a]:
5.6746268647878795

Вывод программы:



Вывод: With the input, we should choose the right number of segments, because too little will lead to insufficient number of points needed to infer the original function, too much will result in incorrect calculation of points (x, y). (great deviation).