Автор: Смирнов Алексей Владимирович ИСУ: 409578 Группа: R3242

Отчет по расчетно-графической работе №1 «Численное интегрирование дифференциальных уравнений первого порядка»

Санкт-Петербург, 2024 г.

Вариант 16

Численно решить дифференциальное уравнение

$$y' = y - x^2,\ y(1) = 2$$

на отрезке $[0; 2]$ с шагом $h = 0.2$ методом Эйлера, модифицированным методом Эйлера и методом Рунге-Кутта. Найти точное решение $y = y(x)$ и сравнить значения точного и приближенных решений в точке $x = 2$. Найти абсолютную и относительную погрешности в этой точке для каждого метода. Вычисления вести с четырьмя десятичными знаками.

Решение

Для численного решения уравнения удобно использовать средства языка программирования Python в совокупности с библиотекой NumPy. Для каждого из методов была написана программа, исходный код которых можно увидеть в Листингах 1–3.

Метод Эйлера

i	x_i	y_i	$f(x_i, y_i)$	Δy_i
1	0.0000	0.8825	0.0000	0.0000
2	0.2000	1.0931	1.0531	−0.2106
3	0.4000	1.3264	1.1664	−0.2333
4	0.6000	1.5680	1.2080	−0.2416
5	0.8000	1.8000	1.1600	−0.2320
6	1.0000	2.0000	1.0000	0.2000
7	1.2000	2.2000	0.7600	0.1520
8	1.4000	2.3520	0.3920	0.0784
9	1.6000	2.4304	−0.1296	−0.0259
10	1.8000	2.4045	−0.8355	−0.1671
11	2.0000	2.2374	0.0000	0.0000

Таблица 1.

модифицированный метод Эйлера

i	x_i	$x_{i+\frac{1}{2}}$	y	$y_{i+\frac{1}{2}}$	$f(x_i, y_i)$	$f\left(x_{i+\frac{1}{2}}, y_{i+\frac{1}{2}}\right)$	Δy_i
1	0.0000	0.1000	1.0407	0.0000	0.0000	0.0000	0.0000

2	0.2000	0.3000	1.2482	1.1274	1.2082	1.0374	−0.2075
3	0.4000	0.5000	1.4652	1.3346	1.3052	1.0846	−0.2169
4	0.6000	0.7000	1.6760	1.5444	1.3160	1.0544	−0.2109
5	0.8000	0.9000	1.8620	1.7398	1.2220	0.9298	−0.1860
6	1.0000	1.1000	2.0000	2.1000	1.0000	0.8900	0.1780
7	1.2000	1.3000	2.1780	2.2518	0.7380	0.5618	0.1124
8	1.4000	1.5000	2.2904	2.3234	0.3304	0.0734	0.0147
9	1.6000	1.7000	2.3050	2.2795	−0.2550	−0.6105	−0.1221
10	1.8000	1.9000	2.1829	2.0772	−1.0571	−1.5328	−0.3066
11	2.0000	2.1000	1.8764	0.0000	0.0000	0.0000	0.0000

Таблица 2.

Метод Рунге-Кутта

i	x	y	K	Δy_i
0	0.0000	1.0326		
1	0.2000	1.2401	−0.2400	−0.2400
	0.3000	1.1201	−0.2060	−0.4120
	0.3000	1.1371	−0.2094	−0.4188
	0.4000	1.0307	−0.1741	−0.1741
				−0.2075
2	0.4000	1.4578	−0.2596	−0.2596
	0.5000	1.3281	−0.2156	−0.4312
	0.5000	1.3500	−0.2200	−0.4400
	0.6000	1.2378	−0.1756	−0.1756
				−0.2177
3	0.6000	1.6703	−0.2621	−0.2621
	0.7000	1.5393	−0.2099	−0.4197
	0.7000	1.5654	−0.2151	−0.4302
	0.8000	1.4553	−0.1631	−0.1631
				−0.2125
4	0.8000	1.8587	−0.2437	−0.2437
	0.9000	1.7369	−0.1854	−0.3707

	0.9000	1.7661	−0.1912	−0.3824
	1.0000	1.6675	−0.1335	−0.1335
				−0.1884
5	1.0000	2.0000	0.2000	0.2000
	1.1000	2.1000	0.1780	0.3560
	1.1000	2.0890	0.1758	0.3516
	1.2000	2.1758	0.1472	0.1472
				0.1758
6	1.2000	2.1758	0.1472	0.1472
	1.3000	2.2494	0.1119	0.2237
	1.3000	2.2317	0.1083	0.2167
	1.4000	2.2841	0.0648	0.0648
				0.1087
7	1.4000	2.2845	0.0649	0.0649
	1.5000	2.3170	0.0134	0.0268
	1.5000	2.2912	0.0082	0.0165
	1.6000	2.2928	−0.0534	−0.0534
				0.0091
8	1.6000	2.2937	−0.0533	−0.0533
	1.7000	2.2670	−0.1246	−0.2492
	1.7000	2.2314	−0.1317	−0.2635
	1.8000	2.1619	−0.2156	−0.2156
				−0.1303
9	1.8000	2.1634	−0.2153	−0.2153
	1.9000	2.0557	−0.3109	−0.6217
	1.9000	2.0080	−0.3204	−0.6408
	2.0000	1.8430	−0.4314	−0.4314
				−0.3182
10	2.0000	1.8452		

Таблица 3.

Точное решение

Исходное уравнение

$$y' = y - x^2$$

переносом y в левую часть преобразуется в линенейное неоднородное уравнение первого порядка.

$$y' - y = -x^2$$

Характеристический многочлен

$$\lambda - 1$$

Решение соответствующего однородного уравнения

$$\lambda - 1 = 0 \Longrightarrow \lambda = 1$$
$$y_0 = C_1 e^x$$

Подберем частное решение методом неопределенных коэффициентов. Правая часть

$$-x^2$$

тогда частное решение имеет вид

$$\tilde{y} = Ax^2 + Bx + C$$
$$\tilde{y}' = 2Ax + B$$
$$\tilde{y}' - \tilde{y} = (-A)x^2 + (2A - B)x + (B - C) = -x^2$$

$$\begin{cases} A = 1 \\ 2A - B = 0 \Longrightarrow B = 2 \\ B - C = 0 \Longrightarrow C = 2 \end{cases}$$

$$\tilde{y} = x^2 + 2x + 2$$

Общее решение уравнения

$$y = C_1 e^x + x^2 + 2x + 2$$

Начальные условия: $y(1) = 2$:

$$2 = C_1 \cdot e + 1 + 2 + 2$$
$$C_1 = -\frac{3}{e}$$

Финальное решение

$$y(x) = -\frac{3}{e}e^x + x^2 + 2x + 2$$

Сравнение результатов

Решение	$x = 1.0$	$x = 1.2$	$x = 1.4$	$x = 1.6$	$x = 1.8$	$x = 2.0$	В точке $x = 1.0$	
							$\delta_{\text{Абс}}$	$\delta_{\text{Отн}}$
Точное	2.0000	2.1758	2.2845	2.2936	2.1634	1.8452		

Метод Эйлера	2.0000	2.2000	2.3520	2.4304	2.4045	2.2374	0.3922	21.257%
Модиф. метод Эйлера	2.0000	2.1780	2.2904	2.3050	2.1829	1.8764	0.0312	1.693%
Метод Рунге-Кутта	2.0000	2.1758	2.2845	2.2937	2.1634	1.8452	0.0000	0.002%

Таблица 4.

Python

```python
import numpy as np
np.set_printoptions(4)
XS = np.linspace(0,2,11)
I = np.where(XS == 1)[0][0]
n = N = XS.shape[0]
h = np.float64('.2')
f = lambda x,y: y - x*x
cellformat = lambda x: "[{:5.4f}]".format(x)

table1_i = np.array(range(1, n+1))
table1_x = XS
table1_y = np.zeros((n,), dtype=np.float64)
table1_y[I] = 2
table1_f = np.zeros((n,), dtype=np.float64)
table1_deltay = np.zeros((n,), dtype=np.float64)

for i in range(I, 0, -1):
    table1_f[i] = f(table1_x[i], table1_y[i])
    table1_deltay[i] = -h*table1_f[i]
    table1_y[i-1] = table1_y[i] + table1_deltay[i]
for i in range(I, N-1):
    table1_f[i] = f(table1_x[i], table1_y[i])
    table1_deltay[i] = h*table1_f[i]
    table1_y[i+1] = table1_y[i] + table1_deltay[i]

table1 = np.vstack((table1_i, table1_x, table1_y, table1_f,
table1_deltay)).T
```

Листинг 1. Решение уравния методом Эйлера

Python

```python
# ...

table2_i = np.arange(1, n+1)
table2_x = XS
table2_y = np.zeros((n,), dtype=np.float64)
table2_f = np.zeros((n,), dtype=np.float64)

table2_x2 = table2_x + h / 2
table2_y2 = np.zeros((n,), dtype=np.float64)
table2_f2 = np.zeros((n,), dtype=np.float64)

table2_deltay = np.zeros((n,), dtype=np.float64)

table2_y[I] = 2


for i in range(I, 0, -1):
    table2_f[i] = f(table2_x[i], table2_y[i])

    # table2_x2
    table2_y2[i] = table2_y[i] -h/2 * table2_f[i]
    table2_f2[i] = f(table2_x2[i], table2_y2[i])

    table2_deltay[i] = -h*table2_f2[i]
    table2_y[i-1] = table2_y[i] + table2_deltay[i]

for i in range(I, n-1):
    table2_f[i] = f(table2_x[i], table2_y[i])

    # table2_x2
    table2_y2[i] = table2_y[i] + h/2 * table2_f[i]
    table2_f2[i] = f(table2_x2[i], table2_y2[i])

    table2_deltay[i] = h*table2_f2[i]
    table2_y[i+1] = table2_y[i] + table2_deltay[i]

table2 = np.vstack((table2_i,
```

```
38                          table2_x, table2_x2,
39                          table2_y, table2_y2,
40                          table2_f, table2_f2,
41                          table2_deltay)).T
```

Листинг 2. Решение уравнения модифицированным методом Эйлера

```python
# ...

table3_i  = np.arange(0, N)
table3_x1 = XS
table3_x2 = table3_x1 + h / 2
table3_x3 = table3_x1 + h / 2
table3_x4 = table3_x1 + h

table3_y1 = np.zeros((n,),dtype=np.float64)
table3_y1[I] = 2

table3_y2 = np.zeros((n,),dtype=np.float64)
table3_y3 = np.zeros((n,),dtype=np.float64)
table3_y4 = np.zeros((n,),dtype=np.float64)

table3_K1 = np.zeros((n,),dtype=np.float64)
table3_K2 = np.zeros((n,),dtype=np.float64)
table3_K3 = np.zeros((n,),dtype=np.float64)
table3_K4 = np.zeros((n,),dtype=np.float64)

table3_Dy = np.zeros((n,),dtype=np.float64)

for i in range(I, 0, -1):
    table3_K1[i] = f(table3_x1[i], table3_y1[i]) * -h

    table3_y2[i] = table3_y1[i] + table3_K1[i] / 2
    table3_K2[i] = f(table3_x2[i], table3_y2[i]) * -h

    table3_y3[i] = table3_y1[i] + table3_K2[i] / 2
    table3_K3[i] = f(table3_x3[i], table3_y3[i]) * -h

```

```
32      table3_y4[i] = table3_y1[i] + table3_K3[i]
33      table3_K4[i] = f(table3_x4[i], table3_y4[i]) * -h
34
35      table3_Dy[i] = 1/6 * (table3_K1[i] + 2*table3_K2[i] +
36                            2*table3_K3[i] + table3_K4[i])
37
38      table3_y1[i-1] = table3_y1[i] + table3_Dy[i]
39
40  for i in range(I, n-1):
41      table3_K1[i] = f(table3_x1[i], table3_y1[i]) * h
42
43      table3_y2[i] = table3_y1[i] + table3_K1[i] / 2
44      table3_K2[i] = f(table3_x2[i], table3_y2[i]) * h
45
46      table3_y3[i] = table3_y1[i] + table3_K2[i] / 2
47      table3_K3[i] = f(table3_x3[i], table3_y3[i]) * h
48
49      table3_y4[i] = table3_y1[i] + table3_K3[i]
50      table3_K4[i] = f(table3_x4[i], table3_y4[i]) * h
51
52      table3_Dy[i] = 1/6 * (table3_K1[i] + 2*table3_K2[i] +
53                            2*table3_K3[i] + table3_K4[i])
54
55      table3_y1[i+1] = table3_y1[i] + table3_Dy[i]
56
57  table3 = np.vstack((table3_i,
58                      table3_x1, table3_x2, table3_x3, table3_x4,
59                      table3_y1, table3_y2, table3_y3, table3_y4,
60                      table3_K1, table3_K2, table3_K3, table3_K4,
61                      table3_Dy)).T
```

Листинг 3. Решение уравнения методом Рунге-Кутта