MOwNiT - Laboratorium 10: Rozwiązywanie równań różniczkowych zwyczajnych

Wojciech Dąbek

21 maja 2024

1 Treści zadań

1. Dane jest równanie różniczkowe (zagadnienie początkowe):

$$y' + y\cos x = \sin x \cos x \qquad y(0) = 0$$

   Znaleźć rozwiązanie metodą Rungego-Kutty i metodą Eulera.
   Porównać otrzymane rozwiązanie z rozwiązaniem dokładnym:

$$y(x) = e^{-\sin x} + \sin x - 1$$

2. Dane jest zagadnienie brzegowe:

$$y'' + y = x \qquad y(0) = 1 \qquad y\left(\frac{\pi}{2}\right) = \frac{\pi}{2} - 1$$

   Znaleźć rozwiązanie metodą strzałów.
   Porównać otrzymane rozwiązanie z rozwiązaniem dokładnym:

$$y(x) = \cos x - \sin x + x$$

2 Rozwiązania

2.1 Zadanie 1 - zagadnienie początkowe

Metoda Rungego-Kutty opiera się na wzorze rekurencyjnym:

$$y_{k+1} = y_k + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

gdzie $h = y_{k+1} - y_k$ oraz

$$k_1 = f(x_k, y_k)$$
$$k_2 = f(x_k + \frac{h}{2}, y_k + \frac{h}{2}k_1)$$
$$k_3 = f(x_k + \frac{h}{2}, y_k + \frac{h}{2}k_2)$$
$$k_3 = f(x_k + h, y_k + hk_3)$$
$$f(x, y) = y'$$

Metoda Eulera opiera się na prostszym wzorze:

$$y_{k+1} = y_k + hf(x_k, y_k)$$

Obliczenia zrealizowałem następującym kodem w języku Python:

```python
from math import exp, sin, cos, fabs


def exact(x: float) -> float:
    return exp(-sin(x)) + sin(x) - 1.

def fun(x: float, y: float) -> float:
    return sin(x) * cos(x) - y * cos(x)

def runge_kutta(x_0, y_0, n, h, f):
    x = x_0
    y = y_0
    for _ in range(n):
        k1 = f(x, y)
        k2 = f(x + h/2, y + k1 * h/2)
        k3 = f(x + h/2, y + k2 * h/2)
        k4 = f(x + h,   y + k3 * h)
        y += (k1 + 2*k2 + 2*k3 + k4) * h/6
        x += h
    return y

def euler(x_0, y_0, n, h, f):
    x = x_0
    y = y_0
    for _ in range(n):
        y += h * f(x, y)
        x += h
    return y

```

```python
if __name__ == '__main__':
    for k in range(1, 6):
        n = 10**k
        print(f'For {n = }:')
        rk_method = runge_kutta(0., 0., n, 1./ n, fun)
        e_method = euler(0., 0., n, 1. / n, fun)
        expect = exact(1.)
        print(f'    Expected value: {expect}')
        print(f'Runge-Kutta method: {rk_method}', end='\t\t')
        print(f'Difference: {fabs(rk_method - expect):.5e}')
        print(f"    Euler's method: {e_method}", end='\t\t')
        print(f'Difference: {fabs(e_method - expect):.5e}')
        print()
```

Jako wartość h przyjąłem $\frac{1}{n}$, aby $x_n = 1$. Rezultatem działania tego programu jest:

```
For n = 10:
    Expected value: 0.27254693545348885
Runge-Kutta method: 0.2725471473929363    Difference: 2.11939e-07
    Euler's method: 0.26442725830740726   Difference: 8.11968e-03

For n = 100:
    Expected value: 0.27254693545348885
Runge-Kutta method: 0.2725469354731914    Difference: 1.97026e-11
    Euler's method: 0.2718062028296542    Difference: 7.40733e-04

For n = 1000:
    Expected value: 0.27254693545348885
Runge-Kutta method: 0.27254693545349107   Difference: 2.22045e-15
    Euler's method: 0.2724735390106294    Difference: 7.33964e-05

For n = 10000:
    Expected value: 0.27254693545348885
Runge-Kutta method: 0.27254693545348246   Difference: 6.38378e-15
    Euler's method: 0.27253960254408427   Difference: 7.33291e-06

For n = 100000:
    Expected value: 0.27254693545348885
Runge-Kutta method: 0.27254693545351943   Difference: 3.05866e-14
    Euler's method: 0.2725462022298938    Difference: 7.33224e-07
```

Wnioski: Jak widać, metoda Rungego-Kutty daje wyraźnie większą dokładność od prostszej metody Eulera. Rośnie ona wraz ze wzrostem n aż do pewnej dużej dokładności (co może wynikać z dokładności reprezentacji zmiennoprzecinkowej liczb). W przypadku metody Eulera przy eksponencjolnym wzroście n dokładność wyników rośnie liniowo.

2.2 Zadanie 2 - zagadnienie brzegowe

Metoda strzałów polega na zastąpieniu zagadnienia brzegowego postaci

$$\begin{aligned} y'' &= f(x,\ y,\ y') \\ y(x_0) &= y_0 \\ y(x_k) &= y_k \end{aligned}$$

zagadnieniem początkowym postaci

$$\begin{aligned} y_a'' &= f(x,\ y_a,\ y_a') \\ y_a(x_0) &= y_0 \\ y_a'(x_0) &= a \end{aligned}$$

z odpowiedno dobranym parametrem a. W naszym przypadku:

$$y'' = f(x,\ y,\ y') = x - y$$

Zatem $y' = a = 0$. Zdefiniujmy:

$$u_1(x) = y(x), \qquad u_2(x) = y'(x) = 0$$

Problem sprowadza się więc do układu:

$$\begin{bmatrix} u_1' \\ u_2' \end{bmatrix} = \begin{bmatrix} u_2 \\ f(x,y) \end{bmatrix} = \begin{bmatrix} 0 \\ f(x,y) \end{bmatrix}$$

Który można rozwiązać jak metodami dla równań pierwszego rzędu, jak w pierwszym zadaniu. Niestety nie starczyło mi na to czasu.

3 Bibliografia

Prof. M. T. Heath - *Scientific Computing: An Introductory Survey*, ch. 9, 10