Tudor Vladut-Alexandru

An 2, Grupa 3, Informatică

Referat 2 – Ecuații diferențiale și cu derivate parțiale

Referat 2:

Considerați o problemă Cauchy pentru o ecuație diferențială a cărei soluție se poate obține explicit. Aproximați soluția folosind metoda lui Euler și metoda Rune-Kutta de ordin patru. Reprezentați grafic soluția exactă și cele două soluții obținute cu metodele precizate. Calculați erorile.

Cerință:

Considerăm problema Cauchy: y' = 3*(x**2)*y, y(1) = 1.

Ecuatia este omogena de ordinul intai (p(x) = 3x**2, q(x) = 0). Solutia generala este y(x) = Ke la integrala din p(x) dx = K*e la x**3.

Cum y(1) = 1, obtinem K = e la −1/3, deci solutia exacta este y(x) = e la x**3-1/3, x ∈ R.

Rezolvare:

1) Metoda lui Euler: alegem xmax = 1.5, h = 0.1.

Tabelul:

x	y_exact	y_euler	error
1.0	1.0000	1.0000	0.0000
1.1	1.1800	1.3000	0.1200
1.2	1.4391	1.7719	0.3328
1.3	1.8194	2.5374	0.7180
1.4	2.3917	3.8238	1.4321
1.5	3.2789	6.0722	2.7933
1.6	4.7021	10.1709	5.4689

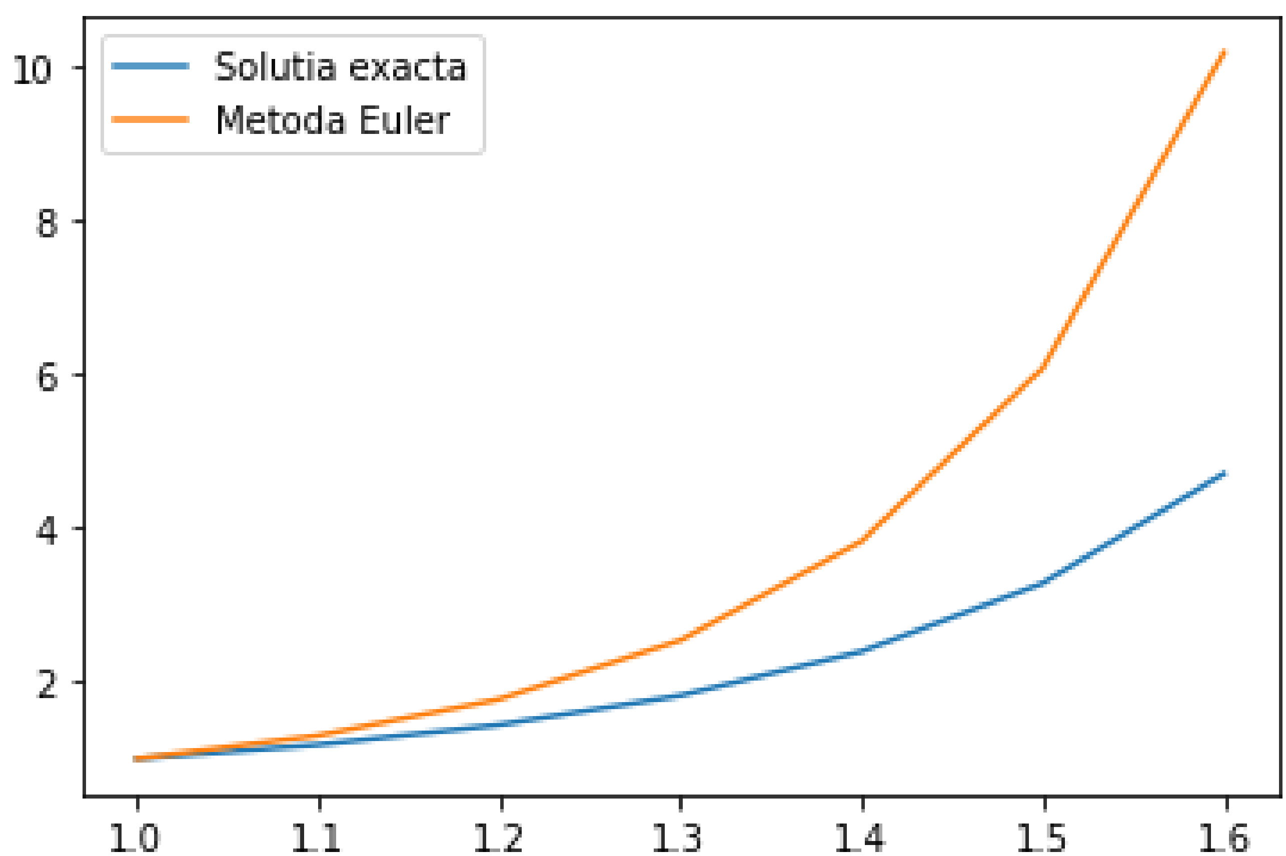
Codul in python:

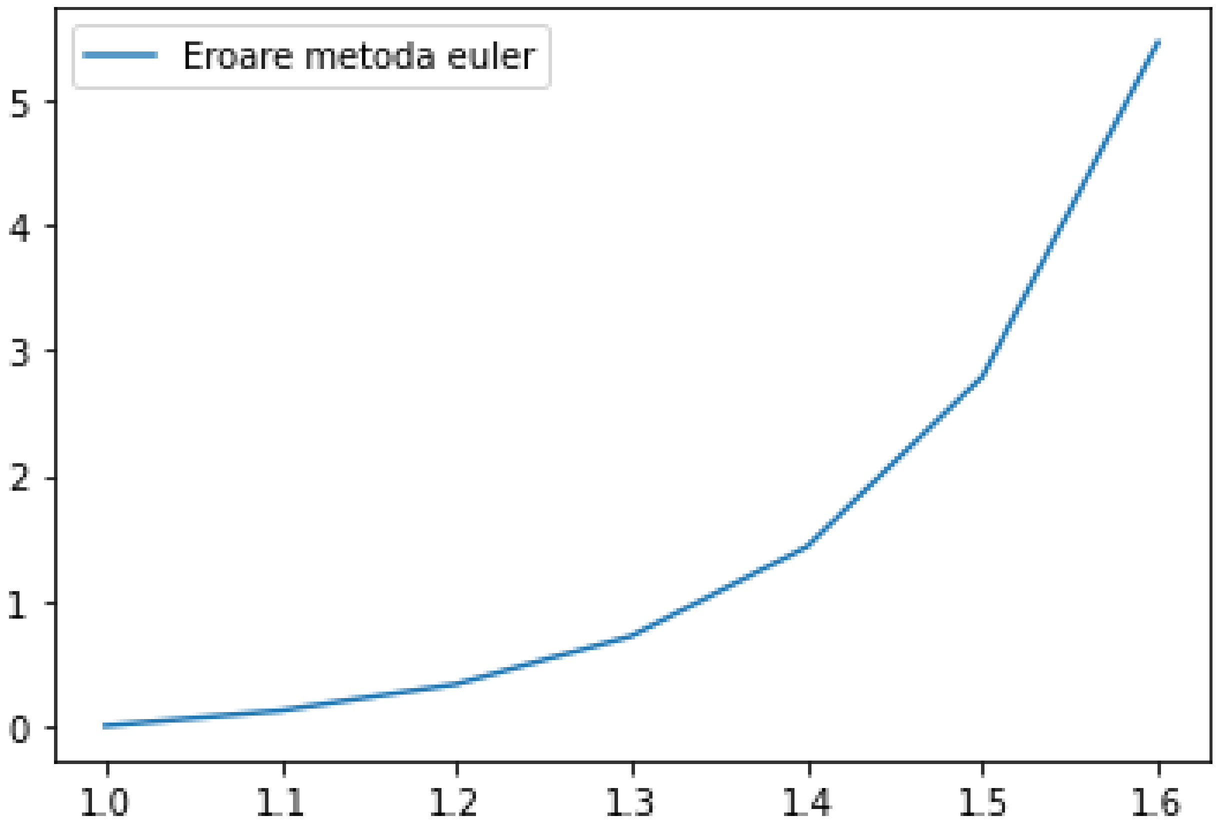
```
import numpy as np
import matplotlib.pyplot as plt


def f(x, y):
    return 3*x**2*y
```

```
x_min, x_max = 1, 1.5
h = 0.1

x = np.arange(x_min, x_max + h, h)
y_euler = np.zeros(len(x))
y_euler[0] = 1

for i in range(len(x) - 1):
    y_euler[i+1] = y_euler[i] + h*f(x[i], y_euler[i])

y_exact = np.exp(0.5*(x**3 - 1))

error_euler = np.abs(y_exact - y_euler)

print(" x    | y_exact | y_euler |  error  ")
print("------|---------|---------|--------")
for i in range(len(x)):
    print(" {:.1f}  | {:.4f}  | {:.4f}  | {:.4f}".format(x[i], y_exact[i], y_euler[i],
error_euler[i]))

# Plot the solutions
plt.plot(x, y_exact, label='Exact solution')
plt.plot(x, y_euler, label='Euler method')
```

```
plt.legend()
plt.show()

# Plot the error
plt.plot(x, error_euler, label='Euler method error')
plt.legend()
plt.show()
```

Reprezentare grafică:





Eroare metoda euler
5
4
3
2
1
0
1.0
1.1
1.2
1.3
1.4
1.5
1.6

2) Metoda Runge Kutta de ordin 4: alegem xmax = 1.5, h = 0.1

Tabelul:

x	y_exact	y_rk4	error
1.0	1.000000	1.000000	0.000000
1.1	1.179983	1.392321	0.212338
1.2	1.439074	2.070745	0.631671
1.3	1.819388	3.309422	1.490035
1.4	2.391689	5.717343	3.325654
1.5	3.278874	10.740218	7.461345
1.6	4.702057	22.066947	17.364890

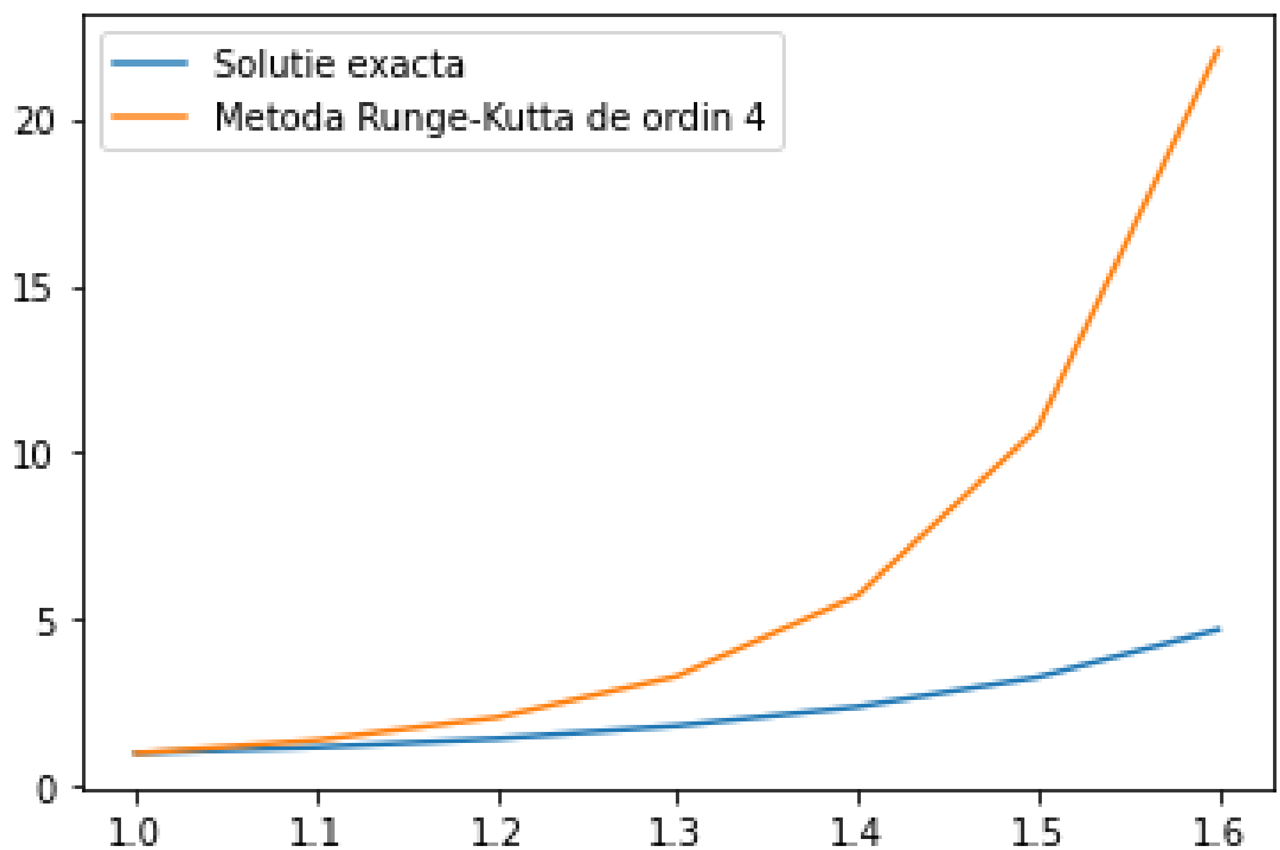
Codul in python:

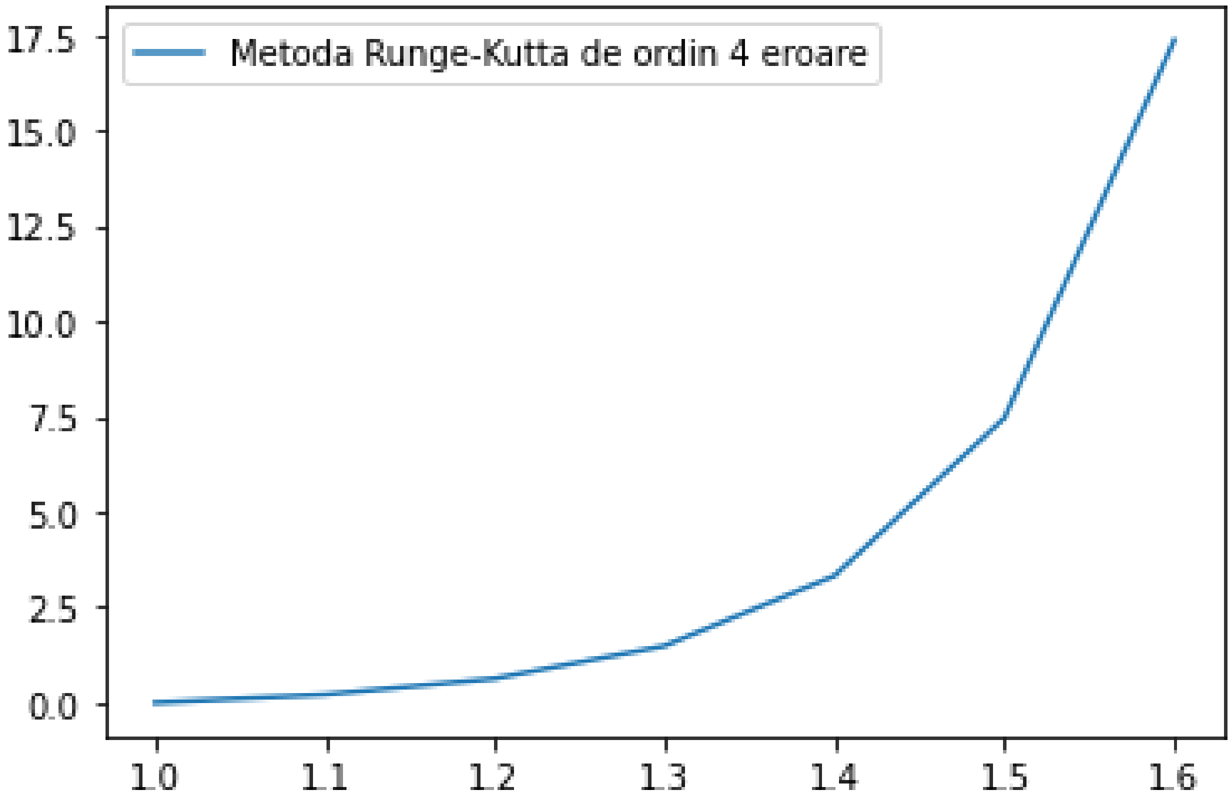
```
import numpy as np
import matplotlib.pyplot as plt


def f(x, y):
    return 3*x**2*y
```

```
x_min, x_max = 1, 1.5
h = 0.1

x = np.arange(x_min, x_max + h, h)
y_rk4 = np.zeros(len(x))
y_rk4[0] = 1

for i in range(len(x) - 1):
    k1 = f(x[i], y_rk4[i])
    k2 = f(x[i] + h/2, y_rk4[i] + (h/2)*k1)
    k3 = f(x[i] + h/2, y_rk4[i] + (h/2)*k2)
    k4 = f(x[i] + h, y_rk4[i] + h*k3)
    y_rk4[i+1] = y_rk4[i] + (h/6)*(k1 + 2*k2 + 2*k3 + k4)

y_exact = np.exp(0.5*(x**3 - 1))
error_rk4 = np.abs(y_exact - y_rk4)

print(" x    | y_exact | y_rk4   |  error  ")
print("------|---------|---------|--------")
for i in range(len(x)):
    print("{:.1f}  | {:.6f}  | {:.6f}  | {:.6f}".format(x[i], y_exact[i], y_rk4[i],
error_rk4[i]))
```

```
plt.plot(x, y_exact, label='Solutia exacta')
plt.plot(x, y_rk4, label='Metoda Runge-Kutta de ordin 4')
plt.legend()
plt.show()

plt.plot(x, error_rk4, label='Metoda Runge-Kutta de ordin 4 eroare')
plt.legend()
plt.show()
```

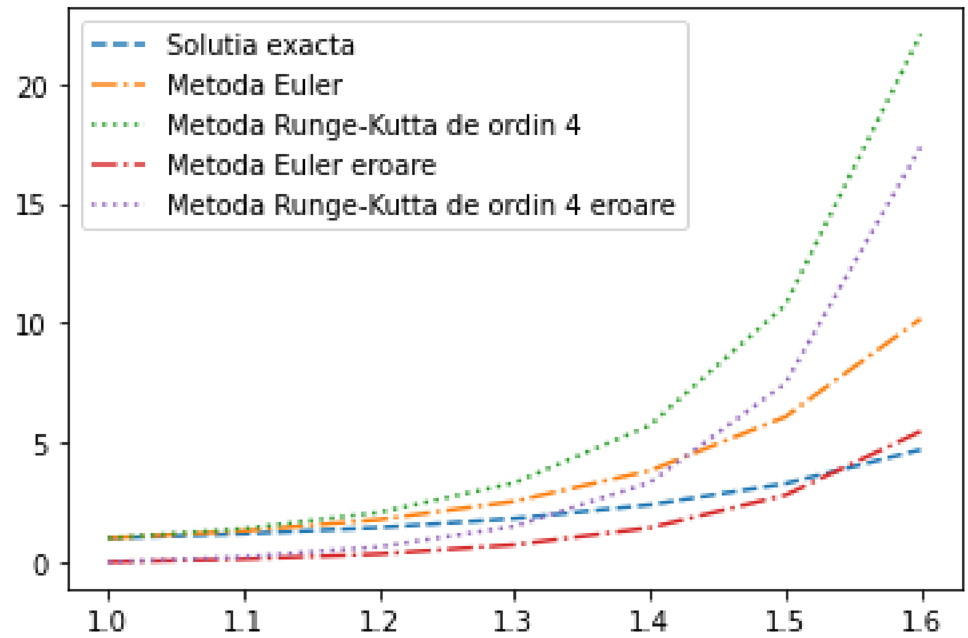
Reprezentare grafică:





2) Compararea metodelor

x	y_exact	y_euler	y_rk4	y_taylor_4	error_euler	error_rk4	error_taylor_4
1.0	1.0000	1.0000	1.0000	1.0000	0.0000	0.0000	0.0000
1.1	1.1800	1.3000	1.3923	1.3707	0.1200	0.2123	0.1907
1.2	1.4391	1.7719	2.0707	2.0762	0.3328	0.6317	0.6371
1.3	1.8194	2.5374	3.3094	3.7977	0.7180	1.4900	1.9783
1.4	2.3917	3.8238	5.7173	12.0516	1.4321	3.3257	9.6599
1.5	3.2789	6.0722	10.7402	480.2822	2.7933	7.4613	477.0033
1.6	4.7021	10.1709	22.0669	1568111214.8584	5.4689	17.3649	1568111210.1564



Metoda Taylor, cat si eroarea sa, au valoarea mult prea mare pentru a fi reprezentate grafic cu celelalte metode. Reprezentarea arata asa: