Homework 3
Marlen Akimaliev
BIL622-Numerical Analysis II

Friday 17th March, 2017

1 Problem

Given the second order differential equation $(x^2 + 1) \times y'' = 2 \times x \times y'$ with initial conditions $y(0) = 1, y(0) = 3$ at the interval $[0; 1]$, find the solution using Runge-Kutta 4^{th} order method with $h = 0.2$. Compare results with the solution of the equation given by $y = -x^3 + 3 \times x + 1$.

2 Solution

Solution for 5 points as a table in the interval $[0, 0.5]$ is as follows:

x	y
0.0	0.0000000000000000
0.1	0.0000000000000000
0.2	0.0010000000000000
0.3	0.0050401000000000
0.4	0.0143450462608010
0.5	0.0315132279968875

We can use the following expression to evaluate the absolute error, which is the sum of the absolute values of the residuals:

$$\varepsilon_{abs} = \sum_{i=1}^{N} |y(x_i) - w_i|$$

I have used the following Python code [1] to evaluate values and plot the graph.

```
import numpy as np
from matplotlib import pyplot as plt
def euler( f, x0, y0, x1, n):
        # determine step-size
        h = (x1-x0)/float(n)
        t = np.arange(x0, x1+h, h )
        w = np.zeros((n+1,))
        t[0] = x0
```

```
    w[0] = y0
    for i in range(1,n+1):
        w[i] = w[i-1] + h * f(t[i-1], w[i-1])
        t[i] = x0 + i * h
    return t,w
def f(x,y): return (y+x)**2
def y(x): return np.tan(x)-x
vx,vy = euler(f, 0, 0, 0.5, 5)
yx = []
for v in vx:
    yx.append(y(v))
dummy, w = euler(f, 0, 0, 0.5, 5)
error_abs = lambda y, w: np.sum(np.abs(y - w))
err = error_abs(yx, w)
print err
for x, y in list(zip(vx, vy))[::1]:
  print("%4.1f %10.16f" % (x, y))
plt.plot(vx, vy, label='approximation')
plt.plot( vx, yx, label='exact' )
plt.title( "Euler's Method Example")
plt.xlabel('Value of x')
plt.ylabel('Value of y')
```

Error value equals to 0.029578291528.
If we try to plot these values and the exact function as a graph result is as follows:

References

[1] Numerical Solutions to ODEs,
http://connor-johnson.com/2014/02/21/numerical-solutions-to-odes/