

```

1  # -*- coding: utf-8 -*-
2  """
3  Created on Mon Nov  4 14:18:51 2019
4
5  @author: Alexm
6  """
7
8  import numpy as np
9  import matplotlib.pyplot as plt
10 print ("Alexandra Mulholland 17336557")
11 #assingment 4
12 #solving and analysing numerical solution to ordinary differential equation
13 
$$\#(1 + t)x + 1 - 3t + t^2$$

14 #must plot direction field (x versus t)
15 #consists of arrows of gradient dx/dt (given by ODE)
16
17 #PART 1
18 #producing said direction field
19
20 def f(T,X):
21     return ((1+T)*X)+1-(3*T)+T**2
22 #defining ranges of grid axes
23 xmin, xmax = -3, 3
24 tmin, tmax = 0, 5
25
26 #defining number of steps
27 dimpts=25
28 x=np.linspace(xmin, xmax, dimpts)
29 t=np.linspace(tmin, tmax, dimpts)
30 T, X = np.meshgrid(t, x)
31
32
33
34 #25x25 grid points
35 #arrows are tangent to the solutions
36 plt.figure(1)
37 plt.quiver(T,X,1,f(T,X), scale=50)
38 plt.ylabel('x(t)')
39 plt.xlabel('t')
40 plt.title('Direction field for dx/dt')
41 plt.ylim(-3.0,3.0)
42 plt.xlim(0,5.0)
43 plt.show()
44
45 #PART 2
46
47 #Defining approximations for the ODE
48 #I have not produced an analytical solution as i think this ODE is unsolvable
49 #not linear nor separable
50 #defining the simple euler
51 def seuler(T,X,step):
52     X_new = X + step*f(T,X)
53     return X_new
54
55 #PART 3
56
57 #improved euler method
58 def ieuler(T,X,step):
59     X_new = X + 0.5*step*(f(T,X) + f(T + step,X+step*f(T,X)))
60     return X_new
61
62 #runge kutta method
63 def rk(T,X,step):
64     k1 = f(T,X)
65     k2 = f(T+ 0.5*step,X + 0.5*step*k1)
66     k3 = f(T + 0.5*step,X + 0.5*step*k2)
67     k4 = f(T + step,X + step*k3,)
68     X_new = X + step/6.0*(k1 + 2.0*k2 + 2.0*k3 + k4)
69     return X_new
70
71 #initial step size =0.04
72 #then reduced in part 3 to 0.02, called step1

```

```

73 #initial condition x(t=0)=x0=0.0655
74 step = 0.04
75 start = 0
76 end = 5
77 X_zero = 0.0655
78 t_zero= 0
79 step1=0.02
80
81 n = int((end-start)/step)
82 t1 = np.arange(start,end,step)
83
84 seul = np.zeros(n)
85 ieul = np.zeros(n)
86 ruku = np.zeros(n)
87
88 seul[0] = X_zero
89 ieul[0] = X_zero
90 ruku[0] = X_zero
91
92 for i in range(1,n):
93     seul[i] = seuler(t1[i-1], seul[i-1], step)
94     ieul[i] = ieuler(t1[i-1], ieul[i-1], step)
95     ruku[i] = rk(t1[i-1], ruku[i-1], step)
96
97 #PART 3b-reducing step size
98
99 seul1 = np.zeros(n)
100 ieul1 = np.zeros(n)
101 ruku1 = np.zeros(n)
102
103 seul1[0] = X_zero
104 ieul1[0] = X_zero
105 ruku1[0] = X_zero
106
107 for i in range(1,n):
108     seul1[i] = seuler(t1[i-1], seul1[i-1], step1)
109     ieul1[i] = ieuler(t1[i-1], ieul1[i-1], step1)
110     ruku1[i] = rk(t1[i-1], ruku1[i-1], step1)
111
112
113
114
115 plt.figure(2)
116 plt.quiver(T, X, 1, f(T,X),scale=50)
117 plt.plot(t1,seul,color='red')
118 plt.ylim(-3,3)
119 plt.title('Euler method atop direction field for step size=0.04')
120 plt.ylabel("x")
121 plt.xlabel("t")
122 plt.show()
123
124 plt.figure(3)
125 plt.quiver(T, X, 1, f(T,X),scale=50)
126 plt.plot(t1,seul,color='red',label='Euler method')
127 plt.plot(t1,ieul,color='green',label='Improved Euler method')
128 plt.plot(t1,ruku,color='blue',label='Runge-Kutta method')
129 plt.ylim(-3,3)
130 plt.title('Plot of the three numerical methods atop the direction field,step=0.04')
131 plt.ylabel("x")
132 plt.xlabel("t")
133 plt.legend()
134 plt.show()
135
136
137 #PART 3b- step size halved
138
139
140
141
142 plt.figure(8)
143 plt.quiver(T, X, 1, f(T,X),scale=50)
144 plt.plot(t1,seul1,color='orange',label='Euler method')

```

```

145 plt.plot(t1,ieul1,color='brown',label='Improved Euler method')
146 plt.plot(t1,ruku1,color='purple',label='Runge-Kutta method')
147 plt.ylim(-3,3)
148 plt.title('Plot of the three numerical methods atop the direction field,step=0.02')
149 plt.ylabel("x")
150 plt.xlabel("t")
151 plt.legend()
152 plt.show()
153
154
155 plt.figure(6)
156 plt.quiver(T, X, 1, f(T,X),scale=50)
157 plt.plot(t1,seul1,color='orange',label='step size 0.02')
158 plt.ylim(-3,3)
159 plt.title('Euler method atop direction field for step size=0.02 and 0.04')
160 plt.plot(t1,seul,color='red',label='step size 0.04')
161 plt.legend(loc='upper right')
162 plt.ylabel("x")
163 plt.xlabel("t")
164 plt.show()
165
166
167 plt.quiver(T, X, 1, f(T,X), scale=50)
168 plt.plot(t1,ieul,color='green', label='step size 0.04')
169 plt.plot(t1,ieul1,color='brown', label='step size 0.02')
170 plt.ylim(-3,3)
171 plt.title('Improved Euler method atop direction field for step size=0.02 and 0.04')
172 plt.ylabel("x")
173 plt.xlabel("t")
174 plt.legend(loc='upper right')
175 plt.show()
176
177 plt.figure()
178 plt.quiver(T, X, 1, f(T,X), scale=50)
179 plt.plot(t1,ruku,color='blue', label='step size 0.04')
180 plt.plot(t1,ruku1,color='purple', label='step size 0.02')
181 plt.ylim(-3,3)
182 plt.title('Runge-Kutta method for step size=0.02 and 0.04')
183 plt.ylabel("x")
184 plt.xlabel("t")
185 plt.legend(loc='upper right')
186 plt.show()

```