|  |  |
| --- | --- |
|  | Atividade 9 CA4322 TURMA 135    Nome N. Matr.  Nome N. Matr. |

1. Resolver a equação diferencial, y’=x\*y2, y(0)=1 h=0.2 e 0≤x≤1.0. Resolver pelo método de Euler e pela função solve. Fazer os gráficos das soluções apresentadas.

Respostas:

Método de Euler

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0 | 0.200 | 0.4000 | 0.6000 | 0.8000 | 1.5000 |
| 1.0000 | 1.0000 | 1.0400 | 1.1265 | 1.2788 | 1.5405 |

solve:

-1/(x^2/2 - 1)

Comandos:

def odeEuler(f,y0,t):

y = np.zeros(len(t))

y[0] = y0

for n in range(0,len(t)-1):

y[n+1] = y[n] + f(y[n],t[n])\*(t[n+1] - t[n])

return y

t = np.linspace(0,1,6)

y0 = 1

f = lambda y,t: y\*t

y = odeEuler(f,y0,t)

print(y)

plt.plot(t,y,'b.-')

plt.legend(['Euler'])

plt.axis([0,1,1,2])

plt.grid(True)

plt.show()

import sympy as sym

import scipy as sp

import matplotlib.pyplot as plt

x = sym.symbols('x')

y= sym.symbols('y', cls=sym.Function)

sol=sym.Eq(y(x).diff(x)-y(x)\*\*2\*x, 0)

sol

soln=sym.dsolve(sol,y(x))

soln

constants = sym.solve([soln.rhs.subs(x,0) - 1]) #condição inicial y(0)=1

display(constants)

resposta=soln.subs(constants)

display(resposta)

import numpy as np

from scipy.integrate import odeint

import matplotlib.pyplot as plt

# function that returns dy/dt

def model(y,t):

dydt = y\*\*2\*t

return dydt

# initial condition

y0 = 1

# time points

t = np.linspace(0,1,6)

# solve ODE

y = odeint(model,y0,t)

print(y)

# plot results

plt.plot(t,y)

plt.xlabel('x')

plt.ylabel('y(x)')

plt.grid(True)

plt.show()

Resultados:

[1. 1. 1.04 1.126528 1.27881584 1.54047503]

[[1. ]

[1.02040831]

[1.08695673]

[1.2195125 ]

[1.47058883]

[2.00000139]]

Tela de computador com texto preto sobre fundo branco

Descrição gerada automaticamente

Gráficos:

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Gráfico

Descrição gerada automaticamente