APMA 0350 – Programming Assignment Solutions

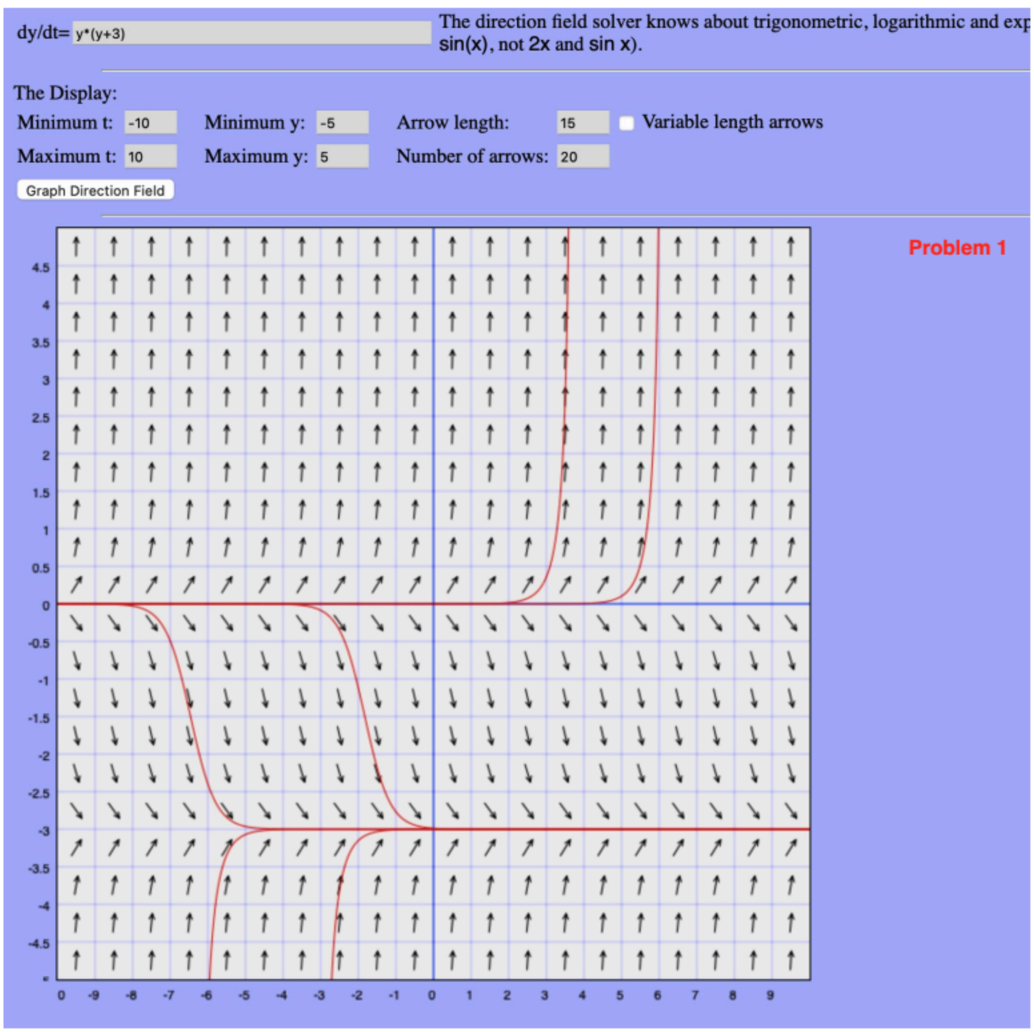
February 26, 2024

1. Use the dfield app to draw the direction field of

$$y' = y(y+3)$$

On that direction field, please click on three solutions, one in the region $y > 0$, one in the region $-3 < y < 0$, and one in the region $y < -3$.

Solution:



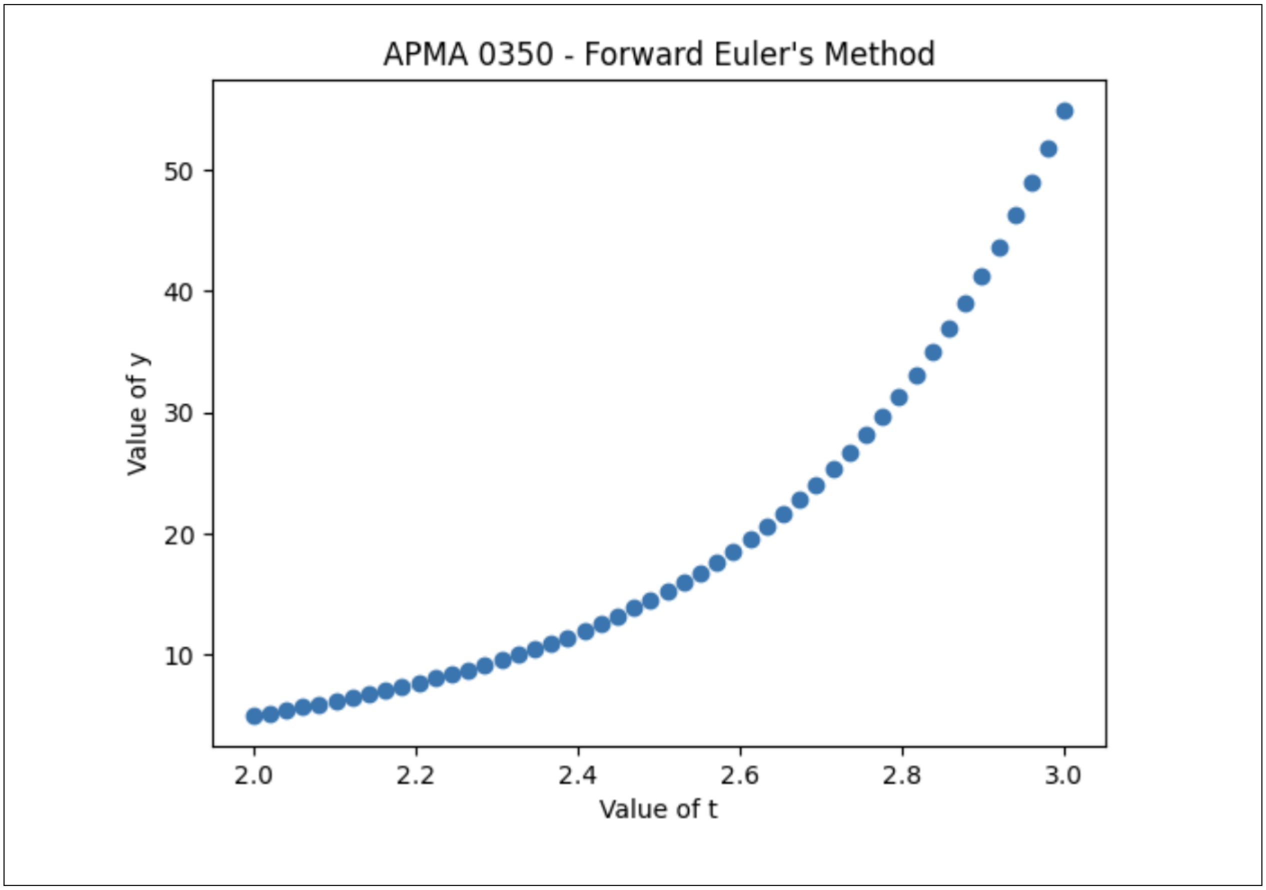
2. Use Python to apply Euler's Method with $N = 50$ on $[2, 3]$ where

$$\begin{cases} y' = \cos(y) + ty \\ y(2) = 5. \end{cases}$$

No need to print the (t, y) values but please plot the points on a graph.

Solution:

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

#Initial condition values
t0 = 2
y0 = 5

#Final value of interval you are solving over
tf = 3

#Stepsize = deltat
n = 50
deltat = (tf - t0)/n

#Time and y variable arrays
t = np.linspace(t0,tf,n)
y = np.zeros([n])

#Initial condition
y[0] = y0

#Method calculation
for i in range(1,n):
    y[i] = deltat*(np.cos(y[i-1]) + (y[i-1]*t[i-1])) + y[i-1]

#Printing results - not necessary for your homework
for i in range(n):
    print(t[i], y[i])

#Plotting
plt.plot(t,y,'o')
plt.xlabel('Value of t')
plt.ylabel('Value of y')
plt.title("APMA 0350 - Forward Euler's Method")
plt.show()
```



3. Use the dsolve command in Python to solve the following. Don't solve them by hand.

(a)

$$y' + y = 3\cos(2t)$$

Solution:

```
#Problem 3
from sympy import *
```

```
t = symbols('t')
y = Function('y')
deq = diff(y(t),t) + y(t) - 3*cos(2*t)
ysoln = dsolve(deq,y(t))
print(ysoln)
```

```
Eq(y(t), C1*exp(-t) + 6*sin(2*t)/5 + 3*cos(2*t)/5)
```

(b)

$$\begin{cases} y' + 2y = 2te^{2t} \\ y(0) = 1 \end{cases}$$

Solution:

Problem 3 - Using dsolve on function in b)

```
#PROBLEM 2#
t=symbols('t')
y=Function('y')
deq=diff(y(t),t)+2*y(t)-2*t*exp(2*t)
ysoln=dsolve(deq,y(t),ics={y(0):1})
print('Solution is',ysoln)
```

```
Solution is Eq(y(t), ((4*t - 1)*exp(4*t)/8 + 9/8)*exp(-2*t))
```

(c)

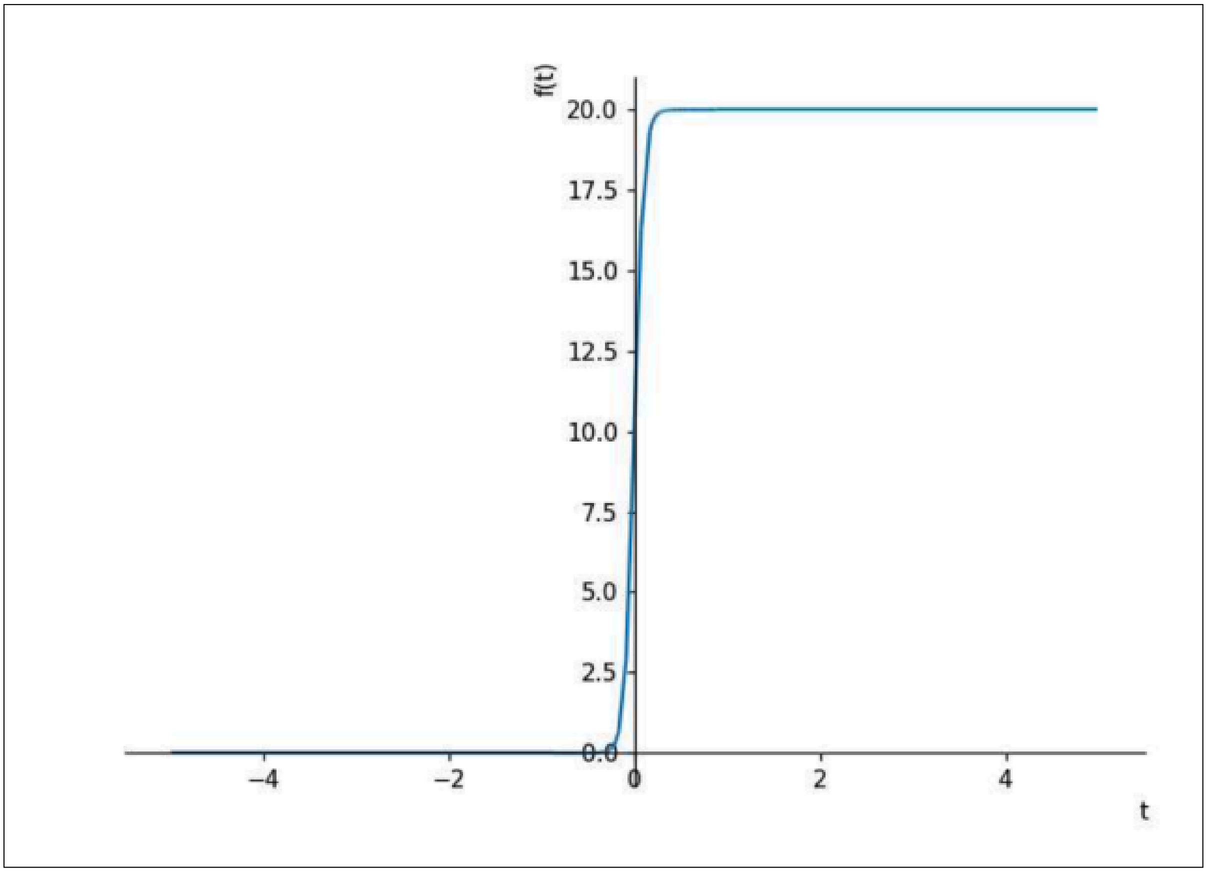
$$\begin{cases} y' = 20y(1 - \frac{y}{20}) \\ y(0) = 10. \end{cases}$$

Please also plot the solution in (c), using -5 and 5 as the t limits and -1 and 21 as the y limits.

Solution:

Problem 3 - Using dsolve on function in c)

```
t=symbols('t')
y=Function('y')
deq=diff(y(t),t) - 20*y(t)*(1-y(t)/20)
ysoln=dsolve(deq,y(t),ics={y(0):10})
plot(ysoln.rhs,(t,-5,5),ylim=[-1,21])
```


f(t)
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
−4
−2
0
2
4
t