import numpy as np  
import matplotlib.pyplot as plt  
import scipy as sci   
import pandas as pd   
import math   
  
# set up constants   
mu = 1.0e-3 #kg/m s viscocity   
rho = 1000.0 #Kg/m^3 fluid density   
d\_tube = 0.953E-3 # inner diameter of the tube   
l\_tube = 2.000 #Length of the tubing   
tc\_area = np.pi\*(100.0\*d\_tube/2)\*\*2 #Ac tube crossectional area   
p1 = 101325.0 #Atmospheric pressure in pascals   
p2 = 102391.6 #Venous pressure in pascals   
grav = 9.81 # gravity in m/s^2   
sa = (500.0/30.0)\*(1/100.0)\*\*2 #bag crossectional area   
vtot = 500.0 # total volume of the IV bag   
icount = 0   
h0 = 0.3 #initial height of the fluid in the IV bag in meters   
H = 1.0 #height of the IV bag relative to the patients arm   
h1 = 2.0 #initialize the the equation   
  
t = 0 # set up time values   
dt = .05 #set up time steps   
  
sigfigs = 4 #set sig figs for the problem   
  
epi = 0.000000001 #dictate a tolerance   
tol = 20 #define an initial tol   
  
x = h0   
  
def v2(x): #define a function to solve for velocity   
 a = 1.0   
 b = (32.0\*mu\*l\_tube/(rho\*d\_tube\*\*2))  
 c = -(grav\*(x+H))  
 vel = (-b + math.sqrt(b\*\*2-4.0\*a\*c))/(2.0\*a)  
 return vel   
  
vmax = v2(h0)\*60.0\*100.0 #solve max velocity   
vmin = v2(0.0)\*60.0\*100 #solve min velocity   
  
# print(vmax)  
# print(vmin)  
# print(tc\_area)  
  
qmax = vmax\*tc\_area #solve for q max  
qmin = vmin\*tc\_area #solve for q min   
  
tmin = vtot/qmax #solve for t min   
tmax = vtot/qmin #solve for t max   
  
#creating a final table   
  
L\_time = ['%.\*g'%(sigfigs,tmin),'%.\*g'%(sigfigs,tmax)] #format time output  
  
#create lists for plotting every minute   
  
L\_timep =['%.\*g'%(sigfigs-1,t/60.0)]  
L\_heightp =['%.\*g'%(sigfigs-1,h0\*100.0)]  
  
# print("Minimum time: %f " % tmin )  
# print(f'Maximum time :{tmax:.4f} minutes')  
  
print(f'Initial Fluid Height :{h0:.4f} m')  
  
ptime = 0 #initialize time   
jcount = 0 #initialize count  
  
while (tol>epi): #Euler's Method   
 icount = icount + 1 #initiate icount  
 jcount = jcount + 1 #initiate icount  
 dd = -(v2(h0)\*np.pi\*(d\_tube/2)\*\*2)/sa #derivitive of function  
 h1 = h0 + dt\*dd #establish new height   
   
 tol = abs(h1-h0)/h1 #calculate tolerance   
 h0 = h1 ##establish new height  
 time = icount\*dt #add to count  
 ptime = jcount\*dt #add to count  
 if (ptime == 120.0): #set up if statement   
 L\_timep.append('%.\*g'%(sigfigs-1,time/60)) #add to time list   
 #print(time/60)  
 L\_heightp.append('%.\*g'%(sigfigs-1,h0\*100.0)) #add to height list   
 jcount = 0   
   
  
L\_timep.append('%.\*g'%(sigfigs-1,time/60)) #append time list   
L\_heightp.append('%.\*g'%(sigfigs-1,h0\*100.0)) #append height list   
L\_time.append('%.\*g'%(sigfigs-1,time/60))  
  
L\_rheightp = list(reversed(L\_heightp)) #reverse height list   
L\_rtimep = list(reversed(L\_timep)) #reverse time list   
x\_ax = np.linspace(0,74,10) #set up x axis   
fig = plt.figure(figsize = (14,8)) #create figure size   
plt.plot(L\_timep,np.zeros(len(L\_timep))) #plot zero line   
plt.plot(L\_rtimep,L\_rheightp) #plot reversed time and height values   
plt.grid(True) #add grid   
plt.title(" Bag Height (cm) vs Time (m) ") #create title   
plt.xlabel(' Time [min] ') #label x axis   
plt.ylabel(' Height (cm) ') #label y axis   
fig.savefig('Initial Value.jpeg',dpio=300,bbox\_inches='tight') #save figure as a \*jpeg  
#print(x\_ax)  
#print(np.zeros(len(x\_ax)))  
  
df = pd.DataFrame({" Actual Time [m] ":L\_timep, #create a dataframe of values   
  
 " Height [cm] ": L\_heightp,})  
  
   
   
df1 = pd.DataFrame({"Min Time [m]": tmin, #create second dataframe of values   
 "Max Time [m]": tmax,  
 "Max Flow Rate[m^3/s]": qmax,  
 "Min Flow Rate [m^3/s]": qmin}, index= [0])  
list\_of\_dfs = [df,df1] #compiled the two data frames   
  
with open('Initial Value Problem.csv','w',) as f: #write csv file   
  
 df.to\_csv('Initial Value Problem.csv', mode='w', index=False, header=True) #write dataframe 1 to the csv file   
  
 f.write("\n") #writes a carriage reture after each data frame to create space   
  
 df1.to\_csv('Initial Value Problem.csv',mode='a', index=False, header=True) #write dataframe 2 to the csv file   
  
#print(df)  
#print(df1)  
#print(L\_timep)  
#print(L\_heightp)

Initial Fluid Height :0.3000 m  
  
  
/var/folders/fg/w2t7p9sx1\_s43rj6b6h5mdqw0000gn/T/ipykernel\_12778/3609721746.py:102: MatplotlibDeprecationWarning: savefig() got unexpected keyword argument "dpio" which is no longer supported as of 3.3 and will become an error in 3.6  
 fig.savefig('Initial Value.jpeg',dpio=300,bbox\_inches='tight') #save figure as a \*jpeg

png

png