Homework 11

Colt Bradley

1 Introduction

This entire lesson was about finding zeros to functions via two methods. We approached this via two methods, I'll illustrate these through the next two examples.

2 Linear Drag

In this example, we find zeros by guessing numbers on either side of where we think the zero will be. I've defined the drag function and guessed two values. The code then calculates the midpoint between them, inputs the three numbers into the drag function, and compares the sign. We know that if the sign is different, there is a zero between the two values, so we can work closer and closer until the values are different within a certain user defined tolerance. This process works, but can take a while.

3 Rocket Example

This method uses a truncated taylor series to calculate the zero. We use the given equation and can begij guessing at any value. The program then runs the code and appends the value to a list. If that value is different from the last guess by only a certain user defined tolerance, the code stops. Otherwise, the code tries its previous answer as the next guess. This is a quick method, but requires calculating the derivative of your function.

4 Code

```
#Colt Bradley
#2.23.16
#Homework Lesson 11
```

```
#import modules
import numpy as n

################################################################
#Linear Drag Exercise
################################################################

#define values used in function
k = .02
g = 9.8
v = 35.

#define our linear drag function
def dragtime(t):
    ans = -(g*t/k) + (v+g/k)/k*(1-n.e**(-k*t))
    return ans

#assign a, b initially, a<b
a = 1.
b = 50.

#put in tolerancefrom problem
tolerance = .001

#define a while loop that works until tolerance is met
while abs(a-b)>= tolerance:
    c = (a+b)/2
    if n.sign(dragtime(c))==n.sign(dragtime(a)):
        a = c
    elif n.sign(dragtime(c))==n.sign(dragtime(b)):
        b = c
#print the final answer
print c
print

#########################################################################
#Rocket Example
###
###
```

```
###
###
#############################################################################


#define values
m0 = 5000. #in kg
rho = 200. #in kg/s
u = 2000. #in m/s
xf = 4000. #in m
v0 = 0
x0 = 0

#define various functions
def mass(t):
    return m0 - rho*t

def velocity(t):
    vel = v0 - u*n.log(mass(t)/m0)
    return vel

def position(t):
    pos = x0+(u + v0)*t + u*(m0/rho - t)*n.log(1 - rho*t/m0)
    return pos

def posprime(t):
    pos = (u + v0) - u*n.log(1 - rho*t/m0)+ \
    u*(m0/rho - t)*((- rho/m0)/(1 - rho*t/m0))
    return pos

#use Newton method
#define an initial guess and a list of guesses
guess = 5
guesses = [0]
guesses.append(guess)

#iterate once to check the value adds correctly, define tolerance
x = guesses[1] - (position(guesses[1])-xf)/posprime(guesses[1])
tolerance = .0000000001
```

```
#create a while loop that runs while answers are outside a specified tolerance
while abs(guesses[-1]-guesses[-2])>= tolerance:
    x = guesses[-1] - (position(guesses[-1])-xf)/posprime(guesses[-1])
    guesses.append(x)

#plug in answer to defined functions, print them out
ans = guesses[-1]
massf = mass(ans)
velf = velocity(ans)
print ans
print massf
print velf
print position(ans)
```