Wanjin Yoo (V00832396)

1/18/2018

CSC349A

Question1)

A)

```
function [t,v] = Euler(m,c,g,t0,v0,tn,n)
% print headings and initial conditions
fprintf('values of t approximations v(t)\n')
fprintf('%8.3f',t0),fprintf('%19.4f\n',v0)
% compute step size h
h=(tn-t0)/n;
% set t,v to the initial values
t = [t0:h:tn] ;
v = zeros(1,n);
v(1)=v0;
% compute v(t) over n time steps using Euler's method
for i=1:n
   v(i+1)= v(i)+ (g-c/m*v(i))*h;
   fprintf('%8.3f',t(i+1)),fprintf('%19.4f\n',v(i+1))
end
end
```

B)

Function call:

[t,v] = Euler(73.5,13.1,9.81,0,0,16,80)

values of t approximations v(t)

0.000	0.0000
0.200	1.9620
0.400	3.8541
0.600	5.6787
0.800	7.4383
1.000	9.1351
1.200	10.7715
1.400	12.3495
1.600	13.8713
1.800	15.3388
2.000	16.7541
2.200	18.1188
2.400	19.4350
2.600	20.7042

2.800	21.9282
3.000	23.1085
3.200	24.2468
3.400	25.3445
3.600	26.4030
3.800	27.4239
4.000	28.4083
4.200	29.3577
4.400	30.2732
4.600	31.1560
4.800	32.0074
5.000	32.8285
5.200	33.6203
5.400	34.3838
5.600	35.1202
5.800	35.8303
6.000	36.5151
6.200	37.1754
6.400	37.8123
6.600	38.4264
6.800	39.0187
7.000	39.5898
7.200	40.1406
7.400	40.6717
7.600	41.1839
7.800	41.6779
8.000	42.1542
8.200	42.6136
8.400	43.0565
8.600	43.4837
8.800	43.8957
9.000	44.2930
9.200	44.6761

9.400	45.0456
9.600	45.4019
9.800	45.7455
10.000	46.0768
10.200	46.3963
10.400	46.7045
10.600	47.0016
10.800	47.2882
11.000	47.5646
11.200	47.8311
11.400	48.0881
11.600	48.3359
11.800	48.5749
12.000	48.8054
12.200	49.0277
12.400	49.2420
12.600	49.4487
12.800	49.6481
13.000	49.8403
13.200	50.0257
13.400	50.2044
13.600	50.3768
13.800	50.5431
14.000	50.7034
14.200	50.8580
14.400	51.0071
14.600	51.1509
14.800	51.2896
15.000	51.4233
15.200	51.5523
15.400	51.6766
15.600	51.7965
15.800	51.9122

16.000 52.0237

t =

Columns 1 through 10

0 0.2000 0.4000 0.6000 0.8000 1.0000 1.2000 1.4000 1.6000 1.8000

Columns 11 through 20

2.0000 2.2000 2.4000 2.6000 2.8000 3.0000 3.2000 3.4000 3.6000 3.8000

Columns 21 through 30

4.0000 4.2000 4.4000 4.6000 4.8000 5.0000 5.2000 5.4000 5.6000 5.8000

Columns 31 through 40

6.0000 6.2000 6.4000 6.6000 6.8000 7.0000 7.2000 7.4000 7.6000 7.8000

Columns 41 through 50

8.0000 8.2000 8.4000 8.6000 8.8000 9.0000 9.2000 9.4000 9.6000 9.8000

Columns 51 through 60

10.0000 10.2000 10.4000 10.6000 10.8000 11.0000 11.2000 11.4000 11.6000 11.8000

Columns 61 through 70

12.0000 12.2000 12.4000 12.6000 12.8000 13.0000 13.2000 13.4000 13.6000 13.8000

Columns 71 through 80

14.0000 14.2000 14.4000 14.6000 14.8000 15.0000 15.2000 15.4000 15.6000 15.8000

Column 81

16.0000

v =

Columns 1 through 10

0 1.9620 3.8541 5.6787 7.4383 9.1351 10.7715 12.3495 13.8713 15.3388

Columns 11 through 20

16.7541 18.1188 19.4350 20.7042 21.9282 23.1085 24.2468 25.3445 26.4030 27.4239

Columns 21 through 30

28.4083 29.3577 30.2732 31.1560 32.0074 32.8285 33.6203 34.3838 35.1202 35.8303

Columns 31 through 40

36.5151 37.1754 37.8123 38.4264 39.0187 39.5898 40.1406 40.6717 41.1839 41.6779

Columns 41 through 50

42.1542 42.6136 43.0565 43.4837 43.8957 44.2930 44.6761 45.0456 45.4019 45.7455

Columns 51 through 60

46.0768 46.3963 46.7045 47.0016 47.2882 47.5646 47.8311 48.0881 48.3359 48.5749

Columns 61 through 70

48.8054 49.0277 49.2420 49.4487 49.6481 49.8403 50.0257 50.2044 50.3768 50.5431

Columns 71 through 80

50.7034 50.8580 51.0071 51.1509 51.2896 51.4233 51.5523 51.6766 51.7965 51.9122

Column 81

52.0237

C)

```
function [ v ]= EulerA(m,c,g,t,n)
```

```
v = (g*m/c) * (1-exp(-(c*t/m)));
fprintf('values of t approximations v(t)\n')
for i=1:n+1
   fprintf('%8.3f',t(i)),fprintf('%19.4f\n',v(i))
end
end
```

Function call:

[v2] = EulerA(73.5,13.1,9.81,t,80)

values of t approximations v(t)

0.000	0.0000
0.200	1.9274
0.400	3.7874
0.600	5.5822
0.800	7.3142
1.000	8.9855
1.200	10.5983
1.400	12.1546
1.600	13.6564
1.800	15.1056
2.000	16.5041
2.200	17.8536
2.400	19.1558
2.600	20.4124
2.800	21.6251
3.000	22.7952
3.200	23.9244
3.400	25.0141
3.600	26.0656
3.800	27.0802
4.000	28.0594
4.200	29.0042
4.400	29.9160
4.600	30.7958

4.800	31.6448
5.000	32.4641
5.200	33.2547
5.400	34.0176
5.600	34.7538
5.800	35.4643
6.000	36.1498
6.200	36.8113
6.400	37.4497
6.600	38.0657
6.800	38.6602
7.000	39.2338
7.200	39.7873
7.400	40.3215
7.600	40.8369
7.800	41.3343
8.000	41.8143
8.200	42.2775
8.400	42.7244
8.600	43.1557
8.800	43.5719
9.000	43.9736
9.200	44.3611
9.400	44.7351
9.600	45.0960
9.800	45.4442
10.000	45.7803
10.200	46.1046
10.400	46.4175
10.600	46.7195
10.800	47.0109

11.000	47.2921
11.200	47.5634
11.400	47.8253
11.600	48.0780
11.800	48.3218
12.000	48.5571
12.200	48.7841
12.400	49.0032
12.600	49.2147
12.800	49.4187
13.000	49.6156
13.200	49.8055
13.400	49.9889
13.600	50.1658
13.800	50.3365
14.000	50.5012
14.200	50.6602
14.400	50.8136
14.600	50.9616
14.800	51.1045
15.000	51.2423
15.200	51.3754
15.400	51.5037
15.600	51.6276
15.800	51.7471
16.000	51.8624

v2 =

Columns 1 through 10

0 1.9274 3.7874 5.5822 7.3142 8.9855 10.5983 12.1546 13.6564 15.1056

Columns 11 through 20

16.5041 17.8536 19.1558 20.4124 21.6251 22.7952 23.9244 25.0141 26.0656 27.0802

Columns 21 through 30

28.0594 29.0042 29.9160 30.7958 31.6448 32.4641 33.2547 34.0176 34.7538 35.4643

Columns 31 through 40

36.1498 36.8113 37.4497 38.0657 38.6602 39.2338 39.7873 40.3215 40.8369 41.3343

Columns 41 through 50

41.8143 42.2775 42.7244 43.1557 43.5719 43.9736 44.3611 44.7351 45.0960 45.4442

Columns 51 through 60

45.7803 46.1046 46.4175 46.7195 47.0109 47.2921 47.5634 47.8253 48.0780 48.3218

Columns 61 through 70

48.5571 48.7841 49.0032 49.2147 49.4187 49.6156 49.8055 49.9889 50.1658 50.3365

Columns 71 through 80

50.5012 50.6602 50.8136 50.9616 51.1045 51.2423 51.3754 51.5037 51.6276 51.7471

Column 81

51.8624

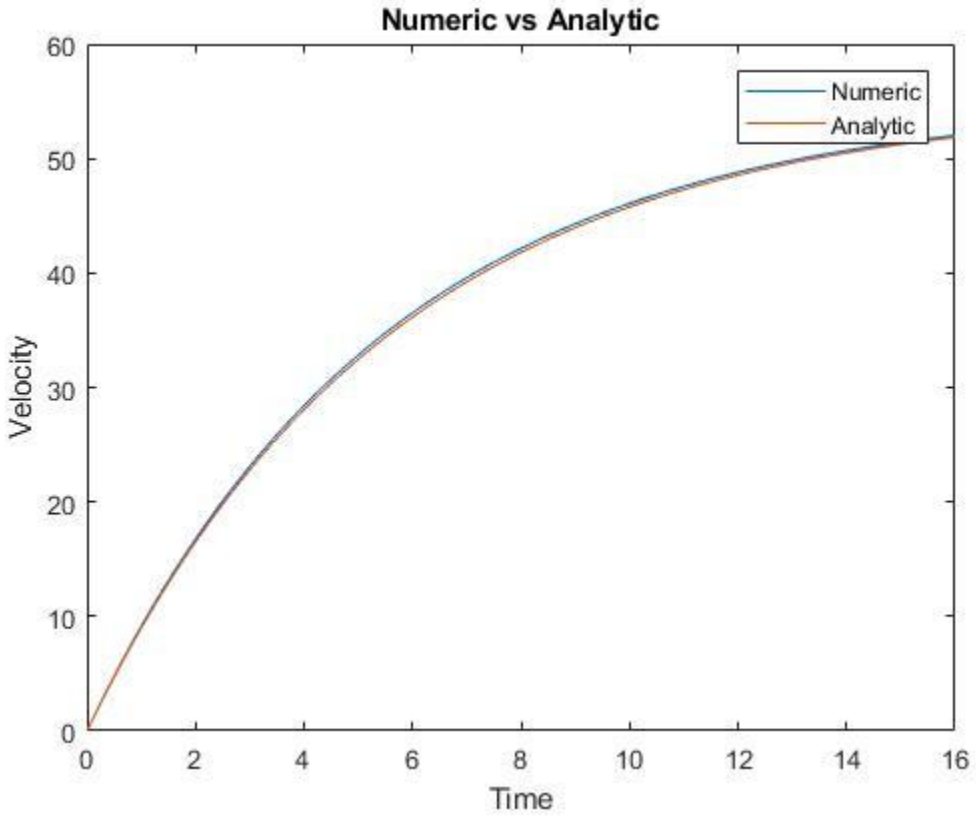
D)

I changed the name of variables for velocity

V -> Numeric

V2 -> Analytic

plot(t,Numeric,t,Analytic)



E)

Terminal velocity is gm/c since time (t) goes to infinity

By calculating g*m/c with given parameters, we get 73.5 * 98.1 / 13.1 = 55.0408

And result from function

Numeric approach:

[t,Numeric] = Euler(73.5,13.1,9.81,0,0,500,80)

Analytic approach:

[Analytic] = EulerA(73.5,13.1,9.81,t,80)

values of t approximations v(t)

0.000	0.0000
6.250	36.9731
12.500	49.1099
18.750	53.0939
25.000	54.4017
31.250	54.8311
37.500	54.9720
43.750	55.0182
50.000	55.0334
56.250	55.0384
62.500	55.0400
68.750	55.0406
462.500	55.0408
468.750	55.0408
475.000	55.0408
481.250	55.0408
487.500	55.0408
493.750	55.0408
500.000	55.0408

Where t = 500 and n = 80

I only included the result from analytic approach because they both have the same result, and it takes up too much space if I include two results .

Question 2)

(a)

```
function Euler2(m , k , g, t0 , v0 , tn , n)
h =(tn-t0)/n;
t = [t0:h:tn];
v = v0;
fprintf('values of t approximations dv/dt\n')
for i=1:n+1
   fprintf('%8.3f',t(i)),fprintf('%19.4f\n',v)
   v = v +( g - ((k/m)*v.^2))*h;

end
end
```

(B)

Euler2(73.5,0.234,9.81,0,0,18,72)

values of t approximations dv/dt

0.000	0.0000
0.250	2.4525
0.500	4.9002
0.750	7.3336
1.000	9.7433
1.250	12.1202
1.500	14.4558
1.750	16.7420
2.000	18.9714
2.250	21.1374
2.500	23.2343
2.750	25.2572
3.000	27.2019
3.250	29.0655
3.500	30.8456
3.750	32.5408
4.000	34.1505
4.250	35.6748
4.500	37.1143
4.750	38.4705
5.000	39.7450
5.250	40.9402
5.500	42.0587
5.750	43.1033
6.000	44.0770
6.250	44.9832
6.500	45.8252
6.750	46.6063
7.000	47.3300
7.250	47.9995
7.500	48.6182
7.750	49.1894
8.000	49.7161
8.250	50.2013
8.500	50.6480
8.750	51.0588

9.000	51.4363
9.250	51.7831
9.500	52.1013
9.750	52.3933
10.000	52.6609
10.250	52.9062
10.500	53.1309
10.750	53.3366
11.000	53.5249
11.250	53.6971
11.500	53.8547
11.750	53.9988
12.000	54.1305
12.250	54.2509
12.500	54.3608
12.750	54.4613
13.000	54.5531
13.250	54.6369
13.500	54.7134
13.750	54.7833
14.000	54.8471
14.250	54.9053
14.500	54.9584
14.750	55.0069
15.000	55.0512
15.250	55.0915
15.500	55.1284
15.750	55.1620
16.000	55.1926
16.250	55.2206
16.500	55.2461
16.750	55.2693
17.000	55.2905
17.250	55.3099
17.500	55.3275
17.750	55.3436
18.000	55.3583

c)

```
function  Analytic(t)

v = sqrt(9.81*73.5/0.234) * tanh(sqrt(9.81*0.234/73.5)*t);
fprintf('%19.4f\n',v)
```

Function call:

>> Analytic(18)

55.3186

The value of v from analytic approach when t =18 is 55.3186

Relative error is = absolute value of (1-55.3583/55.3186)

Which is 7.1766e-04 -> 0.00072

Question 3)

I create matlab functions to solve the question**.**

Here are my functions for the two different approximations :

First approximation

```
function[result1] = Mclaurin(n)
sum =0;
for i=0:n
    sum = sum + (-1)^i * 2^i / factorial(i);
end
result1= sum;
end
```

Second Approximation

```
function[result2] = Mclaurin2(n)
sum = 0;
for i=0:n
    sum = sum + 2^i / factorial(i);
end
result2 = 1/sum;
end
```

Simple function to find relative error

```
function [error] = findrelativeE(p,pstar)
error = abs(1-pstar / p);
end
```

Relative error from 1 to 5 for the two different approximations:

P is 0.1353 since e^-2

N=1
Mclaurin(1)
-1
Relative error:
findrelativeE(exp(-2),Mclaurin(1))
8.3891

Mclaurin2(1)
0.3333
Relative error:
findrelativeE(exp(-2),Mclaurin2(1))
1.4630

N=2
Mclaurin(2)
1
Relative error:
findrelativeE(exp(-2),Mclaurin(2))
6.3891

Mclaurin2(2)
0.2000
Relative error:
findrelativeE(exp(-2),Mclaurin2(2))
0.4778

N=3
Mclaurin(3)
-0.3333
Relative error:
findrelativeE(exp(-2),Mclaurin(3))
3.4630

Mclaurin2(3)
0.1579
Relative error:
findrelativeE(exp(-2),Mclaurin2(3))
0.1667

N=4
Mclaurin(4)
0.3333
Relative error:
findrelativeE(exp(-2),Mclaurin(4))
1.4630

Mclaurin2(4)
0.1429
Relative error:
findrelativeE(exp(-2),Mclaurin2(4))
0.0556

N=5
Mclaurin(5)
0.0667
Relative error:
findrelativeE(exp(-2),Mclaurin(5))
0.5074

Mclaurin2(5)
0.1376
Relative error:
findrelativeE(exp(-2),Mclaurin2(5))
0.0168

At a quick glance of the relative errors of the two different approximation, the second approach is much better and has lower error than the first approach.
In conclusion. The second approach has less error and closer to its orginal value (e^-2).