Oliver Tonnesen
V00885732

CSC 349A

Assignment 1

September 17, 2019

1

1.a

```
function Euler(m,c,g,t0,v0,tn,n)
fprintf('values of t\tapproximations v(t)\n')
fprintf('%8.3f%19.4f\n', t0, v0)
h=(tn-t0)/n;
t=t0;
v=v0;
for i=1:n
    v=v+(g-c/m*v)*h;
    t=t+h;
    fprintf('%8.3f%19.4f\n', t, v)
end
```

1.b

```
Euler (72.7,12.5,9.81,0,0,10,20)
values of t      approximations v(t)
   0.000             0.0000
   0.500             4.9050
   1.000             9.3883
   1.500            13.4862
   2.000            17.2318
   2.500            20.6554
   3.000            23.7846
   3.500            26.6449
   4.000            29.2592
   4.500            31.6488
   5.000            33.8330
   5.500            35.8294
   6.000            37.6541
   6.500            39.3220
   7.000            40.8465
   7.500            42.2399
   8.000            43.5136
   8.500            44.6777
   9.000            45.7418
   9.500            46.7144
  10.000            47.6034
```

```
diary off
```

1.c

```
Euler (72.7, 12.5, 8.87, 0, 0, 10, 20)
values of t      approximations v(t)
   0.000              0.0000
   0.500              4.4350
   1.000              8.4887
   1.500             12.1940
   2.000             15.5806
   2.500             18.6762
   3.000             21.5056
   3.500             24.0918
   4.000             26.4556
   4.500             28.6162
   5.000             30.5911
   5.500             32.3962
   6.000             34.0461
   6.500             35.5542
   7.000             36.9326
   7.500             38.1925
   8.000             39.3441
   8.500             40.3967
   9.000             41.3588
   9.500             42.2382
  10.000             43.0420
diary off
```

1.d

```
v_approx=47.6034

v_approx =

   47.6034

v_true=(9.81*72.7/12.5)*(1-exp(-1*12.5*10/72.7))

v_true =
```

```
   46.8322

err_relative=abs(1-(v_approx/v_true))

err_relative =

    0.0165

diary off
```

2

2.a

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

2.b

```
Euler2(43.5,0.234,9.81,0,0,15,60)
values of t     approximations v(t)
   0.000             0.0000
   0.250             2.4525
   0.500             4.8969
   0.750             7.3172
   1.000             9.6977
   1.250            12.0237
   1.500            14.2818
   1.750            16.4600
   2.000            18.5481
   2.250            20.5379
   2.500            22.4232
```

2.750	24.1995
3.000	25.8645
3.250	27.4173
3.500	28.8589
3.750	30.1914
4.000	31.4180
4.250	32.5431
4.500	33.5713
4.750	34.5082
5.000	35.3592
5.250	36.1303
5.500	36.8273
5.750	37.4559
6.000	38.0216
6.250	38.5300
6.500	38.9860
6.750	39.3945
7.000	39.7599
7.250	40.0865
7.500	40.3779
7.750	40.6378
8.000	40.8695
8.250	41.0757
8.500	41.2592
8.750	41.4223
9.000	41.5674
9.250	41.6962
9.500	41.8106
9.750	41.9122
10.000	42.0023
10.250	42.0823
10.500	42.1532
10.750	42.2161
11.000	42.2718
11.250	42.3213
11.500	42.3651
11.750	42.4039
12.000	42.4382
12.250	42.4687
12.500	42.4957

```
  12.750          42.5196
  13.000          42.5407
  13.250          42.5595
  13.500          42.5761
  13.750          42.5908
  14.000          42.6038
  14.250          42.6153
  14.500          42.6255
  14.750          42.6346
  15.000          42.6426
diary off
```

2.c

```
v_approx=42.6426

v_approx =

   42.6426

v_true=sqrt(9.81*43.5/0.234)*tanh(sqrt
   (9.81*0.234/43.5)*15)

v_true =

   42.6175

err_relative=abs(1-(v_approx/v_true))

err_relative =

   5.8784e-04

diary off
```

3

Below is a function to approximate e^{-x} directly from its MacLaurin series expansion, and a sample output using it to approximate e^{-3}..

```
function exp_inv_1(x,n)
```

```
s=0;
sgn=1;
fprintf("n\tapproximation e^-x\trelative error\n");
for i=0:n
    s=s+sgn*(x^i)/factorial(i);
    sgn = sgn*-1;
    fprintf("%d\t\t%.4f\t\t\t%.4f\n", i, s, abs(1-s/exp(-1*x)));
end
```

```
exp_inv_1(3,5)
n       approximation e^-x      relative error
0               1.0000                  19.0855
1               -2.0000                 41.1711
2               2.5000                  49.2138
3               -2.0000                 41.1711
4               1.3750                  26.6176
5               -0.6500                 14.0556
diary off
```

Below is a function to approximate e^{-x} by taking the inverse of the MacLaurin series expansion of e^{x}, and a sample output using it to approximate e^{-3}..

```
function exp_inv_2(x,n)
s=0;
fprintf("n\tapproximation e^x\trelative error\n");
for i=0:n
    s=s+(x^i)/factorial(i);
    fprintf("%d\t\t%.4f\t\t\t%.4f\n", i, 1/s, abs(1-(1/s)/exp(-1*x)));
end
```

```
exp_inv_2(3,5)
n       approximation e^x       relative error
0               1.0000                  19.0855
1               0.2500                  4.0214
2               0.1176                  1.3630
3               0.0769                  0.5450
4               0.0611                  0.2266
5               0.0543                  0.0916
diary off
```

We can see when using the first method that the relative error did not appear to begin converging to any value as we increased n, while the second method's relative error appears to be getting smaller as n increases. Note that the first method always gives a polynomial, so its limits as $x \to \pm\infty$ are ∞ or $-\infty$ (except when $n = 0$, of course), but the second method gives the inverse of a polynomial, so its limits as $x \to \pm\infty$ are 0. Overall, the series of functions given by the second method look much more similar to one another than those given by the first, perhaps leading to a more stable approximation.