Table of Contents


```
% Caleb Bibb
% Math 341 H9
```

Section 5.3 #1c

Use Taylor's method of order two to approx the solution for y' = 1+y/t, 1<=t<=2, y(1)=2, with h = .25 T^(2)(t(i),w(i)) = f(t(i),w(i)) + (h/2)*f'(t(i),w(i))+...+h^(n-1)/n! * f^(n-1)(t(i),w(i)) So, f'(t(i),y(t)) = d/dt(1+y/t) = -y'/t^2 = -(1+y/t)/t^2 Then, T^(2)(t(i),w(i)) = f(t(i),w(i)) + (h/2)f'(t(i),w(i)) =

```
f = @(t,w) 1+w./t+(.25/2)*(-(1+w./t)/t.^2); % T^(2)
N = 4; % 4 t-values see below
w(1) = 2; % Initial Condition
t = 1:.25:2; % t is in [1,2] in .25 increments (h = .25)
for i = 1:N
 w(i+1) = w(i)+.25*(f(t(i),w(i)));
end

t = t'; % Row vector -> Column Vector (Transpose)
Taylor2 = w'; % Row vector -> Column Vector (Transpose)
T2 = table(t,Taylor2); % Make Table
disp(T2); % Display Table
```

t	Taylor2
1	2
1.25	2.6563
1.5	3.375
1.75	4.1424
2	4.9498

Section 5.3 #3c

Repeat #1c using Taylor's method of order 4. T^(4) = f(t(i),w(i)) + (h/2)f'(t(i),w(i)) + (h/6)f''(t(i),w(i)) + (h/24)f'''(t(i),w(i))

```
clear all
```

```
f = @(t,w) 1+w./t+(.25/2)*(-(1+w./t)/t.^2)+(.25/6)*(2*(1+w./t)/
t.^3)+(.25/24)*(-6*(1+w./t)/t.^4); % T^(4)
N = 4; % 4 t-values see below
w(1) = 2; % Initial Condition
t = 1:.25:2; % t is in [1,2] in .25 increments (h = .25)
for i = 1:N
 w(i+1) = w(i)+.25*(f(t(i),w(i)));
end

t = t'; % Row vector -> Column Vector (Transpose)
Taylor4 = w'; % Row vector -> Column Vector (Transpose)
T4 = table(t,Taylor4); % Make Table
disp(T4); % Display Table
```

t	Taylor4
1	2
1.25	2.6719
1.5	3.4069
1.75	4.1894
2	5.0108

Section 5.4 #1c

Use the modified Euler method to approx the solution and compare: y' = 1+y/t, 1<=t<=2, y(1)=2, with h = .25; actual solution: y(t) = t*ln(t) + 2

```
clear all
f = @(t,w) 1.+w./t;
g = @(t) t.*log(t)+2.*t;
N = 4; % 4 t-values see below
w(1) = 2; % Initial Condition
t = 1:.25:2; % t is in [1,2] in .25 increments (h = .25)
h = 0.25;

for i = 1:N
 w(i+1) = w(i)+(h/2)*(f(t(i),w(i))+f(t(i+1),w(i)+h*f(t(i),w(i))));
end

act = g(t); % Actual Values
t = t'; % Row vector -> Column Vector (Transpose)
ModifiedEuler = w'; % Row vector -> Column Vector (Transpose)
Actual = act'; % Row vector -> Column Vector (Transpose)
ME = table(t,ModifiedEuler,Actual); % Make Table
disp(ME); % Display Table
```

t	ModifiedEuler	Actual
1	2	2
1.25	2.775	2.7789
1.5	3.6008	3.6082

```
     1.75     4.4688            4.4793
        2     5.3729            5.3863
```

Section 5.4 #5c

Repeat using the Midpoint Method

```
clear all
f = @(t,w) 1.+w./t;
g = @(t) t.*log(t)+2.*t;
N = 4; % 4 t-values see below
w(1) = 2; % Initial Condition
t = 1:.25:2; % t is in [1,2] in .25 increments (h = .25)
h = 0.25;

for i = 1:N
 w(i+1) = w(i) + h*f(t(i)+h/2,w(i)+h/2*f(t(i),w(i)));
end

act = g(t); % Actual Values
t = t'; % Row vector -> Column Vector (Transpose)
Midpoint = w'; % Row vector -> Column Vector (Transpose)
Actual = act'; % Row vector -> Column Vector (Transpose)
MP = table(t,Midpoint,Actual); % Make Table
disp(MP); % Display Table
```

t	Midpoint	Actual
1	2	2
1.25	2.7778	2.7789
1.5	3.6061	3.6082
1.75	4.4763	4.4793
2	5.3824	5.3863

Section 5.4 #13c

Repeat Exercise 1 using Runge-Kutta method of order 4:

```
%RK4 (Function, LeftEndpoint, RightEndpoint, Iterations, InitialVal,
 ActualSoln)
RK4(@(t,w)1.+w./t,1,2,4,2,@(t)t.*log(t)+2.*t);
```

t	RK4	Actual
1.25	2.7789	2.7789
1.5	3.6082	3.6082
1.75	4.4793	4.4793
2	5.3862	5.3863

Section 5.9 #1a

Use the Runge-Kutta for Systems Algorithm to approx the solutions of the following differential equation and compare the result to the actual solution. u1' = 3u1+2u2-(2t^2+1)exp(2t), u1(0) = 1; u2' = 4u1 + u2 + (t2 + 2t?4)exp(2t) , u2(0) = 1; 0 ? t ? 1; h = 0.2;

```
%actual solutions u1(t) = 1/3*exp(5t) ? 1/3*exp(?t) + exp(2t) and
 u2(t) =
%1/3*exp(5t) + 2/3 exp(?t) + t^2exp(2t)
RKSystem ({@(t,u1,u2) 3.*u1+2.*u2-(2.*t.^2+1).*exp(2.*t);@(t,u1,u2)
 4.*u1+u2+(t.^2+2.*t-4).*exp(2.*t)}, {@(t) (1/3.*exp(5.*t)-1/3.*exp(-
t)+exp(2.*t));@(t) (1/3.*exp(5.*t)+2/3.*exp(-t)+t.^2.*exp(2.*t))}, 0,
 1, .2, [0;0], 2);
```

t	w1	Actual1	w2	Actual2
0	0	1	0	1
0.2	-0.58797	2.125	-1.2013	1.5116
0.4	-2.8938	4.4651	-4.0928	3.266
0.6	-10.127	9.8324	-11.702	8.2563
0.8	-31.127	23.003	-32.46	21.669
1	-90.056	56.737	-89.687	57.105

Section 5.9 #3a

Use the Runge-Kutta for Systems Algorithm to approx the solutions of the following differential equation and compare the result to the actual solution. y'' -2y' +y = te^t-t, 0<=t<=1, y(0)=0, y'(0)=0, with h = .1; Actual Solution: y(t) = 1/6*t^3*exp(t)+2*exp(t)-t-2 u1(t) = y(t), u2(t) = y'(t). Then u1' = y'(t) = u2. u2' = y'' = 2y'-y+te^t-t This is a 1st order system of 2 DEs with initial y(0)=0, y'(0)=0

```
clear all
RKSystem({@(t,u1,u2)u2;@(t,u1,u2)2.*u2-u1+t.*exp(t)-t},
{@(t)1/6*t.^3.*exp(t)-t.*exp(t)+2*exp(t)-t-2;@(t)t},0,1,.1,[0;0],2)
```

t	w1	Actual1	w2	Actual2
0	0	0	0	0
0.1	8.9724e-06	8.9395e-06	0.00036392	0.1
0.2	0.00015352	0.0001535	0.0031788	0.2
0.3	0.00083427	0.00083434	0.011719	0.3
0.4	0.0028321	0.0028323	0.030353	0.4
0.5	0.0074297	0.0074303	0.064798	0.5
0.6	0.016561	0.016563	0.12242	0.6
0.7	0.032996	0.032998	0.21261	0.7
0.8	0.060559	0.060562	0.34719	0.8
0.9	0.1044	0.10441	0.54094	0.9
1	0.17132	0.17133	0.81218	1

ans =

```
         0
    0.1000
    0.2000
    0.3000
    0.4000
    0.5000
    0.6000
    0.7000
    0.8000
    0.9000
    1.0000
```

Section 5.9 #9

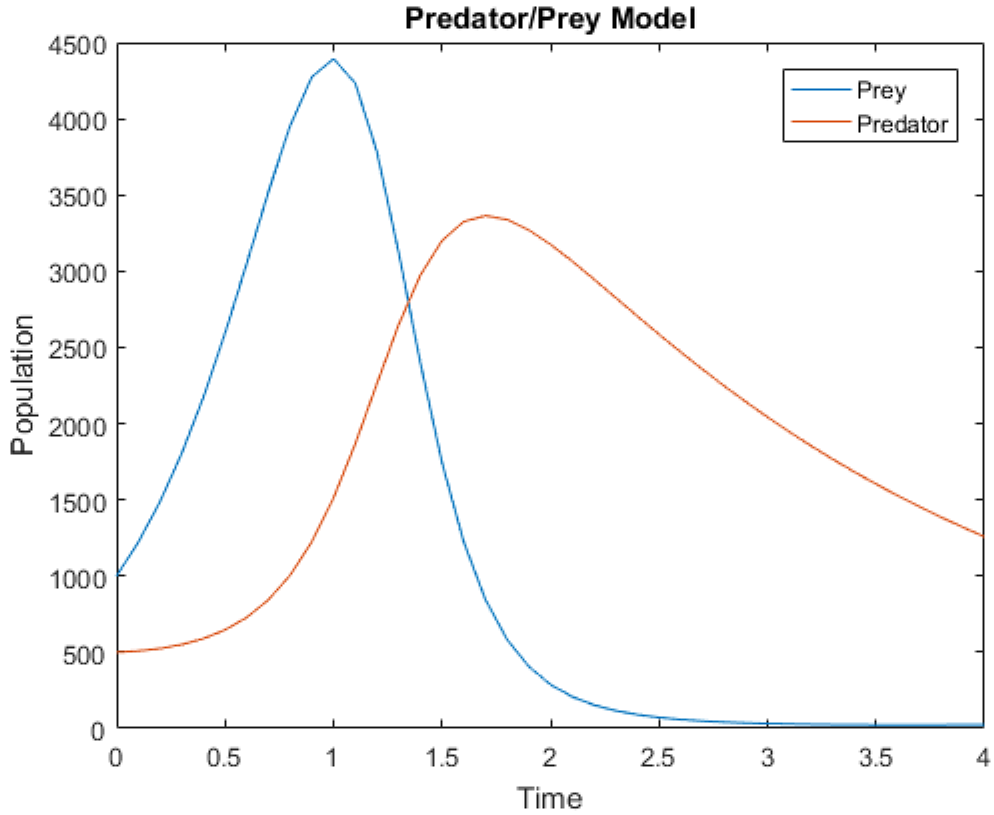
This problem is a predator/prey model. With x1'(t) = k1x1(t)-k2x2(t) for the prey and x2'(t) = k3x1(t)x2(t)-k4x2(t). We want to solve for 0<=t<=4. Assuming Initial prey = 1000 and preditors = 500 and k1 = 3, k2 = .002, k3 = .0006, and k4 = .5. Substituting gets us: x1'(t) = 3*x1(t)-.002x2(t), x2'(t) = .0006x1(t)*x2(t)-.5x2(t)

```
clear all
[t,C,Act1,Act2]=RKSystem({@(t,x1,x2)3.*x1-.002.*x1.*x2;@(t,x1,x2).0006.*x1.*x2-.5.
{@(t)t;@(t)t;},0,4,.1,[1000;500],2);
figure
plot(t,C(:,1),t,C(:,2));
legend('Prey', 'Predator', 'Location', 'NorthEast')
xlabel('Time')
ylabel('Population')
title('Predator/Prey Model')

% We can see that at t=4, the number of prey is approximately 25 and
 the
% number of predators is approximately 1258.  There are two "stable
% solutions", one at x1'=x2'=0 (When there are no predators or prey),
 and
% k1-k2*k2(t)=0 and k3x1(t)-k4=0 which can mean that x1(t)=k4/k3 and
% x2(t)=k1/k2. This occurs ar x1(t)=833.333333 and x2(t)=1500.00.
 (this
% can't ever happen since we can't have a third of an animal).
```

t	w1	Actual1	w2	Actual2
0	1000	0	500	0
0.1	1220.5	0.1	508.27	0.1
0.2	1486.1	0.2	524.25	0.2
0.3	1802	0.3	550.24	0.3
0.4	2171	0.4	589.51	0.4
0.5	2590.7	0.5	646.71	0.5
0.6	3049.2	0.6	728.48	0.6
0.7	3519.4	0.7	843.92	0.7
0.8	3952.3	0.8	1004.8	0.8
0.9	4273.6	0.9	1224.3	0.9
1	4392.9	1	1512.2	1

1.1	4233.9	1.1	1866.2	1.1
1.2	3783.9	1.2	2260.8	1.2
1.3	3124.5	1.3	2647.7	1.3
1.4	2400.9	1.4	2972.5	1.4
1.5	1745.1	1.5	3200.2	1.5
1.6	1224.7	1.6	3325.3	1.6
1.7	845.84	1.7	3363.6	1.7
1.8	583.68	1.8	3338.1	1.8
1.9	406.7	1.9	3269.9	1.9
2	288.09	2	3175.1	2
2.1	208.33	2.1	3065.1	2.1
2.2	154.13	2.2	2947.2	2.2
2.3	116.8	2.3	2826.1	2.3
2.4	90.684	2.4	2705	2.4
2.5	72.123	2.5	2585.6	2.5
2.6	58.731	2.6	2469.1	2.6
2.7	48.935	2.7	2356.2	2.7
2.8	41.688	2.8	2247.4	2.8
2.9	36.28	2.9	2142.8	2.9
3	32.227	3	2042.5	3
3.1	29.195	3.1	1946.4	3.1
3.2	26.949	3.2	1854.6	3.2
3.3	25.327	3.3	1766.9	3.3
3.4	24.214	3.4	1683.2	3.4
3.5	23.531	3.5	1603.4	3.5
3.6	23.227	3.6	1527.4	3.6
3.7	23.269	3.7	1454.9	3.7
3.8	23.644	3.8	1385.9	3.8
3.9	24.35	3.9	1320.2	3.9
4	25.401	4	1257.7	4



Published with MATLAB® R2016b