CRISTIAN B. JETOMO
MATH 175 - H4L

Second Long Exam (Computational Part)

In this problem, we are to solve the Ordinary Differential Equation given by

$$\frac{dx}{dt} = -(1+t+t^2) - (2t+1)x - x^2 \tag{1}$$

on the interval $t \in [0,3]$. Particularly, we apply the Taylor Method with different step sizes h on two IVPs involving Equation 1. We record the average global discretization error (GDE) and average global discretization relative error (GDRE) for each of the runs. Moreover, we solve the IVP using a Runge-Kutta Method and the order 4 Runge-Kutta Method. For this we use the average GDE and GDRE and compare our results from the others. We start with the Taylor Method.

First, we consider the IVP given by

$$\begin{cases} \frac{dx}{dt} & = f(t,x) \\ x(0) & = -\frac{1}{2} \end{cases} \tag{2}$$

where $f(t,x)$ is the right hand side of Equation 1. That is

$$f(t,x) = -(1+t+t^2) - (2t+1)x - x^2 \tag{3}$$

The exact solution of this IVP is then

$$x(t) = -t - \frac{1}{e^t+1}$$

We set up the second order Taylor Method (T2) for Equation 2. Hence, we have

$$\begin{aligned} \frac{w_{i+1}-w_i}{h} &= f(t_i,w_i) + \frac{h}{2}\frac{df}{dt}(t_i,w_i) \\ w_{i+1}-w_i &= hf(t_i,w_i) + \frac{h^2}{2}\frac{df}{dt}(t_i,w_i) \\ w_{i+1} &= w_i + hf(t_i,w_i) + \frac{h^2}{2}\frac{df}{dt}(t_i,w_i) \end{aligned}$$

where w_i's are the approximate values for x_i's and h is the uniform step size. From the given function f, we note that

$$\begin{aligned} \frac{df}{dt} &= \frac{\partial f}{\partial t} + \frac{\partial f}{\partial x}\cdot f \\ &= [-(1+2t) - x(2)] + [-(2t+1) - 2x]\, f \\ &= (-2x-2t-1)(1+f) \end{aligned} \tag{4}$$

Therefore, our marching algorithm for solving the IVP using T2 is

$$\begin{cases} w_0 & = w(0) = -\frac{1}{2} \\ w_{i+1} & = w_i + hf(t_i,w_i) + \frac{h^2}{2}\frac{df}{dt}(t_i,w_i) \end{cases} \tag{5}$$

where $f(t_i,w_i)$ and $\frac{df}{dt}(t_i,w_i)$ are the values of Equations 3 and 4 upon substitution of (t_i,w_i), respectively. We now implement this in MATLAB using stepsizes $h_n = \frac{1}{2^n}$ for $n = 1,2,\ldots,8$. The results of this implementation is presented in Figure 1.

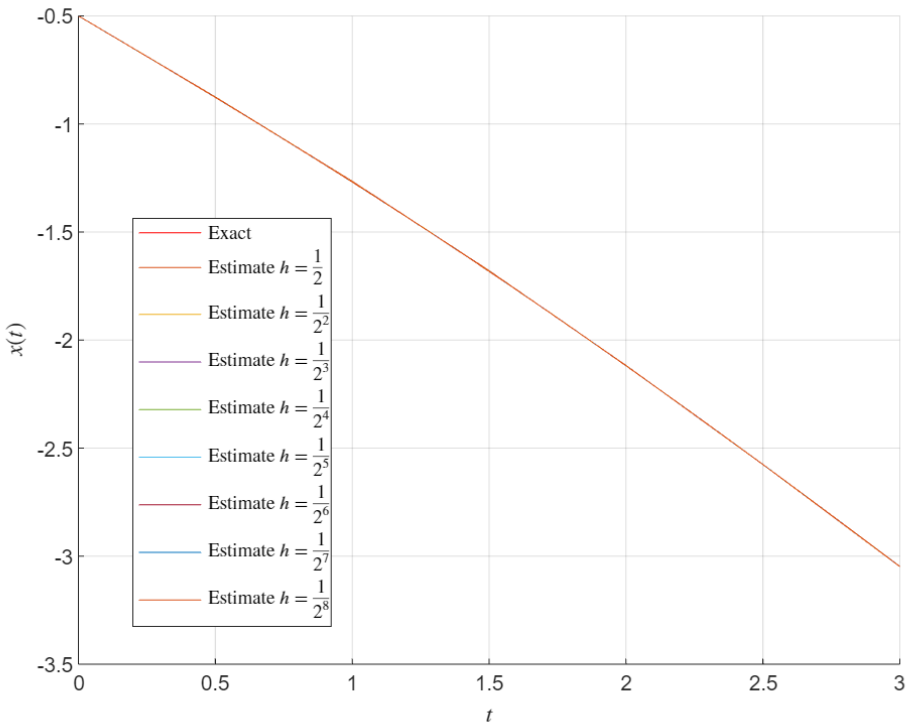

Figure 1: MATLAB result upon implementing T2 on the IVP in Eq 2 using step sizes $h_n = \frac{1}{2^n}$ for $n = \overline{1, 8}$.

Here we observe that all plots of estimates using the different step sizes h_n coincides and almost overlaps with the exact solution. Hence, we say that our estimates are quite good approximation to the exact solution. However, this can be verified analytically using the obtained average GDE. We verify the accuracy of our results by observing the average GDE $e(h_n)$ for each run and computing for the error ratio $\frac{e(h_n)}{e(h_{n-1})}$ for $n = 2, 3, \ldots, 8$. Solving for these values in MATLAB, we obtain the table presented in Figure 2.

The error ratio are

n	h	ave GDE	error ratio
1	0.5000	1.761031e-03	NaN
2	0.2500	4.248923e-04	0.241275
3	0.1250	1.043379e-04	0.245563
4	0.0625	2.587106e-05	0.247955
5	0.0312	6.440304e-06	0.248939
6	0.0156	1.606608e-06	0.249461
7	0.0078	4.012285e-07	0.249736
8	0.0039	1.002536e-07	0.249867

Figure 2: Average GDE and Error Ratio for each run of T2 on IVP in Eq. 2.

Here we notice that the average GDE $e(h_n)$ decreases as n increases or as the step size h_n decreases. From $h_1 = 0.5$ with estimates having average GDE of order 10^{-3} to that of $h_8 = 1/2^8$ with order 10^{-7}, we see a large magnitude in decrease.
This magnitude of decrease is also represented by the error ratio values. Specifically, we observe that the error ratio is approximately 0.25 or $\frac{1}{4}$. For example, since $\frac{e(h_8)}{e(h_7)} \approx 0.25$, then we say that the average GDE $e(h_8)$ is $\frac{1}{4}$ of the average GDE $e(h_7)$. This also implies that as the step size h is halved, the average GDE $e(h)$ is quartered. This indicates that the order of convergence is $\mathcal{O}(h^2)$. That is, the error of the estimate disappears, and thus our estimates approaches the exact solution, as fast as h^2 approaches 0.

Now, we implement the same process for the IVP

$$\begin{cases} \frac{dx}{dt} &= f(t,x) \\ x(0) &= -1 \end{cases} \tag{6}$$

with exact solution

$$x(t) = -t - 1$$

For this IVP, we have the marching algorithm

$$\begin{cases} w_0 &= w(0) = -1 \\ w_{i+1} &= w_i + hf(t_i, w_i) + \frac{h^2}{2}\frac{df}{dt}(t_i, w_i) \end{cases} \tag{7}$$

Implementing this in MATLAB with different step sizes $h := h_n = \frac{1}{2^n}$ for $n = 1, 2, \ldots, 8$, we have the resulting plot in Figure 3.

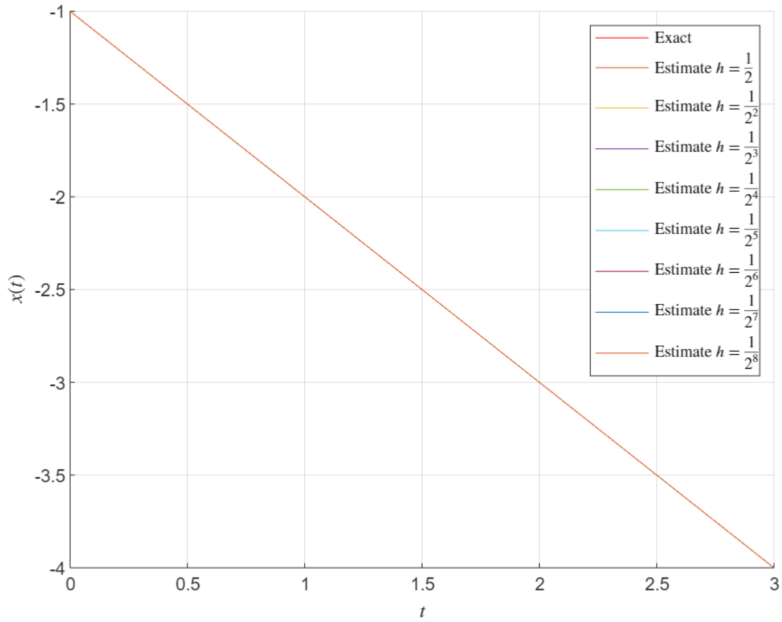

Figure 3: MATLAB result upon implementing T2 on the IVP in Eq 6 using step sizes $h_n = \frac{1}{2^n}$ for $n = \overline{1, 8}$.

From the result, we note that the plots of all estimates coincide with the plot of the exact solution. This means that the obtained estimates are relatively accurate with respect to the exact value. To verify, we also compute for the average GDE $e(h_n)$ in each step size h_n and compute for the error ratio $\frac{e(h_n)}{e(h_{n-1})}$ for $n = 2, 3, \ldots, 8$. Doing so in MATLAB yields Figure 4.

The error ratio are

n	h	ave GDE	error ratio
1	0.5000	0.000000e+00	NaN
2	0.2500	0.000000e+00	NaN
3	0.1250	0.000000e+00	NaN
4	0.0625	0.000000e+00	NaN
5	0.0312	0.000000e+00	NaN
6	0.0156	0.000000e+00	NaN
7	0.0078	0.000000e+00	NaN
8	0.0039	0.000000e+00	NaN

Figure 4: Average GDE and Error Ratio for each run of T2 on IVP in Eq. 6.

Here, we see that the average GDE for all runs is 0 implying that all estimates obtained are exactly the exact solution of the IVP. Because of this, no error ratio is obtained since no changes occurred by decreasing the step size. In such case, the defined initial value could have caused our estimates to approximate the actual solutions exactly, despite what the step size is.

Now, we solve the IVP from Equation 2 using Runge-Kutta Method of order 2 (RK2). We recall that with this method, we use the approximation

$$\begin{aligned}\frac{w_{i+1} - w_i}{h} &= \phi(f, t, x, h) \\ &= (a_1 + a_2)f(t, x) + a_2\alpha_2\frac{\partial f}{\partial t} + a_2\delta_2 f(t, x) + a_2 R_1\end{aligned}$$

where R_1 is the error term and that the constants a_1, a_2, α_2 and δ_2 satisfies

$$\begin{cases} a_1 + a_2 &= 1 \\ \alpha_2 = \delta_2 &= \frac{h}{2a_2} \end{cases}$$

Consequently, we calculate the RK2 approximates using the formula

$$\begin{cases} \widetilde{w} &= w_i + \delta_2 f(t_i, w_i) \\ w_{i+1} &= w_i + ha_1 f(t_i, w_i) + ha_2 f(t_i + \alpha_2, \widetilde{w}) \end{cases} \tag{8}$$

Now, suppose we choose to implement RK2 on the IVP in Equation 2 using $a_2 = \dfrac{1}{4}$ and $h = \dfrac{1}{8}$. This implies that

$$\begin{aligned} a_1 &= \frac{3}{4} \\ \alpha_2 = \delta_2 &= \frac{h}{2 \cdot \frac{1}{4}} = \frac{h}{\frac{1}{2}} = 2h \end{aligned}$$

Therefore, we use the formula

$$\begin{cases} \widetilde{w} &= w_i + 2hf(t_i, w_i) \\ w_{i+1} &= w_i + \frac{3h}{4} f(t_i, w_i) + \frac{h}{4} f(t_i + 2h, \widetilde{w}) \end{cases} \tag{9}$$

for our implementation. Doing so in MATLAB yields the result in Figure 5.
Here, we see that using RK2, the average GDE obtained is $e(h = \frac{1}{8}) \approx 4.51 \times 10^{-4}$. This is larger than with the one we obtained using T2 with the same step size which is $e(h_3) \approx 1.04 \times 10^{-4}$ (see Figure 2, row 3). This means that using RK2 did not improve our approximate to the exact solution. However, the obtained average GDRE using RK2 which is 2.79×10^{-4} still shows that our estimates using the method are still accurate.

i	t_i	w_i	x_i	rel error
0	0.00	-0.500000	-0.5000	0.000000e+00
1	0.12	-0.593872	-0.5938	1.371589e-04
2	0.25	-0.687985	-0.6878	2.354351e-04
3	0.38	-0.782573	-0.7823	3.062940e-04
4	0.50	-0.877853	-0.8775	3.564217e-04
5	0.62	-0.974025	-0.9736	3.901582e-04
6	0.75	-1.071261	-1.0708	4.106134e-04
7	0.88	-1.169706	-1.1692	4.202111e-04
8	1.00	-1.269476	-1.2689	4.209619e-04
9	1.12	-1.370653	-1.3701	4.145984e-04
10	1.25	-1.473293	-1.4727	4.026402e-04
11	1.38	-1.577423	-1.5768	3.864236e-04
12	1.50	-1.683043	-1.6824	3.671146e-04
13	1.62	-1.790135	-1.7895	3.457161e-04
14	1.75	-1.898660	-1.8980	3.230750e-04
15	1.88	-2.008566	-2.0080	2.998902e-04
16	2.00	-2.119789	-2.1192	2.767248e-04
17	2.12	-2.232257	-2.2317	2.540186e-04
18	2.25	-2.345894	-2.3453	2.321031e-04
19	2.38	-2.460619	-2.4601	2.112170e-04
20	2.50	-2.576352	-2.5759	1.915206e-04
21	2.62	-2.693013	-2.6925	1.731106e-04
22	2.75	-2.810525	-2.8101	1.560324e-04
23	2.88	-2.928814	-2.9284	1.402915e-04
24	3.00	-3.047809	-3.0474	1.258637e-04

AveGDE = 4.5090e-04
AveGDRE = 2.7910e-04

Figure 5: Result of implementing RK2 on the IVP in Eq. 2 using step size $h = \frac{1}{8}$.

Now, we solve the same IVP using Runge-Kutta Method of order 4 (RK4). We recall that for this method, we use the formula

$$\begin{cases} w_0 & = \alpha \\ w_{i+1} & = w_i + \frac{1}{6}\left(k_1 + 2k_2 + 2k_3 + k_4\right) \end{cases} \tag{10}$$

where

$$\begin{aligned} k_1 &= hf(t_i, w_i), \\ k_2 &= hf\left(t_i + \frac{h}{2}, w_i + \frac{k_1}{2}\right), \\ k_3 &= hf\left(t_i + \frac{h}{2}, w_i + \frac{k_2}{2}\right), \text{ and} \\ k_4 &= hf(t_i + h, w_i + k_3) \end{aligned}$$

We implement this again in MATLAB using $h = \frac{1}{8}$. For this, we obtain the result shown in Figure 6.

Relative to our results using T2 and RK2, the approximates obtained using RK4 is significantly better than the first two. This is verified by the average GDE obtained for this method which is 7.0638×10^{-8}. Moreover, the approximates are also significantly accurate considering that the obtained average GDRE is 3.6180×10^{-8} which is of order 10^{-8}.

i	t_i	w_i	x_i	rel error
0	0.00	-0.500000	-0.5000	0.000000e+00
1	0.12	-0.593791	-0.5938	6.702495e-09
2	0.25	-0.687824	-0.6878	1.181444e-08
3	0.38	-0.782333	-0.7823	1.616715e-08
4	0.50	-0.877541	-0.8775	2.019726e-08
5	0.62	-0.973645	-0.9736	2.411773e-08
6	0.75	-1.070821	-1.0708	2.800293e-08
7	0.88	-1.169215	-1.1692	3.183874e-08
8	1.00	-1.268941	-1.2689	3.555672e-08
9	1.12	-1.370085	-1.3701	3.906006e-08
10	1.25	-1.472700	-1.4727	4.224377e-08
11	1.38	-1.576813	-1.5768	4.501012e-08
12	1.50	-1.682426	-1.6824	4.727945e-08
13	1.62	-1.789517	-1.7895	4.899662e-08
14	1.75	-1.898047	-1.8980	5.013360e-08
15	1.88	-2.007964	-2.0080	5.068883e-08
16	2.00	-2.119203	-2.1192	5.068425e-08
17	2.12	-2.231691	-2.2317	5.016095e-08
18	2.25	-2.345350	-2.3453	4.917406e-08
19	2.38	-2.460099	-2.4601	4.778769e-08
20	2.50	-2.575858	-2.5759	4.607027e-08
21	2.62	-2.692547	-2.6925	4.409067e-08
22	2.75	-2.810087	-2.8101	4.191508e-08
23	2.88	-2.928403	-2.9284	3.960480e-08
24	3.00	-3.047426	-3.0474	3.721478e-08

AveGDE = 7.0638e-08
AveGDRE = 3.6180e-08

Figure 6: Result of implementing RK4 on the IVP in Eq. 2 using step size $h = \frac{1}{8}$.

Now, we analyze further the results we obtained upon implementation of RK2 and RK4. For this, we plot the graph of our estimates together with the exact solution.

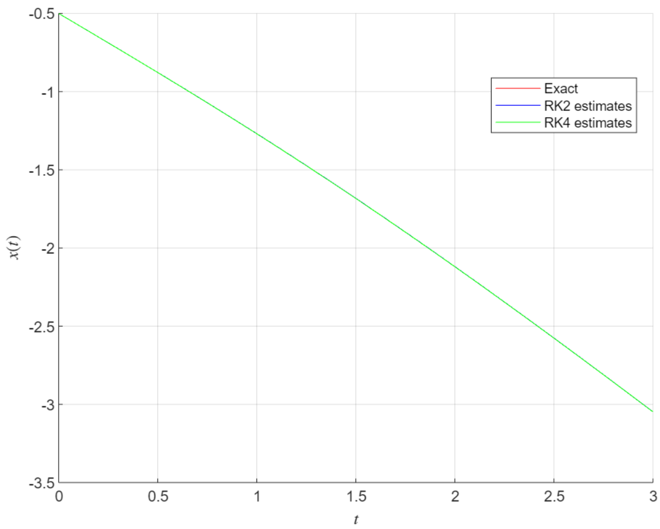

Figure 7: Plot of estimates obtained using RK2 and RK4 and exact solutions.

In Figure 7, we see that the plot of the estimates using RK2 and RK4 completely overlaps with the plot of the exact solution. This verifies our obtained average GDREs for the estimates obtained using RK2 and RK4 which are of order 10^{-4} and 10^{-8}, respectively.

We summarize the average GDE and GDRE of our estimates using T2, RK2, and RK4 in solving the IVP in Equation 2. The summary is shown in Table 1.

n	T2		RK2		RK4	
	ave GDE	ave GDRE	ave GDE	ave GDRE	ave GDE	ave GDRE
1	1.76×10^{-3}	1.29×10^{-3}	-	-	-	-
2	4.25×10^{-4}	3.24×10^{-4}	-	-	-	-
3	1.04×10^{-4}	8.11×10^{-5}	4.51×10^{-4}	2.79×10^{-4}	7.06×10^{-8}	3.62×10^{-8}
4	2.59×10^{-5}	2.03×10^{-5}	-	-	-	-
5	6.44×10^{-6}	5.06×10^{-6}	-	-	-	-
6	1.61×10^{-6}	1.26×10^{-6}	-	-	-	-
7	4.01×10^{-7}	3.16×10^{-7}	-	-	-	-
8	1.00×10^{-8}	7.90×10^{-8}	-	-	-	-

Table 1: Average GDE and GDRE of estimates obtained using T2, RK2, and RK4.

The table further verifies that for $h = \frac{1}{8}$, RK2 performed better than T2, but RK4 performed the best among the three methods in solving the IVP. We note also that it requires $h = \frac{1}{2^8}$ for the average GDE and GDRE of the estimates using T2 to be of the same order with that of the estimates using RK4. Even so, the latter is still better in terms of accuracy relative to the exact solution as seen in the average GDRE.

APPENDIX A

Screenshot of MATLAB program used for the T2 Method to solve the IVPs in Equations 2 and 6.

```
%  Taylor Method for ODE
% find solution to the IVP
% x' = f(t,x)
% x(a) = \alpha

%% PREAMBLE
clc
clear
close all

syms x t % declaration of symbolic variable(s)

%% Initializing Global Variables

% rhs function f(t,x) ----
func = -(1+t+t^2) - (2*t+1)*x - x^2;
diff_func = (-2*x - 2*t -1) * (1+func);

% endpoints -------
a = 0;
b = 3;

% initial value of x  (x(a) = \alpha) ---------
% init_value = -1/2;
init_value = -1;
```

```
for k = 1:3
    % step size --------
    n = k;
    h = 1/(2^n);
    fprintf('for h = 1/2^%d \n', n)

    % estimating function -----
    funcW = x + (h)*func + (h^2/2)*diff_func;

    % vector of t (independent variable)
    T = a:h:b;

    % exact solution
    % f = -t - 1/(exp(t)+1);
    f = -t-1;

    % actual values of x
    X = double(subs(f,T));
    %% Estimates for x_i's (W_i's)

    % placeholders
    W = zeros(length(T),1);
    rel_Error = zeros(length(T),1);
    abs_Error = zeros(length(T),1);

    % initial value
    W(1) = init_value;

    fprintf('i \t t_i \t w_i \t\t x_i \t\t rel error \n')
    for i = 1:length(W)

        if i > 1
            % new estimate
            W(i) = subs(funcW, {t, x}, [T(i-1), W(i-1)]);
        end

        % absolute errors
        abs_Error(i) = abs(W(i) - X(i));
        % relative error of w_i wrt to actual x_i
        rel_Error(i) = abs_Error(i)/abs(X(i));

        % print per iteration
        fp = [i-1, T(i), W(i), X(i), rel_Error(i)];
        fprintf('%d \t %0.2f \t %4.6f \t %4.4f \t %4.6e \n', fp)
    end
```

```
    %% Errors

    % average abs error (ave GDE)
    AveAbsErr = mean(abs_Error);

    % average rel error
    AveRelErr = mean(rel_Error);

    Err = [AveAbsErr, AveRelErr];
    fprintf('AveGDE = %0.4e \nAveGDRE = %0.4e \n\n', Err)

    %% Save Solutions and ave GDE
    soln{k} = W';
    T_pholder{k} = T';
    ave_GDE{k} = AveAbsErr';
    ave_GDRE{k} = AveRelErr';
end
```

```
%% Plotting
figure
hold on
grid on
plot(T,X, 'r')
for j = 1:length(soln)
    plot(T_pholder{j}, soln{j})
end
legend('Exact', 'Estimate $h=\frac{1}{2}$', 'Estimate $h=\frac{1}{2^2}$', ...
     'Estimate $h=\frac{1}{2^3}$', 'Estimate $h=\frac{1}{2^4}$', ...
      'Estimate $h=\frac{1}{2^5}$', 'Estimate $h=\frac{1}{2^6}$', ...
       'Estimate $h=\frac{1}{2^7}$', 'Estimate $h=\frac{1}{2^8}$', ...
       'Interpreter', 'Latex')
xlabel('$t$', 'Interpreter', 'Latex')
ylabel('$x(t)$', 'Interpreter', 'Latex')

%% Error Ratio
fprintf('The error ratio are \n')
fprintf('n \t h \t\t ave GDE \t error ratio \n')

for i = 1:length(ave_GDE)
    if i == 1
        errRatio = nan;
    else
        errRatio = ave_GDE{i}/ave_GDE{i-1};
    end

    stsize = 1/(2^i);
    Err_print = [i, stsize, ave_GDE{i}, errRatio];
    fprintf('%d \t %0.4f \t %0.6e \t %0.6f \n', Err_print)
end
```

APPENDIX B

Screenshot of MATLAB program used for the RK2 Method to solve the IVP in Equation 2.

```
% Runge-Kutta Method of order 2
% find solution to the IVP
% x' = f(t,x)
% x(a) = \alpha


%% PREAMBLE
clc
clear
close all

syms x t

%% Initializing Global Variables

% rhs function f(t,x) ----
func = -(1+t+t^2) - (2*t+1)*x - x^2;

% step size ----
h = 1/8;

% endpoints -----
a = 0;
b = 3;

% constants ---------
a2 = 1/4;
a1 = 1 - a2;
alpha2 = h/(2*a2);
delta2 = alpha2;

% initial value of x  (x(a) = \alpha) ---------
init_value = -1/2;

% vector of t (independent variable)
T = a:h:b;

% exact solution
f = -t - 1/(exp(t)+1);

% actual values of x
X = double(subs(f,T));

```

```
%% Estimates for Xis (Wis) using RK2

% placeholders
W = zeros(length(T),1);
rel_Error = zeros(length(T),1);
abs_Error = zeros(length(T),1);

W(1) = init_value;
fprintf('i \t t_i \t w_i \t\t x_i \t\t rel error \n')

for i = 1:length(W)

    if i > 1
        ftx = subs(func, {t, x}, [T(i-1), W(i-1)]);
        Wtilde = W(i-1) + delta2 * ftx;
        W(i) = W(i-1) + h*a1*ftx + h*a2*subs(func, {t, x}, [T(i-1)+alpha2, Wtilde]);
    end

    % absolute error
    abs_Error(i) = abs(W(i) - X(i));

    % relative error of w_i wrt to actual x_i
    rel_Error(i) = abs_Error(i)/abs(X(i));

    % print per iteration
    fp = [i-1, T(i), W(i), X(i), rel_Error(i)];
    fprintf('%d \t %0.2f \t %4.6f \t %4.4f \t %4.6e \n', fp)

end

%% AVERAGE GDE and GDRE
aveGDE = mean(abs_Error);
aveGDRE = mean(rel_Error);

Err = [aveGDE, aveGDRE];
fprintf('AveGDE = %0.4e \nAveGDRE = %0.4e \n\n', Err)

%% Save Estimates
RK2_result = W;
```

APPENDIX C

Screenshot of MATLAB program used for the RK4 Method to solve the IVP in Equation 2.

```matlab
% Runge-Kutta Method of order 4 (RK4)
% find solution to the IVP
% x' = f(t,x)
% x(a) = \alpha


%% PREAMBLE
% clc
clearvars -EXCEPT RK2_result
close all

syms x t

%% Initializing Global Variables

% rhs function f(t,x) ----
func = -(1+t+t^2) - (2*t+1)*x - x^2;

% step size ----
h = 1/8;

% endpoints -----
a = 0;
b = 3;

% initial value of x  (x(a) = \alpha) ---------
init_value = -1/2;

% vector of t (independent variable)
T = a:h:b;

% exact solution
f = -t - 1/(exp(t)+1);

% actual values of x
X = double(subs(f,T));
```

```matlab
%% Estimates for Xis (Wis) using RK2

% placeholders
W = zeros(length(T),1);
rel_Error = zeros(length(T),1);
abs_Error = zeros(length(T),1);

W(1) = init_value;
fprintf('i \t t_i \t w_i \t\t x_i \t\t rel error \n')

for i = 1:length(W)

    if i > 1

        % constants
        k1 = h*subs(func, {t, x}, [T(i-1), W(i-1)]);
        k2 = h*subs(func, {t, x}, [T(i-1)+(h/2), W(i-1)+(k1/2)]);
        k3 = h*subs(func, {t, x}, [T(i-1)+(h/2), W(i-1)+(k2/2)]);
        k4 = h*subs(func, {t, x}, [T(i-1)+h, W(i-1)+k3]);

        % new estimate
        W(i) = W(i-1) + (1/6)*(k1 + 2*k2 + 2*k3 + k4);
    end

    % absolute error
    abs_Error(i) = abs(W(i) - X(i));

    % relative error of w_i wrt to actual x_i
    rel_Error(i) = abs_Error(i)/abs(X(i));

    % print per iteration
    fp = [i-1, T(i), W(i), X(i), rel_Error(i)];
    fprintf('%d \t %0.2f \t %4.6f \t %4.4f \t %4.6e \n', fp)

end

```

```matlab
%% AVERAGE GDE and GDRE
aveGDE = mean(abs_Error);
aveGDRE = mean(rel_Error);

Err = [aveGDE, aveGDRE];
fprintf('AveGDE = %0.4e \nAveGDRE = %0.4e \n\n', Err)

%% Plotting (RK2 vs RK4 vs exact)
figure
hold on
grid on
plot(T, X, 'r-')
plot(T, W, 'b')
plot(T, RK2_result, 'g')
legend('Exact', 'RK2 estimates', 'RK4 estimates')
xlabel('$t$', 'Interpreter', 'Latex')
ylabel('$x(t)$', 'Interpreter', 'Latex')
hold off
```