GEBZE TEKNİK ÜNİVERSİTESİ

DEPARTMENT OF ELECTRONIC ENGINEERING

MATH214 NUMERICAL METHODS

2020 – 2021 FALL TERM

Project 4

1801022001
Ömer Sarımeşe

Preparation date	Upload date
23.12.2020	25.12.2020

I. Problem Definition and Formulation

The main purpose is finding current values of RL circuit by numerical methods. Source voltage, inductance and resistance values are given already. Current values by time can easily be found by

$$V_s = L\frac{d}{dt}i(t) + Ri(t) \tag{1}$$

Ordinary differential equation(1). For analytical solution that formula can be transform to that:

$$i = \frac{V}{R}(1 - e^{-(R/L).t}) \tag{2}$$

Equation(2) represent change on current by time variable. This formula(2) will be used to find exact values of current. To be able to apply methods, an initial value problem is required. Source voltage is $12\ V$, inductanse is $0.98\ H$, resistance is $14.2\ \Omega$. İnitial values for time and current is given respectively as $0\ s, 100\ mA$.

$$y'(t) = (-14{,}49).y(t) + 12{,}24\ \ ,\ \ 0 \le t \le 0.600\ s\ \ ,\ \ y(0) = 0.1\ A \tag{3}$$

In equation(3), V_s, L, R values already written instead of formula terms. Result of all methods will give, $f(t,y) = y'$, differential equation's(3) solution in all time values. Steps of transformation to these formulas are not required. Generally known four numerical differential method will be used. First of them is Euler Method:

$$y(t_{i+1}) = y(t_i) + h.f(t_i, y(t_i)) \tag{4}$$

$$y(t_{i+1}) = y(t_i) + \frac{h}{2}[f(t_i, y(t_i) + f(t_{i+1}, y(t_i) + h.f(t_i, y(t_i))] \tag{5}$$

Second method is Modified EulerMethod and shown in equation(5). It is looking way too much different than Euler Method. Midpoint Method is shown in equation(6) and basically gives same result as Modified Euler Method.

$$y(t_{i+1}) = y(t_i) + h.[f(t_i + \frac{h}{2}, y(t_i) + f(t_{i+1}, y(t_i)] \tag{6}$$

Last but not least, Runge-Kutte Method Order Four is given by equation(7). This method requires multiple steps to calculate differential equation's solution.

$$K_1 = h.f\big(t_i, y(t_i)\big)\,,$$

$$K_2 = h.f\left(t_i + \frac{h}{2}, y(t_i) + \frac{K_1}{2}\right),$$

$$K_3 = h.f\left(t_i + \frac{h}{2}, y(t_i) + \frac{K_2}{2}\right),$$

$$K_4 = h.f(t_{i+1}, y(t_i) + K_3)\,,$$

$$y(t_{i+1}) = y(t_i) + \frac{1}{6}(K_1 + 2K_2 + 2K_3 + K_4) \tag{7}$$

In all methods, h value is same and step size of time. Additionally, None of the methods need to time value to calculate solution because there is no time variable in differential equation(3).

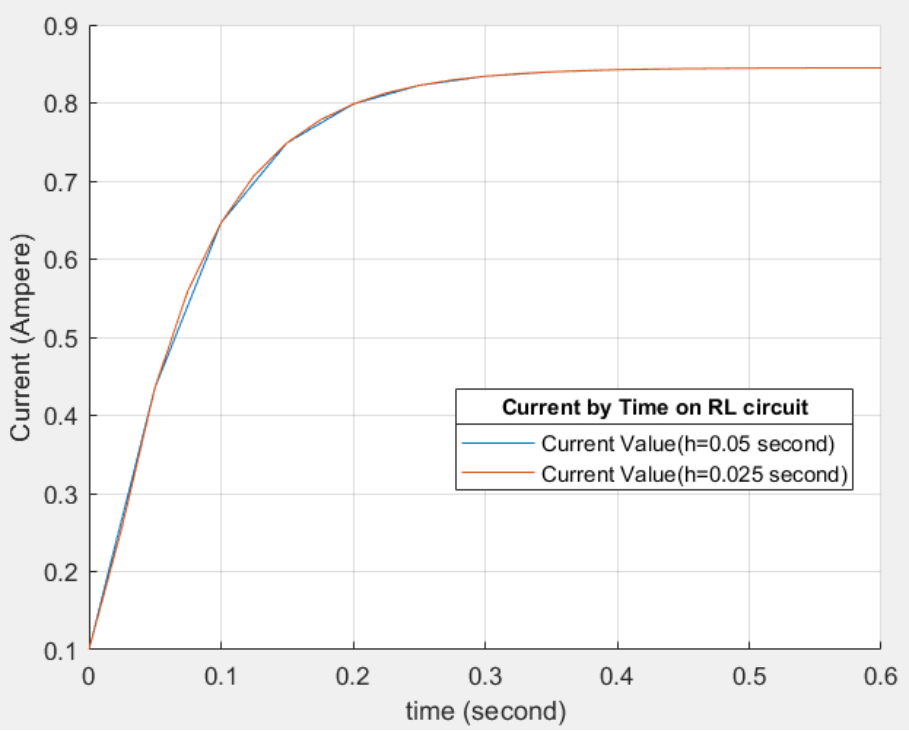
To find error bound equation(9) is used. This equation needs second derivative of y(t) and it is given in equation(8).

$$y(t) = (-0.74472).e^{(-14.49).t} + 0.84472\ ,$$
$$y''(t) = (-156.361).e^{(-14.49).t} \quad (8)$$

$$L = \left|\frac{\partial y'(t)}{\partial y}\right|\ ,\ t \in (0, 0.6)\ ,\ M = (-156.361).e^{-8.694}\ ,$$
$$|yi - wi| \leq \frac{h.M}{2L}(e^{ti.L} - 1) \quad (9)$$

Equation(8) is second derivative of y(t) and used to find M value. When finding M value, second derivative of y(t) is calculated with t equal to 0.6 and written in most simplified way.

II. Analytical Value of Current

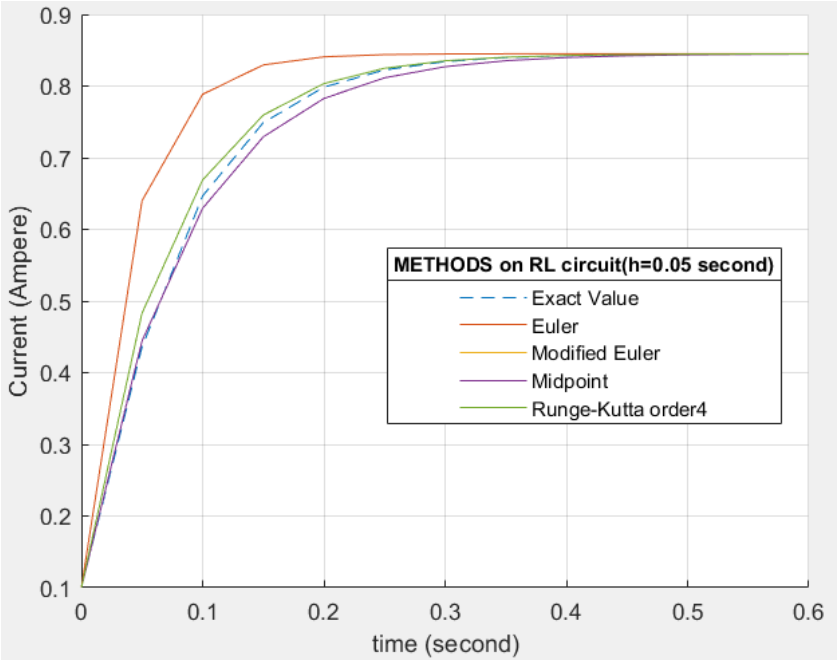


Graph-1: Current by Time on RL circuit

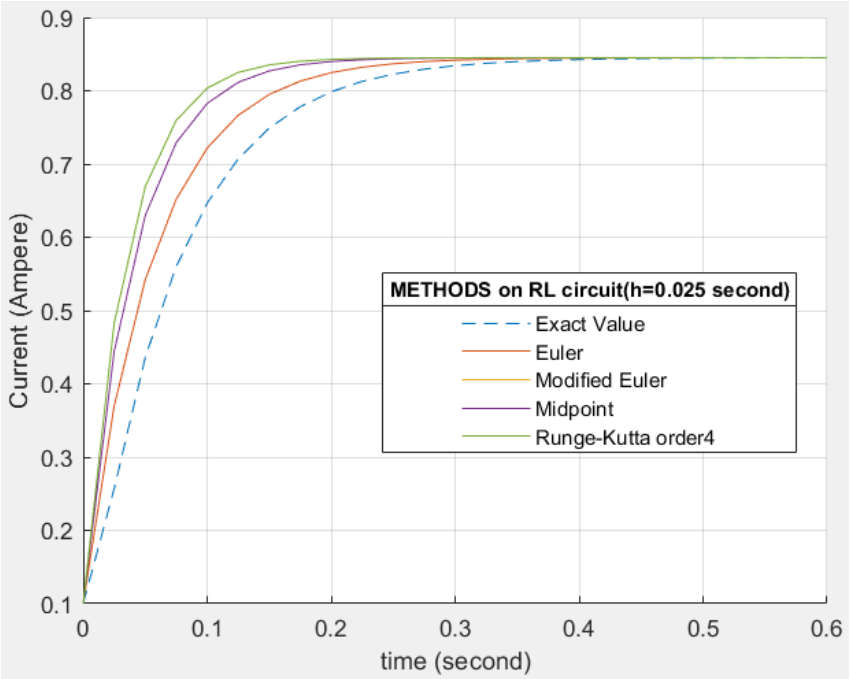
In graph-1, Current values are shown at y-axis. İnitial value is 0.1 mA. It is looking like as supposed to be like. These values found by equation(2). The current is converge to $0.8449\ A$ value.

III. Current by Methods

Modified Euler Method and Midpoint Method gave the similarly same results. For that reason, one of them are overlap the other one in graphs.



Graph-2: Methods Current Values (h=0.05)

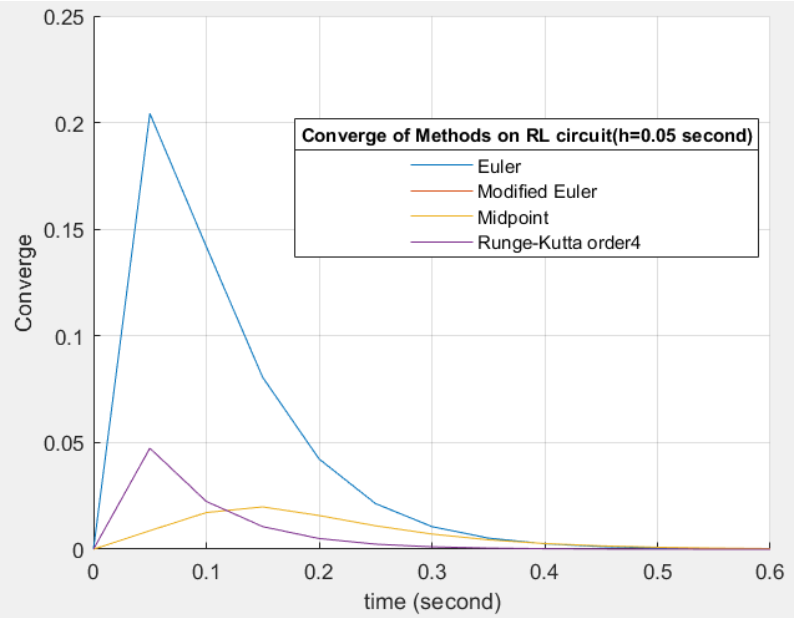


Graph-3: Methods Current Values (h=0.025)

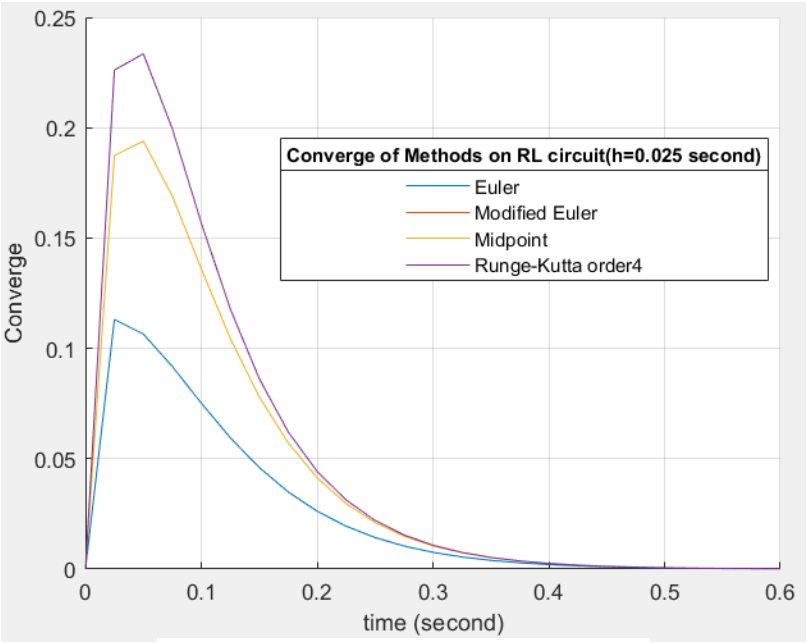
As clearly seen, when stepsize is 0.05 second, Euler method is the less accurate one but in other case when stepsize is 0.025 second, Euler method is the closest method to exact value. Time variable is not included in the differential equation. Therefore, Euler method should be more useful than Runge-kutte method when required iterations, in other words points, are many. Difference between Graph-1 and graph-2 shows us that was provided.

Generally Modified Euler method is more accurate than Euler method. In less stepsize case, that known idea is totally wrong because of lack of time variable in used differential equation.

If we look at the converge of the methods, Accuracy can seen much easily. Converge is founded by substracting exact values and methods results.



Graph-4: Converge of methods (h=0.05)

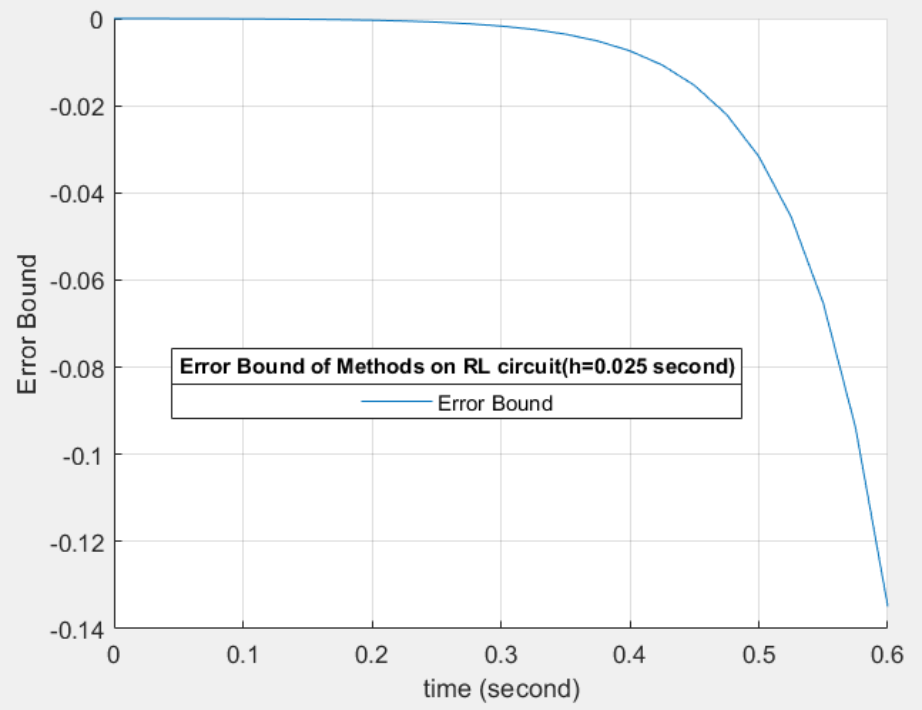


Graph-5: Converge of methods (h=0.025)

In graph-4 and graph-5, converges of methods are given. To remind again, Midpoint and Modified Euler methods are gave literally same results. There is no way to show difference between them or describe. This two cases, in different time stepsizes, methods converge order is exact opposite. Graph-4 shows converges that should be. Runge-kutta method is better than euler because of truncation error is less. Euler is way less accurate than Midpoint and Modified Euler methods because it is first order convergent method. Interestingly,Euler method is be the most convergent in other case when time stepsize is 0.025 second.

Basically, Higher order methods is tend to be more accurate when time stepsize is small but there is no term of time in the differential equation. That can be the reason of mistakes.

IV. Error Bound



Graph-6: Error Bound

In graph-6, Error bound results are shown. Line is moving away rapidly from zero at the ending due to exponantial equation. Error bound is negative so it causes overestimation. Again it is because of lack of time variable in differential equation.

V. Result and analysis of Methods

It is possible to make inferences from relations between exact value and methods' results. Firstly, Midpoint and Modified Euler methods are given exactly same solution. All the examined data tell us, there can be problem when some of the terms in formula is not used. Because of the lack of time variable in differential equation, causes a little wrong results. Runge-kutte order four is clearly better than Euler because of his higher order. Runge-kutta order four is slightly accurate than Modified Euler because it is actually to be considered as order five method. Euler Method is worst one based on his first order feature.

Difference time stepsizes makes things a little confusing because Euler method was the most convergent one in 0.025 second time stepsize. Other methods are comparably close to each other. Small differences are difficulty to interpret because of unused time values. Error bound says solutions founded by methods are overestimated. If we do not look all of these comparison between methods, Results are pretty close to exact value.

VI. Explanation of Matlab Code

Voltage, Inductance, Resistance values are respectively set to V, L, R variables. Lines between 7-19, Code finding analytical solution. T1 is array for time which stepsize is 0.05, T2 is an array for time at stepsize 0.025. sonuc1, sonuc2 arrays store exact values. After declared and set this variables, İnitial result values are set to 0.1. for loops are calculating current value for each time step and store it in certain arrays. Then program plotting the current data.

Between lines 29-91 program executes numerical solutions. All sections are in same logic. Firstly, an array is creating and set all elements zero. Then loop start as second time point and going to all the way last time step. Inside of the loop, There is nothing than formulas. After all of them, Results of methods are plotting as graph.

Between lines 117-133 is about to calculating error bound. An empty array is declared same as before sections. After that, all of the necessary assignments doing. Then, program calculates M value. Loop calculates all error bound values and assign them in an array. After, Program is plotting error bound and stop working successfully.

MATLAB CODE

```
clear all;
close all;
format long;
V=12;
R=14.2;
L=0.98;
%------------analytical solution---------------
T1=0:0.05:0.6;
T2=0:0.025:0.6;
```

```
sonuc1=zeros(0,length(T1)+1);
sonuc2=zeros(0,length(T2)+1);
sonuc1(1)=0.1;
sonuc2(1)=0.1;
for i=2:1:length(T1)
    sonuc1(i)=(V/R)*(1-(exp(-1*(R/L)*T1(i))));
end
for i=2:1:length(T2)
    sonuc2(i)=(V/R)*(1-(exp(-1*(R/L)*T2(i))));
end
figure
grid on;
hold on;
plot(T1,sonuc1);
plot(T2,sonuc2);
ylabel('Current (Ampere)');
xlabel('time (second)');
a=legend('Current Value(h=0.05 second)','Current
Value(h=0.025 second)');
title(a,'Current by Time on RL circuit');
%--------------Euler-h=0.05--------------------------
Euler1=zeros(0,length(T1));
Euler1(1)=0.1;
for i=2:1:length(T1)
    Euler1(i)=Euler1(i-1)+(0.05)*(((-14.4898)*Euler1(i-
1))+12.2449);
end
%--------------Euler-h=0.025-------------------------
Euler2=zeros(0,length(T2));
Euler2(1)=0.1;
for i=2:1:length(T2)
    Euler2(i)=Euler2(i-1)+(0.025)*(((-14.4898)*Euler2(i-
1))+12.2449);
end
%-----------Modified-Euler-h=0.05--------------------
M_Euler1=zeros(0,length(T1));
M_Euler1(1)=0.1;
for i=2:1:length(T1)
    M_Euler1(i)=M_Euler1(i-1)+(0.05/2)*((((-
14.4898)*M_Euler1(i-1))+12.2449)+(((-14.4898)*(M_Euler1(i-
1)+((0.05)*(((-14.4898)*M_Euler1(i-
1))+12.2449))))+12.2449));
end
%-----------Modified-Euler-h=0.025-------------------
M_Euler2=zeros(0,length(T2));
M_Euler2(1)=0.1;
for i=2:1:length(T2)
```

```
    M_Euler2(i)=M_Euler2(i-1)+(0.05/2)*((((-
14.4898)*M_Euler2(i-1))+12.2449)+(((-14.4898)*(M_Euler2(i-
1)+((0.05)*(((-14.4898)*M_Euler2(i-
1))+12.2449))))+12.2449));
end
%----------Midpoint-h=0.05-----------------------------
Midpoint1=zeros(0,length(T1));
Midpoint1(1)=0.1;
for i=2:1:length(T1)
    Midpoint1(i)=Midpoint1(i-1)+(0.05)*(((-
14.4898)*(Midpoint1(i-1)+((0.05/2)*(((-
14.4898)*Midpoint1(i-1))+12.2449))))+12.2449);
end
%----------Midpoint-h=0.025----------------------------
Midpoint2=zeros(0,length(T2));
Midpoint2(1)=0.1;
for i=2:1:length(T2)
    Midpoint2(i)=Midpoint2(i-1)+(0.05)*(((-
14.4898)*(Midpoint2(i-1)+((0.05/2)*(((-
14.4898)*Midpoint2(i-1))+12.2449))))+12.2449);
end
%--------Runge-Kutte-Order4-h=0.05---------------------
RKO1=zeros(0,length(T1));
RKO1(1)=0.1;
for i=2:1:length(T1)
    k1=(0.05)*(((-14.4898)*RKO1(i-1))+12.2449);

    k2=(0.05)*(((-14.4898)*(RKO1(i-1)+(k1/2)))+12.2449);

    k3=(0.05)*(((-14.4898)*(RKO1(i-1)+(k2/2)))+12.2449);

    k4=(0.05)*(((-14.4898)*(RKO1(i-1)+(k3)))+12.2449);
    RKO1(i)=RKO1(i-1)+(1/6)*(k1+2*k2+2*k3+k4);
end
%--------Runge-Kutte-Order4-h=0.025--------------------
RKO2=zeros(0,length(T2));
RKO2(1)=0.1;
for i=2:1:length(T2)
    k1=(0.05)*(((-14.4898)*RKO2(i-1))+12.2449);

    k2=(0.05)*(((-14.4898)*(RKO2(i-1)+(k1/2)))+12.2449);

    k3=(0.05)*(((-14.4898)*(RKO2(i-1)+(k2/2)))+12.2449);

    k4=(0.05)*(((-14.4898)*(RKO2(i-1)+(k3)))+12.2449);
    RKO2(i)=RKO2(i-1)+(1/6)*(k1+2*k2+2*k3+k4);
end
%---------------methods-graphs--------------------
```

```
figure
hold on;
grid on;
plot(T1,sonuc1,'--');
plot(T1,Euler1);
plot(T1,M_Euler1);
plot(T1,Midpoint1);
plot(T1,RKO1);
ylabel('Current (Ampere)');
xlabel('time (second)');
a=legend('Exact Value','Euler','Modified
Euler','Midpoint','Runge-Kutta order4');
title(a,'METHODS on RL circuit(h=0.05 second)');

figure
hold on;
grid on;
plot(T2,sonuc2,'--');
plot(T2,Euler2);
plot(T2,M_Euler2);
plot(T2,Midpoint2);
plot(T2,RKO2);
ylabel('Current (Ampere)');
xlabel('time (second)');
a=legend('Exact Value','Euler','Modified
Euler','Midpoint','Runge-Kutta order4');
title(a,'METHODS on RL circuit(h=0.025 second)');
%---------bound error-------------------------------------
------------
B_Error=zeros(0,length(T2));
for i=1:1:length(T2)
    h2=0.025;
    Lipt=14.49;
    M=(-156.361)*exp(-8.694);
    B_Error(i)=((h2*M)/(2*Lipt))*((exp(T2(i)*Lipt))-1);
end
figure
hold on;
grid on;
display(B_Error);
plot(T2,B_Error);
ylabel('Error Bound');
xlabel('time (second)');
a=legend('Error Bound');
title(a,'Error Bound of Methods on RL circuit(h=0.025
second)');
```