Prepared by
Muhammet Asim Uyanik

In this project, the value of $\mathcal{E}(t)$ in figure 1 will be calculated with different methods and the results will be compared. While doing this calculation;

$$\mathcal{E}(t) = L\frac{d}{dt}i(t) + Ri(t)$$

we will use the formula. It is given to us as L = 0.98 H and R = 14.2 Ω. But in order to use the formula, we need to find the derivative of the current in addition to these. We will use 3 methods to find the derivative of the current. These methods are Forward Difference, Backward Difference and Three Point Midpoint Formula. Since we made these calculations based on four different Δt values, we will compare the results we found.

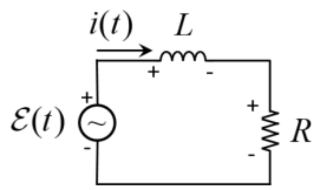


Figure 1: A RL circuit.

First of all, at the very beginning of the code, we will upload the current1.dat, current2.dat, current3.dat and current4.dat files to the matlab program. As shown in Figure 2, we are loading these files with the "load" command.

```
load current1.dat;
load current2.dat;
load current3.dat;
load current4.dat;
```

(Figure 2)

Later, certain definitions are made in the code. These;

```
L=0.98;
R=14.2;
t1=75*10^-3;
t2=50*10^-3;
t3=25*10^-3;
t4=10*10^-3;
```

Here

t1= measurement range in current1.dat

t2= measurement range in current2.dat

t3= measurement range in current3.dat

t4= measurement range in current4.dat

L=inductance

R=Resistance

A: Forward Difference Formula

In this section, we will calculate Ɛ (t) using the time and current values in the .dat files given to us. But we need the derivative of the current to do the calculations. To get this derivative;

$$\frac{di(t)}{dt} = \frac{\mathrm{f(i\ +\ 1)\ -\ f(i)}}{\Delta \mathrm{t}}$$

we will use the formula. In this method of derivation, the derivative is found by subtracting the value in this step from the value in the next step as seen in the formula.
If the calculations are made for the data in the current1.dat file first, the time interval measured in this data is given as 75 ms. So Δt = t1 = 0.075. Using this and the formula in figure 2, the calculations were made in the code snippet in figure 3.

```
for i=1:8
f=(current1(i+1,2)-current1(i,2))/t1;
gf1(i)=f;
Ef1(i)=L*f+R*current1(i,2);
end
```

(Figure 3)

There are 9 current values in the current1.dat file given to us. Since we are using the Forward Difference Form, there will be 8 steps. So the code goes into a loop that will be repeated 8 times. It then calculates the derivative of the current and assigns it to an array named gf1. Then he calculates Ɛ(t) using the derivative of the calculated current and assigns it to an array named Ef1. The calculated values for Current1.dat are shown in table 1.

STEP	TIME(S)	CURRENT(A)	DERIVATE OF CURRENT	$\mathcal{E}(t)$
1	0.000	0.1	6.5833	7.8716
2	0.075	0.593746	2.2206	10.6074
3	0.15	0.760295	0.7491	11.5303
4	0.225	0.816474	0.2527	11.8416
5	0.3	0.835424	0.0852	11.9466
6	0.375	0.841817	0.0287	11.9820
7	0.45	0.843973	0.0097	11.9939
8	0.525	0.844700	0.0033	11.9979

(Table 1)

The same operations in current2.dat, current3.dat and current4.dat are made in the code snippets shown in figure 4, figure 5 and figure 6, but the table cannot be made because there are too many values. Instead, graph 1 and graph 2 are analyzed.

```
for i=1:12
f=(current2(i+1,2)-current2(i,2))/t2;
Ef2(i)=L*f+R*current2(i,2);
gf2(i)=f;
end
```

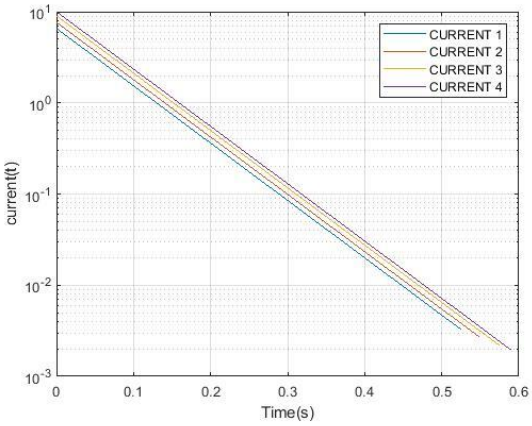
(Figure 4: Current 2)

```
for i=1:24
f=(current3(i+1,2)-current3(i,2))/t3;
Ef3(i)=L*f+R*current3(i,2);
gf3(i)=f;
end
```

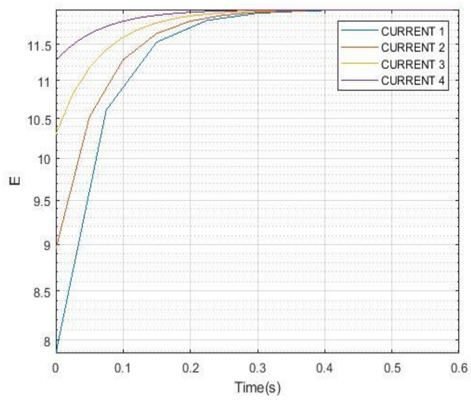
(Figure 5: Current 3)

```
for i=1:60
f=(current4(i+1,2)-current4(i,2))/t4;
Ef4(i)=L*f+R*current4(i,2);
gf4(i)=f;
end
```

(Figure 6: Current 4)



(Graph 1: Derivative of i Values With Forward Difference Formula)



(Graph 2: Ɛ(t) Values With Forward Difference Formula)

Graph 1 shows the variation of derivatives of current values over time, and graph 2 shows the change of Ɛ (t) over time.

When Graph 1 is examined, the change in current values decreases since the measurement range from current 1 to current 4 is constantly narrowing. Therefore, the derivative values decrease linearly.

When Graph 2 is examined, since the measurement range is constantly narrowed from current 1 to current 4, the values we reach are closer to the truth.

When the last values obtained in the calculations are examined;

current 1 (75 ms) = 11.9979

current 2 (50 ms) = 11.9989

current 3 (25 ms) = 11.9996

current 4 (10 ms) = 11.9999

As you can see, since the approached value is 12, we obtained the closest value to 12 at current 4, which is the smallest measurement range. In short, the more we measure the more we keep the measurement interval short, the closer we get to the real value.

B: Backward Difference Formula

In this method of derivation, the derivative is found by subtracting the value in the previous step from the value in this step, which is also seen in the formula.

$$\frac{di(t)}{dt} = \frac{\mathrm{f(i)} - \mathrm{f(i-1)}}{\Delta \mathrm{t}}$$

If we first make the calculations for the data in the current1.dat file, the time interval measured in this data is given as 75 ms. So Δt = t1 = 0.075. Using this and the formula in figure 2, calculations are made in the code snippet in figure 7.

```
for i=2:9
f=(current1(i,2)-current1(i-1,2))/t1;
gb1(i-1)=f;
Eb1(i-1)=L*f+R*current1(i,2);
end
```

(Figure 7)

As seen in this code snippet, first of all the program enters a loop. Then he assigns the derivative of the current inside the loop to an array named gb1. Then, using the derivative of the calculated current, it calculates Ɛ (t) and assigns it to an array named Ef2. These calculated values for Current1.dat are shown in table 2.

STEP	TIME(S)	CURRENT(A)	DERIVATE OF CURRENT	$\mathcal{E}(t)$
1	0	0.1	0	0
2	0.075	0.593746	6.583282	14.882812
3	0.15	0.760295	2.220647	12.972419
4	0.22	0.816474	0.7490601	12.328013
5	0.3	0.835424	0.2526701	12.110644
6	0.375	0.841817	0.08522969	12.037322
7	0.45	0.843973	0.02874935	12.012589
8	0.525	0.844700	0.009697620	12.004247
9	0.60	0.844946	0.003271164	12.001432

(Table 2)

The same operations for current2.dat, current3.dat and current4.dat are done in the code snippets in figure 8, figure 9 and figure 10. But because there are too many values, the table cannot be made. Instead, graph 3 and graph 4 are examined.

```
for i=2:12
f=(current2(i,2)-current2(i-1,2))/t2;
Eb2(i-1)=L*f+R*current2(i,2);
gb2(i-1)=f;
end
```

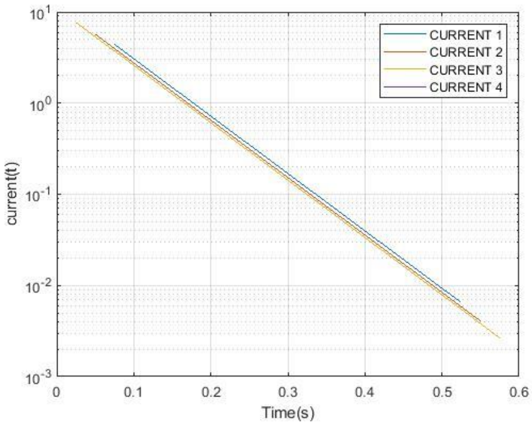
(Figure 8: Current 2)

```
for i=2:24
f=(current3(i,2)-current3(i-1,2))/t3;
Eb3(i-1)=L*f+R*current3(i,2);
gb3(i-1)=f;
end
```

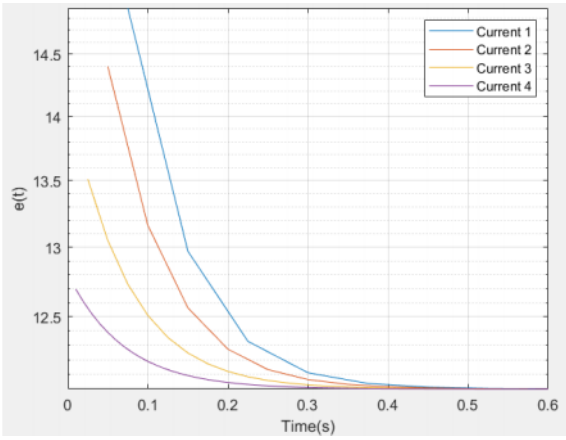
(Figure 9: Current 3)

```
for i=2:60
f=(current4(i,2)-current4(i-1,2))/t4;
Eb4(i-1)=L*f+R*current4(i,2);
gb4(i-1)=f;
end
```

(Figure 10: Current 4)



(Graph 3: Derivative of i Values With Backward Difference Formula)



(Graph 4: Ɛ(t) Values With Backward Difference Formula)

Graph 1 shows the variation of derivatives of current values over time, and graph 2 shows the change of Ɛ(t) over time.

When Graph 1 is examined, the change in current values decreases since the measurement range from current 1 to current 4 is constantly narrowing. Therefore, the derivative values decrease linearly.

When Graph 2 is examined, since the measurement range is constantly narrowed from current 1 to current 4, the values we reach are closer to the truth.

current 1 (75 ms) = 12.0014

current 2 (50 ms) = 12.0017

current 3 (25 ms) = 12.0005

current 4 (10 ms) = 12.0002

As you can see, since the approached value is 12, we obtained the closest value to 12 at current 4, which is the smallest measurement range.

In short, the more we measure the more we keep the measurement interval short, the closer we get to the real value.

C: Three Point Midpoint Formula

In this formula, the derivative is calculated by taking the difference of the value in the previous step from the value in the next step.

$$\frac{di(t)}{dt} = \frac{\mathrm{f(i + 1) - f(i - 1)}}{2\Delta \mathrm{t}}$$

If we first make the calculations for the data in the current1.dat file, the time interval measured in this data is given as 75 ms. So t = t1 = 0.075. Using this and the formula in figure 2, the calculations are made in the code snippet in figure 11.

```
for i=2:8
f=(current1(i+1,2)-current1(i-1,2))/(2*t1);
gm1(i)=f;
Em1(i)=L*f+R*current1(i,2);
end
```

(Figure 11)

As seen in this code snippet, the program first goes into a loop. Inside the loop, it first calculates the derivative of the current and then assigns it to an array with the name gm1. Then it calculates Ɛ (t) using this current value and assigns it to an array named Em1. The calculated values for Current1.dat are shown in table 3.

STEP	TIME(S)	CURRENT(A)	DERIVATE OF CURRENT	$\mathcal{E}(t)$
1	0	0.100	0	0
2	0.075	0.593746	4.401965	12.745121
3	0.15	0.760295	1.484854	12.251341
4	0.225	0.816474	0.5008651	12.084781
5	0.3	0.835424	0.1689499	12.028598
6	0.375	0.841817	0.05698952	12.009647
7	0.45	0.843973	0.01922349	12.003254
8	0.525	0.844700	0.006484392	12.001098

(Table 3)

The same operations for current2.dat, current3.dat and current4.dat are done in the code snippets in figure 12, figure 13 and figure 14. But because there are too many values, the table cannot be made. Instead, graph 5 and graph 4 are examined.

```
for i=2:12
f=(current2(i+1,2)-current2(i-1,2))/(2*t2);
Em2(i)=L*f+R*current2(i,2);
gm2(i)=f;
end
```

(Figure 12: Current 2)

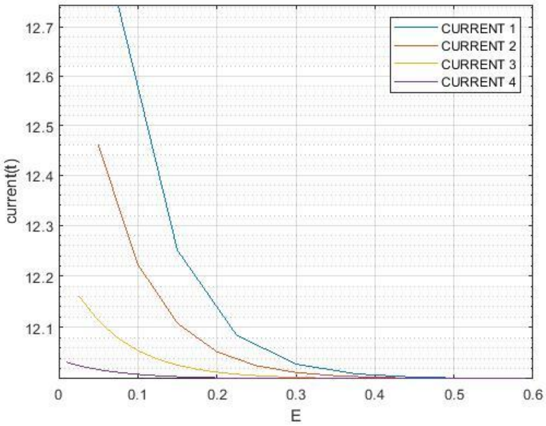
```
for i=2:24
f=(current3(i+1,2)-current3(i-1,2))/(2*t3);
Em3(i)=L*f+R*current3(i,2);
gm3(i)=f;
end
```

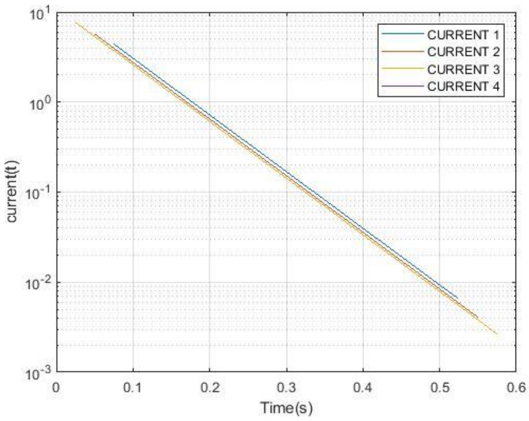
(Figure 13: Current 3)

```
for i=2:60
f=(current4(i+1,2)-current4(i-1,2))/(2*t4);
Em4(i)=L*f+R*current4(i,2);
gm4(i)=0;

end
```

(Figure 14: Current 4)





(Graph 5: Derivative of i Values With Backward Difference Formula) **(Graph 6: Ɛ(t) Values With Backward Difference Formula)**

Graph 1 shows the variation of derivatives of current values over time, and graph 2 shows the change of Ɛ(t) over time.

When Graph 1 is examined, the change in current values decreases since the measurement range from current 1 to current 4 is constantly narrowing. Therefore, the derivative values decrease linearly.

When Graph 2 is examined, since the measurement range is constantly narrowed from current 1 to current 4, the values we reach are closer to the truth.

current 1 (75 ms) = 12.0011

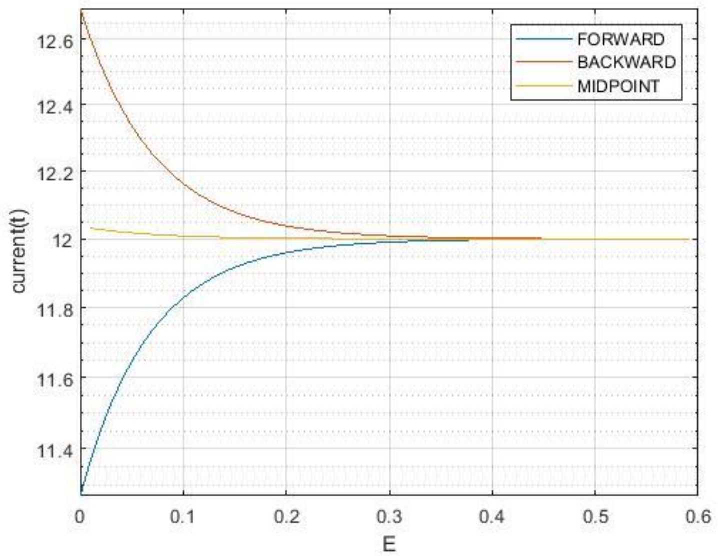
current 2 (50 ms) = 12.0003

current 3 (25 ms) = 12.0001

current 4 (10 ms) = 12.0000

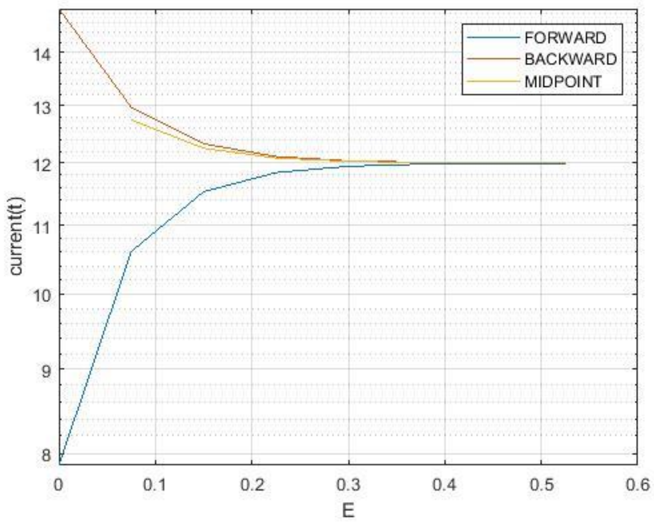
As can be seen, since the approached value is 12, the closest value to 12, even the direct 12 value, was obtained in current 4, which is the smallest measurement range.

In short, if we keep the measurement interval short and measure as much as we can, the closer we get to the real value.



(Graph 7)

When these calculations and graphs were examined, it was Three Point Midpoint Formula that reached the most accurate result. As seen in the graph 7 for Current 4, the formula that approached the real value 12 with the least error was the Three Point Midpoint Formula. Another factor is the measurement range. When we draw the same graph for current 1 as seen in graph 8, the actual value of 12 cannot be approached as much as current 4. This is because of the measuring ranges. The smaller the measuring range, the less error rate.



(Graph 8)

PROGRAM CODES:

```
clear;
clear all;
load current1.dat;
load current2.dat;
load current3.dat;
load current4.dat;

L=0.98;
```

```
R=14.2;
t1=75*10^-3;
t2=50*10^-3;
t3=25*10^-3;
t4=10*10^-3;
...for current1...
...(a)Forward-difference formula
for i=1:8
f=(current1(i+1,2)-current1(i,2))/t1;
gf1(i)=f;
Ef1(i)=L*f+R*current1(i,2);
end

...(b)Backward difference formula
for i=2:9
f=(current1(i,2)-current1(i-1,2))/t1;
gb1(i-1)=f;
Eb1(i-1)=L*f+R*current1(i,2);
end

...(c)Midpoint formula
for i=2:8
f=(current1(i+1,2)-current1(i-1,2))/(2*t1);
gm1(i)=f;
Em1(i)=L*f+R*current1(i,2);
end

...for current2...
...(a)Forward-difference formula
for i=1:12
f=(current2(i+1,2)-current2(i,2))/t2;
Ef2(i)=L*f+R*current2(i,2);
gf2(i)=f;
end

...(b)Backward difference formula
for i=2:12
f=(current2(i,2)-current2(i-1,2))/t2;
Eb2(i-1)=L*f+R*current2(i,2);
gb2(i-1)=f;
end


...(c)Midpoint formula
for i=2:12
f=(current2(i+1,2)-current2(i-1,2))/(2*t2);
Em2(i)=L*f+R*current2(i,2);
gm2(i)=f;
end

...for current3...
...(a)Forward-difference formula
for i=1:24
f=(current3(i+1,2)-current3(i,2))/t3;
Ef3(i)=L*f+R*current3(i,2);
gf3(i)=f;
end

...(b)Backward difference formula
for i=2:24
```

```
f=(current3(i,2)-current3(i-1,2))/t3;
Eb3(i-1)=L*f+R*current3(i,2);
gb3(i-1)=f;
end


...(c)Midpoint formula
for i=2:24
f=(current3(i+1,2)-current3(i-1,2))/(2*t3);
Em3(i)=L*f+R*current3(i,2);
gm3(i)=f;
end

...for current4...
...(a)Forward-difference formula
for i=1:60
f=(current4(i+1,2)-current4(i,2))/t4;
Ef4(i)=L*f+R*current4(i,2);
gf4(i)=f;
end

...(b)Backward difference formula
for i=2:61
f=(current4(i,2)-current4(i-1,2))/t4;
Eb4(i-1)=L*f+R*current4(i,2);
gb4(i-1)=f;
end

...(c)Midpoint formula
for i=2:60
f=(current4(i+1,2)-current4(i-1,2))/(2*t4);
Em4(i)=L*f+R*current4(i,2);
gm4(i)=0;
end


 time1=0:0.075:0.525;
 time2=0:0.050:0.55;
 time3=0:0.025:0.575;
 time4=0:0.01:0.59;
 ...ÇİZDİRMEK İSTEDİĞİNİZ GRAFİĞİN YORUMUNU KALDIRIN

%  ...FORWARD
%
%  semilogy(time1,gf1,time2,gf2,time3,gf3,time4,gf4);
%  xlabel('Time(s)')
%  ylabel('current(t)')
%  legend('CURRENT 1','CURRENT 2','CURRENT 3','CURRENT 4')
%  grid on
%
% ...BACKWARD
%  semilogy(time1,gm1,time2,gm2,time3,gm3,time4,gm4);
%  xlabel('Time(s)')
%  ylabel('current(t)')
%  legend('CURRENT 1','CURRENT 2','CURRENT 3','CURRENT 4')
%  grid on
%
%  ...MIDPOINT
%  semilogy(time1,gm1,time2,gm2,time3,gm3,time4,gm4);
%  xlabel('Time(s)')
```

```matlab
%  ylabel('current(t)')
%  legend('CURRENT 1','CURRENT 2','CURRENT 3','CURRENT 4')
%  grid on
%
%  ...E İÇİN GRAFİKLER
%  ...FORWARD
%
%  semilogy(time1,Ef1,time2,Ef2,time3,Ef3,time4,Ef4);
%  xlabel('Time(s)')
%  ylabel('E')
%  legend('CURRENT 1','CURRENT 2','CURRENT 3','CURRENT 4')
%  grid on
%
% ...BACKWARD
%  semilogy(time1,Eb1,time2,Eb2,time3,Eb3,time4,Eb4);
%  xlabel('E)')
%  ylabel('current(t)')
%  legend('CURRENT 1','CURRENT 2','CURRENT 3','CURRENT 4')
%  grid on
%
%  ...MIDPOINT
%  semilogy(time1,Em1,time2,Em2,time3,Em3,time4,Em4);
%  xlabel('E')
%  ylabel('current(t)')
%  legend('CURRENT 1','CURRENT 2','CURRENT 3','CURRENT 4')
%  grid on
%
%
 semilogy(time4,Ef4,time4,Eb4,time4,Em4);
 xlabel('E')
 ylabel('current(t)')
 legend('FORWARD','BACKWARD','MIDPOINT')
 grid on

%  semilogy(time1,Ef1,time1,Eb1,time1,Em1);
%  xlabel('E')
%  ylabel('current(t)')
%  legend('FORWARD','BACKWARD','MIDPOINT')
%  grid on
```