Differential Equation Model.

Series LR circuit with time dependent imposed voltage is

\[ LI' + RI = V = V_0 \sin(\omega t). \]

L = inductance,

I = current,

R = resistance and

V = imposed voltage.
Method of Solution.

Use variable step sizes.

We would like to adjust the step size, $dt = h$, so that

$$\text{error} = u^{k+1} - u((k+1)h) \text{ is “small.”}$$

In general we do NOT know $u((k+1)h)$….this is the reason for using numerical methods.

Therefore, we do not know the error!
One way around this is to replace the exact solution by a higher order numerical solution.

For example, in order to estimate the error for Euler’s method, approximate the exact solution by the Euler-trapezoid method:

\[ |U^{k+1} - u(t^{k+h})| \leq Bh^3 \]

\[ |u^{k+1} - u(t^{k+h})| \leq Ah^2. \]

In our calculations we will know both \( u^{k+1} \) and \( U^{k+1} \), but not \( u(t^{k+h}) \).

\[ u^{k+1} - U^{k+1} = u^{k+1} - u(t^{k+h}) + u(t^{k+h}) - U^{k+1}. \]

\[ |u^{k+1} - U^{k+1}| \leq |u^{k+1} - u(t^{k+h})| + |u(t^{k+h}) - U^{k+1}| \]

\[ \leq Ah^2 + Bh^3 \]

\[ \leq Ah^2 (1 + B/A h). \]
B/A h is the error made by replacing $u(t^k+h)$ with $U^{k+1}$.

Thus, we may approximate $A$ by

$$A = \frac{|u^{k+1} - U^{k+1}|}{h^2}.$$

Furthermore, the local error for the lower order scheme can be approximated by

$$|u^{k+1} - u(t^k+h)| \approx A h^2 = \left( \frac{|u^{k+1} - U^{k+1}|}{h^2} \right) h^2.$$

Let tol (tolerance) be a positive real number for which we want the error to be less than. We should choose $h$ so that

$$A h^2 \leq \text{tol}.$$
Matlab Implementation.

Euler:

$$\text{newy} = Y + hf(t,Y)$$

Improved Euler:

$$\text{newY} = Y + \frac{h}{2} \left( f(t,Y) + f(t+h,\text{newy}) \right)$$

Approximate Truncation Error:

$$\frac{\text{newY} - \text{newy}}{h} = Y + \frac{h}{2} \left( f(t,Y) + f(t+h,\text{newy}) \right) - (Y + hf(t,Y)) \newline \newline = \frac{h}{2} \left( -f(t,Y) + f(t+h,\text{newy}) \right)$$
Use the function file flr.m and the m-file eulerv.m

\[
\text{function flr = flr(t,y)} \\
\quad \text{flr} = 2*(70 + 30*\sin(\pi*t) - y);
\]

% your name, your student number, lesson number

clear;
y(1) = 200;
tol = 5;
h = .01;
t(1) = 0.;
k = 1;
while \( t(k)<20. \) *(k<2000)
    newt = t(k) + h;
    oldfff = flr(t(k),y(k));
    newy = y(k) + h*oldfff;
    terr = abs(.5*(-oldfff + flr(newt,newy)));
    if terr<tol*1.1
        t(k+1) = newt;
        y(k+1) = newy;
        k = k+1;
    end
scale = tol/terr;
if scale<.1
    h = .1*h;
end
if scale>10.
    h = 10.*h;
end
if ((scale>=.1)*(scale<=10.))
    h = scale*h;
end
steps = size(t,2)
dt = diff(t);
plot(t(1:steps-1),y(1:steps-1),t(1:steps-1),300*dt)
title('your name, your student number, lesson number')
xlabel('time')
ylabel('current and time step size')
If tol gets small, then the time step will get smaller and more steps will be required to get to the final time. For example,

Tol = 10 requires 150 time steps,

Tol = 5 requires 270 time steps and

Tol = 1 requires 1134 time steps!