Numerical Analysis MTH614

Spring 2012, Korea University

Initial value problem

Suppose that $D = \{(y,t) | a \le t \le b, -\infty < y < \infty\}$ and the $f(y,t)$ is continuous on D. If f satisfies a Lipschitz condition on D in the variable y then the initial value problem

$$y'(t) = f(t,y),\ a \le t \le b,\ y(a) = \alpha. \tag{1}$$

Has a unique solution $y(t)$ for $a \le t \le b$.

Def) A function $f(y,t)$ is said to have a Lipschitz condition in the variable on a set $D \subset \mathbb{R}^2$ if a constant $\gamma > 0$ exists with

$$|f(y_1,t) - f(y_2 - t)| \le \gamma |y_1 - y_2| \tag{2}$$

whenever $(y_1,t),\ (y_2,t) \in D$.

Euler method

The object of Euler's method is to get an approximation on initial value problem:

$$\frac{dy}{dt} = f(t, y),\ a \le t \le b,\ y(a) = \alpha. \qquad (1)$$

In fact, a continuous numerical solution never can be obtained, instead, we discretize the closed interval $[a, b]$ such that

$$t_i = a + ih, \quad \text{for each } i = 0, 1, 2, \ldots, N. \qquad (2)$$

The distance between the points $h = \dfrac{b-a}{N} = t_{i+1} - t_i$ is called the step size.

We use Taylor's Theorem to derive Euler's method.

Suppose that $y(t)$ has the following equations on $[a, b]$.

$$y(t_i + 1) = y(t_i) + (t_{i+1} - t_i)y'(t_i) + \frac{(t_{i+1} - t_i)^2}{2}y''(\xi_i),$$
$$= y(t_i) + (t_{i+1} - t_i)y'(t_i) + \frac{h^2}{2}y''(\xi_i), \quad (1)$$

Since $y(t)$ satisfies the initial differential equation $dy/dt = f(t, y)$,

$$= y(t_i) + (t_{i+1} - t_i)f(t_i, y(t_i)) + \frac{h^2}{2}y''(\xi_i).$$

Euler's method is constructed for each $i = 0, 1, 2, \ldots, N$ by deleting the remainder term.

$$y(t_{i+1}) = y(t_i) + (t_{i+1} - t_i)f(t_i, y(t_i)). \quad (2)$$

We let $\omega_i \approx y(t_i)$, be a estimated solution then Euler's method is

$$\omega_0 = \alpha,$$
$$\omega_{i+1} = \omega_i + hf(w_i, t_i). \quad (3)$$

For example, we approximate the solution of the initial value problem such as

$$y' = 1 - 2t + 5y, \quad y(0) = 2. \tag{1}$$

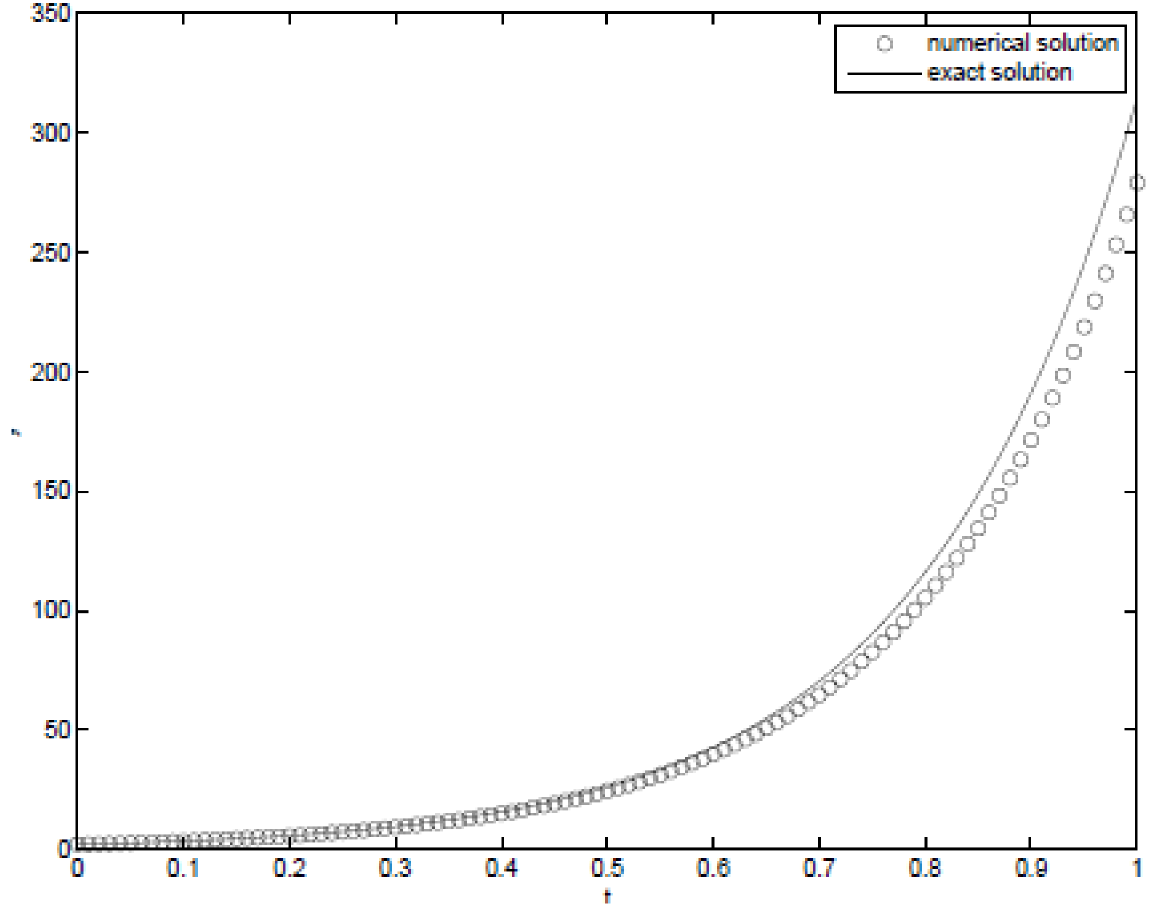
Then $h = 0.001, \; t_i = 0.01i, \;$ for $i = 0, 1, \ldots, 100.$

The exact solution is $y(t) = \dfrac{53}{25}e^{5t} + \dfrac{2}{5}t - \dfrac{3}{25}.$ (2)

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%% Eulerex.m %%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear; clc; clf;
h=0.01;    % time step size
t0=0; T=1;  t=[t0:h:T];  % time step
N=length(t);
y(1)=2;   % inital value
f=inline('1-2*ft+5*fy','ft','fy');

for i=1:N-1
    y(i+1) = y(i) + h * f(t(i),y(i));
end
exy = 53/25*exp(5*t)+2/5*t-3/25; % exact solution
plot(t,y,'ko',t,exy,'k');
xlabel('t'); ylabel('y')
legend('numerical solution','exact solution')
```

MATLAB code result



Error analysis

Error = |True value – Approximation|

$$\text{Relative error} = \frac{|\text{true vale-approximation}|}{\text{true value}}$$

Truncation, or discretization errors caused by the nature of the methods employed to approximate values of y

Round-off errors caused by the limited numbers of significant digits that can be inherited in a computer.

Improvements of Euler's method

The first step in improving the method is to see the Taylor series

$$y(t_{i+1}) = y(t_i) + hy'(t_i) + \frac{h^2}{2}y''(t_i) + O(h^3). \tag{1}$$

Since $y' = f(t,y)$, we use the chain rule of differentiation

$$y''(t_i) = f_t(t_i, y_i) + f_y(t_i, y_i)y'(t_i) = f_t(t_i, y_i) + f_y(t_i, y_i)f(t_i, y_i). \tag{2}$$

Then,

$$y(t_{i+1}) = y(t_i) + hf(t_i, y_i) + \frac{h^2}{2}f_t(t_i, y_i) + \frac{h^2}{2}f_y(t_i, y_i)f(t_i, y_i) + O(h^3)\,. \tag{3}$$

The other hand, we let

$$y(t_{i+1}) = y(t_i) + Ahf(t_i, y_i) + Bhf(t_i + Ph, y_i + Qhf(t_i, y_i)). \tag{4}$$

With help from Taylor series

$$f(t_i + Ph, y_i + Qhf(t_i, y_i)) \\ = f(t_i, y_i) + Phf_t(t_i, y_i) + Qhf(t_i, y_i)f_y(t_i, y_i) + O(h^2) \quad (1)$$

and combine this with the previous equation (4)

$$y(t_{i+1}) = y(t_i) + (A+B)hf(t_i, y_i) + BPh^2 f_t(t_i, y_i) \\ + BQh^2 f_y(t_i, y_i) f(t_i, y_i) + O(h^3) \quad (2)$$

Then we get the following by comparing coefficients

$$A + B = 1, \; BP = \frac{1}{2}, \; BQ = \frac{1}{2} \; . \quad (3)$$

If we set $A = 1/2$ then it follows $B = 1/2, \; P = 1, \; Q = 1$ and

$$y(t_{i+1}) = y(t_i) + \frac{h}{2}\left[f(t_i, y(t_i)) + f(t_{i+1}, y(t_i) + hf(t_i, y(t_i))\right] . \quad (4)$$

This is called the modified Euler's method.

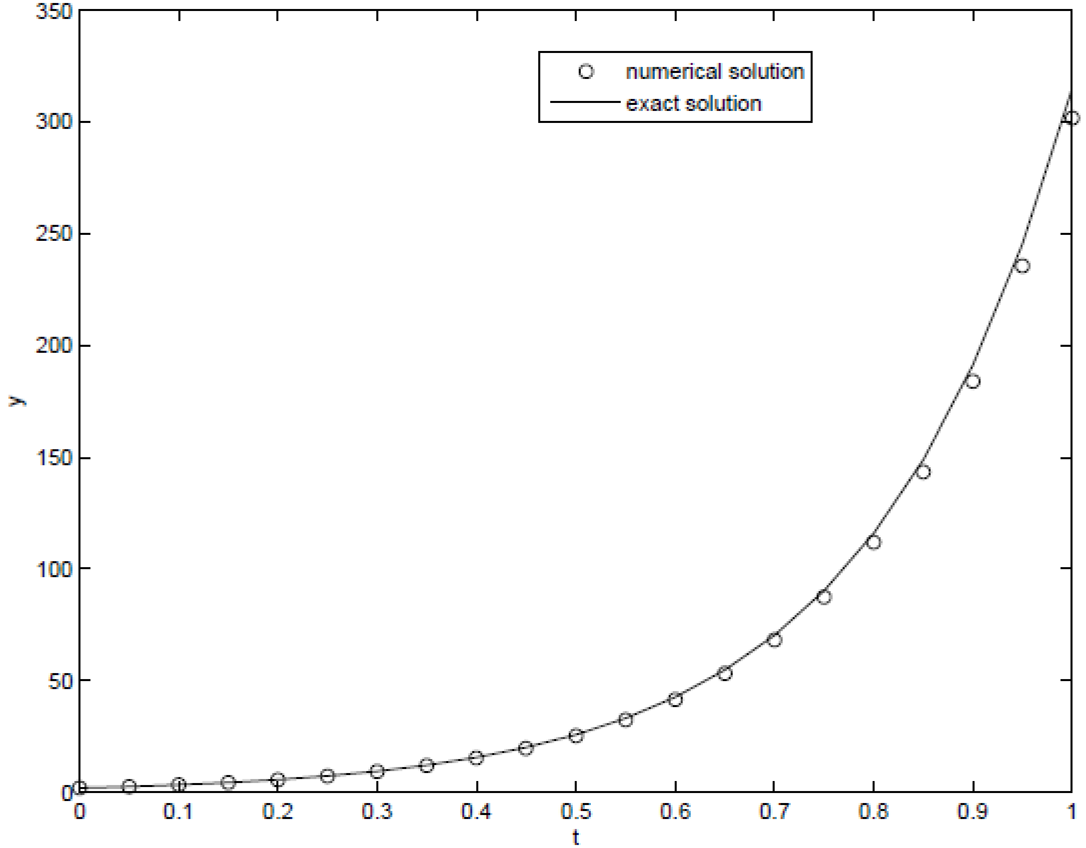
For example, we approximate the solution of the initial value problem such as

$$y' = 1 - 2t + 5y, \ \ y(0) = 2. \quad (1)$$

Then $h = 0.001, \ t_i = 0.01i, \ \text{for } i = 0, 1, \ldots, 100.$

The exact solution is $y(t) = \frac{53}{25}e^{5t} + \frac{2}{5}t - \frac{3}{25}.$ (2)

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% ModiEuler.m %%%%%%%%%%%%%%%%%%%%%%
clear; clc; clf;
h=0.05;    % time step size
t0=0; T=1; t=[t0:h:T];  % time step
N=length(t);
y(1)=2;   % inital value
f=inline('1-2*ft+5*fy','ft','fy');
for i=1:N-1
 y(i+1)=y(i)+0.5*h*(f(t(i),y(i))+f(t(i+1),y(i)+h*f(t(i),y(i))));
end
exy = 53/25*exp(5*t)+2/5*t-3/25; % exact solution
plot(t,y,'ko',t,exy,'k');
xlabel('t'); ylabel('y')
legend('numerical solution','exact solution')
```



numerical solution
exact solution
y
t
350
300
250
200
150
100
50
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

Runge-Kutta

Runge-Kutta methods have the fourth order local truncation error of the Talyor methods while eliminating to compute the derivatives of $f(t, y)$.

Many variations exist but all be cast be in the generalized form of

$$y_{i+1} = y_i + \phi h, \tag{1}$$

where ϕ, is called an increment function, which can be interpreted as a representative slope over the interval. The increment function can be written in general form as

$$\phi = a_1 k_1 + a_2 k_2 + \cdots + a_n k_n$$

where the $a'\mathrm{s}$ are constants and the $k'\mathrm{s}$ are recurrence relationships.

The classical fourth order Runge-Kutta method

$$y_{i+1} = y_i + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)h, \tag{1}$$

where

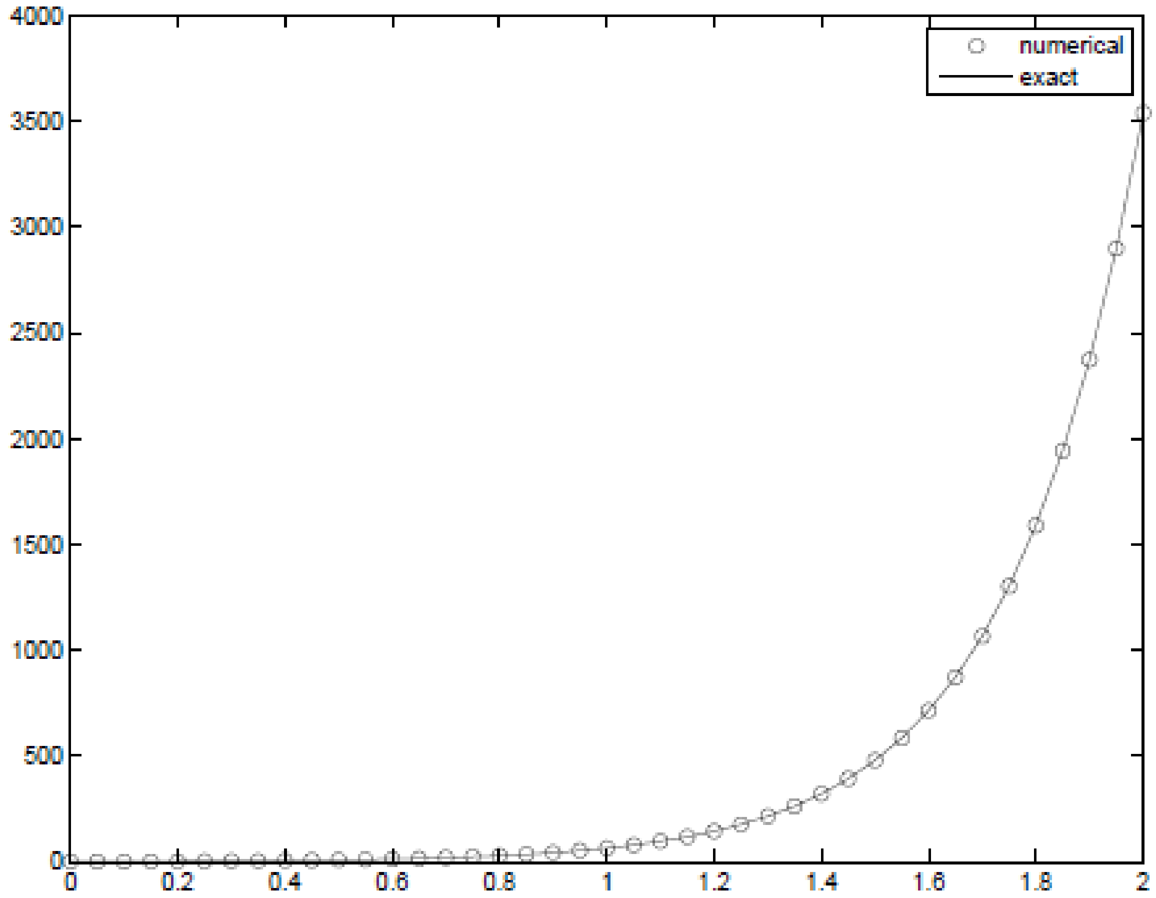
$$\begin{aligned} k_1 &= f(t_i, y_i), \\ k_2 &= f(t_i + \frac{1}{2}h, y_i + \frac{1}{2}k_1 h), \\ k_3 &= f(t_i + \frac{1}{2}h, y_i + \frac{1}{2}k_2 h), \\ k_4 &= f(t_i + h, y_i + k_3 h). \end{aligned} \tag{2}$$

For example, we use the fourth order Runge-Kutta method to solve the same problem

$$y' = 1 - 2t + 5y, \quad y(0) = 2. \tag{3}$$

Then $h = 0.001,\ t_i = 0.01i,\ \text{for } i = 0, 1, \ldots, 100.$

```
%%%%%%%%%%%%%%%%%%%%%%%%%%% RK4.m %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear; clc; clf;
h=0.05;    % time step size
t0=0; T=1; t=[t0:h:T];  % time step
N=length(t);
y(1)=2;   % inital value
f=inline('1-2*ft+5*fy','ft','fy');
for i=1:N-1
    k1=f(t(i),y(i));
    k2=f(t(i) + 0.5*h,y(i) + 0.5*h*k1);
    k3=f(t(i) + 0.5*h,y(i) + 0.5*h*k2);
    k4=f(t(i) + h,y(i) + h*k3);
    y(i+1) = y(i) + h/6 * (k1+ 2*k2 + 2*k3 + k4);
end
exy = 53/25*exp(5*t)+2/5*t-3/25; % exact solution
plot(t,y,'ko',t,exy,'k');
xlabel('t'); ylabel('y')
legend('numerical solution','exact solution')
```



4000
3500
3000
2500
2000
1500
1000
500
0
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
numerical
exact