Initial value problem

1 Initial value problem

IVP (Initial value problem)

2 How to solve initial value problem - Euler's method

Euler's method

2nd order schemes

RK4

Example - Euler's method

Example - RK4

3 How to solve initial value problem - Adam's method

Adam's method

Predictor-Corrector

4 Error

Truncation error - Local error and global error

Round off error

Total (local) error

5 Stability

Stability of scheme

6 `ode45`

`ode45`

Example - `ode45`

Example - `ode45` with parameters

Example - `ode45` with options

IVP (Initial value problem)

$$\begin{cases} \mathbf{y}' = \mathbf{f}(t, \mathbf{y}) & \text{[Differential equation]} \\ \mathbf{y}(t_0) = \mathbf{y_0} & \text{[Initial condition]} \end{cases}$$

Euler's method

$\boxed{\text{Divide time interval}}$

$$t_0 \quad < \quad t_1 \quad < \quad t_2 \quad < \quad \cdots \quad < \quad t_{N-1} \quad < \quad t_N$$

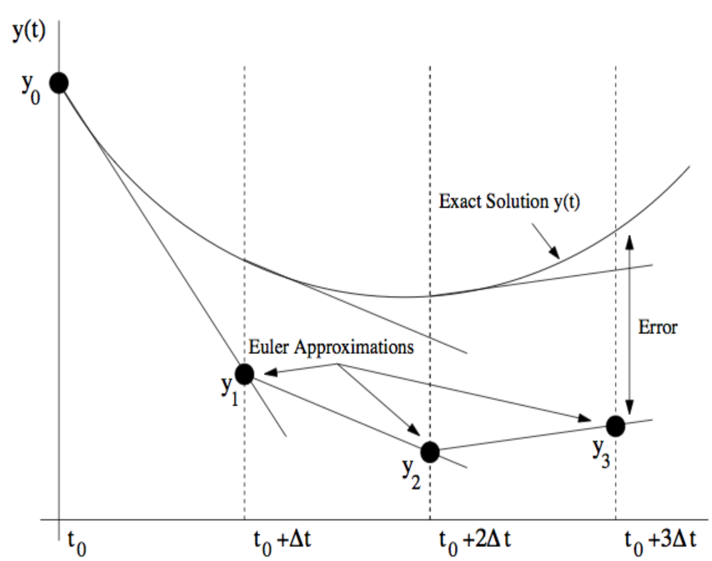
$\boxed{\text{Given info}}$

$$\begin{array}{ccccccccc} \mathbf{y}_0 & & \mathbf{y}_1 & & \mathbf{y}_2 & & \cdots & & \mathbf{y}_n \\ t_0 & < & t_1 & < & t_2 & < & \cdots & < & t_n \end{array}$$

$\boxed{\text{Update info}}$

$$\begin{array}{ccccccccccc} \mathbf{y}_0 & & \mathbf{y}_1 & & \mathbf{y}_2 & & \cdots & & \mathbf{y}_n & & \mathbf{y}_{n+1} \\ t_0 & < & t_1 & < & t_2 & < & \cdots & < & t_n & < & t_{n+1} \end{array}$$

$$\mathbf{y}' = \mathbf{f}(t, \mathbf{y}) \quad \Rightarrow \quad \frac{\mathbf{y}_{n+1} - \mathbf{y}_n}{\Delta t} \approx \mathbf{f}(t_n, \mathbf{y}_n) \quad \Rightarrow \quad \mathbf{y}_{n+1} = \mathbf{y}_n + \mathbf{f}(t_n, \mathbf{y}_n)\Delta t$$



2nd order schemes

Slope setup

$$\mathbf{y}_{n+1} = \mathbf{y}_n + \phi \Delta t$$

$$\phi = A\mathbf{f}(t_n, \mathbf{y}_n) + B\mathbf{f}(t_n + P\Delta t, \mathbf{y}_n + Q\mathbf{f}(t_n, \mathbf{y}_n)\Delta t)$$

Taylor expansion

$$\begin{aligned}
LHS &= \mathbf{y}(t_n + \Delta t) \\
&= \mathbf{y}(t_n) + \left(\frac{d\mathbf{y}}{dt}\right)_{t_n} \Delta t + \frac{1}{2}\left(\frac{d^2\mathbf{y}}{dt^2}\right)_{t_n} (\Delta t)^2 \\
&= \mathbf{y}_n + \mathbf{f}(t_n, \mathbf{y}_n)\Delta t + \frac{1}{2}(\mathbf{f}_t(t_n, \mathbf{y}_n) + \mathbf{H}_{\mathbf{f},\mathbf{y}}(t_n, \mathbf{y}_n)\mathbf{f}(t_n, \mathbf{y}_n))(\Delta t)^2
\end{aligned}$$

$$\begin{aligned}
\phi &= A\mathbf{f}(t_n, \mathbf{y}_n) + B\left[\mathbf{f}(t_n, \mathbf{y}_n) + \mathbf{f}_t(t_n, \mathbf{y}_n)P\Delta t + \mathbf{H}_{\mathbf{f},\mathbf{y}}(t_n, \mathbf{y}_n)Q\mathbf{f}(t_n, \mathbf{y}_n)\Delta t\right] \\
&= \left[A + B\right]\mathbf{f}(t_n, \mathbf{y}_n) + \left[\mathbf{f}_t(t_n, \mathbf{y}_n)BP + \mathbf{H}_{\mathbf{f},\mathbf{y}}(t_n, \mathbf{y}_n)BQ\mathbf{f}(t_n, \mathbf{y}_n)\right]\Delta t
\end{aligned}$$

$$RHS = \mathbf{y}_n + \left[A + B\right]\mathbf{f}(t_n, \mathbf{y}_n)\Delta t + \left[\mathbf{f}_t(t_n, \mathbf{y}_n)BP + \mathbf{H}_{\mathbf{f},\mathbf{y}}(t_n, \mathbf{y}_n)BQ\mathbf{f}(t_n, \mathbf{y}_n)\right](\Delta t)^2$$

Coefficient of Δt	$A + B = 1$
Coefficient of $\mathbf{f}_t(t_n, \mathbf{y}_n)(\Delta t)^2$	$BP = 1/2$
Coefficient of $\mathbf{H}_{\mathbf{f},\mathbf{y}}(t_n, \mathbf{y}_n)\mathbf{f}(t_n, \mathbf{y}_n)(\Delta t)^2$	$BQ = 1/2$

2nd order schemes

[RK2] $A = 0,\ B = 1,\ P = \frac{1}{2},\ Q = \frac{1}{2}$ $\qquad \mathbf{y}_{n+1} = \mathbf{y}_n + \mathbf{f}_2\Delta t + (\Delta t^3)$

[Heun] $A = \frac{1}{2},\ B = \frac{1}{2},\ P = 1,\ Q = 1$ $\qquad \mathbf{y}_{n+1} = \mathbf{y}_n + \left[\frac{\mathbf{f}_1 + \tilde{\mathbf{f}}_4}{2}\right]\Delta t + (\Delta t^3)$

where

$$\begin{aligned}
\mathbf{f}_1 &= \mathbf{f}\left(t_n, \mathbf{y}_n\right) \\
\mathbf{f}_2 &= \mathbf{f}\left(t_n + \frac{\Delta t}{2}, \mathbf{y}_n + \frac{\Delta t}{2}\mathbf{f}_1\right) \\
\tilde{\mathbf{f}}_4 &= \mathbf{f}\left(t_n + \Delta t, \mathbf{y}_n + \Delta t\mathbf{f}_1\right)
\end{aligned}$$

RK4

$$\mathbf{y}_{n+1} = \mathbf{y}_n + \left[\frac{\mathbf{f}_1 + 2\mathbf{f}_2 + 2\mathbf{f}_3 + \mathbf{f}_4}{6}\right]\Delta t + (\Delta t^5)$$

where

$$\begin{aligned}
\mathbf{f}_1 &= \mathbf{f}\left(t_n, \mathbf{y}_n\right) \\
\mathbf{f}_2 &= \mathbf{f}\left(t_n + \frac{\Delta t}{2}, \mathbf{y}_n + \frac{\Delta t}{2}\mathbf{f}_1\right) \\
\mathbf{f}_3 &= \mathbf{f}\left(t_n + \frac{\Delta t}{2}, \mathbf{y}_n + \frac{\Delta t}{2}\mathbf{f}_2\right) \\
\mathbf{f}_4 &= \mathbf{f}\left(t_n + \Delta t, \mathbf{y}_n + \Delta t\mathbf{f}_3\right)
\end{aligned}$$

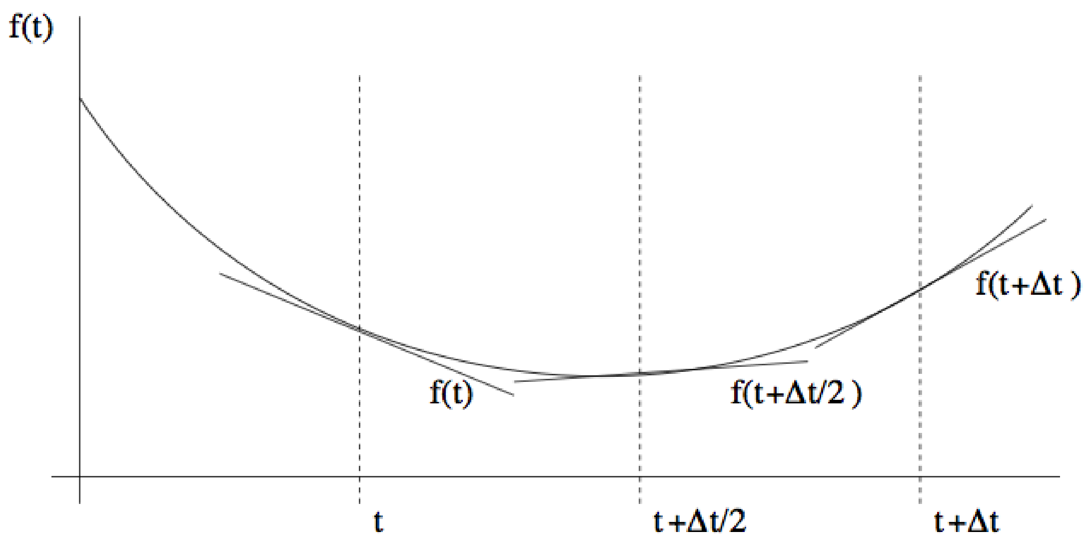
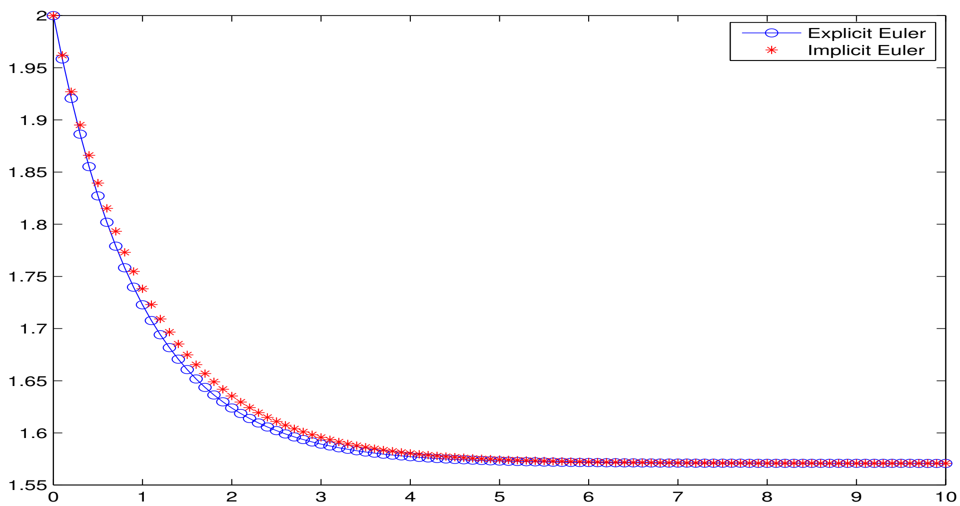


Figure 1: Graphical description of the initial, intermediate, and final slopes used in RK4 scheme over a time Δt. Source: J. N. Kutz

Example - Euler's method

Using Euler's method solve the following initial value problem.

$$y' = \cos(y), \quad y(0) = 2$$

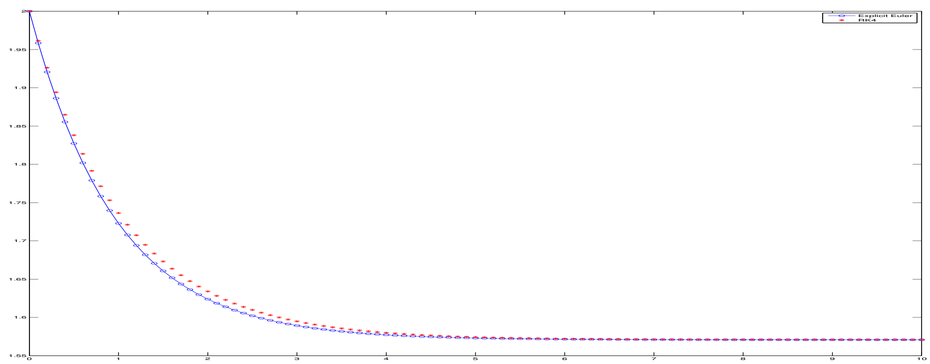



```
clear all; close all; clc;

Nt = 101; t = linspace(0,10,Nt); dt = t(2)-t(1);
ic = 2;

% Euler
y = zeros(size(t)); y(1) = ic;
for i=1:Nt-1
    y(i+1) = y(i) + dt * cos(y(i));
end

plot(t,y,'o-')
```

Example - RK4

Using Euler and RK4 solve the following IVP and compare their solutions.

$$y' = \cos(y), \quad y(0) = 2$$



```
clear all; close all; clc;

Nt = 101; t = linspace(0,10,Nt); dt = t(2)-t(1);
ic = 2;

% Euler
y = zeros(size(t)); y(1) = ic;
for i=1:Nt-1
    y(i+1) = y(i) + dt * cos(y(i));
end

% RK4
z = zeros(size(t)); z(1) = ic;
for i=1:Nt-1
    a1 = z(i); f1 = cos(a1);
    a2 = a1 + (dt/2)*f1; f2 = cos(a2);
    a3 = a2 + (dt/2)*f2; f3 = cos(a3);
    a4 = a3 + dt*f3; f4 = cos(a4);
    z(i+1) = a1 + dt * (f1+2*f2+2*f3+f4)/6;
end

plot(t,y,'o-',t,z,'*r'); legend('Explicit Euler','RK4')
```

Adam's method

$$\mathbf{y}' = \mathbf{f}(t,\mathbf{y}) \quad \Rightarrow \quad \mathbf{y}_{n+1} - \mathbf{y}_n = \int_{t_n}^{t_{n+1}} \mathbf{f}(t,\mathbf{y})dt \quad \Rightarrow \quad \mathbf{y}_{n+1} = \mathbf{y}_n + \int_{t_n}^{t_{n+1}} \mathbf{p}(t,\mathbf{y})dt$$

Adams-Bashforth - Not allow to use future points

Use current point $\Rightarrow \quad \mathbf{y}_{n+1} = \mathbf{y}_n + \mathbf{f}(t_n,\mathbf{y}_n)\Delta t$

Use current and last point $\Rightarrow \quad \mathbf{y}_{n+1} = \mathbf{y}_n + \dfrac{3\mathbf{f}(t_n,\mathbf{y}_n) - \mathbf{f}(t_{n-1},\mathbf{y}_{n-1})}{2}\Delta t$

Adams-Moulton - Allow to use future points - Implicit

Use future point $\Rightarrow \quad \mathbf{y}_{n+1} = \mathbf{y}_n + \mathbf{f}(t_{n+1},\mathbf{y}_{n+1})\Delta t$

Use current and future point $\Rightarrow \quad \mathbf{y}_{n+1} = \mathbf{y}_n + \dfrac{\mathbf{f}(t_n,\mathbf{y}_n) + \mathbf{f}(t_{n+1},\mathbf{y}_{n+1})}{2}\Delta t$

Predictor-Corrector

Adam's method

$$\mathbf{y}' = \mathbf{f}(t, \mathbf{y}) \quad \Rightarrow \quad \mathbf{y}_{n+1} - \mathbf{y}_n = \int_{t_n}^{t_{n+1}} \mathbf{f}(t, \mathbf{y})dt \quad \Rightarrow \quad \mathbf{y}_{n+1} = \mathbf{y}_n + \int_{t_n}^{t_{n+1}} \mathbf{p}(t, \mathbf{y})dt$$

Adams-Bashforth - Not allow to use future points

Use current point $\Rightarrow$ $\mathbf{y}_{n+1} = \mathbf{y}_n + \mathbf{f}(t_n, \mathbf{y}_n)\Delta t$

Use current and last point $\Rightarrow$ $\mathbf{y}_{n+1} = \mathbf{y}_n + \dfrac{3\mathbf{f}(t_n, \mathbf{y}_n) - \mathbf{f}(t_{n-1}, \mathbf{y}_{n-1})}{2}\Delta t$

Adams-Moulton - Allow to use future points - Implicit

Use future point $\Rightarrow$ $\mathbf{y}_{n+1} = \mathbf{y}_n + \mathbf{f}(t_{n+1}, \mathbf{y}_{n+1})\Delta t$

Use current and future point $\Rightarrow$ $\mathbf{y}_{n+1} = \mathbf{y}_n + \dfrac{\mathbf{f}(t_n, \mathbf{y}_n) + \mathbf{f}(t_{n+1}, \mathbf{y}_{n+1})}{2}\Delta t$

Predictor-Corrector modification to Adams-Moulton

Use future point $\Rightarrow$ $\tilde{\mathbf{y}}_{n+1} = \mathbf{y}_n + \mathbf{f}(t_n, \mathbf{y}_n)\Delta t$

$$\mathbf{y}_{n+1} = \mathbf{y}_n + \mathbf{f}(t_{n+1}, \tilde{\mathbf{y}}_{n+1})\Delta t$$

Use current and future point $\Rightarrow$ $\tilde{\mathbf{y}}_{n+1} = \mathbf{y}_n + \mathbf{f}(t_n, \mathbf{y}_n)\Delta t$

$$\mathbf{y}_{n+1} = \mathbf{y}_n + \frac{\mathbf{f}(t_n, \mathbf{y}_n) + \mathbf{f}(t_{n+1}, \tilde{\mathbf{y}}_{n+1})}{2}\Delta t$$

Truncation error - Local error and global error

Local error -Euler

$$\mathbf{y}(t+\Delta t) = \mathbf{y}(t) + \mathbf{y}'(t)\Delta t + \frac{1}{2}\mathbf{y}''(c)\Delta t^2 \approx \mathbf{y}(t) + \mathbf{y}'(t)\Delta t$$

$$\varepsilon_k = \frac{1}{2}\mathbf{y}''(c_k)\Delta t^2 = O(\Delta t^2)$$

Global error - Euler

$$E_N = \sum_{j=1}^{N} \frac{1}{2}\mathbf{y}''(c_j)\Delta t^2 = \frac{1}{2}\mathbf{y}''(\bar{c})\Delta t^2 \cdot N = \frac{1}{2}\mathbf{y}''(\bar{c})\Delta t^2 \cdot \frac{b-a}{\Delta t} = O(\Delta t)$$

Matlab IVP solvers

$$\begin{array}{ccc} \texttt{ode} & \texttt{4} & \texttt{5} \\ & \uparrow & \uparrow \\ & \text{Global error} & \text{Local error} \end{array}$$

	Description
`ode45`	RK4 (First choice)
`ode23`	RK2
`ode113`	Predictor-Corrector
`ode15s`	Stiff (2nd choice)
`ode15i`	Fully implicit differential equation
`ode23s`	Stiff
`ode23t`	Stiff
`ode23tb`	Stiff

Round off error

$$\mathbf{y}_n = \underbrace{\mathbf{Y}_n}_{\text{Representation of } \mathbf{y}_n \text{ in double precision}} + \underbrace{\mathbf{e}_n}_{\text{Round off error}}$$

$$|\mathbf{e}_n| \leq f = 10^{-16}$$

Total (local) error

$$\text{Total (local) error} \quad = \quad \text{Truncation (local) error} \quad + \quad \text{Round off error}$$

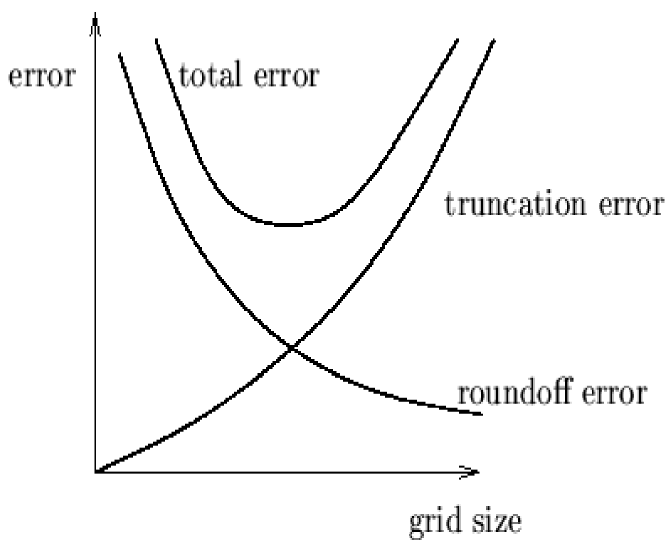
Total (local) error - Euler

$$\mathbf{y}' = \frac{\mathbf{y}_{n+1} - \mathbf{y}_n}{\Delta t} + \varepsilon(\mathbf{y}_n, \Delta t) = \frac{\mathbf{Y}_{n+1} - \mathbf{Y}_n}{\Delta t} + \underbrace{\frac{\mathbf{e}_{n+1} - \mathbf{e}_n}{\Delta t}}_{E_{\text{Round}}} + \underbrace{\varepsilon(\mathbf{y}_n, \Delta t)}_{E_{\text{Truncation}}}$$

With $M = \max|\mathbf{y}''(c)|$

$$E_{\text{Total}} = E_{\text{Round}} + E_{\text{Truncation}} \leq \frac{f+f}{\Delta t} + \frac{1}{2}M\Delta t = \frac{2f}{\Delta t} + \frac{1}{2}M\Delta t$$

Total (local) error starts increasing if you decrease beyond the optimal grid



http://www.me.ucsb.edu/ moehlis/APC591/tutorials/tutorial5/node3.html

Stability of scheme

For $\lambda > 0$

$$y' = \lambda y, \quad y(0) = y_0 \quad \Rightarrow \quad y(t) = y_0 e^{\lambda t}$$

Euler

$$y_{n+1} = y_n + \lambda y_n \Delta t = (1 + \lambda\Delta t) y_n \quad \Rightarrow \quad y_N = (1 + \lambda \Delta t)^N y_0$$

$$y_0 = Y_0 + \varepsilon \quad \Rightarrow \quad E = (1 + \lambda\Delta t)^N \varepsilon$$

$$\begin{array}{llll} \text{[Stable]} & |1 + \lambda\Delta t| < 1 & \Rightarrow & E \to 0 \\ \text{[Untable]} & |1 + \lambda\Delta t| > 1 & \Rightarrow & E \to \infty \end{array}$$

Backward Euler

$$y_{n+1} = y_n + \lambda y_{n+1} \Delta t = \frac{y_n}{1 - \lambda\Delta t} \quad \Rightarrow \quad y_N = \frac{y_0}{(1 - \lambda\Delta t)^N}$$

$$y_0 = Y_0 + \varepsilon \quad \Rightarrow \quad E = \frac{\varepsilon}{(1 - \lambda\Delta t)^N}$$

$$\begin{array}{llll} \text{[Stable]} & \left|\dfrac{1}{1 - \lambda\Delta t}\right| < 1 & \Rightarrow & E \to 0 \\ \text{[Untable]} & \left|\dfrac{1}{1 - \lambda\Delta t}\right| > 1 & \Rightarrow & E \to \infty \end{array}$$

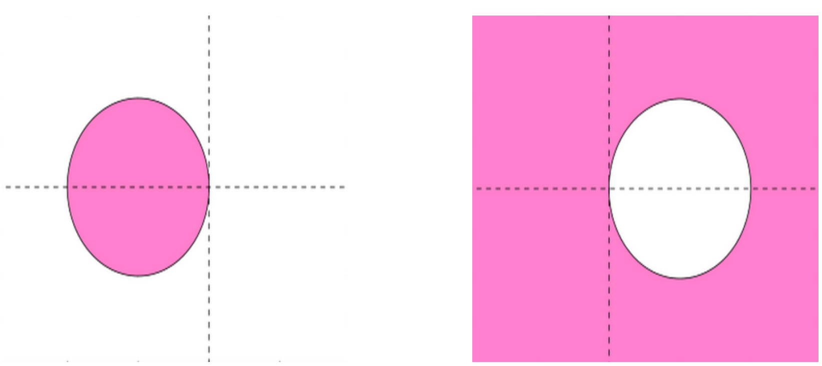


Figure 2: With $z = \lambda\Delta t$ pink region in left is the stability region for Euler, whereas pink region in right is the stability region for backward Euler. Source: wikipedia.

ode45

ode45

```
[t y] = ode45 (rhs, tspan, ic)
               ↑
           rhs=@(t,y)
```

ode45 with parameters

```
[t y] = ode45 (rhs,       tspan, ic, [],                 bt)
                ↑                    ↑                   ↑
         rhs=@(t,y,bt)        Place for options      Parameter
```

ode45 with options

```
options = odeset ('RelTol', 1e-2, 'AbsTol', [1e-4 1e-5])

[t y] = ode45 (rhs, tspan, ic, options);
```

	Description
'RelTol'	Relative tolerance (Default; 1e-3)
'AbsTol'	Absolute tolerance (Default; 1e-6)

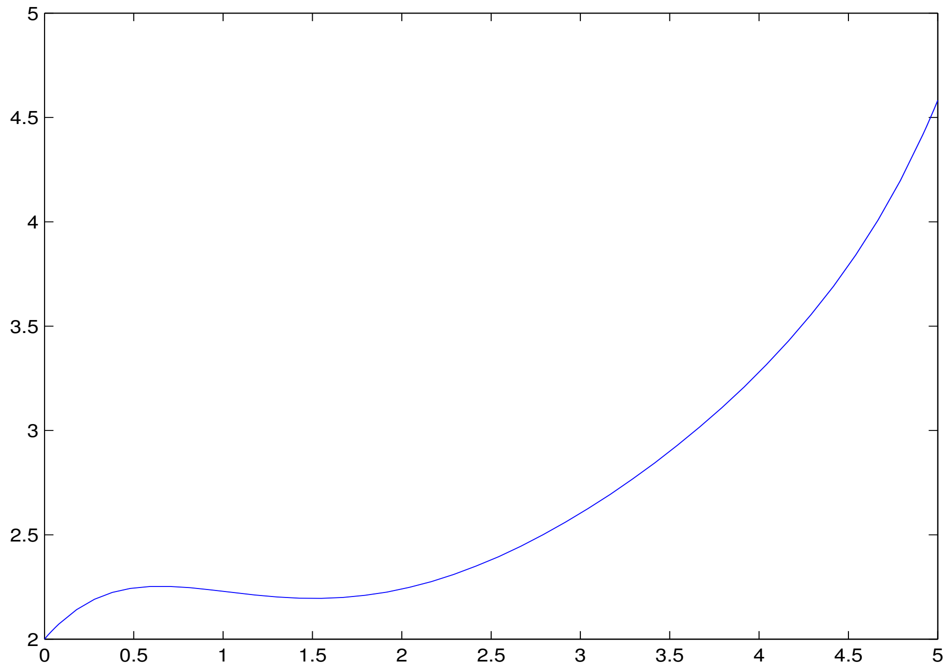
Output

t	[y(:,1)	y(:,2)]
0	3.2791	9.2634
0.01	3.2832	9.2635
0.02	3.2823	9.2637
⋮	⋮	⋮

Example - ode45

Solve the following ODE with IC $y(0) = 2$, $y'(0) = 1$:

$$y'' + 2\sin(y') - 2\cos(y) = t$$

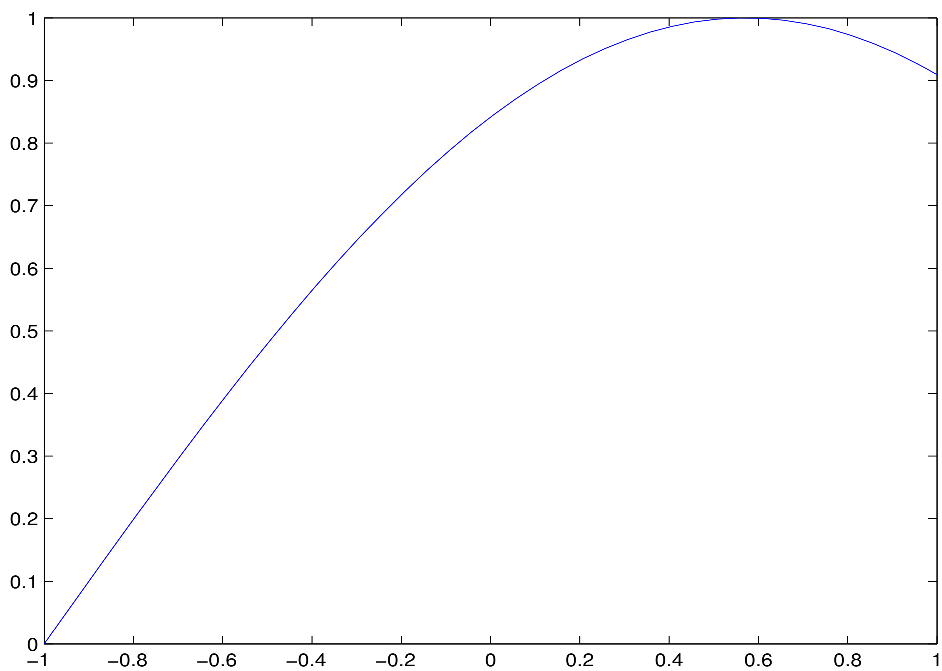



```
rhs = @(t,y) [y(2); 2*cos(y(1)) - 2*sin(y(2)) + t];
tspan = [0; 5];
ic = [2; 1];

[t y] = ode45(rhs,tspan,ic);

plot(t,y(:,1));
```

Example - ode45 with parameters

Solve the following ODE with IC $y(-1) = 0$, $y'(-1) = 1$: With $\beta = 99$

$$y'' + (100 - \beta)y = 0$$

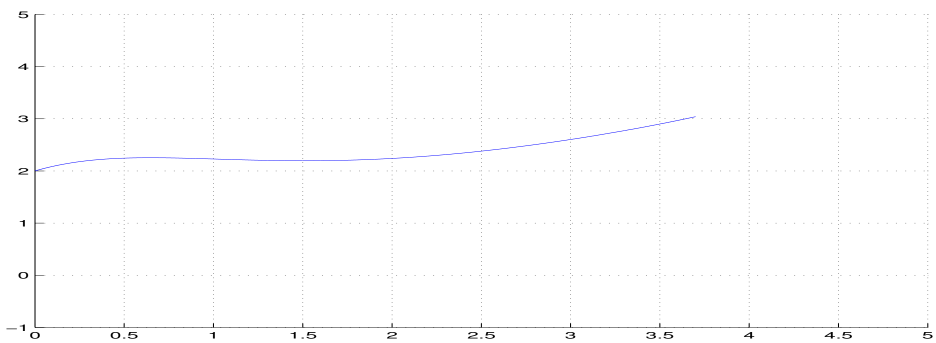


```
rhs = @(t,y,bt) [y(2); (bt-100)*y(1)];
tspan = [-1; 1];
ic = [0; 1];

bt = 99;

[t,y] = ode45(rhs,tspan,ic,[],bt);

plot(t,y(:,1))
```

Example - ode45 with options

Find the hitting time of the level 3, when y moves according to the following ODE:

$$y'' + 2\sin(y') - 2\cos(y) = t, \qquad y(0) = 2,\ y'(0) = 1$$




```
rhs = @(t,y) [y(2); 2*cos(y(1)) - 2*sin(y(2)) + t];
tmin = 0; tmax = 5; dt = 0.1; tspan = [0 dt];
ic = [2; 1];

options = odeset('RelTol', 1e-2, 'AbsTol', [1e-4 1e-5]);

axis([0 5 -1 5]); grid on; hold on;

for i=1:tmax/dt
    [t y] = ode45(rhs,tspan,ic,options);
    plot(t,y(:,1))
    y_end = y(end,:);

    if (y_end(1) >= 3),
        y1 = y(:,1); ind = find(y1>=3, 1, 'first'); Hitting_Time = t(ind)
        return
    end

    tspan = tspan+dt;
    ic = y_end';
end

fprintf('We does not reach 3 up to time %g!',tmax)
```