Solving Ordinary Differential Equations using MATLAB

Initial Value Problems

The typical IVPs are "vector form first order explicit ODEs" given by

$$(IVP): \begin{cases} \dot{\vec{x}}(t) = \vec{f}(t, \vec{x}(t)), \\ \vec{x}(t_0) = \vec{x}_0 \end{cases}$$

where $\vec{x}^T(t) = [x_1(t), \ldots, x_n(t)]$ is the "state vector", $\vec{x}(t_0) = \vec{x}_0 = [x_1(t_0), \ldots, x_n(t_0)]^T$ is the initial value (state), and $f^T = [f_1, \ldots, f_n]$ is a (possibly nonlinear) function. Here we will try to understand and solve numerically for $\vec{x}(t)$, $t \in [t_0, t_f]$, where t_f is referred to as the terminal time.

Given a "small" calculation step h, the LHS of IVP can be approximated by

$$\dot{\vec{x}}(t) \approx \frac{\vec{x}(t_0 + h) - \vec{x}(t_0)}{t_0 + h - t_0}$$

so that the approximate solution at $t_0 + h$ can be written as

$$\underbrace{\hat{\vec{x}}(t_0 + h)}_{\hat{x}_1} = \underbrace{x(t_0)}_{\vec{x}_0} + h \underbrace{\vec{f}(t_0, \vec{x}(t_0))}_{\vec{f}_0}.$$

Of course, this representation contains the error, and it should be written (more generally) as

$$\vec{x}(t_0 + h) = \hat{\vec{x}}(t_0 + h) + \vec{R}_0 = \vec{x}_0 + h \vec{f}(t_0, \vec{x}_0) + R_0$$

where R_0 is the approximation error.

Denoting the state vector at time t_k by $\vec{x}_k$, the "Euler's Algorithm" generates a new state vector

$$(EM): \quad \boxed{\vec{x}_{k+1} = \vec{x}_k + h_k \vec{f}(t_k, \vec{x}_k)} \quad , \quad k = 0, 1, \ldots$$

at time $t_k + h_k$. Note that, here $\vec{x}_k$ represents the approximation for $\vec{x}(t_k)$ {without using the "^" symbol}

Runge-Kutta

Fourth-Order fixed-step Runge-Kutta (RK) algorithm, which is considered to be an effective and easy to implement algorithm. Four additional intermediate variables are introduced such that

$$(RK4)_{k's} \qquad \begin{aligned} \vec{k}_1 &= h \vec{f}(t_k, \vec{x}_k) \\ \vec{k}_2 &= h \vec{f}\left(t_k + \frac{h}{2}, \vec{x}_k + \frac{\vec{k}_1}{2}\right) \\ \vec{k}_3 &= h \vec{f}\left(t_k + \frac{h}{2}, \vec{x}_k + \frac{\vec{k}_2}{2}\right) \\ \vec{k}_4 &= h \vec{f}\left(t_k + h, \vec{x}_k + \vec{k}_3\right) \end{aligned}$$

where h is the fixed step-size; the state vector at the next step is calculated via

$$(RK4): \quad \vec{x}_{k+1} = \vec{x}_k + \frac{1}{6}\left(\vec{k}_1 + 2\vec{k}_2 + 2\vec{k}_3 + \vec{k}_4\right)$$

$$\begin{array}{c|cccc} 0 & & & & \\ 1/2 & 1/2 & & & \\ 1/2 & 0 & 1/2 & & \\ 1 & 0 & 0 & 1 & \\ \hline & 1/6 & 1/3 & 1/3 & 1/6 \end{array} \qquad\qquad \begin{array}{c|c} \alpha & \beta \\ \hline & \gamma^T \end{array} \quad \text{Butcher array}$$

Runge-Kutta-Felhberg

Assuming that the current step-size is h_k, the six intermediate variables $\vec{k}_i$ are evaluated by

$$\vec{k}_i = h_k \vec{f}\left(t_k + \alpha_i h_k,\ \vec{x}_k + \sum_{j=1}^{i-1} \beta_{ij} \vec{k}_j\right), \quad i=1,2,\dots,6$$

where t_k is the current time, and α_i & β_{ij} are parameters, referred to as Dormand-Prince pairs.

The state vector at the next step is obtained from

$$\vec{x}_{k+1} = \vec{x}_k + \sum_{i=1}^{6} \gamma_i \vec{k}_i$$

The algorithm is also known as the "4/5 Runge-Kutta-Felhberg (RK(F)45)" algorithm. The coefficients α_i, β_{ij}, γ_i are given below (Butcher array):

α_i	β_{ij}					
0						
1/4	1/4					
3/8	3/32	9/32				
12/13	1932/2197	−7200/2197	7296/2197			
1	439/216	−8	3680/513	−845/4104		
1/2	−8/27	2	−3544/2565	1859/4104	−11/40	
γ_i	16/135	0	6656/12825	28561/56430	−9/50	2/55
→ γ_i^*	25/216	0	1408/2565	2197/4104	−1/5	0

Although the RKF45 seems to be a fixed-step size algorithm, in practical applications, an error vector

$$\vec{\epsilon}_k = \sum_{i=1}^{6} (\gamma_i - \gamma_i^*) \vec{k}_i$$

can be calculated, and accordingly the step-size h_k can be adjusted. Such an algorithm is known as "adaptive step-size" algorithm.
(variable)

Other RK Methods

General form of a third-order RK method is

$$\vec{k}_1 = h\vec{f}(t_i, \vec{x}_i)$$
$$\vec{k}_2 = h\vec{f}(t_i + \alpha_2 h, \vec{x}_i + \beta_{21}\vec{k}_1)$$
$$\vec{k}_3 = h\vec{f}(t_i + \alpha_3 h, \vec{x}_i + \beta_{31}\vec{k}_1 + \beta_{32}\vec{k}_2)$$

$$\vec{x}_{i+1} = \vec{x}_i + \gamma_1\vec{k}_1 + \gamma_2\vec{k}_2 + \gamma_3\vec{k}_3$$

$$\begin{array}{c|ccc} 0 & & & \\ \alpha_2 & \beta_{21} & & \\ \alpha_3 & \beta_{31} & \beta_{32} & \\ \hline & \gamma_1 & \gamma_2 & \gamma_3 \end{array}$$

— RK3 Kutta's Method

$$\begin{array}{c|ccc} 0 & & & \\ 1/2 & 1/2 & & \\ 1 & -1 & 2 & \\ \hline & 1/6 & 2/3 & 1/6 \end{array}$$

— RK3 Optimal Method

$$\begin{array}{c|ccc} 0 & & & \\ 1/2 & 1/2 & & \\ 3/4 & 0 & 3/4 & \\ \hline & 2/9 & 3/9 & 4/9 \end{array}$$

— RK3 Two-Thirds Rule

$$\begin{array}{c|ccc} 0 & & & \\ 2/3 & 2/3 & & \\ 2/3 & 1/3 & 1/3 & \\ \hline & 1/4 & 0 & 3/4 \end{array}$$

Multi-Step Methods

General (k-step) multistep method has the form

$$\vec{x}_{i+1} = \sum_{j=1}^{k} a_j \vec{x}_{i-j+1} + h \sum_{j=0}^{k} b_j \vec{f}_{i-j+1}$$

when $b_0 \neq 0$, the method is implicit; otherwise explicit.

Two common methods are derived from

— Adams (Multi-step) Methods

$$\vec{x}_{i+1} = \vec{x}_i + h \sum_{j=0}^{k} b_j \vec{f}_{i-j+1}$$

1°) Adams–Bashforth
when $b_0 = 0$ (explicit)

2°) Adams–Moulton
when $b_0 \neq 0$ (implicit).

Eg: Third-Order Adams-Bashforth:

$$\vec{x}_{i+1} = \vec{x}_i + \frac{h}{12}\left[23\vec{f}_i - 16\vec{f}_{i-1} + 5\vec{f}_{i-2}\right]$$

Eg: Third-Order Adams–Moulton:

$$\vec{x}_{i+1} = \vec{x}_i + \frac{h}{12}\left[5\vec{f}_{i+1} + 8\vec{f}_i - \vec{f}_{i-1}\right]$$

Predictor – Corrector Methods

Eg: ABM3:

$$\vec{x}^{*}_{i+1} = \vec{x}_i + \frac{h}{12}\left[23\vec{f}_i - 16\vec{f}_{i-1} + 5\vec{f}_{i-2}\right]$$

$$\vec{x}_{i+1} = \vec{x}_i + \frac{h}{12}\left[5\vec{f}^{*}_{i+1} + 8\vec{f}_i - \vec{f}_{i-1}\right]$$

Note that initially a one-step method, such as RK3 (variant), should be used to compute $\vec{x}_1$ and $\vec{x}_2$, besides $\vec{x}_0$.

BDF (Backward Differentiation Formula) Methods

Such methods arise from

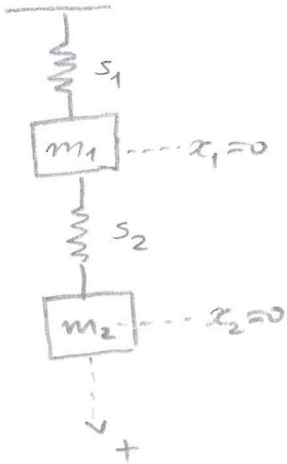
$$\dot{\vec{x}}_{i+1} = \vec{f}(t_{i+1}, \vec{x}_{i+1})$$

with $\dot{\vec{x}}_{i+1}$ is replaced by a backward difference formula, $\dfrac{\vec{x}_{i+1} - \vec{x}_i}{h}$.

BDF k-step Methods has a general form

$$\vec{x}_{i+1} = \sum_{j=1}^{k} a_j \vec{x}_{i-j} + b_0 h \vec{f}_{i+1}$$

Mass-Spring System



$$m_1 x_1'' = -s_1 x_1 + s_2 (x_2 - x_1)$$

$$m_2 x_2'' = -s_2 (x_2 - x_1)$$

$m_1 = 10, \quad m_2 = 2$

$s_1 = 100, \quad s_2 = 120$

$x_1(0) = 1/2 \qquad x_1'(0) = 0$

$x_2(0) = 1/4 \qquad x_2'(0) = 0$

Converted to system as follows, $u_1 = x_1$, $u_2 = x_1'$,
$u_3 = x_2$, $u_4 = x_2'$

$$u_1' = u_2$$

$$u_2' = -\frac{s_1}{m_1} u_1 + \frac{s_2}{m_1}(u_3 - u_1)$$

$$u_3 = u_4$$

$$u_4 = -\frac{s_2}{m_2}(u_3 - u_1)$$

$$\vec{u}(0) = \begin{bmatrix} 1/2 \\ 0 \\ 1/4 \\ 0 \end{bmatrix}$$

Lorenz Equation.

$$\dot{x}_1(t) = -\beta x_1(t) + x_2(t)\, x_3(t)$$
$$\dot{x}_2(t) = -\rho x_2(t) + \rho x_3(t)$$
$$\dot{x}_3(t) = -x_1(t)\, x_2(t) + \sigma x_2(t) - x_3(t)$$

where $\beta = 8/3$, $\rho = 10$, $\sigma = 28$. Let the initial state be $x_1(0) = x_2(0) = 0$, $x_3(0) = \epsilon$ (small; $\epsilon = 10^{-10}$).

Van der Pol Equation

$$\ddot{y} + \mu(y^2 - 1)\dot{y} + y = 0$$
$$y(0) = -0.2\,, \quad \dot{y}(0) = -0.7$$

μ is a parameter such as $\mu = 1$ or $\mu = 2$. (then take $\mu = 1000$)

Satellite Equation

$$\ddot{x} = 2\dot{y} + x - \frac{\mu^*(x+\mu)}{r_1^3} - \frac{\mu(x-\mu^*)}{r_2^3},$$

$$\ddot{y} = -2\dot{x} + y - \frac{\mu^* y}{r_1^3} - \frac{\mu y}{r_2^3}$$

where $\mu = \dfrac{1}{82.45}$, $\mu^* = 1 - \mu$, $r_1 = \sqrt{(x+\mu)^2 + y^2}$
$r_2 = \sqrt{(x-\mu^*)^2 + y^2}$. Let the initial conditions be

$$x(0) = 1.2 \qquad y(0) = 0$$
$$\dot{x}(0) = 0 \qquad \dot{y}(0) = -1.04935751.$$

Stiff Equations

$$\dot{y} = \begin{bmatrix} -21 & 19 & -20 \\ 19 & -21 & 20 \\ 40 & -40 & -40 \end{bmatrix} y, \quad y_0 = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}$$

with the exact solution

$$y(t) = \begin{bmatrix} 0.5\, e^{-2t} + 0.5\, e^{-40t}(\cos 40t + \sin 40t) \\ 0.5 e^{-2t} - 0.5\, e^{-40t}(\cos 40t + \sin 40t) \\ e^{-40t}(\sin 40t - \cos 40t) \end{bmatrix}$$

Implicit Equations

$$\begin{cases} (\sin x_1)\, \dot{x}_1 + (\cos x_2)\, \dot{x}_2 + x_1 = 1 \\ (-\cos x_2)\, \dot{x}_1 + (\sin x_1)\, \dot{x}_2 + x_2 = 0 \end{cases}$$

Equivalently, for $\vec{x} = [x_1, x_2]^T$,

$$A(\vec{x})\, \dot{\vec{x}} = B(\vec{x}), \quad A(\vec{x}) = \begin{bmatrix} \sin x_1 & \cos x_2 \\ -\cos x_2 & \sin x_1 \end{bmatrix} \text{ is non-singular}$$

OR $\dot{\vec{x}} = A^{-1}(\vec{x})\, B(\vec{x})$ $\qquad B(\vec{x}) = \begin{bmatrix} 1 - x_1 \\ -x_2 \end{bmatrix}$

OR Really Implicit Equation

$$\begin{cases} \ddot{x} \sin \dot{y} + \ddot{y}^2 = -2xy\, e^{-\dot{x}} + x \ddot{x} \dot{y} \\ x \ddot{x} \ddot{y} + \cos \ddot{y} = 3y \dot{x} e^{-x} \end{cases}$$

Equivalently, letting $x_1 = x$, $x_2 = \dot{x}$, $x_3 = y$, $x_4 = \dot{y}$ and denoting $p_1 = \ddot{x}$ and $p_2 = \ddot{y}$ we have

$$\begin{cases} p_1 \sin x_4 + p_2^2 + 2x_1 x_3 e^{-x_2} - x_1 p_1 x_4 = 0 \\ x_1 p_1 p_2 + \cos p_2 - 3 x_3 x_2 e^{-x_1} = 0. \end{cases}$$

Let the initial state be $[\underset{x}{\underset{\parallel}{x_1}}, \underset{\dot{x}}{\underset{\parallel}{x_2}}, \underset{y}{\underset{\parallel}{x_3}}, \underset{\dot{y}}{\underset{\parallel}{x_4}}]\Big|_{t=0} = [1, 0, 0, 1]$