Numerical Methods for Ordinary Differential Equations (ODE)

Introduction

In this course, we focus on the following general **initial-value problem (IVP)** for a first order ODE:

$$\begin{cases} x' = f(t,x) \\ x(a) = x_0 \end{cases} \qquad \text{or} \qquad \begin{cases} \frac{dx(t)}{dt} = f(t,x(t)) \\ x(a) = x_0 \end{cases}$$

In many applications, the closed-form solution for the above IVP may be very complicated and difficult to evaluate or there is no closed-form solution. So we want a numerical solution. A computer code for solving an ODE produces a sequence of points (t_i, x_i), $i = 0, 1, \ldots, n$ where x_i is an approximation to the true value $x(t_i)$, while mathematical solution is a continuous function $x(t)$.

Q: Suppose you have obtained those (t_i, x_i). Now you want to obtain an approximate value of $x(t)$ for some t which is within the interval $[t_0, t_n]$ but is not equal to any t_i, what can you do?

The Euler method

We would like to find approximate values of the solution to the IVP over the interval $[a, b]$. Use $n+1$ points $t_0, t_1, \ldots, t_n$ to equally partition $[a, b]$. $h = t_{i+1} - t_i = (b-a)/n$ is called the **size step**. Suppose we have already obtained x_i, an approximation to $x(t_i)$. We would like to get x_{i+1}, an approximation to $x(t_{i+1})$. The Taylor series expansion

$$x(t_{i+1}) \approx x(t_i) + (t_{i+1} - t_i)x'(t_i) = x(t_i) + hf(t_i, x(t_i))$$

leads to the Euler method

$$x_{i+1} = x_i + hf(t_i, x_i), \qquad i = 0, 1, \ldots, n-1.$$

Q: Derive the Euler method by the rectangle rule for integration.

Algorithm for the Euler method (given f, a, b, x_0, n).

$h \leftarrow (b-a)/n$
$t_0 \leftarrow a$
for $i = 0 : n-1$
    $x_{i+1} \leftarrow x_i + hf(t_i, x_i)$
    $t_{i+1} \leftarrow t_i + h$
end

Note: In the Euler method, we chose a constant step size h. But it may be more efficient to choose a different step size h_i at each point t_i based on the properties of $f(t, x)$. An adaptive method can be developed.

Example: Use the Euler method to solve $\begin{cases} x' = x \\ x(0) = 1 \end{cases}$ over $[0, 4]$ with $n = 20$. What did you observe? How do you explain what you observed?

Errors for the Euler method

The Taylor series expansion gives

$$x(t_{i+1}) = x(t_i) + hf(t_i, x(t_i)) + \frac{1}{2}h^2 x''(z_{i+1}), \qquad z_{i+1} \in [t_i, t_{i+1}]. \tag{1}$$

the Euler method gives

$$x_{i+1} = x_i + hf(t_i, x_i). \tag{2}$$

From (1) and (2)

$$x(t_{i+1}) - x_{i+1} = x(t_i) - x_i + h[f(t_i, x(t_i)) - f(t_i, x_i)] + \frac{1}{2}h^2 x''(z_{i+1}).$$

$x(t_{i+1}) - x_{i+1}$ is the error at t_{i+1}. This is called the **global error** at t_{i+1}. It arises from two sources:

1. the **local truncation error**: $\frac{1}{2}h^2 x''(z_{i+1})$. Notice if $x_i = x(t_i)$, then the local truncation error at t_{i+1} is just the global error at t_{i+1}.

2. the **propagation error**: $x(t_i) - x_i + h[f(t_i, x(t_i)) - f(t_i, x_i)]$. This is due to the accumulated effects of all local truncation errors at $t_1, t_2, \ldots, t_i$.

When we perform the computation on a computer with finite precision, there is an additional source of errors: **the rounding error**.

Note: There are a few techniques to determine the step size h according to error analysis results.

The trapezoid-Euler method

$$\begin{cases} \hat{x}_{i+1} = x_i + hf(t_i, x_i), \\ x_{i+1} = x_i + \frac{1}{2}h[f(t_i, x_i) + f(t_{i+1}, \hat{x}_{i+1})] \\ t_i = a + ih \end{cases}$$

In the literature, this is also called the improved Euler's method or Heun's method.

The midpoint-Euler method

$$\begin{cases} x_{i+1/2} = x_i + \frac{1}{2}hf(t_i, x_i), \\ x_{i+1} = x_i + hf(t_i + \frac{1}{2}h, x_{i+1/2}) \\ t_i = a + ih \end{cases}$$

General Taylor series methods

Taylor series expansion gives

$$x(t_{i+1}) \approx x(t_i) + hx'(x_i) + \frac{1}{2!}h^2 x''(t_i) + \cdots + \frac{1}{m!}h^m x^{(m)}(t_i)$$

From $x' = f(t, x)$, we can compute $x'', \ldots, x^{(m)}$. Define $x'_i, x''_i, \ldots, x_i^{(m)}$ as approximations to $x'(t_i), x''(t_i), \ldots, x^{(m)}(t_i)$, respectively. Then we have the Taylor series method of order m:

$$x_{i+1} = x_i + hx'_i + \frac{1}{2!}h^2 x''_i + \cdots + \frac{1}{m!}h^m x_i^{(m)}, \quad x_0 \text{ is known.}$$

Remarks:

1. The Euler method is a Taylor series method of order 1.

2. If $f(t, x)$ is complicated, then high-order Taylor series methods may be very complicated.

Runge-Kutta methods of order 2

$$x_{i+1} = x_i + (1 - \frac{1}{2\alpha})K_1 + \frac{1}{2\alpha}K_2$$

where

$$\begin{aligned} &K_1 = hf(t_i, x_i), \\ &K_2 = hf(t_i + \alpha h, x_i + \alpha K_1), \\ &\alpha \neq 0. \end{aligned}$$

When $\alpha = 1$, we obtain the trapezioid-Euler method, and when $\alpha = 1/2$, we obtain the midpoint-Euler method. The local truncation error of Runge-Kutta methods of order 2 is $O(h^3)$.

Runge-Kutta methods of order 4

The classical fourth-order Runge-Kutta method

$$x_{i+1} = x_i + \frac{1}{6}(K_1 + 2K_2 + 2K_3 + K_4)$$

where

$$\begin{aligned} K_1 &= hf(t_i, x_i), \\ K_2 &= hf(t_i + \frac{1}{2}h, x_i + \frac{1}{2}K_1), \\ K_3 &= hf(t_i + \frac{1}{2}h, x_i + \frac{1}{2}K_2), \\ K_4 &= hf(t_i + h, x_i + K_3). \end{aligned}$$

This method is in common use for solving IVPs. The local truncation error of Runge-Kutta methods of order 4 is $O(h^5)$.

MATLAB tools

1. `ode23`: based on a pair of 2nd and 3rd-order Runge-Kutta methods.
2. `ode45`: based on a pair of 4th and 5th-order Runge-Kutta methods.