Numerical Methods for Ordinary Differential Equations (ODE)

(Reading: Cheney and Kincaid, Sections 7.1 & 7.2)

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 numerical algorithm for solving an ODE produces a sequence of points (t_i, x_i), $i = 0, 1, \ldots,$ where x_i is an approximation to the true value $x(t_i)$, while the 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 a given interval but is not equal to any t_i, what can you do?

Euler's 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 **step size**. 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

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

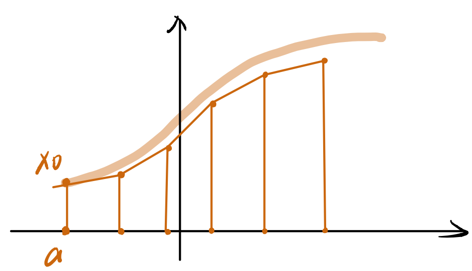
leads to the formula of Euler's method:

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

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

Algorithm for Euler's 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 Euler's 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 Euler's method to solve $\begin{cases} x' = x \\ x(0) = 1 \end{cases}$ over $[0, 4]$ with $n = 20$. What do you observe? How do you explain what you have observed? Error will grow gradually.

Errors for Euler's method

By Taylor's theorem

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

Euler's 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 such that the global error at any point is bounded by a given tolerance.

Trapezoidal Euler's method

From $x'(t) = f(t, x(t))$, we have

$$\int_{t_i}^{t_{i+1}} x'(t)dt = \int_{t_i}^{t_{i+1}} f(t, x(t))dt.$$

Applying the trapezoid rule to the right hand side, we obtain

$$x(t_{i+1}) - x(t_i) \approx \frac{1}{2}h[f(t_i, x(t_I)) + f(t_i, x(t_{i+1}))].$$

This leads to the scheme

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

But this cannot work, because the right hand side involves x_{i+1}. To overcome the difficulty, we use Euler's method to compute x_{i+1} on the right hand side, leading to the formula of the **trapezoidal Euler's method**:

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

In the literature, this method is called the **improved Euler's method** or **Heun's method**. The local truncation error of this method is $O(h^3)$. not testable

Midpoint Euler's method

In deriving the trapezoidal Euler's method, if we use the midpoint rule instead of the trapezoid rule for integration we can obtain the formula of the **midpoint Euler's method**:

$$\begin{cases} \hat{x}_{i+1/2} = x_i + \frac{1}{2}hf(t_i, x_i), \\ x_{i+1} = x_i + hf(t_i + \frac{1}{2}h, \hat{x}_{i+1/2}). \end{cases}$$

The local truncation error is $O(h^3)$. not testable

only read

General Taylor series methods

Taylor series expansion gives

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

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:

$$x_{i+1} = x_i + hx_i' + \frac{1}{2!}h^2 x_i'' + \cdots + \frac{1}{m!}h^m x_i^{(m)}.$$

An example of using the method will be given in class.

e.g. $\begin{cases} x'(t) = x \\ x(0) = 1 \end{cases}$

$\Rightarrow x'(t) = x''(t) = \cdots x^{(m)}(t)$

Notes:

1. Euler's 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

Write

$$x_{i+1} = x_i + w_1K_1 + w_2K_2,$$

where

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

We want to choose w_1, w_2, α and β so that x_{i+1} is close to $x(t_{i+1})$ as much as possible.

Since $x'(t) = f(t, x(t))$,

$$x''(t) = \frac{\partial f(t, x(t))}{\partial t} + \frac{\partial f(t, x(t))}{\partial x}x'(t) = \frac{\partial f(t, x(t))}{\partial t} + \frac{\partial f(t, x(t))}{\partial x}f(t, x(t)).$$

Then by Taylor's theorem we have

$$\begin{aligned} x(t_{i+1}) &= x(t_i) + x'(t_i)h + \frac{1}{2}x''(t_i)h^2 + O(h^3) \\ &= x(t_i) + f(t_i, x(t_i))h + \frac{1}{2}\left[\frac{\partial f(t_i, x(t_i))}{\partial t} + \frac{\partial f(t_i, x(t_i))}{\partial x}f(t_i, x(t_i))\right]h^2 + O(h^3). \qquad (3) \end{aligned}$$

On the other hand, by Taylor's theorem for two variables, we have

$$f(t_i+\alpha h, x_i+\beta K_1) = f(t_i,x_i) + \frac{\partial f(t_i,x_i)}{\partial t}\alpha h + \frac{\partial f(t_i,x_i)}{\partial x}\beta h f(t_i,x_i) + O(h^2).$$

Then

$$\begin{aligned} x_{i+1} &= x_i + w_1 K_1 + w_2 K_2 \\ &= x_i + w_1 h f(t_i,x_i) + w_2 h f(t_i,x_i) + \frac{\partial f(t_i,x_i)}{\partial t} w_2 \alpha h^2 + \frac{\partial f(t_i,x_i)}{\partial x} w_2 \beta h^2 f(t_i,x_i) + O(h^3) \\ &= x_i + (w_1+w_2) f(t_i,x_i) h + \left[w_2 \alpha \frac{\partial f(t_i,x_i)}{\partial t} + w_2 \beta \frac{\partial f(t_i,x_i)}{\partial x} f(t_i,x_i) \right] h^2 + O(h^3). \qquad (4) \end{aligned}$$

We would like the absolute value of the local truncation error (i.e., $|x_{i+1} - x(t_{i+1})|$ when $x_i = x(t_i)$) to be as small as possible. To achieve this goal, we would like x_{i+1} and $x(t_{i+1})$ to have common terms as many as possible when $x_i = x(t_i)$. By comparing (3) and (4), we set

$$w_1 + w_2 = 1, \quad w_2\alpha = 1/2, \quad w_2\beta = 1/2.$$

Thus,

$$w_1 = 1 - \frac{1}{2\alpha}, \quad w_2 = \frac{1}{2\alpha}, \quad \beta = \alpha.$$

Then we obtain the formula of **a class of 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

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

Here α can be any nonzero parameter. Notice that if $x_i = x(t_i)$, then local truncation error $x(t_{i+1}) - x_{i+1} = O(h^3)$.

Note that when $\alpha = 1$, we obtain the trapezioidal Euler's method, and when $\alpha = 1/2$, we obtain the midpoint Euler's method.

Runge-Kutta method of order 4

The formula of the classical **Runge-Kutta method of order 4**:

$$x_{i+1} = x_i + \frac{1}{6}(K_1 + 2K_2 + 2K_3 + K_4), \quad t_{i+1} = t_i + h,$$

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 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.