Lecture 13
Ordinary Differential Equations

Ying-Jer Kao

Department of Physics
National Taiwan University

June 16, 2011

Outline

Introduction

- An ordinary differential equation (ODE) is an equation that involves one or more derivatives of an unknown function.
- A solution of a differential equation is a specific function that satisfies the equation.

$$\begin{aligned} x' - x &= e^t & \qquad x(t) &= te^t + ce^t \\ x'' + 9x &= e^t & \qquad x(t) &= c_1 \sin 3t + c_2 \cos 3t \\ x' + \frac{1}{2x} &= 0 & \qquad x(t) &= \sqrt{c - t} \end{aligned}$$

- The constants c, c_1, c_2 are determined by the initial conditions.

Initial-Value Problem

- We focus on the initial-value problem for a first-order ODE,

$$x' = f(x(t), t); \quad x(a) = s$$

- A numerical solution of a differential equation is usually obtained in the form of a table; the functional form of the solution remains unknown.
- Solving ordinary differential equations numerically requires a large number of steps with small step size, so a significant amount of roundoff error can accumulate.

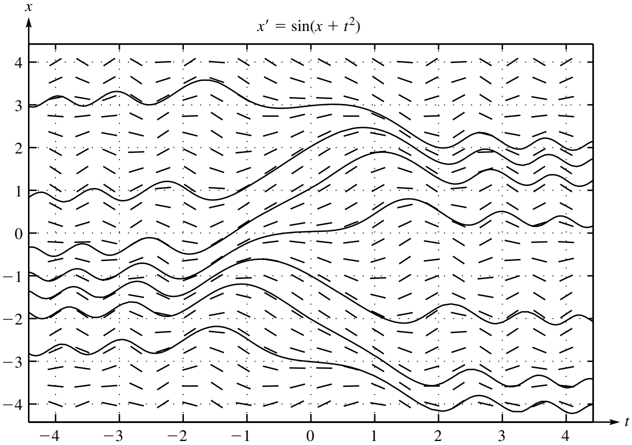
Vector Fields

- Consider a generic first-order differential equation,

$$\begin{cases} \dfrac{dx}{dt} = f(x,t) \\ x(a) = s \end{cases}$$

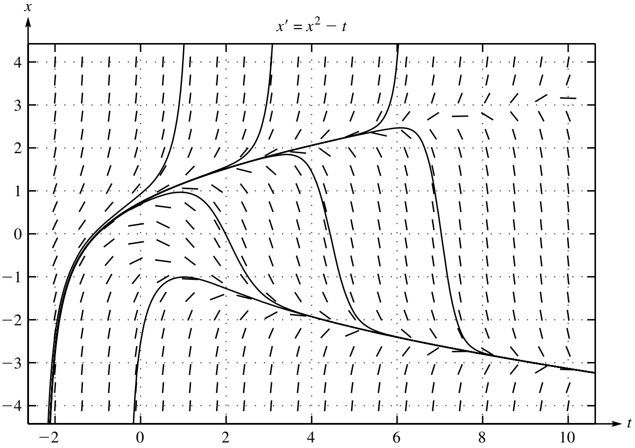
- To get some intuition, we first visualize the vector field of the equation.
- In the tx-plane, at every point $f(x,t)$ is defined, we plot a line segment having the slope $x' = f(x,t)$.
- We hope to understand how the solution function $x(t)$ evolves through these line segments while keeping its slope to the slope of the line segment drawn at that point.

Vector Fields



$$x' = \sin(x + t^2)$$

Vector Fields



$x' = x^2 - t$, very sensitive to the initial condition!!

Solving Differential Equations and Integration

- Consider a generic first-order ODE: $\frac{dx}{dr} = f(x(r), r)$.
- Integrating from t to $t+h$,

$$\int_t^{t+h} dx = \int_t^{t+h} f(x(r), r) dr$$
$$x(t+h) = x(t) + \int_t^{t+h} f(x(r), r) dr$$

- Replacing the integral with one of the numerical integration rules, we obtain a formula for solving the differential equation.
- For example, using the trapezoidal rule, we obtain the implicit formula,

$$x(t+h) = x(t) + \frac{h}{2}\left[f(x(t), t) + f(x(t+h), t+h)\right]$$

Taylor Series Methods

- Assume the solution function $x(t)$ can be represented by a Taylor series

$$x(t+h) = x(t) + hx'(t) + \frac{h^2}{2!}x''(t) + \frac{h^3}{3!}x'''(t) + \frac{h^4}{4!}x^{(4)}(t) + \cdots + \frac{h^m}{m!}x^{(m)}(t) + \cdots$$

- The Taylor series truncated after $m+1$ terms enables us to compute $x(t+h)$ rather accurately if h is small and if $x(t), x'(t), x''(t), ..., x^{(m)}(t)$ are known. This is called the Taylor series method of order m.
- The methods based on Taylor series are the most general, and it is capable of high precision.

Euler's Method

- The Taylor series method of order one ($m = 1$) is called the Euler's method.
- To find the solution of the initial-value problem over the interval $[a, b]$,

$$\frac{dx}{dt} = f(x, t); \quad x(a) = x_a$$

- Using $x(t + h) \approx x(t) + hx'(t)$, we obtain

$$x(t + h) = x(t) + hf(x(t), t)$$

  which can be used to step from $t = a$ to $t = b$ with n steps of size $h = (b - a)/n$.
- Truncation error: $O(h^2)$.

Euler's Method: Pseudo Code

Euler's Method

integer: k, **real**:h, t
$n \leftarrow 100$
external function f
$a \leftarrow 1, b \leftarrow 2, x \leftarrow -4$
$h \leftarrow (b-a)/n$
$t \leftarrow a$
output $0, t, x$
for $k \leftarrow 1, n$ **do**
  $x \leftarrow x + hf(x, t)$
  $t \leftarrow t + h$
  output k, t, x
end for

Taylor Series Method of Order 4

- Consider the initial value problem,

$$\begin{cases} x' = 1 + x^2 + t^3 \\ x(1) = -4 \end{cases}$$

- Differentiation of the equation several times, we obtain

$$\begin{aligned} x' &= 1 + x^2 + t^3 \\ x'' &= 2xx' + 3t^2 \\ x''' &= 2xx'' + 2(x')^2 + 6t \\ x^{(4)} &= 2xx''' + 6x'x'' + 6 \end{aligned}$$

Taylor Series Method of Order 4

- If numerical values of t and $x(t)$ are known, these four formulas, applied in order, yield $x'(t), x''(t), x'''(t)$, and $x^{(4)}(t)$.
- We can use the first five terms in the Taylor series

$$x(t+h) = x(t)+h\left[x'(t) + \frac{1}{2}h\left[x''(t) + \frac{1}{3}h\left[x'''(t) + \frac{1}{4}hx^{(4)}(t)\right]\right]\right]$$

Taylor Series Method of Order 4: Pseudo Code

Taylor Series Method of Order 4

integer: k, **real**:$h, t, x, x', x'', x''', x^{(4)}$
$n \leftarrow 100$
external function f
$a \leftarrow 1, b \leftarrow 2, x \leftarrow -4$
$h \leftarrow (b-a)/n$
$t \leftarrow a$
output $0, t, x$
for $k \leftarrow 1, n$ **do**
  $x' \leftarrow 1 + x^2 + t^3$
  $x'' \leftarrow 2xx' + 3t^2$
  $x''' = 2xx'' + 2(x')^2 + 6t$
  $x^{(4)} = 2xx''' + 6x'x'' + 6$
  $x \leftarrow x + h\left[x' + \frac{1}{2}h\left[x'' + \frac{1}{3}h\left[x''' + \frac{1}{4}hx^{(4)}\right]\right]\right]$
  $t \leftarrow t + h$
  output k, t, x
end for

Error Estimate

- At each step, if $x(t)$ is known and $x(t+h)$ is computed from the first few terms of the Taylor series, an error occurs due to truncation. This is the local truncation error.
- Euler method has truncation error of $O(h^2)$. Taylor series method of order 4 has the truncation error of $O(h^5)$.
- Another type of error is due to the accumulated effects of all local truncation errors.
- The calculated value of $x(t+h)$ is in error because
  - $x(t)$ is in error due to previous truncation errors,
  - local truncation error occurs in the computation of $x(t+h)$.
- Roundoff error may not be serious in a single step of the solution process. After hundreds or thousands of steps, though, it may accumulate and seriously contaminate the calculated solution.

Runge-Kutta Methods

- In Taylor series methods, we need to compute $x'', x''', \ldots$ by differentiating the function f.
- This requires pre-computation of these derivatives analytically before coding.
- Ideally, a method should involve only the evaluation of f.
- The Runge-Kutta methods are designed to imitate the Taylor series method without requiring analytic differentiation of the original differential equation.

Runge-Kutta Method of Order 2

- Consider the functions K_1, K_2,

$$K_1 = hf(t,x), \quad K_2 = hf(t+\alpha h, x+\beta K_1),$$

  and the value of x at $t+h$

$$\begin{aligned} x(t+h) &= x(t) + w_1 K_1 + w_2 K_2 \\ &= x(t) + w_1 hf(t,x) + w_2 hf(t+\alpha h, x+\beta hf(t,x)) \end{aligned}$$

- We want to determine w_1, w_2, α, β so the solution is accurate, i.e., reproducing as many terms as possible in Taylor series,

$$x(t+h) = x(t) + hx'(t) + \frac{h^2}{2!}x''(t) + \frac{h^3}{3!}x'''(t) + \cdot$$

- By the choice of $w_1 = 1$ and $w_2 = 0$ reproduces Euler's method, which agrees to the order h.

Runge-Kutta Method of Order 2

- Expand $f(t+\alpha h, x+\beta hf)$ near (t,x)

$$f(t+\alpha h, x+\beta hf) = f+\alpha hf_t+\beta hff_x+\frac{1}{2}\left(\alpha h\frac{\partial}{\partial t}+\beta hf\frac{\partial}{\partial x}\right)^2 f(\bar{t},\bar{x})$$

- The solution $x(t+h)$ is

$$x(t+h) = x(t) + (w_1+w_2)hf + \alpha w_2 h^2 f_t + \beta w_2 h^2 ff_x + O(h^3)$$

- Taylor series up to h^2 order, rewrite $x'' = f_t + f_x f$ using $x' = f$,

$$x(t+h) = x + hf + \frac{1}{2}h^2 f_t + \frac{1}{2}h^2 ff_x + O(h^3)$$

- The equations for w_1, w_2, α, β are

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

Runge-Kutta Method of Order 2

- A solution (Heun's method) is

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

  and we obtain second-order Runge-Kutta method,

$$x(t+h) = x(t) + \frac{h}{2} f(t,x) + \frac{h}{2} f(t+h, x+hf(t,x))$$

  The solution is computed with two evaluations of $f(t,x)$.
- Other solutions are possible. For arbitrary α,

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

- The error term for Runge-Kutta methods of order 2 is

$$\frac{h^3}{4}\left(\frac{2}{3} - \alpha\right)\left(\frac{\partial}{\partial t} + f\frac{\partial}{\partial x}\right)^2 f + \frac{h^3}{6} f_x \left(\frac{\partial}{\partial t} + f\frac{\partial}{\partial x}\right) f$$

  When $\alpha = \frac{2}{3}$, the first term vanishes. This is the Ralston's method.

Popular Choices

- We list some popular choices of the second-order Runge-Kutta methods,

$$\alpha = \frac{1}{2} \beta = \frac{1}{2} \quad w_1 = 0 w_2 = 1 \quad \text{Modified Euler's Method}$$
$$\alpha = 1 \beta = 1 \quad w_1 = \frac{1}{2} w_2 = \frac{1}{2} \quad \text{Heun's Method}$$
$$\alpha = \frac{2}{3} \beta = \frac{3}{4} \quad w_1 = \frac{1}{3} w_2 = \frac{2}{3} \quad \text{Ralston's Method}$$

- Second-order methods are not popular in applications since the error is only $O(h^3)$.

Runge-Kutta Method of Order 4

- The widely used algorithm is the fourth-order Runge-Kutta method:

$$x(t+h) = x(t) + \frac{1}{6}(K_1 + 2K_2 + 2K_3 + K_4)$$

where

$$K_1 = hf(t, x) \qquad K_2 = hf(t + \frac{1}{2}h, x + \frac{1}{2}K_1)$$

$$K_3 = hf(t + \frac{1}{2}h, x + \frac{1}{2}K_2) \qquad K_4 = hf(t + h, x + K_3)$$

- The solution $x(t+h)$ is obtained by four function evaluations of f.
- It agrees with the Taylor expansion up to and including the term in h^4.

Runge-Kutta Method of Order 4: Pseudo Code

Runge-Kutta Method of Order 4

procedure RK4(f, t, x, h, n)
    integer: j, n, **real**: $K_1, K_2, K_3, K_4, h, t, t_a, x$
    external function f
    $t_a \leftarrow t$
    output $0, t, x$
    for $j \leftarrow 1, n$ **do**
        $K_1 \leftarrow hf(t, x)$
        $K_2 \leftarrow hf(t + \frac{1}{2}h, x + \frac{1}{2}K_1)$
        $K_3 \leftarrow hf(t + \frac{1}{2}h, x + \frac{1}{2}K_2)$
        $K_4 \leftarrow hf(t + h, x + K_3)$
        $x \leftarrow x + \frac{1}{6}(K_1 + 2K_2 + 2K_3 + K_4)$
        $t \leftarrow t_a + jh$
        output j, t, x
    end for
end procedure

Adaptive Runge-Kutta Methods

- Determination of a suitable step size h can be a major issue in numerical integration.
- Large h can cause large truncation error, while small h makes the computation expensive.
- A constant step size may not be appropriate for the entire range of integration.
- Adpative methods estimate the truncation error at each integration step and automatically adjust the step size to keep the error within the prescribed tolerance.

Runge-Kutta-Fehlberg Method

- The Fehlberg method of order 4 is of Runge-Kutta type and uses these formulas:

$$x(t+h) = x(t) + \frac{25}{216}K_1 + \frac{1408}{2565}K_3 + \frac{2197}{4104}K_4 - \frac{1}{5}K_5$$

where

$$\begin{aligned}
K_1 &= hf(t, x) \\
K_2 &= hf\left(t + \frac{1}{4}h, x + \frac{1}{4}K_1\right) \\
K_3 &= hf\left(t + \frac{3}{8}h, x + \frac{3}{32}K_1 + \frac{3}{32}K_2\right) \\
K_4 &= hf\left(t + \frac{12}{13}h, x + \frac{1932}{2197}K_1 - \frac{7200}{2197}K_2 + \frac{7296}{2197}K_3\right) \\
K_5 &= hf\left(t + h, x + \frac{439}{216}K_1 - 8K_2 + \frac{3680}{513}K_3 - \frac{845}{4104}K_4\right)
\end{aligned}$$

Runge-Kutta-Fehlberg Method

- Since this scheme requires five function evaluation, one more than the classical Runge-Kutta method of order 4, it is of little value alone.
- However, with an additional function evaluation

$$K_6 = hf\left(t + \frac{1}{2}h, x - \frac{8}{27}K_1 + 2K_2 - \frac{3544}{2565}K_3 - \frac{1859}{4104}K_4 - \frac{11}{40}K_5\right)$$

we obtain the fifth-order Runge-Kutta method,

$$x(t+h) = x(t) + \frac{16}{135}K_1 + \frac{6656}{12825}K_3 + \frac{28561}{56430}K_4 - \frac{9}{50}K_5 + \frac{2}{55}K_6$$

RK45:Pseudo Code

RK45

procedure RK45(f, t, x, h, n, ϵ)
  integer: j, n, **real**:$K_1, K_2, K_3, K_4, K_5, k_6, h, t, x, x_4$
  external function f
  real $c_{20} \leftarrow 0.25, c_{21} \leftarrow 0.25, c_{30} \leftarrow 0.375, c_{31} \leftarrow 0.09375, c_{32} \leftarrow 0.28125$
  real $c_{40} \leftarrow 12./13., c_{41} \leftarrow 1932./2197., c_{42} \leftarrow -7200./2197., c_{43} \leftarrow 7296./2197.$
  real $c_{51} \leftarrow 439./216., c_{52} \leftarrow -8., c_{53} \leftarrow 3680./513., c_{54} \leftarrow -845./4104.$
  real $c_{60} \leftarrow 0.5, c_{61} \leftarrow -8./27., c_{62} \leftarrow 2., c_{63} \leftarrow -3544./2565., c_{64} \leftarrow 1859./4104.$
  real $c_{65} \leftarrow -0.275$
  real $a_1 \leftarrow 25./216., a_2 \leftarrow 0., a_3 \leftarrow 1408./2565.$
  real $a_4 \leftarrow 2197./4104., a_5 \leftarrow -0.2$
  real $b_1 \leftarrow 16./135., b2 \leftarrow 0., b3 \leftarrow 6656./12825.$
  real $b_4 \leftarrow 28561./56430., b_5 \leftarrow -0.18$
  real $b_6 \leftarrow 2./55.$
  $K_1 \leftarrow hf(t, x)$
  $K_2 \leftarrow hf(t + c_{20}h, x + c_{21}K_1)$
  $K_3 \leftarrow hf(t + c_{30}h, x + c_{31}K_1 + c_{32}K_2)$
  $K_4 \leftarrow hf(t + c_{40}h, x + c_{41}K_1 + c_{42}K_2 + c_{43}K_3)$
  $K_5 \leftarrow hf(t + h, x + c_{51}K_1 + c_{52}K_2 + c_{53}K_3 + c_{54}K_4)$
  $K_6 \leftarrow hf(t + c_{60}h, x + c_{61}K_1 + c_{62}K_2 + c_{63}K_3 + c_{64}K_4 + c_{65}K_5)$
  $x_4 \leftarrow x + a_1K_1 + a_3K_3 + a_4K_4 + a_5K_5$
  $x_5 \leftarrow x + b_1K_1 + b_3K_3 + b_4K_4 + b_5K_5 + b_6K_6$
  $t \leftarrow t + h$
  $\epsilon \leftarrow |x_5 - x_4|$
end procedure

Adaptive Method

- In the RK45 procedure, the fourth- and fifth-order approximations for $x(t+h)$, x_4 and x_5, are computed from six function evaluations and error estimate $\epsilon = |x_4 - x_5|$ is known.
- For given bounds of the error ($\epsilon_{\min} \leq \epsilon \leq \epsilon_{\max}$), the step size h is doubled or halved as needed to keep ϵ within these bounds.
- A range for the allowable step size h can also be specified ($h_{\min} \leq |h| \leq h_{\max}$).

Adaptive RKF Method: Pseudo Code

procedure RK45_ADAPTIVE($f, t, x, h, t_b, itmax, \epsilon_{\max}, \epsilon_{\min}, h_{\min}, h_{\max}, iflag$)
  integer $iflag, itmax, n$;**external function** f
  real $\epsilon, \epsilon_{\max}, \epsilon_{\min}, d, h, h_{\min}, h_{\max}, t, t_b, x, x_{\text{save}}, t_{\text{save}}$
  real $\delta \leftarrow \frac{1}{2} \times 10^{-5}$
  output $0, h, t, x$
  $iflag \leftarrow 1$
  $k \leftarrow 0$
  while $k \leq itmax$ **do**
    $k \leftarrow k + 1$
    if $|h| < h_{\min}$ **then** $h \leftarrow \text{sign}(h) h_{\min}$
    if $|h| > h_{\max}$ **then** $h \leftarrow \text{sign}(h) h_{\max}$
    $d \leftarrow |t_b - t|$
    if $d \leq |h|$ **then**
      $iflag \leftarrow 0$
      if $d \leq \delta \cdot \max(|t_b|, |t|)$ **then** exit loop
      $h \leftarrow \text{sign}(h) d$
    end if
    $xsave \leftarrow x$; $tsave \leftarrow t$
    call RK45(f, t, x, h, ϵ)
    output n, h, t, x, ϵ
    if $iflag = 0$ **then** exit loop
    if $\epsilon < \epsilon_{\min}$ **then** $h \leftarrow 2h$
    if $\epsilon > \epsilon_{\max}$ **then**
      $h \leftarrow h/2$; $x \leftarrow xsave$; $t \leftarrow tsave$;$k \leftarrow k - 1$
    end if
  end while
end procedure