數値分析

Chapter 5

Numerical Solution of Initial-Value Problems

- $\frac{dy}{dt} = f(t, y)$, $a \leq t \leq b$
  $y(a) = \alpha$
  $y(t) = ?$
  $dy = f(t, y)dt$
  目標在求 y, 而非 dy
  $y_{t+dt} = y_t + dy$

  > 跟股價之 process 很像, 只是除了 dt 之外, 還多了 dz
  > $\frac{ds}{s} = \mu dt + \sigma dz$
  > $ds = \mu s dt + \sigma s dz$

5.2 Taylor Methods

- Euler's Method (其實就是一階的 Taylor's method)
  $t_i = a + ih$, for $i = 0, 1, \ldots, N$
  $w_0 = \alpha, w_{i+1} = w_i + hf(t_i, w_i)$,
  with local error $\frac{1}{2}y''(\xi_i)h^2$ for $\xi_i \in [t_i, t_{i+1}]$

- P.180 Fig 5.2
  左圖眞値
  右圖逼近値 (逼近的誤差越來越大)

- P.180 Ex.1 Table 5.1
  local error 會不斷累積成 global error, 所以差距會隨 t 增加而變大
  global error $\approx$ local error$\times N = h^2 \times \frac{b-a}{h} = h(b-a)$

- Euler's Method Error Bound P.183 Example 2, Table 5.2

- 增加精準度的方法:
  1. 讓 h 變小
  2. Taylor's method (多 match 幾階微分)

$$y(t) = y(t_0) + hy'(t_0) + \frac{h^2}{2}y''(t_0) + \frac{h^3}{3!}y'''(t_0) + \cdots$$
$$= y(t_0) + h[\underbrace{y'(t_0)}_{f} + \frac{h}{2}\underbrace{y''(t_0)}_{f'} + \frac{h^2}{3!}\underbrace{y'''(t_0)}_{f''} + \cdots]$$

- Taylor Method of order n
  $w_0 = \alpha$
  $w_{i+1} = w_i + h \cdot T^{(n)}(t_i, w_i)$
  where $T^{(n)}(t_i, w_i) = f(t_i, w_i) + \frac{h}{2}f'(t_i, w_i) + \cdots + \frac{h^{n-1}}{n!}f^{(n-1)}(t_i, w_i)$
  The local error is $\frac{1}{(n+1)!}y^{(n+1)}h^{n+1}$ for some $\xi_i \in (t_i, t_{i+1})$

- P.185 Ex 3 Table 5.3 ($T^{(4)}$ 比 $T^{(2)}$更精準)

- 用線性內插法去算 $y(1.25)$
  取 $t = 1.2$ 和 $t = 1.4$
  (這兩個 $y(t)$ 誤差已經很小, 分別爲 0.0000225與 0.0000321)
  但是 $y(1.25)$ 的誤差變大 (0.0007525)
  (用兩個很接近的點去做內插, error 卻變大)

- 改用 cubic Hermite Interpolation (用兩端點的 f 與 f')
  用 Table 5.4 (使用 P.84 方法), 使 error 變小了 (0.0000286)
  因爲有微分值這個資訊時, 用 cubic Hermite interpolation, 會比一般的 linear Interpolation 來得好.

5.3 Runge-Kutta Methods

- 與 Taylor method 比較, 保持精確度, 但不用作高次微分

- $y(t_{i+1}) = y(t_i) + hy'(t_i) + \frac{h^2}{2}y''(t_i) + \frac{h^3}{3!}y'''(\xi)$
  $= y(t_1) + hf(t_i, y(t_i)) + \frac{h^2}{2}f'(t_i, y(t_i)) + \frac{h^3}{3!}y'''(\xi)$
  $\| f'(t_i, y(t_i)) = \frac{\partial f}{\partial t}(t_i, y(t_i)) + \frac{\partial f}{\partial y}(t_i, y(t_i))y'(t_i)$
  $= y(t_i) + h[f(t_i, y(t_i)) + \frac{h}{2}\frac{\partial f}{\partial t}(t_i, y(t_i)) + \frac{h}{2}\frac{\partial f}{\partial y}(t_i, y(t_i))f(t_i, y(t_i))] + \frac{h^3}{3!}y'''(\xi)$
  Two-dimension Taylor Expression
  [ ] 中很像 $f(t+h, y+h) = [f(t,y) + h\frac{\partial f}{\partial t} + k\frac{\partial f}{\partial y}] + \cdots,$
  所以利用2個變數之 Taylor Expression 來 approximate
  $$a_1 f(t_i + \alpha, y(t_i) + \beta) = a_1[f + \alpha\frac{\partial f}{\partial t} + \beta\frac{\partial f}{\partial y}]$$
  $$= a_1 f + a_1\alpha\frac{\partial f}{\partial t} + a_1\beta\frac{\partial f}{\partial y}$$
  比對後, 得 $a_1 = 1, \alpha = \frac{h}{2}, \beta = \frac{h}{2}f(t_i, y(t_i))$
  $\Rightarrow$ Midpoint Method
  $$w_{i+1} = w_i + h[f(t_i + \frac{h}{2}, w_i + \frac{h}{2}f(t_i, w_i))]$$
  , with local error $O(h^3)$
  (Runge-Kutta method of degree 2, 用 $t_i + \frac{h}{2}$ 之資訊來求 $y(t_i + h)$)

- if 用 $a_1 f(t,y) + a_2 f(t+\alpha, y + \beta f(t,y))$ 來 approximate, 得 $a_1 = a_2 = \frac{1}{2}$, $\alpha = \beta = h$

  Modified Euler Method
  $$w_{i+1} = w_i + \underbrace{\frac{h}{2}}_{\text{把兩個作平均}}[f(t_i, w_i) + \underbrace{f(t_{i+1}, w_i + hf(t_i, w_i))}_{\text{比 Euler 多這項}}]$$

** f 值被帶了三次

- Table 5.5 (三種方法的比較)
  Midpoint 的 error 最小, 所以常用
  (因爲誤差小, 算很快, 且 f 只用了兩次)

- Runge-Kutta Method of Order 4
  $w_0 = \alpha$
  $k_1 = hf(t_i, w_i)$
  $k_2 = hf(t_i + \frac{h}{2}, w_i + \frac{1}{2}k_1)$
  $k_3 = hf(t_i + \frac{h}{2}, w_i + \frac{1}{2}k_2)$
  $k_4 = hf(t_{i+1}, w_i + k_3)$
  $w_{i+1} = w_i + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$
  ** 先算 $k_1 \Rightarrow$再算 $k_2 \Rightarrow k_3 \Rightarrow k_4$
  (f 算4次 $\Rightarrow$ 精準度較高)

- Table 5.6 (跟 P.186 Table 5.3比較)
  比 4 階 Taylor's method 的 Error 還要大一點, but order 相同

- Table 5.7 (f 的計算次數)

order	2	3	4	5	6	7
f 的計算次數	2	3	4	6	?	?
local error	$O(h^3)$	$O(h^4)$	$O(h^5)$	$O(h^5)$	$O(h^6)$	$O(h^7)$

** f 用越多次, “不見得”會提高精準度, 因算 f 的次數多了, error 也跟著上升

** 所以寧可縮小 h, 也不要一直增加 order

- Table 5.8 (比較三種方法)

  Euler 用一次 f, Modified Euler 用二次 f, RK order 4 用四次 f, 即使 h 用的較大, Runge-Kutta order 4 還是最好

5.4 Predictor-Corrector Methods

- 之前的方法都是所謂的 one-step methods, 算 increment 時, 只跟前一點有關, 之前的資訊都沒用, 亦即從 $t_i \to t_{i+1}$, 只考慮 $[t_i, t_{i+1}]$ 的資訊

- 現在從 $t_i \to t_{i+1}$ 考慮 $[t_0 \cdots t_i, t_{i+1}]$ 的資訊

- $y(t_{i+1}) - y(t_i) = \int_{t_i}^{t_{i+1}} y'(t)dt = f * (t_{i+1} - t_i)$ (以前的算法)

  $= \int_{t_i}^{t_{i+1}} f dt$

  (用 $t_0, t_1, \ldots, t_i$ 的資訊去求 polynomial $p(t) \to f$)

- Adams-Bashforth two-step Explicit Method

  $w_0 = \alpha$, $w_1 = \alpha_1$

  (α_1 用 one step 之 RK 導出 (error h^3), 或用簡單的 Euler 導出 (error h^2))

  $w_{i+1} = w_i + \frac{h}{2}(3f(t_i, w_i) - f(t_{i-1}, w_{i-1}))$

  where $i = 1, 2, \ldots, N-1$, with local error $\frac{5}{12}y'''(\mu_i)h^3$ for some μ_i in (t_{i-1}, t_{i+1})

  ** 因爲考慮範圍比較大, 所以 local error 的範圍會比較大 ($\mu_i \in (t_{i-1}, t_{i+1})$)

- Implicit: 用全部的資訊 (比較準), $0 \sim t_{i+1}$

  Explicit: 用部分資訊, $0 \sim t_i$

- Adams-Moulton Two-Step Implicit Method
  (f 用了三次) [Explicit 的 f 只用了兩次]
  $w_0 = \alpha$, $w_1 = \alpha_1$
  $w_{i+1} = w_i + \frac{h}{12}(5f(t_{i+1}, w_{i+1}) + 8f(t_i, w_i) - f(t_{i-1}, w_{i-1}))$
  w_{i+1} 同時在等號的左右邊
  $\Rightarrow$不容易求解

- Ex1. 用 Explicit four-step 跟 Implicit three-step做比較
  (同樣都用 4 次 f)
  Table 5.9 Implicit 的 error 比 Explicit 的 error 小很多
  (but比較複雜, 尤其要解出 w_{i+1} 很困難)

- Predictor-corrector method 結合 Euler 與 Implicit 一起用
  先把 Explicit 的 w_4 當 predictor 算出, 再帶入 Implicit 當 corrector$\Rightarrow$一直算$\Rightarrow$求得 stationary point
  ** 一般來說會比 Implicit 好, 如 P.203 Table 5.1e, 但其實未必

- 未必要從 w_i 出發
  Milne's Method (Explicit 法): 從 w_{i-3} 出發
  Simpson's Method (Implicit 法)
  用 Milne's Method + Simpson's Method 作 predictor-corrector, 效果比用 Adams-Bashforth + Adams-Moulton 的效果來得好

5.5 Extrapolation Methods

- 此節之觀念爲先將 $y(a+h)$ 估的很準, 再估 $y(a+2h)$

- endpoint corretion
  $w_{i-1} = w_i - hf(t_i, w_i)$
  $w_{i+1} = w_i + hf(t_i, w_i)$
  相加除以 2
  $\Rightarrow w_i = \frac{w_{i-1}+w_{i+1}}{2} + O(h^2) + O(h^4) + \cdots$
  因 h, h^3, h^5, $h^7, \cdots$ 都互相消去了

- $w_{i+1} = w_{i-1} + 2hf(t_i, w_i)$ (Midpoint Method)

- $h_0 = \frac{h}{2}$,
  且 $y_{i,j}$ 中 i 表分成 $2i$ 等分, j表第 j 次 approximation.
  $\Rightarrow w_1 = w_0 + h_0 f(a, w_0)$, $w_2 = w_0 + 2h_0 f(a + h_0, w_1)$
  $y_{1,1} = \frac{1}{2}(w_1 + w_3) = \frac{1}{2}(w_1 + w_2 + h_0 f(a + 2h_0, w_2))$
  (w_3 是由 w_2 作 Euler's Method)

- 取 $h_1 = \frac{h}{4}$
  $\Rightarrow w_1 = w_0 + h_1 f(a, w_0)$
  $w_2 = w_0 + 2h_1 f(a + h_1, w_1)$
  $w_3 = w_1 + 2h_1 f(a + 2h_1, w_2)$
  $w_4 = w_2 + 2h_1 f(a + 3h_1, w_3)$
  $y_{2,1} = \frac{1}{2}(w_3 + w_5) = \frac{1}{2}(w_3 + w_4 + h_1 f(a + 4h_1, w_4))$

- $y(a+h) = y_{1,1} + \delta_1(\frac{h}{2})^2 + \delta_2(\frac{h}{2})^4 \cdots (*)$
  $(h_0 = \frac{h}{2})$
  $y(a+h) = y_{2,1} + \delta_1(\frac{h}{4})^2 + \delta_2(\frac{h}{4})^4 \cdots (**)$
  $(h_1 = \frac{h}{4})$
  (*)-4(**)$\Rightarrow$ $y(a+h) = \underbrace{y_{2,1} + \frac{1}{3}(y_{2,1} - y_{1,1})}_{y_{2,2}} - \delta_2 \frac{h^4}{64}$

- P.208 Example 1, Table 5.11
  比較 P.210 Table 5.12 可知 $y(0.25)$ 的值
  $\Rightarrow$ $y_{5,5}$ 最精準

- 把兩個不準的 (有誤差的) 一起算$\Rightarrow$可消去某些 error 且把 Order 降一階

5.6 Adaptive Techniques

- given global error $\varepsilon \in O(h^n)$
  varying step size, (一次未必要跳 h, 而可跳 $q \cdot h$, $q < 1$) 使滿足 local error criterion $\in O(h^{n+1})$, 進而滿足 global error criterion ε

  n-th order Taylor Method
  $\Rightarrow \mid y(t_i) - w_i \mid < Kh^n$ (Global error)
  $(n+1)$-th order Taylor Method
  $\Rightarrow \mid y(t_i) - \widetilde{w}_i \mid < \widetilde{K}h^{n+1}$ (Global error)

- $z(t_i)$ 是假設之前的值都是對的, 所得之 $y(t_i)$ 的 approximation, 所以
  $\mid z(t_{i+1}) - w_{i+1} \mid$ 叫 local error
  $\mid y(t_{i+1}) - w_{i+1} \mid$ 叫 global error

P.213 Figure 5.3

- $z(t_i+h)-w_{i+1}$, 此項是 $O(h^{n+1})$
  $= \underbrace{\widetilde{w}_{i+1}-w_{i+1}}_{O(h^{n+1})} + \underbrace{z(t_i+h)-\widetilde{w}_{i+1}}_{O(h^{n+2})}$
  (因爲 $z(t_i+h)-\widetilde{w}_{i+1}$ 很小, 所以$\widetilde{w}_{i+1}-w_{i+1}$ 爲 $O(h^{n+1})$)
  $\Rightarrow \underbrace{z(t_i+h)-w_{i+1}}_{\text{眞的 local error}} \approx \underbrace{\widetilde{w}_{i+1}-w_{i+1}}_{\text{估計的 local error}}$
  $= Kh^{n+1}$ (global 跟 local 差一階)
  $\Rightarrow K \approx \frac{|\widetilde{w}_{i+1}-w_{i+1}|}{h^{n+1}}$
  (given h, 用 nth-order 與 $(n+1)$th-order Taylor, 可求出 K)
  回頭來看 global error
  $\Rightarrow | y(t_i+qh)-w_{i+1} | < Kq^nh^n = \frac{q^n|\widetilde{w}_{i+1}-w_{i+1}|}{h} < \varepsilon$
  $\Rightarrow q < (\frac{\varepsilon h}{|\widetilde{w}_{i+1}-w_{i+1}|})^{\frac{1}{n}}$ (選 q, 使得 global error $< \varepsilon$)
  ** 先算 $Kh^{n+1} = \frac{|\widetilde{w}_{i+1}-w_{i+1}|}{h}$看是否 $< \varepsilon$
  如果沒有 $< \varepsilon \Rightarrow$ 再算 q

- P.215 Example 1, P.216 Table 5.13

- 是否可以不要用 Taylor method, 而用 Runge-Kutta?
  P.216
  (5.4式) Adams-Bashforth Explicit Four-step method, with local error $\frac{251}{720}z^{(5)}(\widehat{\mu}_i)h^5$, where $\widehat{\mu}_i \in (t_{i-3}, t_i)$
  (5.5式) Adams-Moulton Implicit Three-step method, with local error $\frac{-19}{720}z^{(5)}(\widetilde{\mu}_i)h^5$, where $\widetilde{\mu}_i \in (t_{i-2}, t_{i+1})$
  如果 h 切割的夠小$\Rightarrow z^{(5)}(\widehat{\mu}_i) \approx z^{(5)}(\widetilde{\mu}_i)$
  $\Rightarrow$ 可得 $z^{(5)}(\widetilde{\mu_i}) \approx \frac{8}{3h^5}(w_{i+1}-w_{i+1}^{(0)})$

$$\Rightarrow\mid z(t_{i+1})-w_{i+1}\mid=\frac{19}{720}z^{(5)}(\tilde{\mu}_i)h^5$$
$$=\frac{19}{720}\frac{8}{3h^5}\mid w_{i+1}-w_{i+1}^{(0)}\mid h^5=\frac{19|w_{i+1}-w_{i+1}^{(0)}|}{270}$$
$$\Rightarrow \text{global error}=\frac{|z(t_{i+1})-w_{i+1}|}{h}=\frac{19|w_{i+1}-w_{i+1}^{(0)}|}{270h}$$

- 用 qh 之 global error $\frac{|z(t_i+qh)-\widehat{w}_{i+1}|}{qh}<\varepsilon$
  $$\Rightarrow \frac{19}{720}\mid z^{(5)}(\tilde{\mu}_i)\mid q^4h^4<\varepsilon$$
  $$\approx \frac{19}{720}[\frac{8}{3h^5}\mid w_{i+1}-w_{i+1}^{(0)}\mid q^4h^4<\varepsilon$$
  $$\Rightarrow q<(\frac{270}{19}\frac{h\varepsilon}{|w_{i+1}-w_{i+1}^{(0)}|})^{\frac{1}{4}}$$

- P.218 Table 5.14 因 multi-step, 還要回頭算前幾期 (用 qh 爲 step), 造成浪費, 所以若 q 接近 1, 則 ignore

5.7 Methods for Systems of Equations

- P.222 Rouge-Kutta of order 4, m 個 equations, 一起解

5.8 Stiff Differential Equations

- 可能 $f^{(n)}(\xi)$ 隨著 $n\uparrow$ 而上, 例如 e^{-ct}, 每次微分, c 會降下來, 使得 $f^{(n)}(\xi)$ 越來越大, error 也越來越大

- P.230 Example 1, P.231 Table 5.17, h 不夠小, 會爆掉
  此節在交 h 要取到何範圍內才沒問題

- $y'=\lambda y\Rightarrow y=e^{\lambda t}$
  $w_{i+1}=w_i+h(\lambda w_i)=(1+h\lambda)w_i=(1+h\lambda)^{i+1}w_0$
  Global error $=\mid y(t_j)-w_j\mid$
  $$=\mid e^{jh\lambda}-(1+h\lambda)^j\mid\mid\alpha\mid\ (w_0=\alpha)$$

if $\lambda < 0, j \to \infty \Rightarrow (e^{h\lambda})^j \to 0$

$\mid 1 + h\lambda \mid < 1$

$\Rightarrow h < \frac{2}{|\lambda|}$

(在 Example 1, $\mid \lambda \mid = 39 \Rightarrow h < \frac{2}{39} \approx 0.05$ 才會收斂)

- Implicit Trapezoidal Method
  (用 Newton's method 解 Implicit)

- P.233 Example 2, Table 5.18, 5.19
  (Implicit Trapezoidal Method 表現好)

-

	Local Error	Other Features
Euler's Methods	$O(h^2)$	Simple, intuitive, 但很少在實務上用
Taylor Methods	$O(h^{n+1})$	理論上可用, 但因 $y' = f(t, y)$ 是兩個變數之函數, 多階微分不可行
Runge-Kutta Methods	$\approx O(h^{n+1})$ P.195 Table 5.7	跟 Taylor 之精確度相同, 但只需用不同 (t_i, y_i) 之 f 作weighting, 而不用去計算 $f^{(n)}(t, y)$, 但因 f 要算的次數隨 $n \uparrow$ 而 $\uparrow$ $\Rightarrow$ local error $\uparrow$
Predictor Corrector Methods explicit	$O(h^{n+1})$	P-C methods 比 Implicit 好用, 但並非一定要比較小
Predictor Corrector Methods implicit	$O(h^{n+2})$	

  (explicit $n \sim$ implicit $n-1$)

  implicit 可能需要解 nonlinear function at each step

(所以結合 explicit 與 implicit)

Extrapolation Methods	先將 $y(a+h)$ 估計非常非常準 再以此為基礎算下一個估計值 $y(a+2h)$ 但是光算一個 $y(a+h)$ 就是人家估整個 $y(t)$ 之 effort
Adaptive Techniques	step size 可變, 當 local error 或 implied global error 太大時, 將 $n\downarrow$ 以滿足 given 之 global error