Visual Basic

CHAPTER 6

微分與積分

(Differentiation and Integration)

張 榮 興 博士

豐映科技股份有限公司

E-mail: chang.ronhsin@msa.hinet.net

微分與積分是最基礎的工程數學工具

函數關係若是 $f(x)$ 與 x 的數據要如何進行微分及積分

絕熱分批式反應器

在一個絕熱型分批式反應器中，進行液相反應 $A \rightarrow B$，其反應速率表示式爲

$$-r_A = k_0\ Exp(-E/RT)\ C_A{}^n$$

其中

$(-r_A)$ =反應速率[g-mole/ℓ-hr]，

Arrhenius反應速率常數$k_0 = 1.5\times 10^{11}\,[\frac{\ell}{\text{hr}}\cdot(\frac{\text{g}-\text{mole}}{\ell})^{1-n}]$，

T=反應溫度 [°K]，

C_A = A的濃度 [g-mole/ℓ]，

n =反應次數，n = 1.5；

活化能 E = 15300 cal/g-mole，

氣體定律常數 R = 1.947 cal/g-mole °K 。

開始反應時，溫度T_0 = 293 °K ，A的最初濃度爲C_{A0} = 0.2 g-mole/ℓ ，反應物比熱爲 C_p = 0.95 cal/g °K ，反應物密度ρ = 1,100 g/ℓ ，此反應的反應熱爲 (-⊿Hr) = 35,000 cal/g-mole。若反應在絕熱情況下進行，則其轉化率達到x_A時所需的時間爲 [2]：

$$\theta = C_{A0}\int_0^{x_A}\frac{dx_A}{-r_A} = C_{A0}\int_0^{x_A}\frac{dx_A}{k_0\ Exp(-E/RT)\ C_{A0}^n\ (1-x_A)^n}$$

反應物溫度與轉化率間的關係爲：

$$T = T_0 + \frac{(-\Delta H_r)\,C_{A0}}{\rho C_p}x_A$$

請利用上述方程式撰寫一計算機程式，建立轉化率與反應時間之關係。

函數的微分與積分是相當重要的數學運算，因此，工程學系的學生在修習微積分學時，都曾花了相當長的時間來學習這類技巧，包括各種函數的微分及積分方法，並學會許多嚴密的證明方法。但是，函數關係如果不是用方程式表示，而是一組 *f(x)* 與 *x* 的數據來表示，則欲求函數的微分及積分該如何進行呢？在本章中，我們將介紹這類的技巧，並設計所需的計算機程式。本章所介紹的技巧，在第八章以後各章節所探討的微分方程式解法中將極爲有用。

在稍後的介紹中，我們將發現數值積分相當容易執行，但是數值微分則需面對較多的困難及危險。這與學微積分時，所感受到微分相當容易，積分相當困難正好相反。

第一節　數值微分

Visual Basic

在本節中我們將討論利用數值方法計算一階及二階微分的最簡單最直接的方法。在第二節中再討論更一般化的數值微分方法。

假設在$x = x_0$處，*f(x)* 之導函數定義爲

$$f'(x_0) = \lim_{\Delta x \to 0} \frac{f(x_0 + \Delta x) - f(x_0)}{\Delta x} \qquad \textbf{(6-1.1)}$$

因此，若 Δx 取一相當小的值，則 $f'(x_0)$ 可利用下式得到相當好的近似值。

$$f'(x_0) \cong \frac{f(x_0 + \Delta x) - f(x_0)}{\Delta x} \qquad \textbf{(6-1.2)}$$

當然，Δx 增量可能爲正值或負值，因此，$f'(x_0)$ 亦可用以下的方程式作爲近似值。

$$f'(x_0) \cong \frac{f(x_0 - \Delta x) - f(x_0)}{-\Delta x} \qquad \textbf{(6-1.3)}$$

圖 6.1 中，割線L^+之斜率等於方程式 (6-1.2) 的計算值，割線L^-之斜率等於方程式 (6-1.3) 的計算值。而真正的導函數 $f'(x_0)$ 則等於曲線上 x_0 處之切線斜率。由圖很清楚地可看出當 $\Delta x \to 0$ 時，割線L^+及L^-均會趨近於 x_0 處之切線。但是由圖 6.1，我們可能會推想由P至Q作一直線，所得斜率可能較方程式 (6-1.2) 及方程式 (6-1.3) 更接近 $f'(x_0)$。

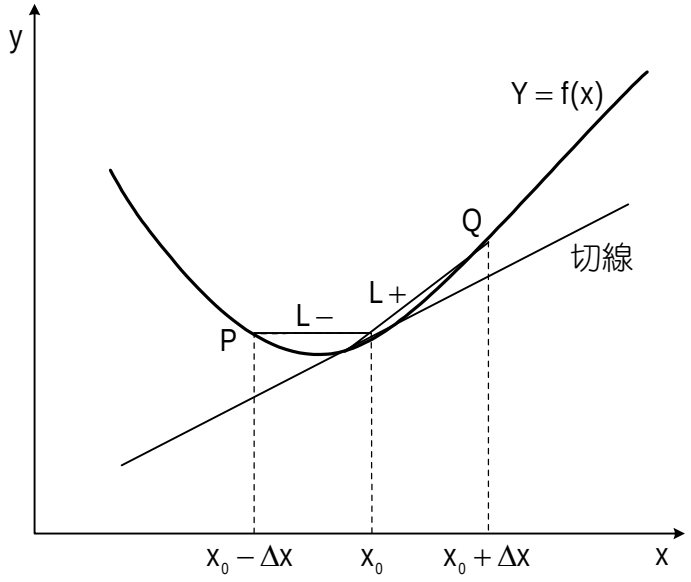


圖 6.1　函數 *f(x)* 的導函數，有兩種可能的近似值

通過 P 及 Q 的直線斜率爲

$$\frac{f(x_0+\Delta x)-f(x_0-\Delta x)}{2\,\Delta x} \qquad \textbf{(6-1.4)}$$

正好爲方程式 (6-1.2) 及方程式 (6-1.3) 所得斜率之平均值。爲了方便起見，令 $h = 2\Delta x$ ，則上式可寫成：

$$f'(x_0) \cong \frac{f(x_0+h/2)-f(x_0-h/2)}{h} \qquad \textbf{(6-1.5)}$$

我們利用此方程式作爲 $f'(x_0)$ 之近似值。

假設我們希望計算 *f(x)*在 $x = x_0$ 處的二次導函數。則由方程式 (6-1.5)，將 *f(x)*代換成 $f'(x_0)$ 得

$$f''(x_0) \cong \frac{f'(x_0+h/2)-f'(x_0-h/2)}{h} \qquad \textbf{(6-1.6)}$$

再利用方程式 (6-1.5) 得一次導函數 $f'(x_0+\frac{h}{2})$ 及 $f'(x_0-\frac{h}{2})$ 分別爲

$$f'(x_0+\frac{h}{2}) \cong \frac{f(x_0+h)-f(x_0)}{h} \qquad \textbf{(6-1.7)}$$

$$f'(x_0-\frac{h}{2})\cong\frac{f(x_0)-f(x_0-h)}{h} \qquad \textbf{(6-1.8)}$$

代入方程式 (6-1.6) 中，經整理後得 $f''(x_0)$ 之近似

$$f''(x_0)\cong\frac{f(x_0+h)-2f(x_0)+f(x_0-h)}{h^2} \qquad \textbf{(6-1.9)}$$

當然，利用這種方法我們可獲得任何階次的導函數近似值，詳見第三節說明。

第二節　微分的計算誤差

Visual Basic

本節中將討論利用方程式 (6-1.5) 及方程式 (6-1.9) 作一次及二次導函數的近似值時，所引入的截尾誤差 (Truncation errors)，並探討利用這些近似表示式時所造成的數值計算誤差。

根據泰勒定理，將*f(x)*對$x = x_0$展開，得到

$$f(x)=f(x_0)+(x-x_0)f'(x_0)+\frac{(x-x_0)^2}{2}f''(x_0)+\frac{(x-x_0)^3}{6}f'''(\xi) \qquad \textbf{(6-2.1)}$$

其中ξ位於x_0及x所界定的封閉區間內。將x用$x_0+\frac{h}{2}$代換，則得

$$f(x_0+\frac{h}{2})=f(x_0)+\frac{h}{2}f'(x_0)+\frac{h^2}{8}f''(x_0)+\frac{h^3}{48}f'''(\xi_1) \qquad \textbf{(6-2.2)}$$

其中 $x_0\le\xi_1\le x_0+\frac{h}{2}$。同理，利用$x=x_0-\frac{h}{2}$，得到

$$f(x_0-\frac{h}{2})=f(x_0)-\frac{h}{2}f'(x_0)+\frac{h^2}{8}f''(x_0)-\frac{h^3}{48}f'''(\xi_2) \qquad \textbf{(6-2.3)}$$

其中$x_0-\frac{h}{2}\le\xi_2\le x_0$。

將方程式 (6-2.2) 減去方程式 (6-2.3)，經整理後得到

$$f'(x_0)=\frac{f(x_0+\frac{h}{2})-f(x_0-\frac{h}{2})}{h}-\frac{h^2}{24}\left[\frac{f'''(\xi_1)+f'''(\xi_2)}{2}\right] \qquad \textbf{(6-2.4)}$$

若 $f'''(x)$ 爲一連續函數，則根據中間值定理 (Intermediate Value Theorem) 必存在ξ值，使得$\xi_2\le\xi\le\xi_1$，且

$$\frac{f'''(\xi_1)+f'''(\xi_2)}{2}=f'''(\xi) \qquad \textbf{(6-2.5)}$$

因此，方程式 (6-2.4) 可以被改寫成

$$f'(x_0)=\frac{f(x_0+\frac{h}{2})-f(x_0-\frac{h}{2})}{h}-\frac{h^2}{24}f'''(\xi) \qquad \textbf{(6-2.6)}$$

其中 $x_0-\frac{h}{2}\le\xi\le x_0+\frac{h}{2}$ 。與方程式 (6-1.5) 比較，可以得到截尾誤差 (真值 - 近似值) 爲

$$\varepsilon_t=\frac{h^2}{24}f'''(\xi)$$

因此，我們若將間距 h 減半，則截取誤差可減爲 1/4 倍。理論上，h 愈小則結果可能愈佳，但事實上不然。由於計算機的有效位數及準確度問題，當 h 值小到某一程度以後，由於 $f(x_0+h/2)$ 及 $f(x_0-h/2)$ 變得極爲接近，若其差別小於計算機之捨入誤差，則二者的差就機器而言就變成零了，因此，計算結果可能完全錯誤。

相同的分析可適用於二次導函數，仿以上縯導，得到二次導函數表示式 (6-1.9) 的截尾誤差爲

$$\varepsilon_t=-\frac{h^2}{12}f^{iv}(\xi) \qquad \textbf{(6-2.8)}$$

注意此截尾誤差亦與 h^2 成比例，但卻與 $f^{iv}(\xi)$ 而非與 $f'''(\xi)$ 成比例。

例題 6-1　函數之一次及二次導函數

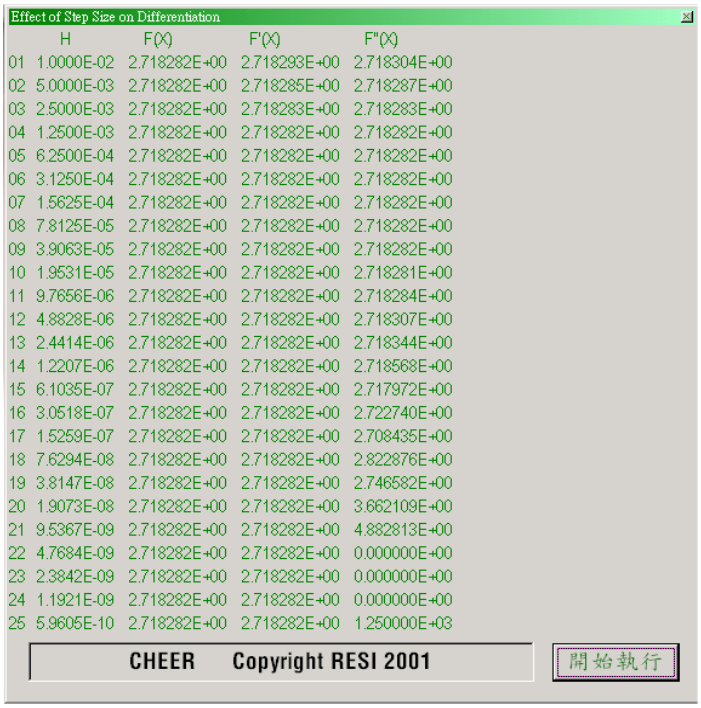
試寫一程式求 $f(x)=e^x$ 在 $x = 1$ 處的一次及二次導函數。並比較間距 h 對結果之影響。

程式列印：

```
' **********************************
' DIFFERENTIATION
' **********************************
'
' Program Developed by Enya Chang
' Copyright 2001 CHEER
'
```

```
Private Sub Differentiation()
'
'       H = STEP SIZE
'     FP = F'(X)
'     FD = F''(X)
'
Print "         H                F(X)              F'(X)                    F''(X)"
X = 1
H = 0.01
For I = 1 To 25
        FP = (F(X + H / 2) - F(X - H / 2)) / H
        FD = (F(X + H) - 2 * F(X) + F(X - H)) / H / H
        Print Format(I, "      00       ");
        Print Format(H, " 0.0000E+00        ");
        Print Format(F(X), "0.000000E+00        ");
        Print Format(FP, "0.000000E+00        ");
        Print Format(FD, "0.000000E+00")
        H = H / 2
Next
End Sub


Private Function F(X)
F = Exp(X)
End Function
```

測試結果：



結果討論：

由於 $f(x)=e^x$ ，故 $f''(x)=f'(x)=e^x$ 。由以上測試結果顯示當 h 值較小的時候，導函數之計算誤差會逐漸顯著，因此，計算導函數的方法應設法求取同一個 h 值情況下使計算誤差減小的方法，而不是希望減小 h 值提高準確度。

第三節　再論數值微分

Visual Basic

在本節中，我們將介紹多點微分計算法，以解決例 6-1 所述之減法消去所產生誤差的難題，並續導一至四階導函數的一般近似方程式。在本節中，我們將發現計算 $f'(x)$ 時，所用點數愈多，則截尾誤差愈小，使我們不必減小 h 值即可減少截尾誤差，因此，即不必擔心 h 值過小所產生的問題。

$f'(x_0)$ 的三點表示式

首先我們先續導 $f'(x_0)$ 的三點表示式。假設用於計算的個點爲x_{-1}，x_0及x_1。其中 $x_{-1}\le x_0\le x_1$ ，令

$$x_{-1}=x_0-h_1 \qquad \textbf{(6-3.1)}$$

$$x_1=x_0+h_2 \qquad \textbf{(6-3.2)}$$

其中 $h_1>0$ ， $h_2>0$ 。則我們可仿照方程式 (6-1.5) 的型式，將 $f'(x_0)$ 表示成爲 $f(x_{-1})$ ， $f(x_0)$ 及 $f(x_1)$ 之線性函數

$$\begin{aligned} f'(x_0) &\cong C_{-1}f(x_{-1})+C_0f(x_0)+C_1f(x_1) \\ &= C_{-1}f(x_0-h_1)+C_0f(x_0)+C_1f(x_0+h_2) \end{aligned} \qquad \textbf{(6-3.3)}$$

我們希望決定最佳的C_{-1}， C_0及C_1值。首先考慮當 *f(x) = 1* 時，我們當然希望方程式 (6-3.3) 完全正確，此時 $f'(x_0)=0$ ，因此，由方程式(6-3.3)可以得到

$$0=C_{-1}+C_0+C_1 \qquad \textbf{(6-3.4)}$$

其次考慮當 $f(x)=x-x_0$ 時，也希望方程式 (6-3.3) 爲完全正確，此時 $f'(x_0)=1$ ，故由方程式 (6-3.3) 可以得到

$$1=-h_1C_{-1}+h_2C_1 \qquad \textbf{(6-3.5)}$$

最後考慮當 $f(x)=(x-x_0)^2$ 時，希望方程式 (6-3.3) 也能滿足此拋物線方程式。此時 $f'(x)=2(x-x_0)$，由方程式 (6-3.3) 可以得到

$$0=h_1^{\,2}C_{-1}+h_2^{\,2}C_1 \qquad \textbf{(6-3.6)}$$

係數 C_{-1}，C_0 及 C_1 必須滿足聯立方程式 (6-3.4)， (6-3.5) 及 (6-3.6)。其解爲

$$C_{-1}=-\frac{h_2}{h_1(h_1+h_2)} \qquad \textbf{(6-3.7)}$$

$$C_0=\frac{h_2^{\,2}-h_1^{\,2}}{h_1h_2(h_1+h_2)} \qquad \textbf{(6-3.8)}$$

$$C_1=\frac{h_1}{h_2(h_1+h_2)} \qquad \textbf{(6-3.9)}$$

需注意因爲 $h_1>0$，$h_2>0$，因此上列諸式必然存在。代回方程式 (6-3.3)，得到一次導函數的一般表示式爲

$$f'(x)=\frac{h_1^{\,2}f(x_0+h_2)+(h_2^{\,2}-h_1^{\,2})f(x_0)-h_2^{\,2}f(x_0-h_1)}{h_1h_2(h_1+h_2)} \qquad \textbf{(6-3.10)}$$

微分的五點表示式

其次再考慮五點的表示式。假設此五點分別爲 x_0-h_1，x_0-h_2，x_0，x_0+h_3 及 x_0+h_4，其中 $h_1>h_2>0$，$h_4>h_3>0$。利用這五點的函數值 *f(x)* 求取 $f'(x_0)$ 之近似值，仿以上處理方式，將 $f'(x_0)$ 寫成五個函數值的線性組合：

$$\begin{aligned}f'(x_0)\cong\ & C_{-2}f(x_0-h_1)+C_{-1}f(x_0-h_2)+C_0f(x_0)\\ & +C_1f(x_0+h_3)+C_2f(x_0+h_4)\end{aligned} \qquad \textbf{(6-3.11)}$$

我們希望找出 C_{-2}，C_{-1}，C_0，C_1 及 C_2 的值。

如前述，令 *f(x) = 1*，即 $f'(x_0)=0$，則由方程式 (6-3.11) 可以得到

$$0=C_{-2}+C_{-1}+C_0+C_1+C_2 \qquad \textbf{(6-3.12)}$$

其次，令 $f(x)=x-x_0$，即 $f'(x_0)=1$；由方程式 (6-3.11) 得到

$$1=-h_1C_{-2}-h_2C_{-1}+h_3C_1+h_4C_2 \qquad \textbf{(6-3.13)}$$

又令 $f(x)=(x-x_0)^2$，得到

$$0 = h_1^{\,2}C_{-2} + h_2^{\,2}C_{-1} + h_3^{\,2}C_1 + h_4^{\,2}C_2 \qquad \textbf{(6-3.14)}$$

令 $f(x) = (x - x_0)^3$，得到

$$0 = -h_1^{\,3}C_{-2} + h_2^{\,3}C_{-1} + h_3^{\,3}C_1 + h_4^{\,3}C_2 \qquad \textbf{(6-3.15)}$$

最後令 $f(x) = (x - x_0)^4$，得到

$$0 = h_1^{\,4}C_{-2} + h_2^{\,4}C_{-1} + h_3^{\,4}C_1 + h_4^{\,4}C_2 \qquad \textbf{(6-3.16)}$$

方程式 (6-3.12) 至 (6-3.16) 計有五個方程式，並含五個未知數 C_{-2}，C_{-1}，C_0，C_1 及 C_2。其解較爲繁複，爲了簡化分析起見，假設 $h_1 = 2h_2$，$h_4 = 2h_3$，且 $h_2 = h_3 = h$，則可以得到

$$C_{-2} = \frac{1}{12\,h}$$

$$C_{-1} = -\frac{2}{3\,h}$$

$$C_0 = 0$$

$$C_1 = \frac{2}{3\,h}$$

$$C_2 = -\frac{1}{12\,h}$$

故方程式 (6-3.11) 變成

$$f'(x_0) \cong \frac{1}{12h}\left[f(x_0 - 2h) - 8f(x_0 - h) + 8f(x_0 + h) - f(x_0 + 2h) \right] \qquad \textbf{(6-3.17)}$$

此表示式對任何四次以下的多項式均爲正確表示式。

同理，我們可以證明二次、三次及四次導函數導的等間距五點表示式分別爲

$$f''(x_0) \cong \frac{1}{12\,h^2}\left[-f(x_0 - 2h) + 16f(x_0 - h) - 30f(x_0) + 16f(x_0 + h) - f(x_0 + 2h) \right] \qquad \textbf{(6-3.18)}$$

$$f'''(x_0) \cong \frac{1}{2\,h^3}\left[-f(x_0 - 2h) + 2f(x_0 - h) - 2f(x_0 + h) + f(x_0 + 2h) \right] \qquad \textbf{(6-3.19)}$$

$$f^{iv}(x_0) \cong \frac{1}{h^4}[f(x_0-2h)-4f(x_0-h)+6f(x_0)-4f(x_0+h) + f(x_0+2h)] \quad \textbf{(6-3.20)}$$

這種續導相當簡單，對於更高階次的微分或更多點的近似表示式均可仿以上步驟續導之。請注意以上的導函數近似表示式的係數和都等於零，因此，如果 h 值相當小時，都會產生減法消去所造成的嚴重錯誤，使用時必須特別謹慎。

應用本節所介紹差分法解微分方程式的方法，詳見第九章及第十一章。

第四節　數值積分

Visual Basic

在以下各節中，我們將介紹三種常用的數值積分方法，分別建立 Visual Basic 程式，並比較其執行效率。考慮函數 *f(x)* 之定積分式

$$\int_a^b f(x)dx = F(b)-F(a) \quad \textbf{(6-4.1)}$$

方程式 (6-4.1) 的計算值，可解釋爲在 [*a* , *b*] 區間內，在函數 *f(x)* 曲線下之面積，如圖 6.2 所示。積分式的計算對某些函數而言可能相當簡單，可是有時候也有可能相當複雜，或無法直接積分，而必須藉助於適當的近似方法，即數值積分 (Numerical integration)。

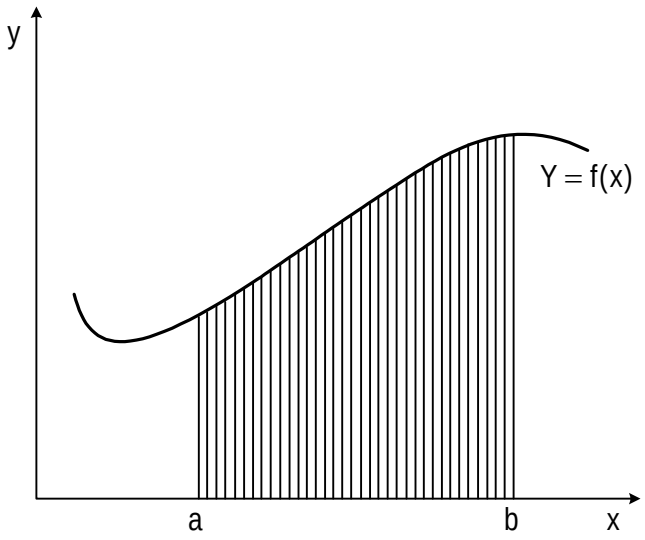


圖 6.2　函數 *f(x)* 之定積分

常用的數值積分方法通常是將原函數 *f(x)* 用一易於積分的函數取代，這種新函數可能爲一多項式，如直線、拋物線、或是超越函數，如正弦函數或餘弦函數等。數值積分計算結果的準確性就決定於替代函數與原函數的近似程度。

下一節中，我們首先介紹梯形積分法，並利用此方法介紹函數替代的原理。

第五節　梯形積分法

Visual Basic

梯形積分法是所有數值積分法中最簡單的一種。這種方法基本上是利用一組直線線段來替代原函數。將積分範圍分割成 *n* 個等間距的子區間，其寬度 Δx 爲

$$\Delta x = \frac{b-a}{n} \qquad \textbf{(6-5.1)}$$

其中 *n* 爲子區間數，*b* 及 *a* 分別爲積分式的上下限。首先考慮只有單一子區間的情況，如圖 6.3 所示，若整個區間用單一直線作爲原函數之近似，則所求得面積爲

$$A = \frac{(b-a)[f(a)+f(b)]}{2} \qquad \textbf{(6-5.2)}$$

在此式中，*f(a)* 爲積分式下限處的函數值，*f(b)* 爲積分式上限處的函數值。

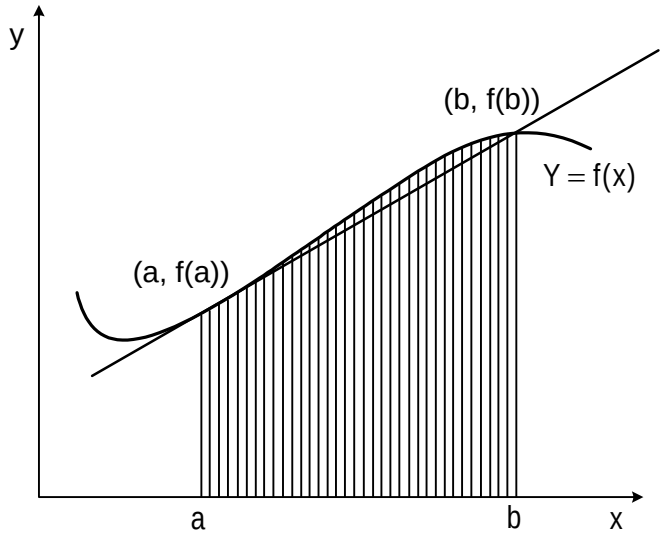


圖 6.3　單一子區間之梯形積分

對於更一般化的情況，若將整個區間 [a , b] 分成 n 個子區間，則第 i + 1 個子區間的積分上下限分別爲 $x_i + \Delta x$ 及 x_i ，如圖 6.4 所示。在每一子區間上的函數曲線分別用一直線方程式取代，由於每一子區間的形狀均成梯形，故稱爲「梯形積分法」。

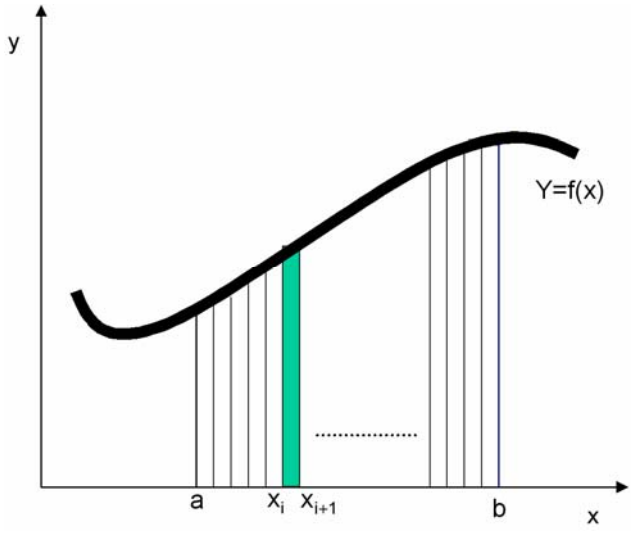


圖 6.4 n 個子區間之梯形積分

利用梯形積分法時，第 i + 1 個子區間的面積爲

$$A_{i+1} = \frac{[f(i) + f(i+1)]\,\Delta x}{2} \qquad \textbf{(6-5.3)}$$

其中 $f(i) = f(x_i)$ ， $f(i+1) = f(x_{i+1})$ ，i = 0 ，1 ，2 ，…… ，分別爲第 i + 1 個子區間之左側及右側函數值。我們所要求解的積分式 (6-4.1) 即爲各個子區間面積的總和：

$$\begin{aligned} A &= \sum_{i=1}^{n} A_i = \sum_{i=1}^{n} \frac{\Delta x}{2}[f(i-1) + f(i)] \\ &= \frac{\Delta x}{2}\{\,[f(a) + f(1)] + [f(1) + f(2)] + \cdots\cdots + [f(n-1) + f(b)]\,\} \\ &= \frac{\Delta x}{2}[f(a) + 2\,f(1) + 2\,f(2) + \cdots\cdots + 2\,f(n-2) + 2\,f(n-1) + f(b)] \end{aligned}$$

(6-5.4)

利用梯形積分法所造成的截尾誤差，等於函數區間 $y = f(x)$ 與 $(x_i\,,\,f(x_i))$ 及 $(x_{i+1}\,,\,f(x_{i+1}))$ 所形成的弦之間的扇形區域面積和。當子區間的數目愈大，則此扇形面積和愈小，即截尾誤差愈小。考慮函數$y = f(x)$ 對$x = x_i$點之泰勒級數展開式：

$$f(x) = f(x_i) + (x - x_i) f'(x_i) + \frac{(x - x_i)^2}{2} f''(x_i) + \cdots\cdots \quad \textbf{(6-5.5)}$$

同理，對$x = x_{i+1}$點作泰勒級數展開爲

$$f(x) = f(x_{i+1}) + (x - x_i - \Delta x) f'(x_{i+1}) + \frac{(x - x_i - \Delta x)^2}{2} f''(x_{i+1}) + \cdots\cdots \quad \textbf{(6-5.6)}$$

以上二方程式雖然都是正確的，但卻無法用於和方程式 (6-5.4) 作比較。因此，首先取上二式的平均值，得到

$$\begin{aligned} f(x) = {} & \frac{1}{2}\left[f(x_i) + f(x_{i+1}) \right] + \frac{(x - x_i)}{2}\left[f'(x) + f'(x_{i+1}) \right] \\ & - \frac{\Delta x}{2} f'(x_{i+1}) + \frac{(x - x_i)^2}{4}\left[f''(x) + f''(x_{i+1}) \right] \\ & - \frac{\Delta x}{2}(x - x_i) f''(x_{i+1}) + \frac{(\Delta x)^2}{4} f''(x_{i+1}) + \cdots\cdots \end{aligned} \quad \textbf{(6-5.7)}$$

將$f(x)dx$由x_i積分至x_{i+1}，並以$f(i)$ 代表$f(x_i)$，得到

$$\begin{aligned} \int_{x_i}^{x_{i+1}} f(x)dx = {} & \frac{\Delta x}{2}\left[f(i) + f(i+1) \right] + \frac{(\Delta x)^2}{4}\left[f'(i) + f'(i+1) \right] \\ & - \frac{(\Delta x)^2}{2} f'(i+1) + \frac{(\Delta x)^3}{12}\left[f''(i) + f''(i+1) \right] \\ & - \frac{(\Delta x)^3}{4} f''(i+1) + \frac{(\Delta x)^3}{4} f''(i+1) + \cdots\cdots \\ = {} & \frac{\Delta x}{2}\left[f(i) + f(i+1) \right] + \frac{(\Delta x)^2}{4}\left[f'(i) - f'(i+1) \right] \\ & + \frac{(\Delta x)^3}{12}\left[f''(i) + f''(i+1) \right] + \cdots\cdots \end{aligned} \quad \textbf{(6-5.8)}$$

此方程式爲第 $i + 1$ 個子區間之真正積分近似值，與方程式 (6-5.3) 比較，顯示梯形積分法將 $(\Delta x)^2$ 以上諸項捨去。因此，梯形積分法之截尾誤差爲

$$E_{T_i} = \frac{(\Delta x)^2}{4}\left[f'(i) - f'(i+1) \right] + \frac{(\Delta x)^3}{12}\left[f''(i) + f''(i+1) \right] + \cdots\cdots \quad \textbf{(6-5.9)}$$

當 Δx 非常小時，上式可用第一項近似之。由於 $f'(x) = df / dx$ 利用泰勒級數展開，並乘以 $(\Delta x)^2$ ，可以得到

$$(\Delta x)^2 f'(i+1) = (\Delta x)^2 f'(i) + (\Delta x)^3 f''(i) + \cdots\cdots \quad \textbf{(6-5.10)}$$

表示 $(\Delta x)^3 f''(i)$ 與方程式 (6-5.9) 的第一項仍呈某種比例關係，依此類推，較高次項亦與第一項成某種比例關係。故可將梯形積分法的截尾誤差寫成

$$E_{T_i} \cong K(\Delta x)^2 [f'(i) - f'(i+1)] \quad \textbf{(6-5.11)}$$

其中 K 爲一未定係數。在此假設 K 大約成一定值。

考慮函數$f(x) = x^2$。則代入積分式中，可以得到

$$I_{i+1} = \int_{x_i}^{x_{i+1}} x^2 dx = \frac{1}{3}[x_{i+1}^3 - x_i^3] \quad \textbf{(6-5.12)}$$

而由方程式 (6-5.8) 則得到

$$I_{i+1} = \frac{\Delta x}{2}[x_i^2 + x_{i+1}^2] + E_{T_i} \quad \textbf{(6-5.13)}$$

令方程式 (6-5.12) 與方程式 (6-5.13) 兩式相等，則得到

$$\begin{aligned} E_{T_i} &= \frac{1}{3}[x_{i+1}^3 - x_i^3] - \frac{(x_{i+1} - x_i)}{2}[x_i^2 + x_{i+1}^2] \\ &= \frac{1}{6}(x_i - x_{i+1})^3 \quad = -\frac{1}{6}(\Delta x)^3 \end{aligned} \quad \textbf{(6-5.14)}$$

與截尾誤差的一般式方程式 (6-5.11) 比較，得到

$$K = \frac{1}{12} \quad \textbf{(6-5.15)}$$

即

$$E_{T_i} = \frac{(\Delta x)^2}{12}[f'(i) - f'(i+1)] \quad \textbf{(6-5.16)}$$

由於總截尾誤差爲各個子區間截尾誤差之和，故得

$$E_T = \sum_{i=1}^{n} E_{T_i} = -\frac{(\Delta x)^2}{12}[f'(b) - f'(a)] \quad \textbf{(6-5.17)}$$

例題 6-2 絕熱型分批式反應器設計

在一個絕熱型分批式反應器中進行液相反應 $A \rightarrow B$。其反應速率表示爲

$$-r_A = k_0 \, Exp(-E/RT) \, C_A^{\,n} \quad \textbf{(6-5.18)}$$

其中

$-r_A$ 爲反應速率 [g-mole/ℓ-hr]，

Arrhenius 常數 $k_0 = 1.5 \times 10^{11}〔\frac{\ell}{\text{hr}}(\frac{\text{g}-\text{mole}}{\ell})^{1-n}〕$，

T 爲反應溫度 [°K]，

C_A 爲 A 的濃度 [g-mole/ℓ]，

n 爲反應次數，n = 1.5；

活化能 E = 15,300 cal/g-mole，

反應物比熱爲 C_p = 0.95 Cal/g·°K ，

反應物密度 ρ = 1,100 g/ℓ ，

氣體定律常數 R = 1.987 cal/g-mole。

開始反應時，溫度 T = 293 °K ，A 的最初濃度爲 C_{AO} = 0.2 g-mole/ℓ ，此反應的反應熱爲 $(-\Delta Hr)$ = 35000 cal/g-mole。若反應在絕熱情況下進行，則其轉化率達到 x_A 時所需要的時間爲 [2]：

$$\theta = C_{AO}\int_o^{x_A}\frac{dx_A}{-r_A} = C_{AO}\int_o^{x_A}\frac{dx_A}{k_0\ Exp(-E/RT)\ C_{AO}^n\ (1-x_A)^n} \qquad \textbf{(6-5.19)}$$

反應物溫度與轉化率間的關係爲

$$T = T_0 + \frac{(-\Delta Hr)C_{AO}}{\rho C_p} \qquad \textbf{(6-5.20)}$$

試求轉化率達到 $x_A = 0.8$ 需要多少時間？

解：

TOP-DOWN 設計：

梯形積分法的計算式可以寫成

$$I = \frac{\Delta x}{2}〔f(a) + 2f(1) + 2f(2) + \cdots\cdots + 2f(n-2) + 2f(n-1) + f(b)〕 + \frac{(\Delta x)^2}{12}〔f'(a) - f'(b)〕 \qquad \textbf{(6-5.21)}$$

其中最後一項稱爲端點校正。

主 程 式

@設定參數
@執行梯形積分
Call *Sub Trapezoidal(Sum, Upper, Lower, EPS, TolErr)*
Time = $C_{AO}\int_{o}^{x_A}\frac{dx_A}{-r_A}$

副程式 **Trapezoidal(Sum, Upper, Lower, EPS, TolErr)**

@啓動及設定
PCS% = 1
DH = (Upper – Lower)/PCS%
ESUM = f(Lower) + f(Upper)
SUM = ESUM * DH / 2
@計算端點修正值
DFX(1) = (f(Lower+EPS) – f(Lower))/EPS
DFX(2) = (f(Upper) – f(Upper-EPS))/EPS
EDCR = (DFX(2) – DFX(1))/12
@增加區塊數，重複計算至收斂
SSUM = 0
PCS% = PCS% * 2
HOLD = SUM
DH = (Upper – Lower)/PCS%
For I = 1 to PCS%/2
X = Lower + DH * (2*I -1)
SSUM = SSUM + F(X)
SUM = (ESUM + 2*SSUM) * DH/2
SUM = SUM – DH*DH*EDCR
Repeat While ABS((SUM-HOLD)/SUM) >= TolErr
RETURN

符號說明：

CAO： 最初濃度 C_{AO} [g-mole/ℓ]

CP： 熱容量 C_p [cal/g·°K]

DFX： $\partial f/\partial x$ ；DFX(1) = $f'(a)$ ，DFX(2) = $f'(b)$

DH： 子區間的寬度 Δx

E： 反應的活化能 [cal/g-mole]

EDCR： 末端校正項；$\equiv \frac{1}{12}[f'(b)-f'(a)]$

EPS： 利用數值微分求 $f'(a)$ 及 $f'(b)$ 之微小增量 ε

F(x)： 所欲進行積分的函數

HR： 反應熱 $(-\Delta Hr)$ ；[cal/g-mole]

KO： Arrhenius 常數；$[\frac{\ell}{\text{hr}}(\frac{\text{g}-\text{mole}}{\ell})^{1-n}]$

Lower： 積分下限：a

PCS%： 子區間數目

R： 理想氣體常數；$[\equiv 1.987\ \text{cal/g}-\text{mole}\cdot{}^\circ\text{K}]$

RHO： 流體密度；ρ

SUM： 積分結果

SSUM： 子區間面積和

TO： 最初溫度； (293°K)

TIME： 反應時間；θ

TolErr： 誤差容許度

Upper： 積分上限，b

程式列印：

```
'   ********************************************
'      INTEGRATION BY TRAPEZOIDAL RULE
'   ********************************************
'
Private Sub TrapezoidalIntegration(Xpos, Ypos)
'
'     Define Parameters
'
TolErr = 0.000001
Lower = 0
Upper = 0.8
EPS = 0.0001
CA0 = 0.2:            ' INITIAL CONCN
'
Cls
Print
'
' CALL TRAPEZOIDAL
'
```

```
Call Trapezoidal(Sum, Upper, Lower, EPS, TolErr)
RxnTime = CA0 * Sum
Print
Print "REACTION TIME = ";
Print Format(RxnTime, "       0.000000E+00    HR")
End Sub


Private Function T(XA)
CA0 = 0.2:            ' INITIAL CONCN
CP = 0.95:            ' HEAT CAPACITY
HR = 35000!:          ' HEAT OF REACTION
RHO = 1100:           ' DENSITY
T0 = 293:             ' INITIAL TEMP.
T = T0 + HR * CA0 * XA / RHO / CP
End Function


Private Function F(XA)
'
' User Defined Function
'
CA0 = 0.2:            ' INITIAL CONCN
N = 1.5:              ' REACTION ORDER
R = 1.987:            ' GAS CONST
K0 = 150000000000#: ' ARRHENIUS CONST
E = 15300:            ' ACTIVATION ENERGYU
F = 1 / (K0 * Exp(-E / R / T(XA)) * CA0 ^ N * (1 - XA) ^ N)
End Function


Public Sub Trapezoidal(Sum, Upper, Lower, EPS, TolErr)
'
'     -------------------------
'        SUGBROUTINE TRAPEZOID
'     -------------------------
'
'        Sum = RESULT
'     Upper = UPPER LIMIT
'     Lower = LOWER LIMIT
'       EPS    = DX IN DERIVE DFX
'    TolErr = TOLERANCE
'
'        DH    = SUBINTERVAL WIDTH
'       DFX    = DERIVATIVE
'     ESUM    = END SUM
'     EDCR    = END CORRECTION
'     F(X)    = FUNCTION TO INTEGRATE
```

```
'       PCS% = NO. OF SUBINTERVALS
'       SSUM = SUIBINTERVAL SUM
'
Dim DFX(2)
        Print "No. Pcs       ";
        Print "    W/O End Corr ";
        Print "    With End Corr"
'
'   ENTRY POINT
'   PASS #1
'
PCS% = 1
DH = (Upper - Lower) / PCS%
ESUM = F(Lower) + F(Upper)
Sum = ESUM * DH / 2!
Print Format(PCS%, "          00000       ");
Print Format(Sum, "        #####.000000")
'
'CALCULATE END CORRECTION
'
DFX(1) = (F(Lower + EPS) - F(Lower)) / EPS
DFX(2) = (F(Upper) - F(Upper - EPS)) / EPS
EDCR = (DFX(2) - DFX(1)) / 12
'
'   PASS #2
'
SSUM = 0

Do
        PCS% = PCS% * 2
        HOLD = Sum
        DH = (Upper - Lower) / PCS%
        For I = 1 To PCS% / 2
                X = Lower + DH * (2 * I - 1)
                SSUM = SSUM + F(X)
        Next I
        Print Format(PCS%, "          00000       ");
        Sum = (ESUM + 2 * SSUM) * DH / 2
        Print Format(Sum, "        #####.000000");
        Sum = Sum - DH * DH * EDCR
        Print Format(Sum, "                    #####.000000")
Loop While (Abs(Sum - HOLD) > Abs(TolErr * Sum))
'
' CHEER-1986-2001    BY RON-HSIN CHANG
'
End Sub
```

副程式使用說明：

1. 副程式 **Public Sub Trapezoidal(Sum, Upper, Lower, EPS, TolErr)**
   (1) 使用者需輸入積分上限 Upper，積分下限 Lower，導函數計算差量 EPS，及積分誤差容忍度 TolErr。
   (2) 使用者需自行定義要積分的函數 Function F(X)。
   (3) 呼叫副程式 Call Trapezoidal(Sum, Upper, Lower, EPS, TolErr)
   (4) 積分結果爲 Sum。

執行結果：

結果討論：

1. 執行結果顯示若未利用端點校正誤差，需取 8192 個子區間，才可達到小數以下五位準確度；但是若採用端點校正誤差，執行速度約可提高 32 倍，只要取 64 個子區間，就能達到相同的準確度。
2. 在端點處如果切線斜率爲無窮大，則不可使用端點校正法。此外，若端點處的切線斜率均爲零，則端點校正法並無作用。

第六節 理查遜展延法

Visual Basic

梯形積分法爲一相當簡單的數值積分法，最大缺點是計算時需分割成相當多的子區間進行積分，才能獲得滿意的準確度。因此，本節將介紹一種簡單的改良方法，以提高其準確度。

梯形積分法的截尾誤差 (Truncation error) 如方程式 (6-5.17) 所示，與子區間寬度 Δx 的平方成正比，因此，可以表示成：

$$E_T = \alpha(\Delta x)^2 \tag{6-6.1}$$

其中截尾誤差比率係數α的表示式爲

$$\alpha = -\frac{1}{12}\left[f'(b) - f'(a)\right] \tag{6-6.2}$$

由於*f'(a)* 及*f'(b)* 並不明確，但假設 α 約成定值，則可以利用以下做法推估之。首先，令I_h爲 $\Delta x = h$ 時利用梯形積分法所得到的結果，I_k爲 $\Delta x = k$ 時利用梯形積分法所得到的結果；由於積分截尾誤差可以用方程式 (6-6.1) 表示，因此，真正的積分值*I*分別可以寫成

$$I = I_h + \alpha h^2 \tag{6-6.3}$$

$$I = I_k + \alpha k^2 \tag{6-6.4}$$

將所得到的兩個方程式相減，整理得到截尾誤差比率係數α的表示式爲

$$\alpha = \frac{I_h - I_k}{k^2 - h^2} \tag{6-6.5}$$

將此結果代入方程式 (6-6.3)，可以得到

$$I = I_h + \frac{I_h - I_k}{(k/h)^2 - 1} \tag{6-6.6}$$

利用這種方法可以求得比I_h及I_k更好的近似值。這種方法稱爲理查遜展延法 (Richardson's Deferred Approach)。使用上，通常以半間距作展延，即*k=h/2*。

第七節　辛普森積分法

Visual Basic

辛普森積分法是最廣泛被使用的數值積分法，基本上與梯形積分法相似，都需要將積分的範圍分割成許多的小區間，然後計算在各子區間端點處的函數值。辛普森積分法與梯形積分法比較，唯一的區別在於函數曲線下面積的近似方法不同。梯形積分法是將子區間的面積用一梯形作近似，辛普森法則是將兩個相臨的子區間的函數曲線用一拋物線作近似。因此，梯形積分法只在函數爲一次多項式時爲正確，而辛普森法則在函數爲三次或三次以下的多項式時均爲正確。故辛普森積分法是一種計算方法與梯形積分法相仿，但準確度較高的數值積分方法。

以下我們根據梯形積分法及前節所介紹的理查遜展延法來縯導辛普森積分方程式。在梯形積分法中，假設子區間的數目爲

$$n = \frac{(b-a)}{\Delta x} \tag{6-7.1}$$

假設 n 爲偶數，並令

$$\delta x = 2\Delta x \tag{6-7.2}$$

則由梯形積分法方程式 (6-5.4) 得到

$$\mathrm{I}_{\Delta} = \frac{\Delta x}{2}\left[f(a) + 2f(1) + 2f(2) + \ldots\ldots + 2f(n-2) + 2f(n-1) + f(b)\right] \tag{6-7.3}$$

$$\mathrm{I}_{\delta} = \Delta x\left[f(a) + 2f(2) + \ldots\ldots + 2f(n-2) + f(b)\right] \tag{6-7.4}$$

將方程式 (6-7.2)、方程式 (6-7.3) 及方程式 (6-7.4) 代入理查遜展延法之方程式 (6-6.6) 中，可以得到

$$\begin{aligned}
\mathrm{I} &= \mathrm{I}_{\Delta} + \frac{I_{\Delta} - I_{\delta}}{(\frac{\delta x}{\Delta x})^2 - 1} = \frac{4}{3} I_{\Delta} - \frac{1}{3} I_{\delta} \\
&= \frac{2}{3}\Delta x\left[f(a) + 2f(1) + 2f(2) + \ldots\ldots 2f(n-2) + 2f(n-1) + f(b)\right] \\
&\quad - \frac{1}{3}\Delta x\left[f(a) + 2f(2) + \ldots\ldots + 2f(n-2) + f(b)\right] \qquad \text{(6-7.5)} \\
&= \frac{\Delta x}{3}\left[f(a) + 4f(1) + 2f(2) + 4f(3) + 2f(4) + \ldots\ldots + 2f(n-4)\right. \\
&\quad \left. + 4f(n-3) + 2f(n-2) + 4f(n-1) + f(b)\right]
\end{aligned}$$

方程式 (6-7.5) 即稱爲辛普森積分法則 (Simpson's Rule)。其截尾誤差爲

$$E_T \cong -\frac{(\Delta x)^4}{180}(b-a)f^{iv}(\xi) \qquad a<\xi<b \qquad \textbf{(6-7.6)}$$

辛普森法的截尾誤差與子區間寬度 Δx 的四次方成正比，而梯形積分法則與 $(\Delta x)^2$ 成正比。

辛普森積分法程式設計時，亦可仿梯形積分法作端點校正誤差。如方程式 (6-7.6) 所示，利用辛普森法端點校正需使用函數的四次導函數 $f^{iv}(\xi)$，而本章稍前曾談及方程式之數值微分常會引入較大的誤差，因此，就程式設計及應用目的而言，端點校正最好只用一次導函數。含一次端點校正誤差的辛普森積分法則爲：

$$I=\{7[f(a)+f(b)]+14\sum_{i=2}^{n-2}f(i)+16\sum_{j=1}^{n-1}f(j)+\Delta x\,[f'(a)-f'(b)]\}\frac{\Delta x}{15} \qquad \textbf{(6-7.7)}$$

其中 i 爲偶數，j 爲奇數。

例題 6-3　辛普森積分法

利用辛普森積分法重做例題 6-2。

由於辛普森積分法計算方法與梯形積分法相仿，故 TOP-DOWN 設計予以省略，請參考例 6-2。

符號說明：(沿用例 6-2 之符號，以下只列出不同者)

EVSUM：偶數項的和，$\sum_{i=2}^{n-2}f(i)$，i 爲偶數。

ODSUM：奇數項的和，$\sum_{j=2}^{n-1}f(j)$，j 爲奇數。

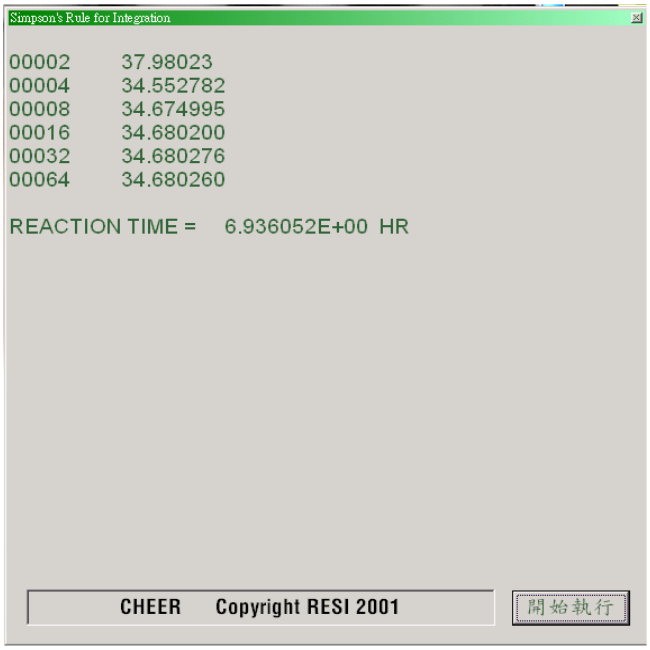
程式列印：

```
' ******************************************************
'       INTEGRATION BY SIMPSON'S RULE
' ******************************************************
'
Private Sub SimpsonRuleIntegration(Xpos, Ypos)
'
'   Define Parameters
'
```

```
TolErr = 0.000001
Lower = 0
Upper = 0.8
EPS = 0.0001
CA0 = 0.2:            ' INITIAL CONCN
'
Cls
Print
'
' CALL SIMPSON
'
Call Simpson(Sum, Upper, Lower, EPS, TolErr)
RxnTime = CA0 * Sum
Print
Print "REACTION TIME = ";
Print Format(RxnTime, "       0.000000E+00    HR")
End Sub


Private Function T(XA)
CA0 = 0.2:            ' INITIAL CONCN
CP = 0.95:            ' HEAT CAPACITY
HR = 35000!:          ' HEAT OF REACTION
RHO = 1100:           ' DENSITY
T0 = 293:             ' INITIAL TEMP.
T = T0 + HR * CA0 * XA / RHO / CP
End Function


Private Function F(XA)
'
' User Defined Function
'
CA0 = 0.2:            ' INITIAL CONCN
N = 1.5:              ' REACTION ORDER
R = 1.987:            ' GAS CONST
K0 = 150000000000#: ' ARRHENIUS CONST
E = 15300:            ' ACTIVATION ENERGYU
F = 1 / (K0 * Exp(-E / R / T(XA)) * CA0 ^ N * (1 - XA) ^ N)
End Function


Public Sub Simpson(Sum, Upper, Lower, EPS, TolErr)
'
'    INTEGRATION BY SIMPSON RULE
'
'          Sum = RESULT
'      Upper = UPPER LIMIT
'      Lower = LOWER LIMIT
'        EPS    = DX IN DERIVE DFX
'    TolErr = TOLERANCE
'
```

```
'         DH    = SUBINTERVAL WIDTH
'        DFX    = DERIVATIVE
'      ESUM     = END SUM
'      EDCR     = END CORRECTION
'      F(X)     = FUNCTION TO BE INTEGRATED
'       PCS% = NO. OF SUBINTERVALS
'       SSUM = SUIBINTERVAL SUM
'
Dim DFX(2)
'
'    ENTRY POINT
'    PASS #1
'
PCS% = 2
DH = (Upper - Lower) / PCS%
ESUM = F(Lower) + F(Upper)
ODSUM = F(Lower + DH)
EVSUM = 0
Sum = (ESUM + 4 * ODSUM) * DH / 3
Print Format(PCS%, "00000          ");
Print Format(Sum, "        #####.00000")
'
'    Calculate End Correction
'
DFX(1) = (F(Lower + EPS) - F(Lower)) / EPS
DFX(2) = (F(Upper) - F(Upper - EPS)) / EPS
EDCR = DFX(1) - DFX(2)
'
'    PASS #2
'
Do
        PCS% = PCS% * 2
        HOLD = Sum
        DH = (Upper - Lower) / PCS%
        EVSUM = EVSUM + ODSUM
        ODSUM = 0
        For I = 1 To PCS% / 2
                X = Lower + DH * (2 * I - 1)
                ODSUM = ODSUM + F(X)
        Next I
        Sum = 7 * ESUM + 14 * EVSUM + 16 * ODSUM + EDCR * DH
        Sum = Sum * DH / 15
        Print Format(PCS%, "00000          ");
        Print Format(Sum, "        #####.000000")
Loop While (Abs(Sum - HOLD) > Abs(TolErr * Sum))
'
' CHEER-1986-2001    BY RON-HSIN CHANG
'
End Sub
```

執行結果：



結果討論：

1. 一般而言，要達到相同準確度，辛普森積分法略較梯形積分法優。
2. 在端點處如果切線斜率爲無窮大，則不可使用端點校正法。此外，若端點處的切線斜率均爲零，則端點校正法並無作用。

第八節　龍勃格積分法

Visual Basic

利用梯形積分法計算以下的積分式時

$$I = \int_a^b f(x)dx \qquad \textbf{(6-8.1)}$$

若只取一個區間 $\Delta x = b - a = \ell$，則得到

$$I \equiv A(1,1) \cong \frac{[f(a) + f(b)]}{2}\Delta x = \frac{1}{2}[f(a) + f(b)]\ \ell \qquad \textbf{(6-8.2)}$$

若將積分範圍分成兩個子區間，即 $\Delta x = \frac{\ell}{2}$，則得到

$$I \equiv A(2,1) \cong \frac{1}{4}\left[f(a) + 2f(1) + f(b) \right] \ell \qquad \textbf{(6-8.3)}$$

同理，取四個子區間時， $\Delta x = \frac{\ell}{4}$ ，得到

$$I \equiv A(3,1) \cong \frac{1}{8}\left[f(a) + 2f(1) + 2f(3) + f(b) \right] \ell$$

依此類推得到當 $\Delta x = \frac{\ell}{2^{k-1}}$； 子區間數爲 2^{k-1} 個時

$$\begin{aligned} A(k,1) &= \frac{\ell}{2^k}\left[f(a) + 2\sum_{i=1}^{2^{k-1}-1} f(a + i\Delta x) + f(b) \right] \\ &= \frac{1}{2}A(k-1,1) + \frac{\ell}{2^{k-1}}\sum_{j=1}^{2^{k-2}} f\left[a + \frac{(2j-1)}{2^{k-1}}\ell \right] \end{aligned} \qquad \textbf{(6-8.4)}$$

在以上諸方程式中，積分近似值 *A (I，J)* 的 *I* 表示子區間數爲 2^{I-1}，*J = 1* 表積分時利用線性函數取代原函數 *f(x)*。

仿以上的處理方式，若利用拋物線方程式取代原函數 *f(x)*，即利用辛普森積分法，可得 $\Delta x = \frac{\ell}{2^k}$； 子區間數爲 2^k

$$A(k,2) = \frac{\Delta x}{3}\left\{ f(a) + 2f(b) + 4\sum_{j=1}^{2^{k-1}} f\left[a + (2j-1) \right] + 2\sum_{j=i}^{2^{k-1}-1} f\left[a + 2j\Delta x \right] \right\} \qquad \textbf{(6-8.5)}$$

比較方程式 (6-8.4) 及方程式 (6-8.5)，得到梯形積分法與辛普森積分法之關係爲

$$A(k,2) = \frac{4A(k+1,1) - A(\mathrm{k},1)}{3} \qquad \textbf{(6-8.6)}$$

同理類推得 *m* 次曲線近似值方法所得結果爲

$$A(n,1) = \frac{1}{2}A(n-1,1) + \frac{\ell}{2^{n-1}}\sum_{j=1}^{2^{n-2}} f\left[a + \frac{(2j-1)}{2^{n-1}}\ell \right] \qquad \textbf{(6-8.7)}$$

$$A(n,m) = \frac{4^{m-1}A(n+1,m-1) - A(n,m-1)}{4^{m-1}-1} \qquad \textbf{(6-8.8)}$$

我們利用這種外插法可很快地得到收斂的正確積分值。這種方法即稱爲龍勃格積分法，事實上，這種方法與理查遜展延法完全相同。實際計算時，只需先求出梯形積分法所得的結果 *A(I , 1)*，然後利用 (6-8.8) 外插即可求得正確的結果。

例題 6-4 龍勃格積分法

利用龍勃格積分法重解例 6-2。

解：

符號說明：

A： 矩陣，定義如 (6-8.7) 及 (6-8.8)

DH： 子區間寬度 Δx ，DH = $(b - a)/n$

DFX： 用於求端點校正用的導函數值 $f'(a)$ 及 $f'(b)$

EDCR： 端點校正錯誤值，式 (6-5.17)，

$$\text{EDCR} = \frac{1}{12}〔f'(b) - f'(a)〕$$

ESUM： 端點函數和，ESUM = $\frac{1}{12}〔f(a) + f(b)〕$

EPS： 利用數值微分求 $f'(a)$ 及 $f'(b)$ 之微小增量 ε

F(X)： 所要進行積分的函數 $f(x)$

HR： 反應熱 ($-\Delta Hr$)

K0： Arrhenius 常數

KFLAG： 錯誤旗幟，0 爲未收斂，1 表已收斂

Lower： 積分下限；a

N： 反應次數；在本例中 n = 1.5

PCS%： 子區間數目

R： 理想氣體定律常數，R = 1.987 cal/g-mole.^{o}K

RHO： 流體密度；ρ

ROW%： 行列式指標

RxnTime： 反應時間

SUM： 積分結果

TO： 最初溫度

TolErr： 誤差容許度

Upper： 積分上限，b

TOP-TOWN 設計：

主　程　式

```
LOWER = 0
    Upper = Lower + 0.1
    Call Romberg(Sum, Upper, Lower, EPS, TolErr, KFLAG)
    TIME = CA0 * Sum + TIME
    Print Result
    Lower = Lower + 0.1
Repeat While Upper < 0.9
```

副程式 **Romberg(Sum, Upper, Lower, EPS, TolErr, KFLAG)**

```
@啓動及設定
    PCS% = 1
    DH = (Upper – Lower)/PCS%
    ESUM = (f(Lower) + f(Upper))/2
    KFLAG = 1
@計算端點修正值
    DFX(1) = (f(Lower+EPS) – f(Lower))/EPS
    DFX(2) = (f(Upper) – f(Upper-EPS))/EPS
    EDCR = (DFX(2) – DFX(1))/12
@方程式(6-8.2)
    A(1,1) = DH * ESUM
    ROW% = 1
    SUM = ESUM
    F4 = 4
@Romberg's Rule
    ROW% = ROW% +1
    PCS% = PCS% * 2
    DH = (Upper-Lower)/PCS%
    For I = 1 to PCS%/2
        J = I*2 – 1
        X = Lower + J * DH
        SUM = SUM + F(X)
    @方程式(6-8.4)
        A(ROW%,1) = DH * SUM – DH * DH * EDCR
        For M = 2 to ROW%
            @方程式(6-8.8)
            A(Row%,4) = (F4*A(ROW%,M-1)-A(ROW%-1,M-1)/(F4-1)
            F4 = F4 * 4
    @Check Convergence
Repeat While Not Convergent
```

程式列印：

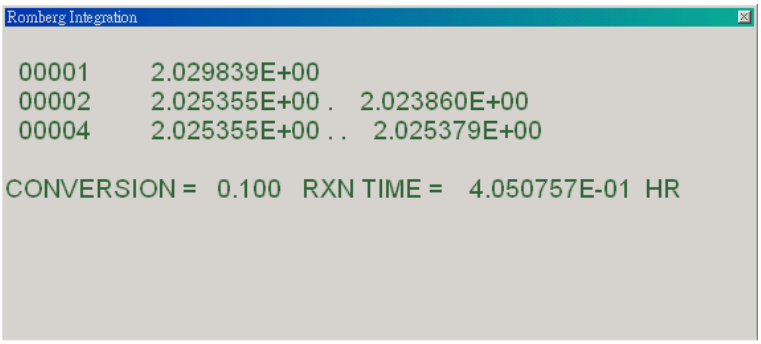
```
'   ***********************************************
'   INTEGRATION    BY      ROMBERG'S    RULE
'   ***********************************************
'
Private Sub RombergIntegration(Xpos, Ypos)
'
' Define Parameters
'
TolErr = 0.000001
MErr = 1E-16
EPS = 0.0001
CA0 = 0.2: ' INITIAL CONCN.
Lower = 0
'
Do
        Cls
        Print
        Upper = Lower + 0.1
        Call Romberg(Sum, Upper, Lower, EPS, TolErr, KFLAG)
        If (KFLAG <> 0) Then
                RxnTime = CA0 * Sum + RxnTime
                Print
                Print "CONVERSION = ";
                Print Format(Upper, "    0.000");
                Print "      RXN TIME = ";
                Print Format(RxnTime, "      0.000000E+00    HR")
                Print
        End If
        If Upper < 0.9 - MErr Then
                Lower = Lower + 0.1
                LoopID = 1
        Else
                LoopID = 0
        End If
        MsgBox ("Continue")
Loop While LoopID = 1
End Sub


Private Function T(XA)
CA0 = 0.2:          ' INITIAL CONCN
CP = 0.95:          ' HEAT CAPACITY
HR = 35000!:        ' HEAT OF REACTION
RHO = 1100:          ' DENSITY
```

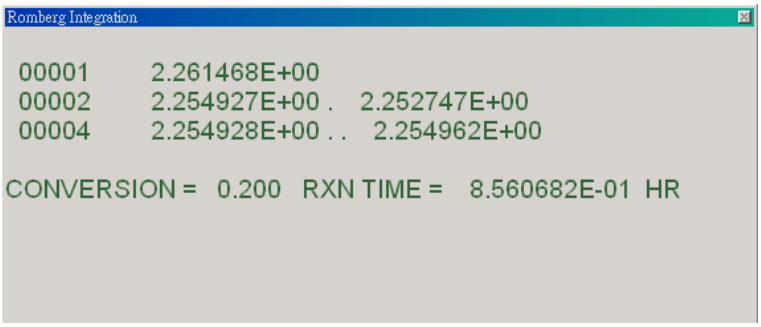
```
T0 = 293:              ' INITIAL TEMP.
T = T0 + HR * CA0 * XA / RHO / CP
End Function


Private Function F(XA)
CA0 = 0.2:             ' INITIAL CONCN
N = 1.5:               ' REACTION ORDER
R = 1.987:             ' GAS CONST
K0 = 150000000000#: ' ARRHENIUS CONST
E = 15300:             ' ACTIVATION ENERGYU
F = 1 / (K0 * Exp(-E / R / T(XA)) * CA0 ^ N * (1 - XA) ^ N)
End Function


Public Sub Romberg(Sum, Upper, Lower, EPS, TolErr, KFLAG)
'
' Integration By Romberg's Method
'
'         SUM = RESULT
'      UPPER = UPPER LIMIT
'      LOWER = LOWER LIMIT
'         EPS = DX for calculate DFX
'    TolErr = TOLERANCE
'
'            A = A(I,J)
'         DH    = SUBINTERVAL WIDTH
'        DFX    = DERIVATIVE
'      EDCR    = END CORRECTION
'      ESUM    = END SUM
'      F(X)    = FUNCTION TO INTEGRATE
'      KFLAG = ERROR FLAG
'                    0: NON-CONVERGENT
'                    1: ALL OK
'        PCS% = NO. OF SUBINTERVAL
'        ROW% = ROW IDENTIFIER
'
'    ENTRY POINT
'
NMAX = 16
Dim A(100, 100), DFX(2)
'
' CLEAR ERROR FLAG
'
KFLAG = 1
'
' INITIALIZE
'
```

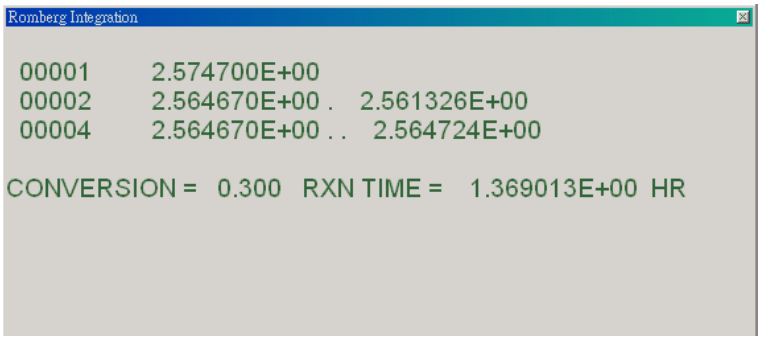
```
PCS% = 1
DH = (Upper - Lower) / PCS%
ESUM = (F(Lower) + F(Upper)) / 2
DFX(1) = (F(Lower + EPS) - F(Lower)) / EPS
DFX(2) = (F(Upper) - F(Upper - EPS)) / EPS
EDCR = (DFX(2) - DFX(1)) / 12
A(1, 1) = DH * ESUM
Print Format(PCS%, "    00000          ");
Print Format(A(1, 1), "        0.000000E+00")
Row% = 1
Sum = ESUM
'
'F4 = 4^(M-1)
'
F4 = 4
'
' TRAPEZOIDAL RULE
' WITH END CORRECTIONS
'
Do
        Row% = Row% + 1
        PCS% = PCS% * 2
        DH = (Upper - Lower) / PCS%
        For I = 1 To PCS% / 2
                J = I * 2 - 1
                X = Lower + J * DH
                Sum = Sum + F(X)
        Next I
        A(Row%, 1) = DH * Sum - DH * DH * EDCR
        Print Format(PCS%, "    00000          ");
        Print Format(A(Row%, 1), "        0.000000E+00");
'
' ROMBERG RULE
' DEFERRED APPROACH
'
        For M = 2 To Row%
                A(Row%, M) = (F4 * A(Row%, M - 1) - A(Row% - 1, M - 1)) / (F4 - 1)
                F4 = F4 * 4
                Print " .";
        Next M
        Print Format(A(Row%, Row%), "        0.000000E+00")
'
' CHECK CONVERGENCE
'
        If (Row% <= 2) Then
                KFLAG = 1
        ElseIf (A(Row%, 1) = 0) Then
                KFLAG = 3
```

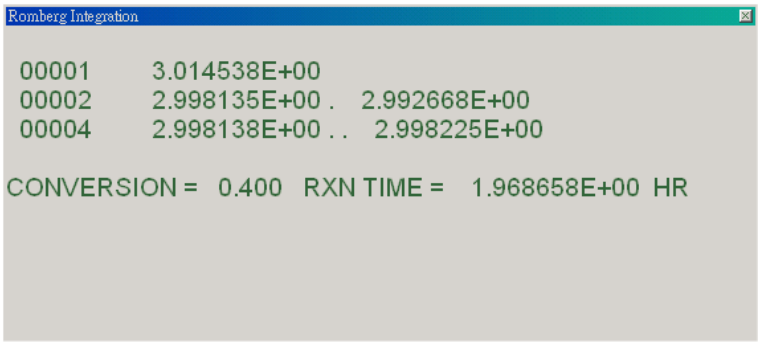
```
        ElseIf (Abs((A(Row% - 1, 1) - A(Row%, 1)) / (A(Row%, 1))) <= TolErr) Then
            KFLAG = 2
        ElseIf (Abs((A(Row% - 1, Row% - 1) - A(Row%, Row%)) / (A(Row%, Row%))) <= TolErr) Then
            KFLAG = 2
        ElseIf (Row% > NMAX) Then
            KFLAG = 0
        Else
            KFLAG = 1
        End If
Loop While KFLAG = 1 Or KFLAG = 3
'
' CONVERGENT
'
If KFLAG = 2 Then
        Sum = A(Row%, Row%)
'     NORMAL RETURN
Else
'     NO CONVERGENT
        Print "NO CONVERGENT"
        Print "NMAX = "; NMAX
End If
'
' CHEER-1986~2001    by    Ron-Hsin Chang
'
End Sub
```

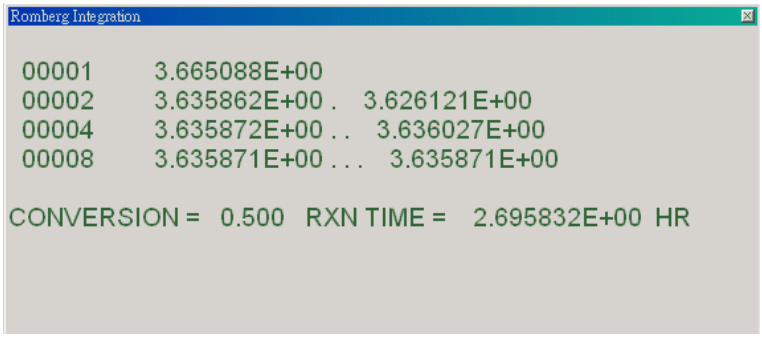
執行結果：

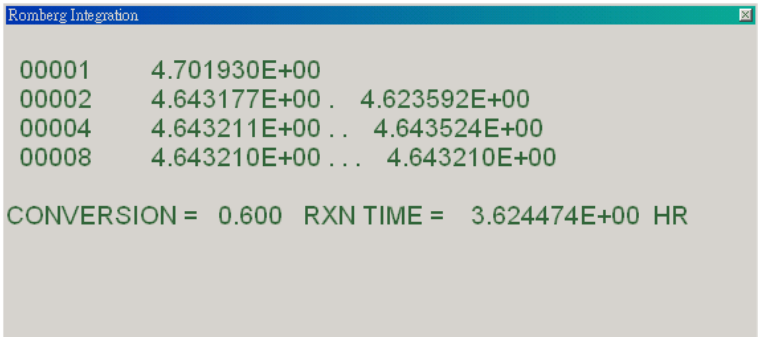


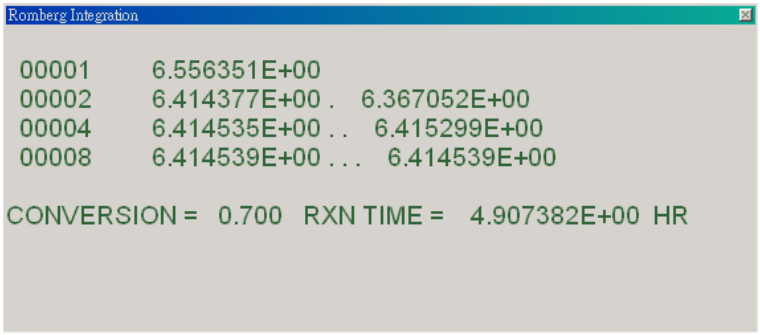

Romberg Integration
00001 2.261468E+00
00002 2.254927E+00 . 2.252747E+00
00004 2.254928E+00 . . 2.254962E+00
CONVERSION = 0.200 RXN TIME = 8.560682E-01 HR

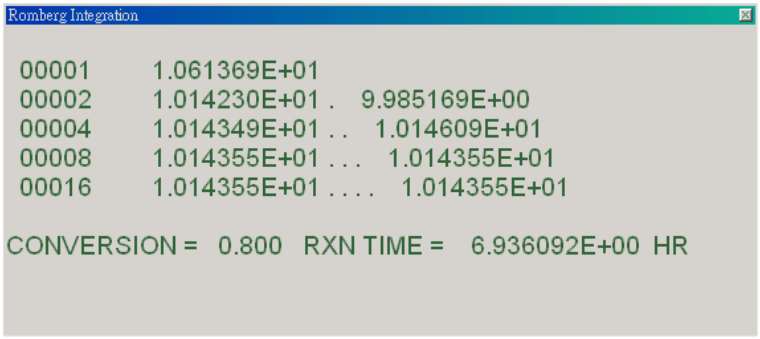

Romberg Integration
00001 2.574700E+00
00002 2.564670E+00 . 2.561326E+00
00004 2.564670E+00 . . 2.564724E+00
CONVERSION = 0.300 RXN TIME = 1.369013E+00 HR

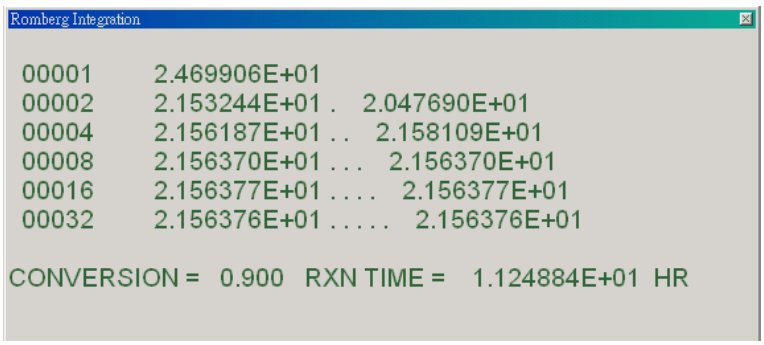

Romberg Integration
00001 3.014538E+00
00002 2.998135E+00 . 2.992668E+00
00004 2.998138E+00 . . 2.998225E+00
CONVERSION = 0.400 RXN TIME = 1.968658E+00 HR


Romberg Integration
00001 3.665088E+00
00002 3.635862E+00 . 3.626121E+00
00004 3.635872E+00 . . 3.636027E+00
00008 3.635871E+00 . . . 3.635871E+00
CONVERSION = 0.500 RXN TIME = 2.695832E+00 HR


Romberg Integration
00001 4.701930E+00
00002 4.643177E+00 . 4.623592E+00
00004 4.643211E+00 . . 4.643524E+00
00008 4.643210E+00 . . . 4.643210E+00
CONVERSION = 0.600 RXN TIME = 3.624474E+00 HR

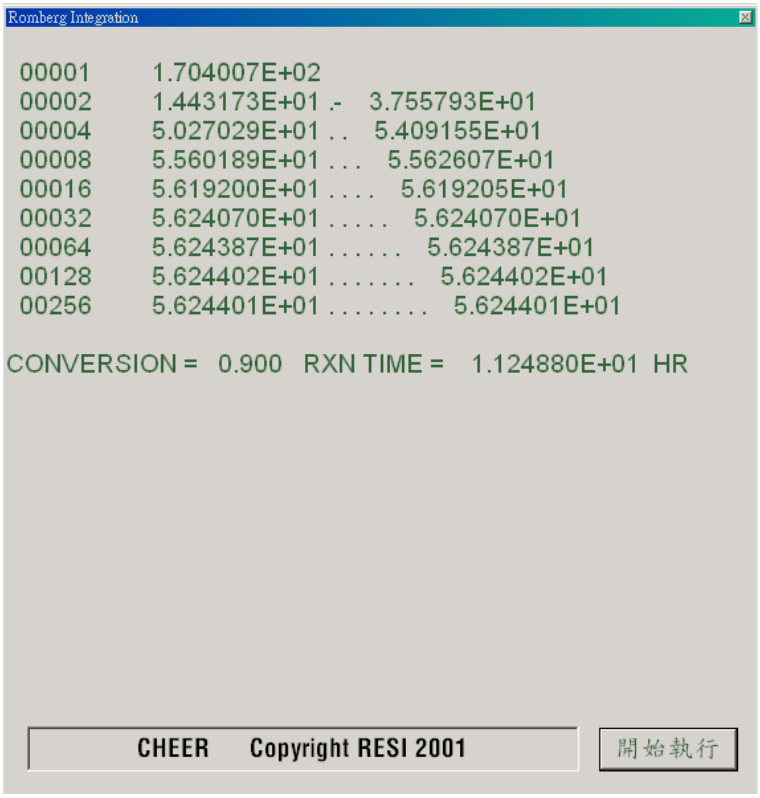

Romberg Integration
00001 6.556351E+00
00002 6.414377E+00 . 6.367052E+00
00004 6.414535E+00 . . 6.415299E+00
00008 6.414539E+00 . . . 6.414539E+00
CONVERSION = 0.700 RXN TIME = 4.907382E+00 HR


Romberg Integration
00001 1.061369E+01
00002 1.014230E+01 . 9.985169E+00
00004 1.014349E+01 . . 1.014609E+01
00008 1.014355E+01 . . . 1.014355E+01
00016 1.014355E+01 . . . . 1.014355E+01
CONVERSION = 0.800 RXN TIME = 6.936092E+00 HR


Romberg Integration
00001 2.469906E+01
00002 2.153244E+01 . 2.047690E+01
00004 2.156187E+01 . . 2.158109E+01
00008 2.156370E+01 . . . 2.156370E+01
00016 2.156377E+01 . . . . 2.156377E+01
00032 2.156376E+01 . . . . . 2.156376E+01
CONVERSION = 0.900 RXN TIME = 1.124884E+01 HR

結果討論：

以上的積分結果是以 $x_A = 0.1$ 爲區間，逐次由 $x_A = 0$ 開始積分。由於積分區間小，很明顯的積分速度相當快。爲了比較起見，直接由 $x_A = 0$ 積分至 $x_A = 0.9$ 所得結果亦列於下：



前者積分至 $x_A = 0.9$ 只需分割成 88 個子區間；而一次積分則需分割成 256 個子區間，故執行時所需時間略長。

將積分所得反應時間對轉化率作圖，如圖 6.5 所示。由圖可明顯地看出 x_A 較大時，反應所需時間急速增加，因此，積分所需分割區間亦隨著增加，此時若連低轉化率部分也同時增加分割區間數，只是徒然增加計算執行時間，對準確度並沒有任何益處。因此，積分時最好是先將原積分範圍略作分割再執行數值積分，可能使執行速度略快。這種技巧亦可用於端點函數值趨近於無窮大的情況。

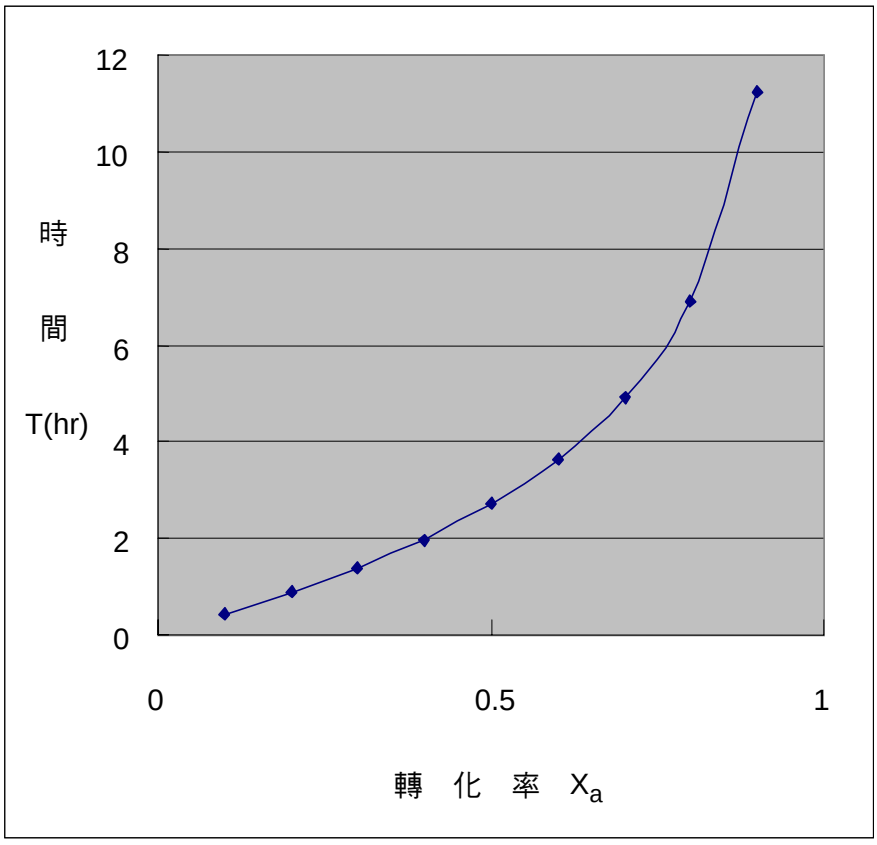


圖 6.5　反應時間與轉化率的關係

第九節　高斯積分法

Visual Basic

以上各節所介紹的數值積分法，使用者通常可隨意地自行選擇子區間寬度 Δx。而在實際使用上，我們通常選用相同的 Δx。假使我們放棄這種自由選擇子區間寬度的權利，而利用精確的子區間最佳決定策略，理論上應該可以獲得較高準確度的回報，高斯積分法 (Gauss Quadrature) 就是這類方法中的一個典型的例子。

試回想前述三種積分方法：梯形積分法、辛普森積分法及龍勃格積分法，基本上均可將這些積分方法寫成

$$I \equiv \int_a^b f(x)dx = \sum_{i=1}^{N+2} \omega_i f(x_i) \qquad \begin{matrix} \text{梯形積分法} & N=0 \\ \text{辛普森積分法} & N=1 \end{matrix} \qquad \textbf{(6-9.1)}$$

其中 ω_i 稱爲配重係數， $f(x_i)$ 則是在$x = x_i$處的函數值。例如，辛普森法中， $\omega_1 = \omega_3 = 1/6$， $\omega_2 = 2/3$。

爲了方便處理起見，如果將積分式的上下限利用變數轉換，使上下限變成 0 與 1：

$$u = \frac{x-a}{b-a} \qquad \textbf{(6-9.2)}$$

則方程式 (6-9.1) 可以被轉換成以下的積分型式，

$$I \equiv \int_0^1 f(x)dx = \sum_{i=1}^{N} \omega_i f(x_i) \qquad \textbf{(6-9.3)}$$

假設函數 *f(x)* 可以用一個多項式表示，並令 $f_i = x^{i-1}$，則可以將函數 *f(x)* 寫成

$$f(x) = \sum_{i=1}^{N} a_i \ f_i \qquad \textbf{(6-9.4)}$$

由於

$$\int_0^1 f_i \, dx = \int_0^1 x^{i-1} dx = \frac{1}{i} \equiv F_i \qquad \textbf{(6-9.5)}$$

而又由方程式 (6-9.3) 得到

$$\int_0^1 f_i \, dx = \sum_{j=1}^{N} \omega_j \, f_i(x_j) = \sum_{J=1}^{N} \omega_j \ x_j^{i-1} \qquad \textbf{(6-9.6)}$$

定義 $Q_{ji} = x_j^{i-1}$，則由方程式 (6-9.5) 及方程式 (6-9.6) 相等，可以得到

$$\sum_{j=1}^{N} \omega_j Q_{ji} = F_i \qquad \textbf{(6-9.7)}$$

或改寫成矩陣型式爲

$$\begin{bmatrix} w_1 & w_2 & \cdots & w_N \end{bmatrix} \begin{bmatrix} Q_{11} & Q_{12} & \cdots & Q_{1N} \\ Q_{21} & Q_{22} & \cdots & Q_{2N} \\ \vdots & \vdots & & \vdots \\ Q_{N1} & Q_{N2} & \cdots & Q_{NN} \end{bmatrix} = \begin{bmatrix} F_1 & F_2 & \cdots & F_N \end{bmatrix} \qquad \textbf{(6-9.8)}$$

$$\underline{W}\underline{\underline{Q}} = \underline{F} \qquad \textbf{(6-9.9)}$$

由於 $f_i = x^{i-1}$ 及 $Q_{ji} = x_j^{i-1}$ ，由方程式 (6-9.9) 直接可以求解配重係數矩陣 $\underline{W}$ 的關係式爲：

$$\underline{W} = \underline{F}\,\underline{\underline{Q}}^{-1} \qquad \textbf{(6-9.10)}$$

計算策略

1. 決定配置點數目 N。
2. 決定需要求解的未知數數目 m，未知數包括：配置點 x_i 及配重係數 w_j。
3. 利用方程式 (6-9.6) 建立未知數的關係式。其中基礎函數 $f_i = x^{i-1}$ 的最大 i 值，等於 m。
4. 由以上所建立的關係式，求得配置點 x_i 及配重係數 w_j。
5. 利用方程式 (6-9.3) 求得函數 $f(x)$ 的積分值。

例題 6-5　高斯積分法

利用高斯積分法時，若採用 $N = 3$，試求配重係數爲何。

解：

1. 當 $N = 3$ 時，總共需要求解的未知數包括：

   配置點：　x_1 , x_2 及 x_3
   配重係數：　w_1 , w_2 及 w_3

   亦即需要求解六個未知數。爲了方便說明起見，假設 $x_1=0$ ，$x_3=1$，則未知數只剩下四個，要解四個未知數需要有四組方程式，即 $N = 4$，亦即，函數積分可以準確到 x^3 項。若假設 $f(x) = f_1 = 1$，則方程式(6-9.3)必須仍能成立，亦即

   $$f(x) = f_1 = 1 \quad : \quad \int_0^1 f(x)dx = 1 = \sum_{i=1}^{3} w_i f(x_i) = w_1 + w_2 + w_3$$

   同理，若令 $f(x) = f_2 = x$ ，$f(x) = f_3 = x^2$ ，$f(x) = f_4 = x^3$，則方程式 (6-9.3) 也必須都能成立，亦即

   $$f(x) = f_2 = x : \quad \int_0^1 f(x)dx = 1/2 = \sum_{i=1}^{3} w_i f(x_i) = w_1 \cdot (0) + w_2 \cdot x_2 + w_3 \cdot (1)$$

$$f(x)=f_3=x^2\ :\ \int_0^1 f(x)dx=1/3=\sum_{i=1}^{3}w_i f(x_i)=w_1\cdot(0)+w_2\cdot x_2^2+w_3\cdot(1)$$

$$f(x)=f_4=x^3\ :\ \int_0^1 f(x)dx=1/4=\sum_{i=1}^{3}w_i f(x_i)=w_1\cdot(0)+w_2\cdot x_2^3+w_3\cdot(1)$$

在以上的演導中，選擇$x_1=0$，$x_3=1$，x_2爲未知的配置點，加上配重係數w_1，w_2，及w_3，我們總計有四個未知數及四組方程式，

$$w_1+w_2+w_3=1$$

$$w_2x_2+w_3=1/2$$

$$w_2x_2^2+w_3=1/3$$

$$w_2x_2^3+w_3=1/4$$

以上四組方程式的解爲$x_2=1/2$，$w_1=1/6$，$w_2=2/3$，$w_3=1/6$，結果與辛普森法完全相同，可將積分式寫成

$$I=\int_0^1 f(x)dx=\tfrac{1}{6}\left[f(0)+4f(\tfrac{1}{2})+f(1)\right]$$

2. 若不假設$x_1=0$，$x_3=1$ 時，總共需要求解的未知數包括：

配置點：　x_1, x_2 及 x_3
配重係數：　w_1, w_2 及 w_3

亦即需要求解六個未知數。要解六個未知數需要有六組方程式，即$N = 6$，亦即，函數積分可以準確到x^5項。仿上演導，可以得到下列方程式：

$$w_1+w_2+w_3=1$$

$$w_1x_1+w_2x_2+w_3x_3=1/2$$

$$w_1x_1^2+w_2x_2^2+w_3x_3^2=1/3$$

$$w_1x_1^3+w_2x_2^3+w_3x_3^3=1/4$$

$$w_1x_1^4+w_2x_2^4+w_3x_3^4=1/5$$

$$w_1x_1^5+w_2x_2^5+w_3x_3^5=1/6$$

以上方程式的解爲

$x_1 = 0.112\ 701\ 6654$
$x_2 = 0.500\ 000\ 0000$
$x_3 = 0.887\ 298\ 3346$
$w_1 = 0.277\ 777\ 7778$
$w_2 = 0.444\ 444\ 4444$
$w_3 = 0.277\ 777\ 7778$

利用方程式 (6-9.10) 求解配重係數$\underline{w}$時，必須先知道配置點$\underline{x}$，而由以上簡單的實例，我們可以發現如果利用$f_i(x) = x^{i-1}$代入方程式 (6-9.3)，聯立求解$\underline{x}$ 及$\underline{w}$，會得到一組非線性聯立方程式，其求解過程將變得相對複雜。但如果使用雅可必 (Jacobi) 多項式之類的正交多項式 (Orthogonal Polynomial) 來代替f_i，並以n次正交多項式的根作爲配置點，則可輕易的求得配重係數$\underline{w}$。有關配置點及配重係數相關問題，請閱讀本書第十章數學配置法，有更詳細的討論及說明。

第十節 高斯 - 雅可必積分法

Visual Basic

高斯 - 雅可必積分法 (Gauss-Jacobi Quadrature) 是利用正交多項式之一的雅可必多項式 (Jacobi polynomial) 取代高斯積分法中的基礎函數$f_i(x) = x^{i-1}$，並利用雅可必多項式的根作爲配置點，以簡化求解配重係數的方法。

表 6.1 典型的雅可必多項式

α	β	$N = 0$	$N = 1$	$N = 2$	$N = 3$
0	0	1	$2x-1$	$6x^2-6x+1$	$20x^3-30x^2+12x-1$
1	0	1	$3x-1$	$10x^2-8x+1$	$35x^3-45x^2+15x-1$
2	0	1	$4x-1$	$15x^2-10x+1$	$56x^3-63x^2+18x-1$
0	1	1	$\frac{3}{2}x-1$	$\frac{25}{6}x^2-5x+1$	$\frac{35}{4}x^3-15x^2+\frac{15}{2}x-1$
1	1	1	$2x-1$	$6x^2-6x+1$	$14x^3-21x^2+9x-1$
2	1	1	$\frac{5}{2}x-1$	$\frac{49}{6}x^2-7x+1$	$21x^3-28x^2+\frac{21}{2}x-1$

表 **6.2**　正交多項式 $(\alpha=\beta=0)$ 的根及方程式 (6-9.3) 所定義的配重係數

N	x_i	ω_i
1	0.50000 00000	0.66666 66667
2	0.21132 48654 0.78867 51346	0.50000 00000 0.50000 00000
3	0.11270 16654 0.50000 00000 0.88729 83346	0.27777 77778 0.44444 44444 0.27777 77778
4	0.06943 18442 0.33000 94783 0.66999 05218 0.93056 81558	0.17392 74226 0.32607 25774 0.32607 25774 0.17392 74226
5	0.04691 00771 0.23076 53450 0.50000 00000 0.76923 46551 0.95308 99230	0.11846 34425 0.23931 43353 0.28444 44444 0.23931 43353 0.11846 34425
6	0.03376 52429 0.16939 53086 0.38069 04070 0.61930 95931 0.83060 46933 0.96623 47571	0.08566 22462 0.18038 07865 0.23395 69678 0.23395 69678 0.18038 07865 0.08566 22462

註：$\int_0^1 x^\beta(1-x)^\alpha P_j(x)P_N(x)dx=0 \quad j=0,1,2,\cdots,N-1$

(1) 上列雅可必方程式中，假設 $\alpha=\beta=0$ 。

(2) 配置點爲$x_1=0$, $x_{N+2}=1$ ，其他x_2 , x_3 , … , x_{N+1}如上表所列。

(3) N=1 時，$\omega_1=\omega_3=1/6$, $\omega_2=2/3$。

(4) N≥2 時，$\omega_1=\omega_{N+2}=0$。

雅可必多項式是一種正交多項式，其定義爲

$$\int_0^1 x^\beta(1-x)^\alpha P_j(x)P_N(x)dx=0 \qquad j=0,1,2,\cdots,N-1 \qquad \textbf{(6-10.1)}$$

或寫成

$$P_N^{(\alpha,\beta)}(x)=\sum_{i=0}^{N}(-1)^{N-i}\gamma_i\ x^i \qquad \textbf{(6-10.2)}$$

其中

$$\gamma_i = \frac{N-i+1}{i} \cdot \frac{N+i+\alpha+\beta}{i+\beta} \cdot \gamma_{i-1} \qquad i = 1, 2, \cdots, N$$

$$\gamma_0 = 1$$

典型的雅可必多項式如附表 6.1 所示。方程式 (6-10.1) 中，$x^{\beta}(1-x)^{\alpha}$ 稱爲積分配重函數，以$W(x)$ 表示之。除了特殊目的以外，通常採用$\alpha = \beta = 0$。此時，典型的配置位置x_i及配重係數ω_i分別如表 6.2 所示。而當α與β不爲零的情況，請參考 [6 , 7] 或本書第十章。

例題 6-6　重解設計問題 D-VI

試以高斯 - 雅可必積分法重解設計問題 D-VI。

解：

TOP-DOWN 設計：

主程式結構請參照例 6-4。

副程式 **GaussQuadrature(MQUAD, KeyPoint, Weight, XZ, Upper, Lower, Sum, KFLAG)**

If 0<MQUAD<=6 Then	Else
KFLAG = 1	KFLAG = 0
JFIRST = KeyPoint(MQUAD)	
JLAST = KeyPoint(MQUAD+1)-1	
PANEL = Upper – Lower	
SUM = 0	
For I = JFIRST to JLAST	
X = PANEL *XZ(I) + Lower	
SUM = SUM + F(X)	
SUM = PANEL * SUM	
RETURN	

符號說明：

大部分符號請參照例 6-4

KEY：　w_j及x_j之存放位置指示

MQUAD：正交多項式的次數，N

WEIGHT：配重係數，ω_i

XZ： 配重位置， x_i

程式列印：

```
'*****************************************************
' GAUSS QUADRATURE INTEGRATION
'*****************************************************
'
Private Sub GaussQuadratureIntegration(Xpos, Ypos)
'
' F(X)=FUNCTION TO INTEGRATE
' KeyPoint=POLYNOMIAL ID.
' KFLAG=ERROR FLAG
'            0:NON-CONVERGENT
'            1:ALL OK
' LOWER=LOWERLIMIT
' SUM=RESULT
' MQUAD=ORDER OF POLYNOMIAL
' UPPER=UPPER LIMIT
' WEIGHT=WEIGHT FUNCTION
' XZ=COLLOCATION POINTS
'
Dim KeyPoint(7), XZ(23), Weight(23)
MErr = 1E-16
CA0 = 0.2: ' INITIAL CONCN.
Do
      Do
            Cls
            Lower = 0
            Print "Degree of Quadrature Polynomial MQUAD= ";
            MQUAD = Val(InputBox("Degree of Polynomial (<= 6); MQUAD= ", "", MQUAD, Xpos,
Ypos))
            Print MQUAD
      Loop While MQUAD > 6 Or MQUAD < 1
      Print
      Print "*************************************************************"
      Print "    CONVERSION        TIME(HR)    TEMPERATURE(C)"
      Print "-------------------------------------------------------------------"

'
'== Initialize Quadrature
'
      Call Quadrature(KeyPoint, XZ, Weight)
      RxnTime = 0

      Do
            Upper = Lower + 0.1
            Call GaussQuadrature(MQUAD, KeyPoint, Weight, XZ, Upper, Lower, Sum, KFLAG)
```

```
            If (KFLAG <> 0) Then
                RxnTime = CA0 * Sum + RxnTime
                Print Format(Upper, "          0.000E+00    ");
                Print Format(RxnTime, "          0.0000E+00    ");
                Print Format(T(Upper), "          0.0000E+00 ")
            End If
            If Upper < 0.9 - MErr Then
                Lower = Lower + 0.1
                RangeID = 0
            Else
                RangeID = 1
            End If
        Loop While RangeID = 0
        Print "************************************************************"
        NEXTJOB$ = InputBox("Run Next Job <Y/N>?", "NEXT JOB", "N", Xpos, Ypos)

    Loop While NEXTJOB$ <> "N"
    End Sub


    Private Function F(XA)
    R = 1.987: ' GAS CONST.
    E = 15300: ' ACTIVATION ENERGY
    N = 1.5:    ' REACTION ORDER
    CA0 = 0.2: ' INITIAL CONCN.
    K0 = 150000000000#: 'ARRHENIUS CONSTANT
    F = 1 / (K0 * Exp(-E / R / T(XA)) * CA0 ^ N * (1 - XA) ^ N)
    End Function


    Private Function T(XA)
    T0 = 293: ' INITIAL TEMP.
    CA0 = 0.2: ' INITIAL CONCN.
    CP = 0.95: ' HEAT CAPACITY
    HR = 35000!: ' HEAT OF REACTION
    RHO = 1100: ' DENSITY
    T = T0 + HR * CA0 * XA / RHO / CP
    End Function


    Public Sub Quadrature(KeyPoint, XZ, Weight)
    '
    ' ==>INITIALIZE GQUAD
    KeyPoint(1) = 1
    KeyPoint(2) = 4
    KeyPoint(3) = 6
    KeyPoint(4) = 9
    KeyPoint(5) = 13
```

```
KeyPoint(6) = 18
KeyPoint(7) = 24
'
' COLLOCATION POINTS
XZ(1) = 0
XZ(2) = 0.5
XZ(3) = 1
XZ(4) = 0.2113248654
XZ(5) = 0.7886751346
XZ(6) = 0.1127016654
XZ(7) = 0.5
XZ(8) = 0.8872983346
XZ(9) = 0.0694318442
XZ(10) = 0.3300094783
XZ(11) = 0.6699905218
XZ(12) = 0.9305681558
XZ(13) = 0.0469100771
XZ(14) = 0.230765345
XZ(15) = 0.5
XZ(16) = 0.7692346551
XZ(17) = 0.953089923
XZ(18) = 0.0337652429
XZ(19) = 0.1693953068
XZ(20) = 0.380690407
XZ(21) = 0.6193095931
XZ(22) = 0.8306046933
XZ(23) = 0.9662347571
'
' QUADRATURE WEIGHT FUNCTIONS
Weight(1) = 0.1666666667
Weight(2) = 0.6666666667
Weight(3) = 0.1666666667
Weight(4) = 0.5
Weight(5) = 0.5
Weight(6) = 0.2777777778
Weight(7) = 0.4444444444
Weight(8) = 0.2777777778
Weight(9) = 0.1739274226
Weight(10) = 0.3260725774
Weight(11) = 0.3260725774
Weight(12) = 0.1739274226
Weight(13) = 0.1184634425
Weight(14) = 0.2393143353
Weight(15) = 0.2844444444
Weight(16) = 0.2393143353
Weight(17) = 0.1184634425
Weight(18) = 0.0856622462
Weight(19) = 0.1803807856
```

```
Weight(20) = 0.2339569678
Weight(21) = 0.2339569678
Weight(22) = 0.1803807856
Weight(23) = 0.0856622462
End Sub


Public Sub GaussQuadrature(MQUAD, KeyPoint, Weight, XZ, Upper, Lower, Sum, KFLAG)
'
' Integration By Gauss Qudrature
'
' MQUAD    =ORDER OF POLYNOMIAL
' KeyPoint        =POLYNOMIAL ID.
' Weight          =WEIGHT FUNCTION
' XZ              =COLLOCATION POINTS
' Upper           =UPPER LIMIT
' Lower           =LOWERLIMIT
' Sum             =RESULT
' KFLAG           =ERROR FLAG
'                 0:NON-CONVERGENT
'                 1:ALL OK
' F(X)            =External FUNCTION for Integration
'
If (MQUAD > 6 Or MQUAD <= 0) Then
    KFLAG = 0
    Print "INVALID ORDER USED"
Else
    KFLAG = 1
    JFIRST = KeyPoint(MQUAD)
    JLAST = KeyPoint(MQUAD + 1) - 1
    PANEL = Upper - Lower
    Sum = 0
    For I = JFIRST To JLAST
        Sum = Sum + Weight(I) * F(PANEL * XZ(I) + Lower)
    Next I
    Sum = PANEL * Sum
End If
End Sub
```

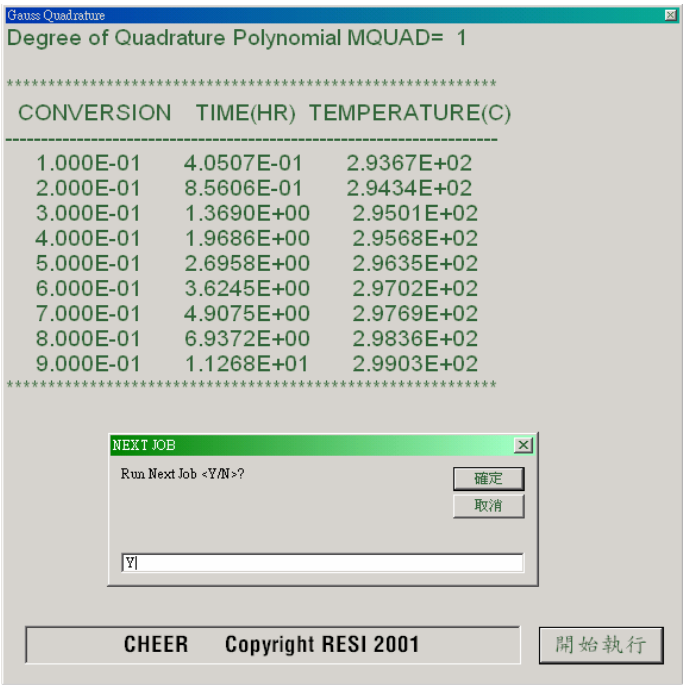
註：同時產生配置點 x_i 及配置係數 ω_i 的程式設計法詳見本書第十章。

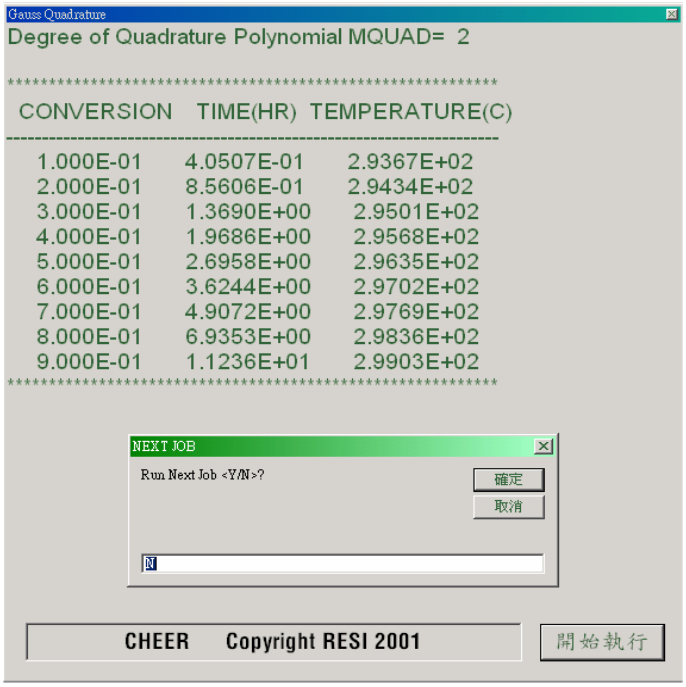
副程式使用方法：

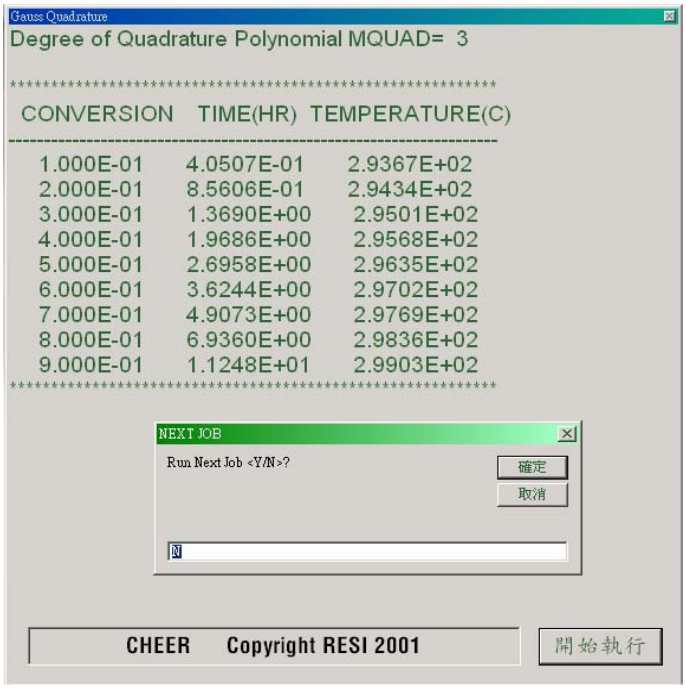
1. 副程式 **Public Sub Quadrature(KeyPoint, XZ, Weight)** 使用方法
   (1) 直接呼叫 Call Quadrature(KeyPoint, XZ, Weight)。
   (2) 副程式會傳回多項式次數指標 KeyPoint、配置點 XZ 及配重函數 Weight。

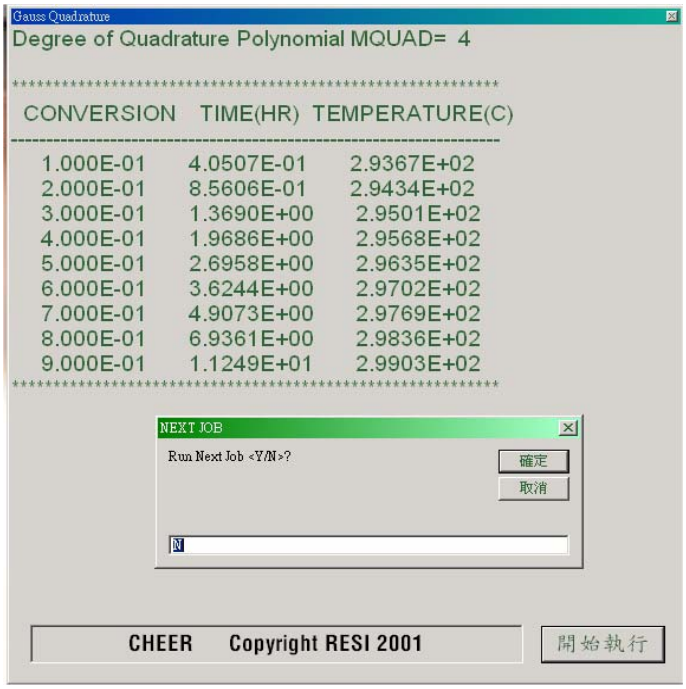
2. 副程式 **Public Sub GaussQuadrature(MQUAD, KeyPoint, Weight, XZ, Upper, Lower, Sum, KFLAG)** 使用方法：
   (1) 決定配置點數 MQUAD、多項式次數指標 KeyPoint、配重函數 Weight、配置點 XZ、積分上限 Upper、積分下限 Lower。
   (2) 呼叫副程式，Call GaussQuadrature(MQUAD, KeyPoint, Weight, XZ, Upper, Lower, Sum, KFLAG)。
   (3) 副程式於計算後傳回積分結果 Sum 及狀態指標 KFLAG。

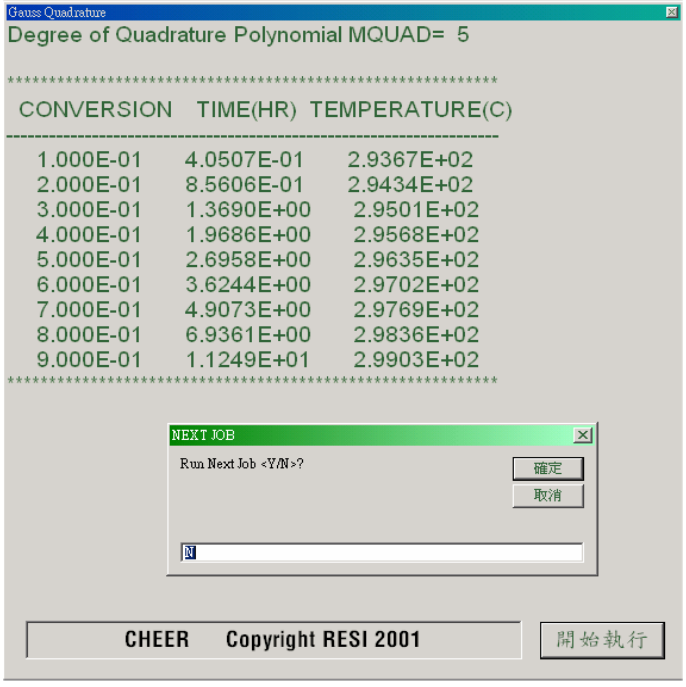
執行結果：

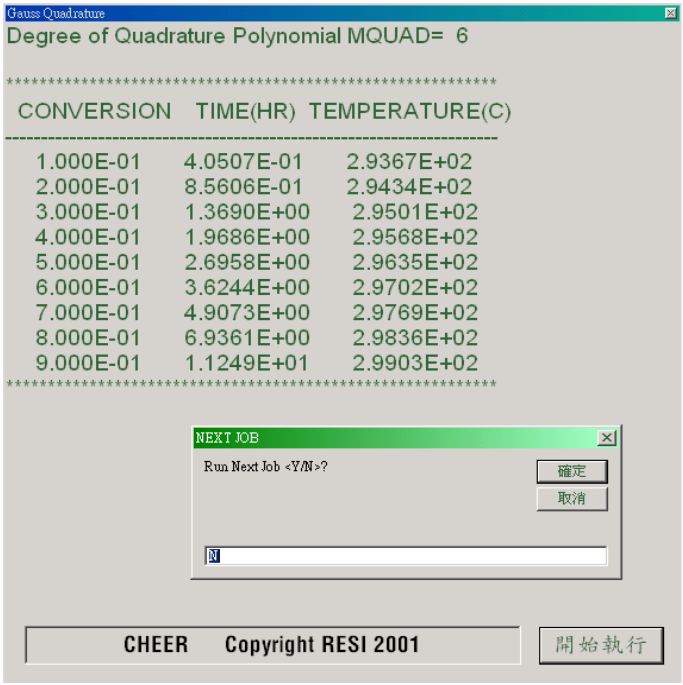



Gauss Quadrature
Degree of Quadrature Polynomial MQUAD= 2
CONVERSION TIME(HR) TEMPERATURE(C)
1.000E-01 4.0507E-01 2.9367E+02
2.000E-01 8.5606E-01 2.9434E+02
3.000E-01 1.3690E+00 2.9501E+02
4.000E-01 1.9686E+00 2.9568E+02
5.000E-01 2.6958E+00 2.9635E+02
6.000E-01 3.6244E+00 2.9702E+02
7.000E-01 4.9072E+00 2.9769E+02
8.000E-01 6.9353E+00 2.9836E+02
9.000E-01 1.1236E+01 2.9903E+02
NEXT JOB
Run Next Job <Y/N>?
確定
取消
N
CHEER Copyright RESI 2001
開始執行


Gauss Quadrature
Degree of Quadrature Polynomial MQUAD= 3
CONVERSION TIME(HR) TEMPERATURE(C)
1.000E-01 4.0507E-01 2.9367E+02
2.000E-01 8.5606E-01 2.9434E+02
3.000E-01 1.3690E+00 2.9501E+02
4.000E-01 1.9686E+00 2.9568E+02
5.000E-01 2.6958E+00 2.9635E+02
6.000E-01 3.6244E+00 2.9702E+02
7.000E-01 4.9073E+00 2.9769E+02
8.000E-01 6.9360E+00 2.9836E+02
9.000E-01 1.1248E+01 2.9903E+02
NEXT JOB
Run Next Job <Y/N>?
確定
取消
N
CHEER Copyright RESI 2001
開始執行


Gauss Quadrature
Degree of Quadrature Polynomial MQUAD= 4
CONVERSION TIME(HR) TEMPERATURE(C)
1.000E-01 4.0507E-01 2.9367E+02
2.000E-01 8.5606E-01 2.9434E+02
3.000E-01 1.3690E+00 2.9501E+02
4.000E-01 1.9686E+00 2.9568E+02
5.000E-01 2.6958E+00 2.9635E+02
6.000E-01 3.6244E+00 2.9702E+02
7.000E-01 4.9073E+00 2.9769E+02
8.000E-01 6.9361E+00 2.9836E+02
9.000E-01 1.1249E+01 2.9903E+02
NEXT JOB
Run Next Job <Y/N>?
確定
取消
N
CHEER Copyright RESI 2001
開始執行


Gauss Quadrature
Degree of Quadrature Polynomial MQUAD= 5
CONVERSION TIME(HR) TEMPERATURE(C)
1.000E-01 4.0507E-01 2.9367E+02
2.000E-01 8.5606E-01 2.9434E+02
3.000E-01 1.3690E+00 2.9501E+02
4.000E-01 1.9686E+00 2.9568E+02
5.000E-01 2.6958E+00 2.9635E+02
6.000E-01 3.6244E+00 2.9702E+02
7.000E-01 4.9073E+00 2.9769E+02
8.000E-01 6.9361E+00 2.9836E+02
9.000E-01 1.1249E+01 2.9903E+02
NEXT JOB
Run Next Job <Y/N>?
確定
取消
N
CHEER Copyright RESI 2001
開始執行



結果討論：

1. 由以上測試結果顯示 MQUAD = 3 即可得到相當準確的結果，此時共只需計算函數 3x9＝27 次，即可積分到 $x_A = 0.9$ 。
2. 幾種積分方法效率比較如下：

積　分　法	*f(x)* 計算次數
梯形積分法	64
辛普森積分法	32
龍勃格積分法	64
高斯積分法	27

3. 如果函數並不是用方程式表示，而是利用數據表的型式表示時，利用高斯積分法亦極爲方便。計算策略如下：
   (1) 由高斯雅可必積分法決定配置位置 x_i 。
   (2) 利用 Lagrange 內插法求得 $f(x_i)$ 。
   (3) $I = \int f(x)dx = \sum \omega_i f(x_i)$

參考文獻

1. Perry, J.H. ed., "Chemical Engineer's Handbook" 5th ed., McGraw-Hill, (1973).

2. Holland, C.D. and R.G. Anthony, "Fundamentals of Chemical Reaction Engineering", (1981).

3. Richard, L.F. and J.A. Gaunt, "The Deferred Approach to the Limit," Trans. Roy. Soc. London, 226A, 300 (1927).

4. Foust, A.S., L.A. Wenzel, C.W. Clump, L. Maus, and L.B. Andersen, "Principles of Unit Operations" 2nd ed. John Wiley (1980).

5. Hirschfelder, J.O., C.F. Curtiss, and R.B. Bird, "Molecular Theory of Gas and Liquids".

6. Villadsen, J., and M.L. Michelsen, "Solution of Differential Equation Models by Polynomial Approximation", Prentice Hall, (1978).

7. Finlayson, B. A., "Nonlinear Analysis in Chemical Engineering" (1980).

習　題

1. 試證明方程式 (6-3.10) 的截尾誤差爲

$$\varepsilon_t = -\frac{h_1 h_2}{6(h_1+h_2)}\left[\, h_2 f'''(\xi_1) + h_1 f'''(\xi_2)\right]$$

其中 $x_0 \le \xi_1 \le x_0 + h_2$ ， $x_0 - h_1 \le \xi_2 \le x_0$ 。

2. 續問題 1，試求當 $h_1 = h_2 = h$ 時，方程式 (6-3.10) 及其截尾誤差分別爲何。

3. 試證明方程式 (6-3.17) 的截尾誤差爲

$$\varepsilon_t = \frac{24}{5} h^4 f^{v}(\xi)$$

其中 $x_n - 2h \le \xi \le x_0 + 2h$

4. 試求下列諸式之積分值，並比較梯形積分法，辛普森積分法及龍勃格積分法之優劣。

(a) $\int_0^{10} x^2 e^{-x} dx$　　(b) $\int_0^{10} \frac{dx}{e^x + e^{-x}}$　　(c) $\int_0^{10} e^{-x^2} dx$

(d) $\int_0^1 \sin x\,dx$ (e) $\int_0^1 \frac{\sin x}{x}\,dx$

5. 在統計熱力學中，計算物質之定容比熱時，須利用以下的 Debye 函數[5]：

$$D(x) = 3x^{-3}\int_0^x \frac{y^3}{e^y - 1}dy$$

試作 *D(x)* 對 *x* 之圖形，其中 *0 < x < 100*。

6. 逸壓 (Fugacity) 可用於表示一等溫系統可獲得的功。理想氣體的逸壓 f 等於其壓力，但是對真實氣體而言，逸壓與壓力關係爲

$$\ell n\frac{f}{p} = \int_0^p \frac{C-1}{p}dp$$

其中 *C* 爲壓縮係數，需利用實驗測定。已知甲烷的壓縮係數與壓力之關係[1]爲：

P (atm)	*C*	*P (atm)*	*C*
1	0.9940	80	0.3429
10	0.9370	120	0.4259
20	0.8683	160	0.5252
40	0.7043	250	0.7468
60	0.4515	400	1.0980

試寫一程式可讀入壓力 *P* 及壓縮係數 *C* 的值，並利用這些數據求得對應表中壓力之逸壓 *f*。假設壓縮係數在表列值中成線性關係 (較正確的做法是用一多項式表示，詳見第七章；或利用插值法)。當壓力趨近於 0 時，壓縮係數 $C=1$。

7. 利用飽和溫度爲T_s之蒸氣在管外凝結，而使流量爲m lb/hr流經套管式熱交換器之流體由入口溫度T_1升溫至出口溫度T_2，如附圖 6.6 所示。此熱傳系統之能量平衡方程式爲

$$h(T_s - T)\pi D dL = mC_p dT$$

積分得所需熱交換器長度爲

$$L = \frac{m}{\pi D}\int_{T_1}^{T_2} \frac{C_p dT}{h(T_s - T)}$$

其中 *D* 爲管徑，*h* 爲管側熱傳係數，可利用下試求之 [4]

$$h = \frac{0.023\,k}{D}\left(\frac{4m}{\pi D\mu}\right)^{0.8}\left(\frac{\mu C_p}{k}\right)^{0.4}$$

其中熱容量 C_p，流體黏度 μ 及熱傳導率 k 均爲溫度 T 的函數。試設計一程式以求得所需熱交換器長度。

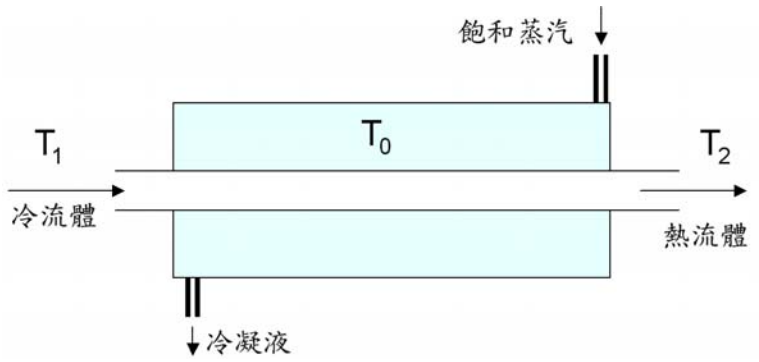


圖 6.6　熱交換系統

數據：

流體：乙二醇 (液體)

流量：m＝45,000 1b/hr

入口溫度：$T_1 = 0°F$

出口溫度：$T_2 = 180°F$

蒸氣溫度：$T_5 = 250°F$

直徑：D = 1.032 吋

熱容量：Cp = 0.53 + 0.00065T　(Btu/lb.°F)

熱傳導率：$k \cong 0.153$　(Btu/hr · ft · °F)

黏度：

μ(lb/ft · hr)	242	82.1	30.5	12.6	5.57
T(°F)	0	50	100	150	200

8. Clapeyron 方程式可用於表示蒸氣壓力與熱力學性質間的關係：

$$\frac{d\ell nP}{dT} = \frac{\Delta H_v}{RT^2}$$

其中　P = 蒸氣壓力

T = 絕對溫度

ΔH_v = 蒸發潛熱

R = 氣體定律常數

此方程式只在極有限的溫度及壓力範圍內方爲正確。將上式改寫，並由已知溫度T_0及壓力P_0開始積分，則可求得對應任何溫度時的蒸氣壓力。

$$\int_{p_0}^{p} d\ell np = \int_{T_0}^{T} \frac{\Delta H_v}{RT^2} dT$$

或

$$\ell n \frac{P}{P_0} = \int_{T_0}^{T} \frac{\Delta H_v}{RT^2} dT$$

欲求蒸氣壓力P，則需計算右側的積分式。但由於ΔH_v通常並不能表示成溫度T的簡單函數，因此，必須藉助於數值積分才能求得P-T關係。已知一物質之ΔH_v與溫度關係如表 6.3 所示。$R = 0.01614$ Btu/1b - °R ，$P_0 = 0.14224$ lb/in^2 ，T_0 = 330°R 。試寫一程式可求出對應於表列溫度之蒸氣壓。(註： 430°R 時，正確的蒸氣壓爲 12.00 1b/in^2)

表 6.3　一物質之ΔH_v與溫度關係

T(°R)	ΔH_v(Btu/lb)	T(°R)	ΔH_v(Btu/lb)
330	81.577	390	75.853
340	80.617	400	74.885
350	79.663	410	73.906
360	78.714	420	72.913
370	77.764	430	71.903
380	76.812		

9. 一名積體電路製造工廠品管工程師，希望求得一製程流量計在一小時內的平均值，如果他只希望

(a) 讀二次數據

(b) 讀三次數據

(c) 讀四次數據

則他應在哪一時刻讀數據，才能獲得最佳的平均值？他應該如何計算？