MATLAB 的積分計算與程式設計觀念

last modified May 27, 2010

積分的數值計算經常出現在數學 (統計) 問題上。單純的積分問題只需一兩個指令便可輕易解決, 複雜的可能需要動要到程式設計的技巧。本單元藉由處理與積分計算相關的問題, 帶出一些程式設計相關的觀念與技巧, 做爲未來解決更複雜問題的基礎。

本章將學到關於程式設計

MATLAB的程式結構、程式的邏輯概念與程式檔案的管理。

⟨本章關於 MATLAB 的指令與語法⟩
操作元 (operators): .^ ./
指令: for, tic, toc, area, alpha, factorial, trapz, quad, pause, inline
語法: for 迴圈的建立與技巧, 匿名函數的介紹 f=@(x)。

1 背景介紹

函數的定積分$\int_a^b f(x)dx$ 可以解釋爲函數圖形在直線 $x = a$及$x = b$ 之間與 X 軸所圍的面積, 如圖1所示的陰影面積。

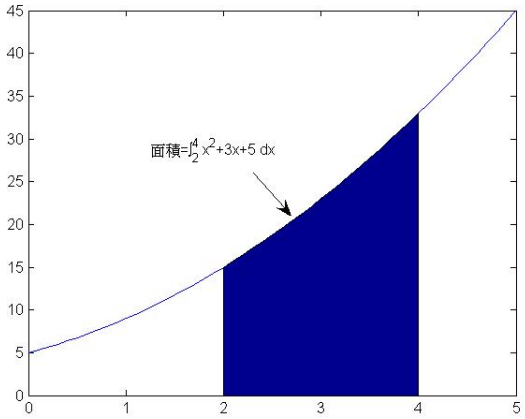

圖 1: 積分的幾何意義

這個不規則形的面積也可以所謂的黎曼積分來定義, 適用於數值積分的計算:

黎曼積分(Riemann Integral) **的定義**:

將一個閉區間$[a, b]$任意分割成 n 個小區間, 其分割點爲 $a < x_1 < x_2 < \cdots < x_{n-1} < b$, 這些小區間的寬度表示爲 $\triangle x_1, \triangle x_2, \cdots, \triangle x_n$。這個數值

$$\sum_{k=1}^{n} f(x_k)\triangle x_k$$

稱爲函數$f(x)$在這些區間的的黎曼和 (Riemann Sum)。當區間寬度$\triangle x_k \to 0$時

$$\lim_{n\to\infty} \sum_{k=1}^{n} f(x_k)\triangle x_k$$

稱爲函數$f(x)$在區間$[a, b]$的黎曼積分。

一般定積分的數值計算上多應用黎曼積分的觀念, 本單元的探討以此爲基礎。爲方便

說明本單元內容, 以下均假設所有小區間的寬度$\triangle x_k$一致, 即

$$h = \triangle x_k = \frac{b-a}{n}$$

黎曼和寫成

$$\sum_{k=1}^{n} f(x_k)h = \sum_{k=1}^{n} f(a+kh)h \tag{1}$$

代表 n 個長方形的面積和。圖2是這黎曼和的示意圖。當 h 越小時 (n 越大), 陰影部分的長方形越窄, 也因此上緣越貼近函數的曲線。當 h 趨近於0時 ($n \to \infty$), 這無限多個長方形的面積和即爲黎曼積分。

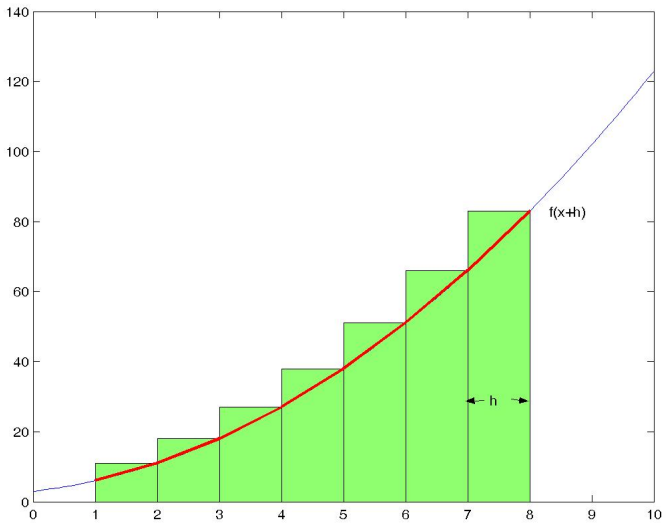

圖 2: 積分的定義: 黎曼和

2 練習

範例1: 利用上述黎曼和公式(1), 寫一支程式計算

$$\int_0^3 x + 3 \, dx$$

其中設 $h = 1$。請比較黎曼和與實際的積分值。如果設 $h = 0.5$, 結果又如何? 當$h = 0.25$是不是更接近了呢?

當 h 值愈趨近0時, 數值計算的黎曼和將與實際值愈接近, 不過程式的寫作越麻煩, 因爲要加入的長方形面積越多。此時上個單元介紹的迴圈技巧將可派上用場。

範例2: 利用迴圈技術計算範例1的數值積分值, 假設 h=0.01、0.001時。

```
p=[1  3]; % 多項式函數的係數
h=0.01;
x=h:h:3; %所有的 x 值
n=length(x);
A=0; %黎曼和, 先預設爲 0
for i=1:n
    A=A+polyval(p,x(i))*h; %面積累加
end
```

上述程式比較關鍵的地方在 x 值的設定, 它決定了長方形位置與大小。這些設定與積分的上下限有關, 因此比較好的程式寫法, 應該納入上下限的變數, 上述的程式可以修改爲

```
p=[1  3];  h=0.01;
a=0; b=3; %積分的下限與上限
x=a+h:h:b;
n=length(x);
A=0; %黎曼和, 先預設爲 0
for i=1:n
    A=A+polyval(p,x(i))*h; %面積累加
end
```

爲節省空間方便觀看, 將一些參數的指令集中在同一行。這個數值積分的方法即俗稱的長方形法。當 x 值略作調整, 可以做出不同位置的長方形。譬如 x=a:h:b-h 或 x=(a+h/2):h:(b-h/2)。不同的設定是否得到不一樣的結果呢? 讀者不妨自己試試看。

利用迴圈的方式, 可以順利解決累加的問題, 但你是否發現當 h 愈小時, 所花費的計算時間愈多。別忘了 MATLAB 的專長, 爲避免過多的迴圈造成較差的執行效率, 請利用矩陣的運算解決函數計算與累加的問題, 並比較與迴圈技術在計算時間上的差別 (h 愈小差別愈明顯)。計算程式執行時間, 參考時間指令:tic, toc, 例如

範例: 計算$\sum_{x=1}^{5}(x+3)$

程式:
```
tic
x=1:5;
answer=sum(x+3)
t=toc %顯示自 tic 後的執行時間
```

Tips: 程式語言的邏輯與數學或一般生活邏輯有些差異, 因此寫程式時, 千萬別一口氣寫完才執行, 這樣會造成除錯上的困難, 浪費不必要的時間。對初學者而言, 也不需要從第一行開始寫起, 可以先寫中間的主體部分或甚至後面的結果, 再逐一根據指令或變數的需求穿插補足。過程中, 可以隨時儲存程式並執行, 確定每個指令與變數的使用都是正確的。

範例3: 計算定積分

$$\int_{-5}^{5} x^2 + 3x + 5 \, dx$$

的黎曼和。建議先畫出函數圖形, 再利用迴圈法與矩陣法分別計算積分值, 並比較當 h=0.0001 時, 兩者運算時間的差異。

所謂矩陣法就是不使用迴圈的方式, 利用向量矩陣的方式直接計算每塊長方形的面積, 並加總得到面積和, 即

$$\begin{aligned} A &= f(-5+h)h + f(-5+2h)h + \cdots + f(5)h \\ &= (f(-5+h) + f(-5+2h) + \cdots + f(5))h \end{aligned}$$

這個面積和 A 相當於函數和乘上 h, 以 MATLAB 指令來表示, 可以寫成

```
p=[1  3  5];  h=0.01;
a=-5; b=5;
x=a+h:h:b;
f=polyval(p,x);
A=sum(f)*h;
```

範例4: 如果能畫出積分計算過程的長方形, 會不會對積分計算的瞭解有幫助呢? 試試看能不能利用迴圈的技巧, 沿著函數的曲線, 畫滿一排長方形。至於長方形的寬度可以從程式來控制。

爲 MATLAB 圖上的曲線塗上顏色的方式如下:

```
x=[4 6];              註: 準備兩個點的值畫一條直線
y=[5 5];
area(x,y)             註:area 與 plot 的不同在於塗色
axis([0 10 0 10])     註: 必要改變座標範圍方便觀察
```

最後一個指令 axis 用來改變圖形的座標範圍,MATLAB 自行決定範圍有時並不符實際的要求, 矩陣裡的四個數字分別代表 [xmin xmax ymin ymax], 其意義如字面所示。當利用迴圈技術計算面積並繪出面積圖時, 由於電腦速度的關係, 感覺上好像每塊面積是同時畫上去的, 實際上並非如此。此時可以利用 pause 指令來緩和速度, 強迫電腦停止運作或停止一定的時間, 譬如在 area(x,y) 後面加上

```
pause(1);
```

強迫電腦停止 1 秒鐘, 在迴圈裡面呈現出來的效果有如動畫般的神奇。不妨試著結合前一個範例積分程式, 配合塗上顏色與時間暫停, 玩玩動畫。試試看下面這段程式碼的表現

```
x=0:0.1:7;
f=@(x) polyval([1  3  5],x);%匿名函數的定義
plot(x,f(x))
hold on
for i=1:5
      area([i i+1], [f(i+1) f(i+1)])
end
alpha(0.2)
hold off
```

這裡介紹了一個指令 alpha, 讓圖形具穿透性, 讀者只要將這個指令拿掉, 讓程式跑一次, 觀察兩者的不同, 便瞭解「穿透性」的意思。alpha 指令裡面的數字代表「穿透程度」, 什麼意思呢? 改改它 (0 到 1 之間), 不就知道了嘛!

上述程式引進了匿名函數的做法, 匿名函數可以當作是函數型的變數, 經過定義後 (如第二行), 直接以函數表示的方式計算函數值, 如第三與第六行的做法。匿名函數爲程式帶來兩個方便: 其一, 程式看起來更乾淨俐落; 其二, 程式的表達更接近紙筆的方式。更多關於匿名函數的使用可以參考線上手册或多留意他人的作品, 常有出乎自己意料之外的做法, 在寫作程式之餘, 憑添樂趣。

範例5: 如上題, 將每個小面積從長方形改爲梯形, 如圖 3, 並改寫黎曼和的數值定義 (1), 重新計算一次。另一方面, 請嘗試逐漸降低 h 值, 觀察計算的結果, 與前面長方形面積有何不同? 兩個公式孰優孰劣?

數值積分的相關計算方法很多, 其中的「辛普森法則 Simpson's Rule」比梯形法更爲準確。讀者有空不妨去找找看什麼是「辛普森法則,」它的原理及最後的公式。從長方

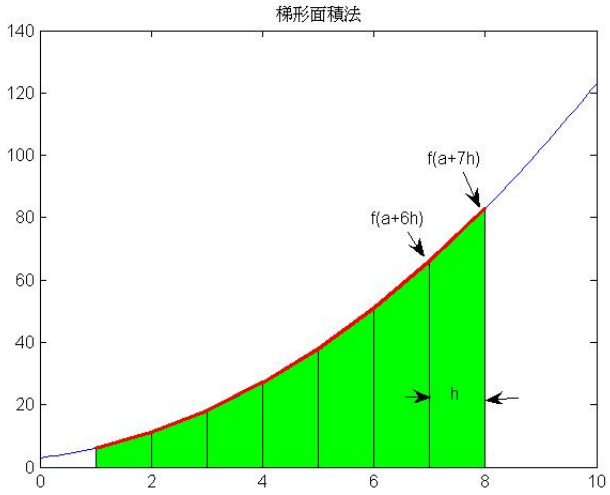

圖 3: 梯形面積的黎曼和

形法或梯形法程式中, 經過適當的修改, 你也可以輕易的寫成利用辛普森法計算的積分程式。

另外,MATLAB 當然也提供了計算積分的指令, 如 trapz 及 quad, 從字面上大約可以猜出是採用了梯形法與辛普森法的積分指令。使用方式如下列程式所示

```
clear
p=[1  3  5];  h=0.01;  a=-5; b=5;
x=a:h:b;
y=polyval(p,x);
A=trapz(x,y);
```

```
a=-5; b=5;%積分下限與上限
f=@(x) x.^2 + 3 * x + 5;
A=quad(f,a,b);
```

這裡的匿名函數將 x 當作向量看待, 因爲後面使用匿名函數計算時, 需要給予向量型態的 x 值。積分指令 quad 根據所定義的函數變數及積分的上下限做積分的計算。

依作者的經驗, 在絕大部分的情況下, quad 的表現稱職, 所以在積分的計算上, quad 是首選。不過, quad 畢竟是內建的指令, 並不操之在我, 當這個指令行不通的時候,[1]我們必須有 B 計畫, 甚至 C 計畫。積分計算的 B 計畫, trapz 梯形法是不錯的選擇, 這個方法不較不受限於函數, 是很好的備胎。

範例6: 計算下列標準常態分配的累積密度函數 CDF

$$P(x) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}} e^{-\frac{z^2}{2}} dz$$

在 $x = -5 : 0.1 : 5$, 繪製 $P(x)$ 的階梯圖 (stairs plot)。

這個問題牽涉到多個積分值的計算, 因此要考慮迴圈技巧的幫忙。在迴圈過程中, 必須儲存每個積分值, 留待迴圈結束後畫圖。另一方面, 每個積分式的下限是 $-\infty$, 在實務上必須找一個夠小的替代值, 至於多小的值才適合, 得要花點時間做些測試, 實務面的東西, 操作時要靈活些, 不能太拘泥, 畢竟解決問題才是最終的目標。建議讀者先試著寫寫看, 再參考下列程式。程式執行結果如圖 4 所示。本題可與指令 normcdf 比較。

```
clear
f=@(x) normpdf(x,0,1);      %定義積分式
x=-5:0.1:5;
n=length(x);
P=zeros(1,n);
for i=1:n
    P(i)=quad(f,-10,x(i));   %以 -10 取代 -∞
end
stairs(x,P)                  %也可以試著採用不同的圖形, 如 plot
ylim([0 1.2])
grid
```

[1]內建指令通常無法考慮到所有可能的情況, 或許對某些特殊的函數出現計算上的問題, 學習程式設計就是要能補這方面的不足, 才不會被一個軟體綁架。

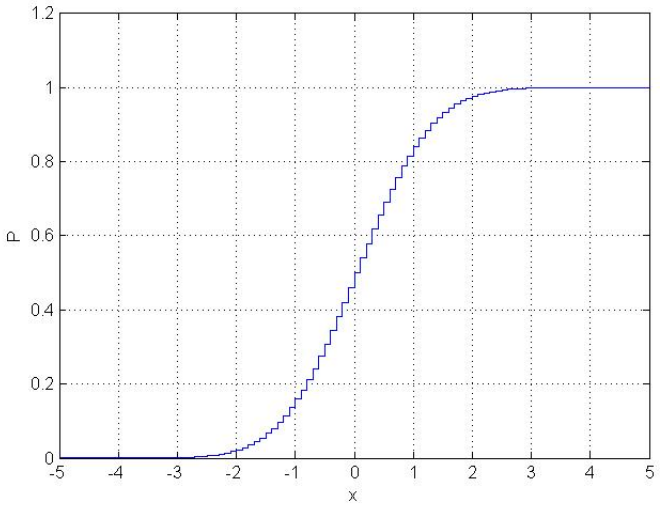

圖 4: 積分計算的應用

3 觀察

1. 數值積分與數值導數一樣都面臨一個 h 值的選擇, 但兩者間還是有所不同。h 值對數值積分的影響的關鍵因素在哪兒? 與數值導數有很大的不同喔!

2. 長方形面積的黎曼和有三種選擇方式,

   - 其一, 長方形範圍超過曲線, 面積總合將超積分值。
   - 其二, 長方形範圍在曲線下面, 面積和低於積分值。
   - 其三, 長方形範圍介於上述兩種之間。這個方法與梯形法的面積是否較爲接近?

4 作業

1. 改寫黎曼和定義 (1), 成爲梯形的面積和。

2. 利用長方形面積的黎曼和計算 $\int_{-5}^{5}(x^3 + 2x^2 + 3x + 4)\ dx$ 並畫出長方形面積,h 值設定爲 0.5。

3. 畫一張圖比較『長方形』法與『梯形』法的準確度。任選一個積分函數與範圍。要怎麼畫? 怎麼比較請自行決定。

4. 寫一支程式採梯形法計算下列的積分: 程式盡量具彈性, 可以讓使用者自行決定上、下限。最好先將圖形畫出來。

   - $\int_0^5 100e^x dx$ (ans: 14741.32)
   - $\int_0^7 \frac{x^2}{2} e^{-x} dx$ (ans:0.9704), Gamma(3,1) 分配
   - $\int_{-1.96}^{1.96} \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} dx$

5. 已知 $p(\mathbf{y}|\theta)$ 爲一概似函數, 其中 $\mathbf{y}$ 及 θ 分別爲樣本與未知參數。假設

$$\begin{aligned} p(\mathbf{y}|\theta) &= (2+\theta)^{125}(1-\theta)^{38}\theta^{34} \\ p(\theta) &= 1, \qquad 0 \le \theta \le 1 \end{aligned}$$

   計算後驗機率的期望值 (即 θ 的貝氏估計) $E(\theta|\mathbf{y})$, 即計算

$$E(\theta|\mathbf{y}) = \frac{\int_0^1 (2+\theta)^{125}(1-\theta)^{38}\theta^{34}\theta d\theta}{\int_0^1 (2+\theta)^{125}(1-\theta)^{38}\theta^{34} d\theta}$$

6. 計算積分還有一種方式叫做「Monte Carlo Integration,」其原理簡單敍述如后,

$$\int_D f(x)dx = \int_D \frac{f(x)}{p(x)} p(x)dx = E\left[\frac{f(x)}{p(x)}\right] \approx \frac{1}{N}\sum_{k=1}^{N} \frac{f(x_k)}{p(x_k)}$$

   其中 $p(x)$ 被當作機率密度函數 (pdf), 而 x_k 是從具 $p(x)$ 分配的母體中抽出的樣本值。積分的問題於是可以寫成函數的期望值。在決定 pdf 函數 $p(x)$ 後, 可以利用抽樣的方式, 取出相當數量的樣本, 再以樣本計算函數值 $f(x)/p(x)$ 及其平均數來近似期望值。其準確度與 $p(x)$ 的選擇、樣本數大小都有關係。當然, 當樣本數趨近無限大時, 不論選擇的 $p(x)$ 爲何, 總能逼近完美的積分值。但適當的選擇卻可以以較少的樣本得到最準確的結果。原則上 $p(x)$ 的選擇與接近函數 $f(x)$ 的外形爲佳。

利用此法, 重做作業4的積分問題, 需說明使用的 $p(x)$ 與樣本數, 譬如計算 $\int_0^7 f(x)dx$, 當選擇均等分配爲其 pdf 函數時, 問題變成

$$\int_0^7 f(x)dx = E\left[\frac{f(x)}{\frac{1}{7}}\right] = 7E\left[f(x)\right] \approx \frac{7}{N}\sum_{k=1}^{N} f(x_k)$$

其中 x_k 爲樣本,N 爲樣本數。