

數學與電腦

汪群超
2005.9



目錄

序	iv
再序：不可變與不可不變：我的程式寫作觀	vii
1 繪製函式圖：認識數學函式	1
1.1 練習	1
1.2 觀察	4
1.3 作業	4
2 MATLAB的數學運算	7
2.1 練習	7
2.2 觀察	9
2.3 作業	9
3 MATLAB的機率分配	13
3.1 練習	13
3.2 觀察	17
3.3 作業	18
4 MATLAB的程式：以微分的計算為例	19
4.1 練習	19
4.2 觀察	22
4.3 作業	22

ii MATLAB 的數學實驗

5	MATLAB程式的迴圈技巧：以積分為例	25
5.1	練習	25
5.2	觀察	28
5.3	作業	28

圖目錄

圖 2.1	散佈圖與相關係數的計算	11
圖 3.1	常態分配的機率密度函數 (pdf)	14
圖 3.2	二項分配的機率密度函數 (pdf), 以指令 (a)stem(b)bar 繪 製的不連續函數圖。	15
圖 5.1	積分的定義: 黎曼和	26

序

如果學習數學相關的學科是痛苦的，那真是一個天大的誤會。數學長久以來被『妖魔』化了。在許多學生心中有一種無形的恐懼，甚至厭惡。有些人選擇提早擺脫數學的糾纏，但有些人卻一直揮之不去，走到哪裡都會碰到數學，或數學相關的話題或應用的領域。

逃避未必求得正果，逃避只是摀著眼睛假裝看不到，一切的逃避或美其名的以不感興趣迴避，都是錯把數學當成表皮摻有農藥的蘋果。儘管知道裡面好吃，卻不敢去碰。但數學能力對一個人的重要性不會因此消失。因為數學能力的展現不一定用來解決數學問題。或許因為如此隱晦，才會引導學生對數學的學習做出零和的決定：學或不學，而且往往是一輩子的賭注。

這本單元式的講義企圖挽回一般學生對數學莫名的恐懼，進而開始喜歡上它。不管你以前多麼痛恨數學，從此刻起，不計前嫌的再一次面對數學。這一次讓電腦來幫幫忙，透過電腦程式的寫作去了解數學的內涵與精神。數學題材不在深，電腦程式不在精闢，一切都是玩票的。學完後，你不會成為電腦程式專家，更不會變成數學家，但是你可能不再討厭數學，且對電腦程式的運作有些概念。或許不知不覺中，數學與電腦會激盪出你未來求知求學的另一番憧憬。幾句話充作參考

用電腦來解決數學問題，比較輕易的化解對數學的厭惡與對電腦的恐懼。

用電腦來解決數學問題，找不到答案也可以觀察到許多未知的領域。

用電腦來解決數學問題，不知不覺中，觀察、解析問題的能力提昇了。

用電腦來解決數學問題，時間似乎流逝的特別快，你已經浸在裡面了。

用電腦來解決數學問題，看問題的角度變大了、變寬廣些了。

對數學的畏懼來自不當的教學或失敗者的恫嚇。不了解其實學習數學是培養各種領域專長的催化劑。數學不見得是第一線的武器，但它永遠是後勤的資源。常常隱而不見，需要時，卻自然流露。不要小看數學的影響力，它無所不在、無孔不入，你只是沒有得到適當的引導！這本講義透過獨立單元介紹一些統計系學生會接觸到的數學，並結合數學軟體MATLAB，將數學的內涵呈現在螢幕上。這本講義的編排方式不是朝向完整教科書的巨細靡遺，僅作為上課練習的腳本與課後作業的參考，上課的過程仍是必須的。部分內容摘自同學的作品。當學生的數學情緒被激發時，我似乎看到潛藏在他們內心理面，受到壓抑的數理能力，他們的發現往往超過我的預期。

汪群超

2002年7月於台北大學

再序

不可變與不可不變：我的程式寫作觀

教學七年了，這本講義也用了三年。其間經過多次的修改，不管擴編還是刪減，多半是依據上課時學生的反應而來。這本講義其實很多地方寫得不夠詳細，本想進一步將所有細節完整呈現，成為一本書。但幾經思量，仍維持原貌，原因是太詳細的內容會養成學生的依賴心，喪失原先期望學生自己去補足不清楚、不詳細的部分。希望學生藉著這門課拾回過去學得不清不楚的微積分、統計學與線性代數。

這本講義企圖將數學原理以電腦數據圖表的方式呈現出來，再要求同學以文字圖案呈現出其間的條理，這樣的訓練是現今大學生十分欠缺的。說穿了就是「表達的能力」的培養。這可不是說、學、逗、唱之類的表達，而是一種試圖將不易說清楚或難以理解的東西，透過文字、圖表或語言將它交代清楚。這樣的能力絕對需要長時間的訓練，有了這項「絕技，」大學畢業生不必急著說自己學非所用。有太多的事實證明，擁有絕佳的表達能力，放諸四海都餓不著肚子。

表達能力的養成必須按部就班，一點都急不得。可惜的是，莘莘學子不是自作聰明，便是固執己見，往往喜歡憑自己過去的經驗來解決未知的問題，缺乏耐心去熟練不熟悉工具，不願將專注力用在問題的觀察。學習過程像極矇著雙眼亂砍亂殺，到頭來學不到東西還怪老師出太多怪怪的功課，既對升學沒有幫助，也無助以後做生意賺大錢，不多久便放棄了，殊是可惜。告訴他這是未來升官發財的利器，他當你在三娘教子，在家裡聽多了。

以寫作程式為例，每一種程式語言都有其語法規範，該怎麼寫怎麼用，一點也

馬虎不得，連錯一點點都不行，沒得商量的。初學者往往輕忽之，不喜歡被「規範」束縛，不顧老師一再地提醒，愛怎麼寫就怎麼寫，天才般的自己編撰起語法來了，結果當然是錯誤百出，急得老師在一旁乾著急。更有甚之，錯了還不認帳，直呼語法太不人性化，不能隨意更動，學它何用，便率性的打起電動或MSN來了。

寫程式首要遵守語法教條，待熟悉語法規範之後，才能漸漸懂得運用，透過寫一些不痛不癢的小程式，一方面熟悉語法，一方面體驗其威力。漸熟，才慢慢從觀察別人寫的「模範程式」中，瞭解死的語言原來也能玩出活把戲，這才一步步進入寫作程式的精髓，進一步玩出樂趣。這道理亙古不變，古今達人不管學習琴棋書畫，還是打拿摔跌等武藝，無不遵循這樣的哲理¹

能力未至不可變也、學識未敷不得變也、功侯未到不能變也。

學於師已窮其法，不可不變也、友古人已悉其意，不得不變也、

師造化已盡其理，不能不變也。

從「不可變」、「不得變」、「不能變」，到「不可不變」、「不得不變」、「不能不變」，可以作為寫作程式的養成過程。學習之初應謹慎遵循所有的規範，一絲不苟，不能濫用自己的小聰明亂抄捷徑，要聽話、要服從，將老師的交代與叮嚀當作聖旨般遵循，務必做到。如此這般一段時日之後，犯錯愈少，進步愈多，自然而然當變則變，逐漸形成自己的風格。

不能急，成就總在不知不覺中「赫然」被別人發現，絕非刻意營造而能得。別人眼中看到的成就，對自己而言永遠都是平常事而已，只不過在許多小地方比別人好一點點罷了。但別小看這一點點，許許多多的一點點累積起來，那可有多少啊！

汪群超

2005年2月於台北大學

¹ 摘自五絕奇人鄭曼青先生名著「曼髯三論。」

第 1 章

繪製函式圖：認識數學函式

能夠在畫面上看到繪製的數學函式，可以拉進與數學間的疏離感。如果能夠親自動手操作電腦畫出在書本上曾經看過的函數圖形，那會是件令人興奮的事。本單元試圖讓學生了解電腦繪製函式圖形的原理，並藉此熟練MATLAB 這個數學軟體的特色，特別是它的矩陣表示法。

1.1 練習

MATLAB以描點法來繪圖。所以對二維圖形而言，必須先產生函式上對應的(x,y) 點。然後將這些點以符號的方式顯示在一定範圍的座標軸上。下面的範例將協助你成功的在 MATLAB 畫出一張簡單的圖。

範例1: 先練習如何在MATLAB 的環境下 (命令視窗) 建立矩陣 (含 scalar, vector and matrix)。分別建立定義如下的 A,B,C,D,E,F,G,H 八個矩陣，並觀察結果。

輸入方式:

$A = [1 \ 2; 3 \ 4] \ B = [1 \ 2 \ 3] \ C = [5 \ 6 \ 7 \ 8]' \ D = 1 : 10$

$E = C(1 : 2) \ F = A(1, 2) \ G = A(:, 1) \ H = A(2, :) \ I = -5 : 0.5 : 5$

2 MATLAB 的數學實驗

注意:

1. 每個指令的最後若加上分號 (;), 則運算的結果將不會顯示在命令視窗上。
2. 請留意符號 (, : ;) 所代表的意涵, 並記錄下來。
3. 變數 E, F, G, H 是取其他變數的部分內容, 請特別留意其產生的方式及結果, 對於往後 MATLAB 的使用非常關鍵。
4. 利用 MATLAB 的指令 whos 觀察每個矩陣的大小。
5. 在 MATLAB 的語法中, 上述代表矩陣的 A, B, \dots, I 稱為變數, 其內容(值)可以透過等號 (=) 賦予, 且隨時可以被改變。
6. MATLAB 也提供一些指令可以更迅速的產生特殊的矩陣, 試試看這些指令: zeros, ones, eye, diag。使用前可以利用 help 的指令了解使用的方式, 譬如 help zeros。

範例2: Let $A = [1 \ 2; 3 \ 4], B = [5 \ 6; 7 \ 8], c = 3$, 做以下的練習, 並觀察結果

$A + B, A - B, A + c, A * B, A / B, A^c, c * A, A / c,$
 $A . * B, A . / B, A.^c, A', B', (A * B)', B' * A'$

請注意 MATLAB 所使用的運算符號, 如 $+, -, *, /, ^, .'$ 等所代表的意涵。從執行的結果去觀察並記錄下來。

範例3: 繪製函式 $y = f(x) = 2x + 1 \quad 0 \leq x \leq 10$

繪圖前首先必須先將要描繪的點, 依指定的範圍產生出來:

1. 先依繪圖範圍產生 x 方向的所有點, 譬如 $x = 0 : 10$
 2. 再根據函式的關係產生所有相對應 y 方向的點, 譬如: $y = 2 * x + 1$
 3. 再利用 plot 繪製不同型態的圖形。請特別注意 plot 提供的各種繪圖的選項, 試著去改變描點的方式、點的符號、顏色等選項。
-

範例4: 繪製函式

1. $y = f(x) = x^2 + 3x + 5 \quad -5 \leq x \leq 5$
 2. $y = f(x) = x^3 - 10x^2 + 29x - 20 \quad 0 \leq x \leq 7$
-

繪圖的方式同前一題, 不過須注意

1. 描繪的點『數』。
 2. plot(x, y) 中, x 軸的點與對應 y 軸上點的產生, 譬如平方與三次方, 不管是多少個點, MATLAB 的特色都是一次產生 (計算) 所有的點, 請仔細回顧範例1、2的矩陣運算做為參考。
-

範例5: 繪製函式 $y = f(x) = x^4 - 8x^3 + 16x^2 - 2x + 8$

請選擇適當的範圍, 務必看到比較完整的圖。

4 MATLAB 的數學實驗

範例6: 依下列範圍繪製函式 $f(x) = x + \sin(x)$

(1) $1 \leq x \leq 1000$ (2) $1 \leq x \leq 500$ (3) $1 \leq x \leq 100$ (4) $1 \leq x \leq 10$

1.2 觀察

1. 電腦繪圖係以描點的方式畫上去, 因此點與點間的距離越近, 所呈現出來的圖形越平滑。請觀察在一定的範圍內, 點的數量多與少的效果, 也就是調整 x 的間距。
2. 利用 MATLAB 指令計算所畫上去的點數。譬如 length, size 等指令。
3. 改變 plot 指令的參數, 讓畫上去的點做不同的表現, 譬如, 畫出散佈圖。利用 help plot 指令, 看看有些繪圖的技巧可以使用。
4. x 軸的範圍取捨非常重要, 除了給定的範圍外, 請試試看其他範圍。寬一點或窄一點。
5. 為這些圖加上些裝飾, 如 x , y 軸的文字、標題或是格線等等。e.g. xlabel, ylabel, title, grid.
6. 練習將結果 (圖) 貼到 WORD 或其他文書編輯工具 (如 Latex 或中文的 cwTex) 裡面, 做好如何呈現結果的工作。將 MATLAB 產生的圖形貼到其他地方的方式有兩種; (1) 利用 File/Export(7.0版以前) 或 Save As(7.0版以後) 的方式將圖形以檔案的方式儲存下來。(2) 利用 Edit/Copy Figure 的方式將圖形 copy 下來, 再到 WORD 或其他使用的軟體, 直接以 paste 的方式貼回。

1.3 作業

畫出下列函式, x 範圍自訂, 盡量畫出比較完整的函式圖, 並加上必要的裝飾文字。

1. 繪製函式圖: 認識數學函式 5

1. $y = f(x) = \sin(x) + \cos(x)$

2. $y = f(x) = \frac{1 - e^{-2x}}{1 + e^{-2x}}$ 代表指數的指令 `exp`

3. $y = f(x) = \sqrt[3]{\frac{4 - x^3}{1 + x^2}}$

先試試看: $-8^{(1/3)}$ 及 $(-8)^{(1/3)}$ 有何不同, 何者才能正確計算 $\sqrt[3]{-8}$?

MATLAB 究竟如何處理開立方根號? $\sqrt[3]{-8}$ 是否等於 $-\sqrt[3]{8}$? 這一題或許需要用到判斷正負號的指令如 `sign` 及 `abs`, 而開根號可以使用 0.5 次方的方式或是指令 `sqrt`

4. $y = f(x) = \frac{1}{x - 1}$

5. $y = f(x) = \frac{1}{2\sqrt{2\pi}} e^{\frac{-(x-1)^2}{8}}$

6. $y = f(x) = x^{2/3} = \sqrt[3]{x^2}$

7. $y = f(x) = 2x^3 - x^4$

8. $y = f(x) = x\sqrt{4 - x^2}$

補充: 下面的指令可以改善圖形呈現的品質

`x = 0 : 0.5 : 5;`

`y = x.^2 + 2 * x + 3;`

`plot(x,y)` 此時出現的 x 軸刻度間隔為 0.5 假設想改間隔為 1

`set(gca,'xtick',0:1:5)` 當然將 `'xtick'` 改為 `'ytick'` 就可以針對 Y

如果想將線條變粗, 使用 `plot(x,y,'LineWidth',2)`, 裡面的 2 代表線條得寬度, 數字越大越寬。

6 MATLAB 的數學實驗

第 2 章

MATLAB 的數學運算

了解 MATLAB 處理數學公式計算的原理，並藉此進一步熟練 MATLAB 的矩陣表示法及其計算。學習在面對數學公式時，如何利用電腦語言有效率的去計算公式的值。MATLAB提供哪些計算的方式？從對電腦語言的探索，去發覺解決問題的方法及其差異性。

2.1 練習

量測自變數 X 與應變數 Y 所得的資料為

X	6	3	5	8	7
Y	2	4	3	7	6

資料準備：依前一個練習的方式，在 MATLAB 的命令視窗裡將這些數值建立成 x, y 矩陣 (向量)，如 ¹

$$x = [6 \ 3 \ 5 \ 8 \ 7]$$

範例1: 依序利用下列的MATLAB 指令，計算變數 X 的 sample mean:

$$\mu_x = \frac{\sum_{k=1}^N x_k}{N}, \text{ where } N = 5$$

¹請注意程式設計中變數的命名不可輕忽，最好能賦予有意義且相關的名稱，有利程式的可讀性，在未來除錯或維護上都會很方便，無形中節省許多不必要的時間浪費。在此以小寫的 x 代表變數 X 的樣本資料，使用小寫是確知這是個向量不是矩陣，來符合以小寫字母代表向量或常數，大寫代表矩陣的慣用語法。當然習慣隨人，只要保持一致即可。

8 MATLAB 的數學實驗

1. $\mu = (x(1) + x(2) + x(3) + x(4) + x(5))/5$
2. $\mu = \text{sum}(x)/5$
3. $\mu = \text{mean}(x)$

其中 sum , mean 為 MATLAB 提供的指令, 使用前請先利用 `help` 瞭解其使用方式。請注意分母的5代表樣本數 N , 在程式設計的技巧上, 會先設定一變數, 譬如 $N = 5$, 代表樣本數, 之後的指令便以 N 取代固定的常數5。

範例2: 依序利用下列的MATLAB 指令, 計算變數 X 的 sample variance:

$$S_x^2 = \frac{\sum_{k=1}^N (x_k - \mu_x)^2}{N - 1}$$

1. 模仿範例1之1的作法, 非常繁瑣, 但請耐心的完成, 其中 μ_x 可以直接利用範例1的結果。
2. 模仿範例1之2的作法, 利用向量的運算並加入 `sum` 指令以減少繁雜程度, 即
$$\sum_{k=1}^N x_k^2 = \text{sum}(x .* x)$$
3. 利用矩陣運算的技巧, 將平方與累加合併計算, 進一步精簡指令, 即
$$\sum_{k=1}^N x_k^2 = x * x'$$
4. 利用 MATLAB 內建函式 `var`。

上述不同的過程應該得到相同的結果, 只是繁瑣與精簡之差別。程式設計當然力求精簡, 除便利可讀性外, 往往效率也會比較高。本練習的目的在讓初學者

體驗 MATLAB 矩陣運算的效率與技巧，當數學式本身非常繁雜時會需要這些技巧。初學者應具耐心細心的輸入與比較才能漸得 MATLAB 程式設計之精髓。

範例3: 計算每一個 x 值的 z-score: $z = \frac{x - \mu_x}{S_x}$

參考指令: `zscore`

範例4: 計算 $S_{xy} = \sum_{k=1}^N (x - \mu_x)(y - \mu_y)$, 你可以用多少種方法來計算這個值呢? 哪一種方法最乾淨俐落?

2.2 觀察

1. 觀察常數在矩陣運算中的角色與結果。
2. 觀察累加 Σ 與矩陣的關係, 並且想像上述的問題如果資料量大的時候, 計算上, 表達上有何困擾? 這是為什麼 MATLAB 強調避免直接使用加法, 而以矩陣的運算取代。
3. 變數的命名必須非常謹慎小心, 不可與 MATLAB 內建的指令相同, 如此將混淆該指令的功能, 使之失效。譬如範例2的 sample variance, 一般直接會取用 `var` 當作變數名稱, 但卻與 MATLAB 內建的指令相同, 會使隨後使用的 `var` 功能失效。

2.3 作業

兩變數間相關性的觀察與相關係數的計算 (注意繪圖時 x, y 軸的 scale)

1. 寫下上述範例4的各種計算方式, 每一種方式都盡量用一條指令完成。

10 MATLAB 的數學實驗

2. 根據網頁上提供的資料 (<http://web.ntpu.edu.tw/~ccw/statmath/book.htm>):
資料1,

- 先畫出 x, y 的散佈圖並以目視約略判斷 x, y 的相關性。
- 計算變數 x, y 的相關係數 (the coefficient of correlation):

$$r = \frac{S_{xy}}{S_{xx}S_{yy}}$$

其中 S_{xx} 及 S_{yy} 分別是變數 x 與 y 的變異, 與之前練習所計算的變異數關係為 $S_{xx} = (N - 1)S_x$, $S_{yy} = (N - 1)S_y$ 。觀察這個 r 值與散佈圖呈現的樣子是否吻合?

- MATLAB 也提供了計算相關係數的指令, 請把它找出來, 並且自行了解它的使用方式, 最後驗證自己的答案。
 - 將 r 值寫在散佈圖上 (如圖2.1), 並寫下計算 r 值的所有指令。
3. 同2, 但使用資料2。觀察這個 r 值與散佈圖呈現的樣子是否吻合?
4. 同2, 但使用資料3。觀察這個 r 值與散佈圖呈現的樣子是否吻合?

補充1:

作業所採用的資料來自檔案, 如 data1.txt。在 MATLAB 中讀取檔案的方式為:

```
load data1.txt
```

當使用 whos 指令察看命令視窗的變數時, 將發現新的變數 data1, 代表資料檔案 data1.txt 的內容。

補充2:

作業中要求將計算出來的 r 值寫在散佈圖上, 這需要用到指令 text 或 gtext,

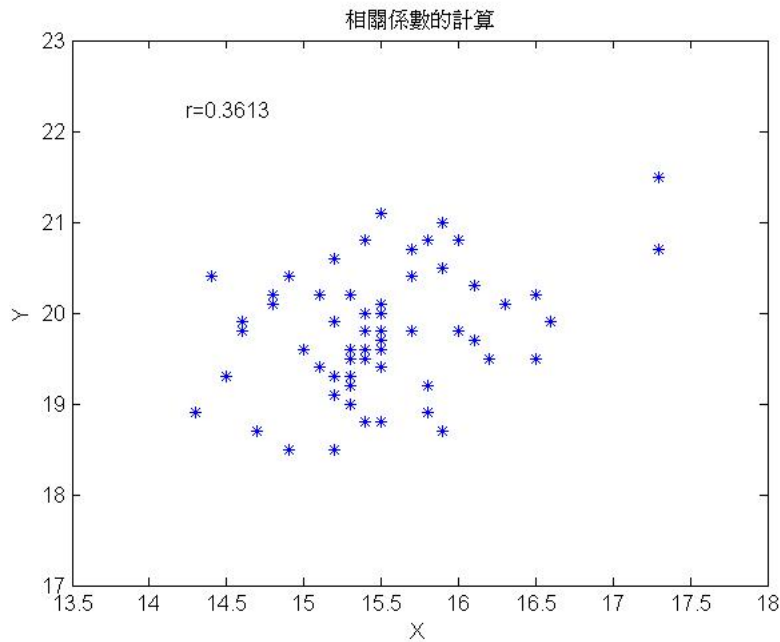


圖 2.1: 散佈圖與相關係數的計算

其差別在 `text` 必須指定 (x,y) 座標, 而 `gtext` 則可以使用滑鼠選定適當的位置, 詳細的使用方式請自行參考使用手冊。勤用 `help` 是必要的, 畢竟每個指令都有不同的參數給定方式, 簡單的 `help` 一下, 立刻便能查知, 無須記憶。

補充3:

散佈圖的觀察會隨著X,Y 軸的座標範圍而在目視上有所不同, 適當的 `zoom-in` 或 `zoom-out` 可以讓圖形的表現更貼切, 如圖2.1並非畫出來就長這樣。畫完圖之後座標軸可以利用指令 `axis` 加以改變, 同樣先以 `help` 查出其使用方式。

12 MATLAB 的數學實驗

第 3 章

MATLAB 的機率分配

機率相關的問題對於統計學門，不僅是基礎的理論，更是應用上不可或缺的工具。對多數初學機率的學生而言，機率是有點抽象的，需要帶點想像的，學習的障礙不算小，不過有了電腦工具（如 MATLAB）的輔助，那些想像與抽象的部分會變的比較具體些。本練習旨在熟悉 MATLAB 處理機率問題的指令及其應用，包括分配函數的繪製與亂數的產生。

3.1 練習

MATLAB提供哪些 pdf 及 cdf 函式呢？你可不可從 MATLAB 提供的查詢功能查到呢？你必須練習找找看。

範例1: 先來畫常態分配 $N(\mu, \sigma^2)$ 的 pdf 及 cdf 圖，其中的參數 $\mu = 0, \sigma = 1$ 。請注意調整 x 軸的範圍，方便看到最完整的分配圖形。

- 練習給予不同的 μ 及 σ 值，看看圖形的變化。
- 固定 μ 值，改變 σ 值，練習將每一張圖都疊在畫面上（如圖3.1所示）。
- 試試看其他的分配？找出相對的指令來，把圖畫出來，並使用不同的參數。

請注意，只要是繪圖就是描點法，要將每一個點的 x, y 值都事先給定或計算好。MATLAB 中 pdf 及 cdf 函式都是用來計算 y 值的。另外本練習為觀

14 MATLAB 的數學實驗

察參數對於一個分配函數的影響, 要求將不同參數的分配圖疊在一起, 方便觀察。MATLAB 利用 `hold on` 指令將目前的圖形保留, 隨後畫上的圖會直接疊在上面, 再利用 `hold off` 取消保留的設定。譬如將兩張圖疊在一起的做法:

```
plot(x1,y1)
hold on
plot(x2,y2)
hold off
```

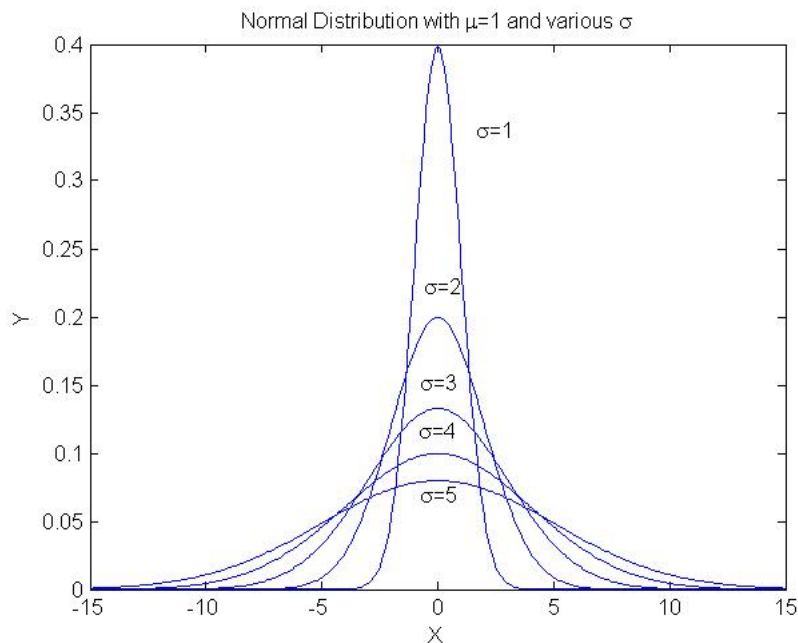


圖 3.1: 常態分配的機率密度函數 (pdf)

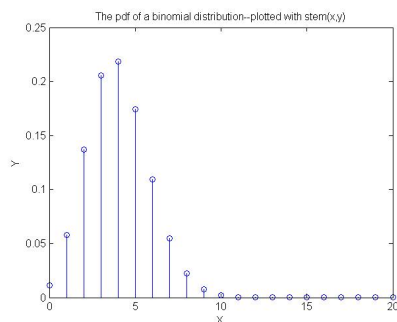
另外, 不連續型分配 (如二項分配) 的繪製要特別謹慎, 要注意 x 軸範圍的限制與間距的特性, 不能以一般繪製連續函數的方式大刺刺的一筆帶過, 畫出來的圖必須合乎學理, 不是畫出圖來便是對的。以二項分配為例, x 座標的選定與母體的選擇息息相關, 假設母體為 $B(20, 0.2)$, 其 pdf 的繪製可以這樣做

```

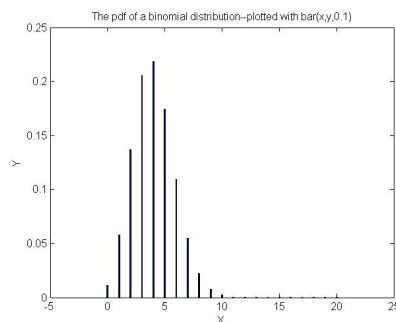
x = 0 : 20;
y = binopdf(x, 20, 0.2);
stem(x, y)

```

其結果如圖3.2(a)所示, x 的範圍與間距準確的呈現出母體的所有可能性, 太大的範圍沒意義 (如 $x = 0 : 50$), 太小的範圍 (如 $x = 0 : 15$) 與非整數的間距 (如 $x = 0 : 0.1 : 20$) 則是對二項分配的不瞭解, 都應該很警覺的避免。MATLAB指令 `stem` 可以畫出漂亮的間距圖形, 但也不是唯一的選擇, 圖3.2(b) 利用寬度比較窄的長條圖 `bar`, 也可以畫出類似的效果, 甚至變化更豐富。至於 cdf 圖, 可以採用 `stairs` 的指令, 畫出如階梯般的機率累積圖。`stairs`指令怎麼用呢? 其實繪圖指令用多了, 猜猜看往往八九不離十, 學習程式語言要有此本事才會越學越輕鬆愉快。



(a)



(b)

圖 3.2: 二項分配的機率密度函數 (pdf), 以指令 (a)`stem`(b)`bar` 繪製的不連續函數圖。

範例2: 如何從MATLAB 產生亂數 (random numbers)? 照例可以從 Help 裡面查到亂數相關的指令及其使用方式。

- 隨意產生 100 個具常態分配的亂數。你如何確定這些亂數值具備常態分配的特性? 用 `plot` 畫出這些值, 看看長什麼樣子? 再試試用直方圖去畫。

16 MATLAB 的數學實驗

- 重複上個練習, 但試著改變所產生的亂數個數 (變多或少), 與不同的分配。觀察畫出來的直方圖有何不同? 跟你原本的認知有什麼差別?

直方圖是統計常用的基本圖形, 常用來觀察資料的分佈情形 (頻率), MATLAB 對應的指令為 `hist`。繪製直方圖需要很有「感覺」, 否則容易畫出連自己都不易看得懂圖, 其中樣本數的多寡牽涉到直方圖選定的組界範圍, 尤其困擾初學者, 不妨多練習並不斷的變更組界範圍並觀察圖形的變化。

範例3:

- 產生兩組資料, 每組各 100 個樣本。這兩組資料來自兩個不相關的變數 (uncorrelated variables) y_1 及 y_2 (可以先假設變數都具常態分配)。畫一個散佈圖來呈現資料間的不相關性 y_1 vs. y_2 。並且計算出兩者間的相關係數 r 。
- 同上, 但資料來自兩個完全相關 (completely correlated) 的變數並觀察相關係數, 例如: $y_1 = cy_2$, c 代表一個常數。
- 同上, 但資料來自兩個部分相關 (partially correlated) 的變數並觀察相關係數。如何模擬「部分相關」的兩組資料呢? 動動腦筋!

本範例假設學習者對於兩變數間的相關性及其散佈圖的模樣已經有相當的概念, 方能藉此逐步模擬出自己所認知的相關性, 特別是模擬部分相關的資料。此外, 當畫出的幾張圖形具有比較意義時, 可以利用 MATLAB 提供的畫面切割技術, 將幾個圖分別呈現在畫面的不同位置, 這個指令是 `subplot`。用來作為繪圖前位置的指定, 使用方式如下

```
subplot(2,1,1),plot(x1,y1)
subplot(2,1,2),plot(x2,y2)
```

至於 subplot 的參數代表的意義, 親自動手畫圖的時候便可以領會, 無須在此贅言。寫得太仔細會把學的人教笨了, 豈不罪過!

3.2 觀察

1. 畫出來的 pdf 與 cdf 圖是否都符合你的預期呢? 如果不是, 去翻翻統計學、機率論的書驗證一下。
2. 釐清楚機率分配的 pdf 圖與依亂數值產生的直方圖。不要搞混了! 兩者間有什麼異同?
3. 畫出來的直方圖若與自己認定的不符時, 請特別注意是畫圖技巧不好, 還是指令的操作錯誤或是觀念的錯誤。通常直方圖的畫圖技巧必須注意樣本數及範圍間距的選擇。
4. 透過這個單元的練習, 當給予一組隨機資料時, 你有多少把握知道其原始的分配是什麼? 這與資料量的多寡有關嗎? 除了畫直方圖之外, 還有沒有其他方式可以提供更多的參考訊息呢?
5. 觀察兩組資料的相關性, 相當具實用價值。本練習進而去模擬產生具某種相關性的兩組實驗資料。產生模擬資料對研究工作而言, 往往是必備的基本動作。
6. 兩變數的相關性不一定是線性的, 這裡只是給予線性的訓練。你也可以試試產生非線性相關的資料, 再去計算 r , 看看發生什麼事了?
7. 請觀察變數資料的產生與關係的形成, 及最後散佈圖的樣子, 要牢牢的將這些東西連結在一起。加強對資料的感覺, 培養與資料間的感情, 即所謂的『資料感』。

3.3 作業

1. 畫出下列分配的 pdf 及 cdf 圖, 並觀察參數的改變與圖形的關係。將相關的圖疊在一起 (至少 5 張圖疊成一張)。所有的參數資料盡量寫在圖形的空白處。
 - Normal Distribution: (1) 固定 μ 改變 σ , (2) 固定 σ 改變 μ
 - Chi Square Distribution: 自行調整參數。
 - Binomial Distribution: 自行調整參數。
 - F Distribution: 自行調整參數。
 - T Distribution: 自行調整參數。
 - β Distribution: 自行調整參數。
 - Exponential Distribution: 自行調整參數。
2. 產生 5 組亂數 (來自 5 個不同的分配), 其個數與分配自行決定, 分別畫出直方圖。觀察畫出來的圖是否符合預期呢? 你必須確定這一點。不能畫了就算!

第 4 章

MATLAB的程式：以微分的計算為例

MATLAB作為一個數學計算的工具，不只是提供許多計算的指令而已，它還是一個電腦語言。利用一般電腦語言的邏輯能力，結合數學相關的指令成為一個程式。藉由執行程式達到解決複雜程度較高的數學問題。微分或積分的計算是利用電腦能力解決數學問題的典型。這點從定義上便可以一窺究竟。

4.1 練習

微分根據其定義：

$$f'(x) = \frac{df(x)}{dx} = \lim_{h \rightarrow 0} \left(\frac{f(x+h) - f(x)}{h} \right)$$

配合函數的型態，推展出許多微分的技巧。這些技巧為人所熟悉，但是電腦可沒這麼聰明，雖有人工智慧的輔助，還是不及人腦來得靈光。而且技巧總是有限，碰到一些「不友善」的函數時，往往不夠用。這時電腦可派上用場了，除了可以迅速並大量計算微分問題外，函數的「長相」通常它是認不得的。撇開像 Mathematica 這類可以根據數學符號就能計算微分的軟體，以電腦程式來計算數值微分，通常會從上述的定義來下手。以下列舉幾個範例來練習，看看電腦如何計算函數 $f(x)$ 在任一點的導數（或稱切線斜率）。¹

¹有一門領域叫做「數值分析」專門研究如何利用電腦來解決數學問題。關於「微分」當然也發展出許多的演算法 (Algorithm)，克服各式各樣可能在電腦上產生的問題，本文僅就最簡單的方式著手，避免涉及太多細節，以免干擾本文真正的目的：介紹 MATLAB 的程式功能。

範例1: 假設 $f(x) = x^2 + 3x + 1$, 求 $f'(3)$ 。

- 定義上的極限 \lim 在數值計算上如何表示?
- h 值該給多少才算趨近於0? 試試看當 $h = 0.1, 0.01, 0.001, 0.0001$ 時, 導數值分別為多少? 與實際值相差多少? 哪一個選擇比較好? 導數值由下列公式計算:

$$f'(3) = \frac{f(3+h) - f(3)}{h}$$

上述的公式需要計算兩個多項函數值, 輸入時稍嫌麻煩, 此時可以使用 MATLAB 提供計算多項式函數的指令 *polyval*, 可以讓整體指令「乾淨」許多。

範例2: 從上一個練習得知, 計算數值導數時, h 值的選擇會影響導數值。對於這個函數你能找出一個最好的 h 值嗎? 畫一張圖看看 h 值的選擇對計算導數的影響。畫圖時 x 軸的選擇很重要, 可以採 $\log_{10}(h)$ (MATLAB 指令為 *log10(h)*)。

MATLAB 的特點是矩陣的運算, 本練習要觀察不同的 h 值計算出的導數值, 便可以利用這項特質。譬如建立一個變數

$$h = \begin{bmatrix} 10^{-4} & 10^{-5} & 10^{-6} & 10^{-7} & 10^{-8} & 10^{-9} & 10^{-10} & 10^{-11} & 10^{-12} \end{bmatrix}$$

建立的方式可以如上式一一輸入, 但最快也最能掌握 MATLAB 精髓的方式為:

$$h = 10.^{-4:-1:-12} \text{ 或 } h = 1./10.^{4:12}$$

利用 MATLAB 提供的運算元 (\cdot) , 可以同時針對矩陣或向量的每一個元素做計算。

範例3：前述的練習需要幾個指令來完成導數的計算，如果每計算一次就要重複執行相同的指令，代入不同的數值，那將會非常沒有效率，更何況面對複雜的數學問題可能要花數十，甚至數百條指令來處理。在這個情況下，MATLAB 允許將這些指令集中起來，存放在一個檔案裡面，藉由在命令視窗輸入檔名，MATLAB會循序執行檔案裡面的每一條指令。這個檔案被稱為 MATLAB 的程式。請將上述範例計算導數的指令集中起來，存成一個檔案（副檔名自動設定為 .m），² 譬如，derivative.m

請注意要 MATLAB 正確的執行程式檔，必須符合下面兩個條件之一：

- 「命令視窗」所在的位置必須相同於程式檔。這可藉由變更命令視窗的「current directory」達到。
- 將程式檔所在的目錄設為 MATLAB 可以自動參考的路徑。這可藉由「File」→「Set Path」→「Add Folder」達到。

將所有必要的指令放在一起，不但方便執行，也容易管理與維護，更提供程式寫作許多可以運用的技巧。

範例4：MATLAB 提供什麼指令計算數值導數呢？(hint: diff)。試著用 MATLAB 指令的方式來計算導數值。觀察看看，與你前面所計算的值相差多少？哪一個比較準？

像 MATLAB 這種等級的數學軟體當然會提供計算導數的功能指令，究竟 MATLAB 計算導數的指令與自己根據定義所寫的程式有何不同？這是令人好奇也值得一探的。如果發現 MATLAB 的指令執行結果令人滿意，未來程式有需要自然可以直接引用，不一定要使用自己寫的，畢竟專業的程式設計還是比較縝密，使用上也比較方便。

²請注意檔名的選擇，不能與 MATLAB 內建的指令名稱相同，那將影響該指令的功能。

4.2 觀察

寫程式解決數學問題常會觀察到一些平常看不到的「內涵。」透過這些觀察與反覆思考查證，往往會對理論有進一步的體會，那可是平常人無法直接從書本獲得的。以下列舉些觀察點供參考。

- h 值是影響數值計算的重要因素，是否在所有的練習中可以歸納出一個適當的 h 值，對所有的函式都適用？對於 h 值的選擇，你有哪些觀察呢？
- MATLAB 利用單一指令解決單純問題。當一個指令無法解決問題時，可藉由指令的「接力」執行，共同合作完成任務。多個指令組合（或稱程式）並且一起被執行，方便解決較複雜的問題。但當指令被組合起來時，必須注意其前後的合作關係，如何安排變數、如何決定運算的次序、如何適當的採用 MATLAB 的內建功能，讓程式執行順暢、有效率，而且易懂、容易變更、擴充，都是非常重要的程式寫作技巧。初學者應細心體會其中的奧妙。多看別人的程式，比較自己的程式，從中獲得真正的寫作技巧。
- 程式的執行遵循從上而下的規則，但程式的寫作可不一定要從第一行寫起。可以從做容易下手（最直覺）的地方著手，再依程式的需要補足前後面的指令，這樣反而比較順手，速度比較快。因為程式語言的邏輯與人腦還是不同的，因此程式寫作的過程常常是上上下下穿插的寫，並且不斷的執行部分過程，以確定指令與邏輯的正確，千萬不可妄想一次寫完再執行，往往會連錯在哪裡都不知道，徒增除錯的困難。

4.3 作業

1. 將數值導數的定義改為

$$f'(x) = \frac{df(x)}{dx} = \lim_{h \rightarrow 0} \left(\frac{f(x+h) - f(x-h)}{2h} \right)$$

與之前的定義做比較，哪一個定義比較準確？舉範例1的函數，取幾個不同的 h 值，譬如： $h = 10^{-2}, 10^{-3}, \dots, 10^{-9}$ ，等，畫一張圖紀錄在每一

個 h 值，不同導數公式的誤差（誤差=|數值導數 - 實際導數|）。即畫一張圖顯示兩條來自不同的導數定義的誤差線。

2. 從上題的結果，你可以判斷在數值計算上哪一個定義比較好嗎？下面這個練習可以幫助釐清一些觀念：

令函數為 $f(x) = x^2 + 3x + 1$ ，將此函數分別代入兩個定義中，經過一番化簡之後，你得到什麼？這個結果可否幫助你判斷在數值計算上哪一個定義比較好？

3. 畫一張圖比較函數 $f(x) = -x^3 + 6x^2$ 的數值導數與理論導數的差別。所謂理論導數，即 $f'(x)$ ；而數值導數則是利用數值方法（例如前述的練習）以電腦程式計算的函數值。請自行選擇適當的導數定義、 h 值與 x 的間距範圍；將理論值的圖與數值計算的圖畫在一起。請注意
- 你可以選擇不同的 h 值，藉以凸顯不當的 h 值會影響數值導數的精確度。
 - x 軸範圍的選擇頗為關鍵。範圍的大小與區域的選擇不當往往會造成視覺上的錯覺，導致錯估兩者間的差距。不妨多觀察幾個不同的區域與範圍，甚至可以去計算誤差的大小。
4. 同上題，但是函式為： $f(x) = \frac{\ln x}{x^3}$ ，其導函數為： $\frac{1-3\ln x}{x^4}$ 。選擇 $x > 0$ 的適當範圍。

第 5 章

MATLAB程式的迴圈技巧: 以積分為例

迴圈技巧是程式語言的生命, 有了它程式才顯示出價值。電腦之應用貴在於能大量且快速的執行相同的計算, 而迴圈便是這項功能的靈魂。MATLAB 程式的迴圈技巧與其他語言類似, 甚至相同, 有其他語言基礎的可以很快的進入狀況。本單元利用計算積分的數值方法來介紹 MATLAB 中迴圈的使用。

積分 (Definite Integral) 的定義:

$$\int_a^b f(x)dx = \lim_{h \rightarrow 0} \sum_{x=a}^{b-h} f(x+h)h$$

圖 5.1 是這項積分定義的示意圖。當 h 很小時, 陰影部分的長方形越窄, 也因此上緣越貼近函數的曲線, 而數值積分正是這些長方形面積的總合。當 h 趨近於 0 時, 這無限多個長方形的面積合即為積分值。

5.1 練習

範例1: 利用上述數值公式, 計算

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

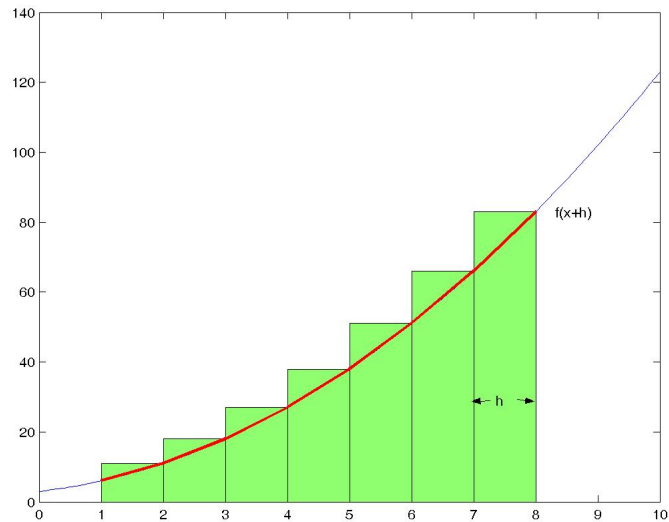


圖 5.1: 積分的定義: 黎曼和

其中設 $h = 1$ 。請比較數值積分的結果與實際的積分值。如果 $h = 0.5$, 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, 數值計算的結果與實際值愈接近。假設 $h=0.01$, 在程式的寫作上你會遭遇什麼困難?

範例2:『迴圈技術』(for loop) 可以輕易解決這個問題。先練習迴圈技巧: 寫一支迴圈程式計算 1 加到 100。程式執行時要在每個迴圈中印出累加值及迴圈數。

迴圈的指令為 for, 請利用 help 找出迴圈的使用方法, 下表的示範程式計算「從 1 加到 10」。

total=0;	註: 必須先將「總和變數」total 的值預設為 0
for i=1:10	註: 迴圈開始
total=total+i;	註: 開始累加, 變數 total 值每次都不同
end	註: 迴圈結束

在上述的範例中嘗試將第一行拿掉, 看看執行結果有何差別。將一行程式暫時拿掉的方式, 是將其變為註解行, 即做前面加上百分比符號%。另外, 寫迴圈程式有個慣例, 在迴圈的第一行與最後一行間的程式碼做適當的縮排。這樣做的原因除方便除錯外, 也讓增加程式的可讀性。

範例3: 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。

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, 請利用矩陣的乘法解決累加的問題。比較看看與迴圈技術在計算時間上的差別。(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: x=1:5;
      answer=sum(x+3)
```

範例4: 計算

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

先畫出函數圖形, 再利用迴圈法與矩陣法分別計算積分值。

範例5: 如果能畫出積分計算過程的長方形, 會有助於對積分計算的瞭解? 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。至於長方形的寬可以從程式來控制。

為 MATLAB 圖上的曲線塗上顏色的方式如下:

```
x=1:5;
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: 改變座標範圍方便觀察
```

範例6: 如上題, 將面積部分改為梯形, 並改寫積分的數值定義, 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』比梯形法更為準確。同學有空不妨去找找看什麼是『辛普森法則』。另外, MATLAB 提供哪些計算積分的指令呢? 採用什麼樣的計算法則呢? 查查看, 並試著用來計算前面的積分式。

5.2 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. 長方形法有三種選擇方式,
 - 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, 面積合低於積分值。
 - 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為接近?

5.3 作業

1. 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!

2. 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)
- $\int_0^7 x^2 e^{-x} dx$ (ans::1.94)
- $\int_{-1.96}^{1.96} \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} dx$

3. 還記得泰勒展開式 (Taylor Expansion or Taylor Series) 嗎? 任何函數都可以展開成一個無限的級數。例如: 函數 $f(x)$ 在常數 a 展開之泰勒級數

$$f(x) = f(a) + f'(a)(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \cdots + \frac{f^{(n)}(a)}{n!}(x-a)^n + \cdots$$

當 $f(x)$ 是一個比較複雜函數 (譬如無理函數) 時, 這個無限級數的展開式便可以派上用場。適當的選擇 a 及無限級數的前幾項, 往往可以得到滿意的近似值。譬如當 $a = 0$ 時, 這個無限級數變成一個簡單的多項式, 當需計算 $f(x)$ 的值時, 便可以用前述的近似方法, 以簡單的計算得到滿意的近似值。這個練習將以迴圈的方式將泰勒級數的前幾項一一加起來。請按下列步驟計算 $f(x) = e^x$

- 選擇一個適當的 a 值, 將函數展開。哪一個 a 值比較有利計算呢?
- 挑好 a 值後, 到底要加幾項才能得到比較滿意的近似值呢? 請建立一個迴圈, 設定20圈, 每繞一圈加一項進來並列印出加總的結果 (即逐圈列印結果)。
- 選擇不同的 a 值對於結果有影響嗎? 說說你的觀察。

註: 計算階乘, 參考指令 factorial。

4. 計算積分還有一種方式叫做「Monte Carlo Simulation,」其原理簡單敘述如后,

$$\int_D f(x)dx = \int_D \frac{f(x)}{p(x)}p(x)dx = E \left[\frac{f(x)}{p(x)} \right]$$

其中 $p(x)$ 為機率密度函數 (pdf)，積分的問題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 $f(x)$ 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 $p(x)$ 為何，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 $f(x)$ 的外形為佳。利用此法，重做作业1的積分問題，需說明使用的 $p(x)$ 與樣本數。