

MATLAB 程式的迴圈技巧

last modified August 27, 2009

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，具備其他語言基礎的可以很快的進入狀況。

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`, `factorial`, `pause`

語法: `for` 迴圈的建立與技巧。

1 背景介紹

一支程式由數個指令組成，執行時是一根腸子通到底，由上而下逐步執行，一個指令執行一個動作。當完成一件事情（計算）需要龐大的運算動作時，有限的指令是行不通的。譬如，計算泰勒展開式

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

當展開的項次 n 很大或不確定大小時，固定指令的寫法並不可行。從上式累加的項目中發現，這些項目具規則性的結構，程式語言中的迴圈技巧 (for loop) 正可以輕巧的解決這個問題。以下列出一些簡單的範例，說明迴圈技巧的使用方式與時機，最後利用泰勒展開式與迴圈技巧計算無理函數 $\exp(x)$ 。

2 練習

範例1: 寫一支迴圈程式計算1加到100，即計算 $\sum_{k=1}^{100} k$ 。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 for』找出迴圈的使用方法，指令如下。

clear	註：程式開始前，先清除命令視窗的所有變數
total=0;	註：必須先將「總和變數」total 的值預設為0
for i=1:100	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
total	註：印出結果

上述程式第三行代表迴圈的開始，第五行表示迴圈結束。當程式開始執行時，由上（第一行）逐步往下，直到進入迴圈後，將重複執行迴圈內的所有指令若干次，次數由指令迴圈索引變數 i 所設定的向量決定。譬如， $i=1:100$ ，迴圈內的指令將被執行 100 次，每次索引變數 i 代表向量中的每個數，由第一個 ($i=1$) 到最後一個 ($i=100$)。

第四行運用了常用的累加技巧，這裡的等號並非數學裡的相等之意，而是將右邊所計算的結果指定給左邊的變數。於是變數 `total` 的值在迴圈重複的過程中每次都不同，因而產生累加的效果。在上述的範例中嘗試將第二行拿掉，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，即在最前面加上百分比符號`%`。¹另外，寫迴圈程式有個慣例，在迴圈開始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，也增加程式的可讀性。

Tips: 學習程式寫作最好的方法是參考正確的示範程式，在瞭解語意後，多模仿，久之，便能得心應手。在模仿上述程式片段前，初學者最好先找出迴圈指令 `for` 的正確語法，再逐行解讀示範程式，完全瞭解後，最後執行示範程式並觀察結果（最好逐行觀察中間的結果）。

累加技巧屬於基本動作，應多做練習，以下練習一個稍複雜些的問題。

範例2: 利用累加技巧計算樣本變異數 $\hat{\sigma}^2 = \sum_{i=1}^N (x_i - \hat{\mu}_x)^2 / (N - 1)$ ，其中的樣本 x_i 自行以亂數產生器 `normrnd(0,1,1,N)` 產生，樣本數 N 設為 30。

樣本變異數的計算有很多方式，在前面的單元也介紹過，MATLAB 也提供計算樣本變異數的指令 `var`，這裡純粹是為練習迴圈的累加技巧，刻意利用這個簡單的問題。問題雖然簡單，但是模式是固定，將來遇到複雜的問題可以根據本題的模式推展。請讀者先試著以上一題的架構，寫出本題的程式，對錯與否可以 `var` 指令驗證。

範例3: 在同一張圖上畫出斜率不同的直線函數 $y = mx$ ，斜率 $m = 1, 2, \dots, 50$ ，如圖 1。

要將 50 個條斜率不同的直線畫在同一張圖上，當然不能手動一張張疊上去，迴圈技巧是最好的選擇。以對迴圈技巧最初淺的想法可以寫成

¹註解指令較快速的方式是先將滑鼠游標指到該行，再按「Ctrl+r」即可，復原時再按「Ctrl+t」，且不管單行或多行都可以。

```
clear
x=-5:5;
for i=1:50
    y=i*x;
    plot(x,y)
end
```

上面的程式藉由迴圈中索引變數 i 的改變，執行迴圈中的兩行指令 50 次，變數 i 的值從 1 到 50。雖然執行的指令不變，但是透過 i 的改變，變數 y 的值 (向量) 也變了，當然作圖時也會產生變化，於是斜率不同的直線便一一產生。上述程式還不能產生圖 1，需要處理一些細節的問題，譬如疊圖。圖 1 便是由下列程式產生的。

```
clear
x=-5:5;
y=x;
plot(x,y)          註: 先畫第一條線, 方便做 hold on
ylim([-250 250])   註: 預先調整 Y 軸範圍
hold on
for i=2:50          從 m=2 畫起
    y=i*x;
    plot(x,y)
    pause(0.1)      製作成動畫, 方便觀察繪圖情況。
end
hold off
```

這裡利用指令 `pause(0.1)` 製造程式執行暫留 0.1 秒，造成繪圖過程中的視覺現象，看起來像動畫，一來增加程式樂趣，再則將迴圈執行的過程依序呈現出來，常用來檢查程式的邏輯是否錯誤。

通常迴圈的索引變數值習慣引用整數值，如上述的 $i=2:50$ 。如果本題的斜率 $m=0.5, 1, 1.5, \dots, 20$ ，索引變數 i 固然可以設為 $i=0.5:0.5:20$ ，但這並非好習慣，非整數的設定有些時候會出錯的。類似的情況非常普遍，解決的方式如下

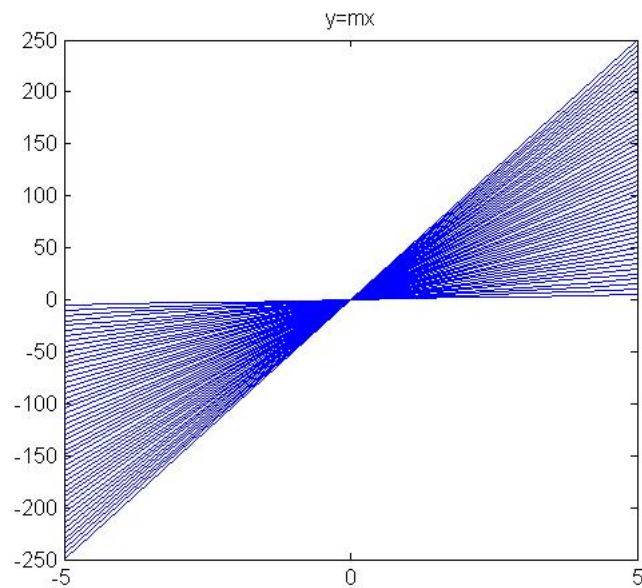


圖 1: 利用迴圈技巧畫圖

```

...          註: 同前
m=0.5:0.5:20;  註: 預設所有的斜率值在一向量裡
n=length(m);   註: 計算斜率個數
for i=1:n
    y=m(i)*x;  註: 依序取得斜率值
    plot(x,y)
    ...        註: 同前
end
...          註: 同前

```

這裡的 i 變成名符其實的索引值, 從向量 m 中逐一挑出預設的斜率值, 共有 n 個。這個技巧非常好用, 請記得常回來看看。

範例3: 承上題, 但函數圖改為

1. $y = ie^x, -5 \leq i \leq 5$

2. $y = i \sin(x), -5 \leq i \leq 5$

3. $y = i(x - 3)^2 + 2, -10 \leq i \leq 10$

其中 i 皆為整數。

自己可以隨意找些函數來玩玩，只要更動某個參數值常常可以形成很動人的圖案，配合動畫的效果，滿有娛樂效果。

範例4: 如果想看到機率密度函數的「長相」與其參數間的關係，可以利用迴圈的技巧來改變參數值，再將函數一個個畫上去。瞬間便可以看許多密度函數的改變，或是利用時間的暫停慢慢欣賞密度函數隨參數改變的變化情況。試著利用迴圈技巧畫出卡方分配的機率密度函數隨自由度改變，譬如從 $\nu = 3, 4, \dots, 20$ 的變化情形。

依前述範例，很容易利用迴圈中索引值做出如圖2所展示的，在不同自由度下的卡方密度函數。

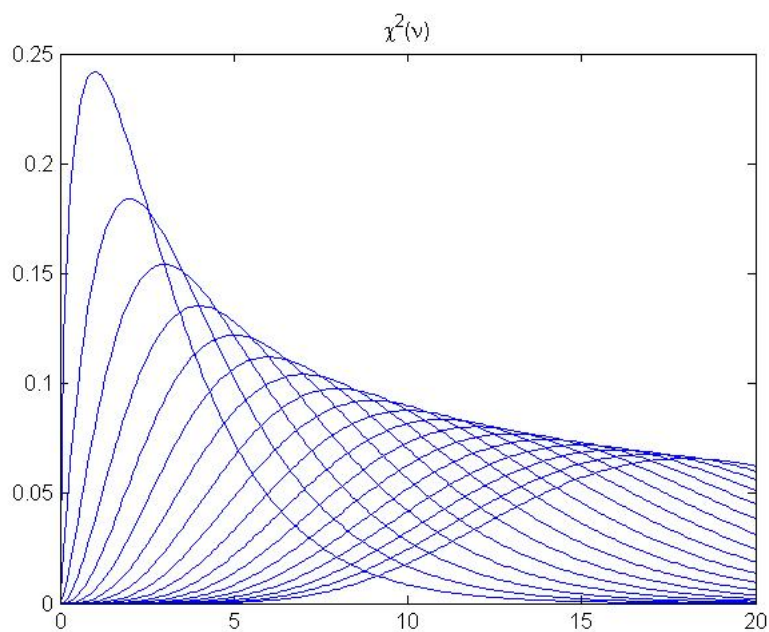


圖 2: 不同自由度下的卡方密度函數

不過如果希望在時間暫留的動態呈現下，也可以將參數值送上畫面，將更有利密度函數改變的觀察。此時牽涉到將不同文字動態呈現出來的技巧，如下列片段程式

```
...
for i=4:20
    ...
    plot(x,y)
    str=strcat('\nu =',num2str(i));
    title(str);
end
```

其中指令 `strcat` 常用來將不同來源的文字串連起來，裡面的逗號可以一直加下去，一次串連多組文字。這裡串連了兩組文字，一組是固定文字 `\nu =`，另一組是動態文字，由迴圈的索引值經過指令 `num2str` 轉來。索引值 `i` 是數字，按 MATLAB 的語法，不得與文字串連，所以需經過數字 (num) 到文字 (str) 的轉化 (2, 音同 to)。因為 `i` 的改變，造成串連出來的字串每次都不同，適合做動態的呈現。這裡將這組動態的文字當做 `title` 展示出來。

範例 5: 設計一個射飛靶的遊戲。首先畫一個正圓在座標中央，利用適當的亂數產生器給出兩個數值，分別代表座標軸上的 `x` 與 `y` 座標。接著在座標軸上的 `(x,y)` 座標點畫上任一個符號，譬如「o」。利用迴圈重複執行隨機座標點的產生與作圖，配合暫停指令 `pause`，完成一個射飛鏢的簡單遊戲。如圖 3 (a)。

這個遊戲用到迴圈技術，但是並沒有使用到索引變數 `i` 的值，純粹重複固定的指令若干次。此外 `pasue` 指令可以加入暫停秒數，如 `pause(1)` 表示暫停 1 秒，在這裡不加任何秒數代表無限期暫停，等待使用這按任意鍵停止暫停，這個按鍵的動作感覺上就好像發射飛鏢。此時若要停止程式，可在命令視窗按「Ctrl+c」，否則將繼續執行到迴圈走完。

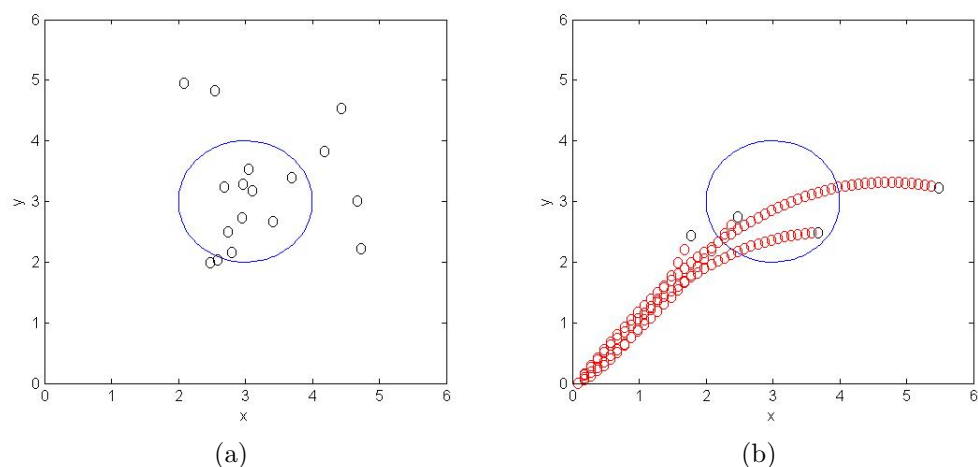


圖 3: 射飛標遊戲

```
clear
ezplot('(x - 3)^2 + (y - 3)^2 = 1') 畫圓
axis([0 6 0 6])                    限制範圍，讓飛靶在中間
axis square                          將座標圖設定為正方形，讓飛靶看起來是正圓
for i=1:100                          設定玩飛靶的次數
    x=normrnd(3,1);                  以常態分配產生隨機亂數，當做 X 座標
    y=normrnd(3,1);                  以常態分配產生隨機亂數，當做 Y 座標
    text(x,y,'O')                    標示字串 O 在 (x,y) 座標點上
    pause                            程式執行無限制暫停，直到按鍵盤任意鍵
end
```

此外，上述程式採常態分配的隨機亂數當做每次射出飛鏢的位置，設計者可以依自己的喜好或遊戲的規則使用不同的方式取得落點。甚至為了增添遊戲的樂趣與畫面的可看性，可以加入不同的元素，譬如，畫出飛鏢射出去的軌跡，如圖 3 (b) 展示的紅色軌跡圖。如果再配合動畫的呈現，便栩栩如生，像看大聯盟的線上轉播一樣，看得到投手投出去的球飛行的軌跡。軌跡的部份可以在飛鏢跑到定點時抹去，才不會讓畫面越來越混亂。至於軌跡繪製方式更是千變萬化，設計者可以憑自己的想像來製作。如圖 3 (b) 的軌跡，是以一個二次函數 $y = ax^2 + bx + c$ 的曲線圖畫出來的，其中 a, b, c 由

三個座標點來決定，分別是 $(0, 0)$, (x, y) , $(x/2, x/2)$ 。也就是先以隨機亂數取得落點 (x, y) ，並假設發射起點為 $(0, 0)$ ，在這兩點間取一點，譬如 $(x/2, x/2)$ ，便可以計算出該二次函數的係數，再利用 text 指令配合另一個迴圈「沿路」寫上符號即可。

範例 6: 假設隨機變數 X 服從均等分配 $\text{Unif}(0,1)$ ，隨機產生 n 個樣本, $x_1, x_2, \dots, x_n$ ，並計算其樣本平均數 $\bar{x} = (\sum_{i=1}^n x_i)/n$ 。如果想知道 $\bar{x}$ 的抽樣分配長什麼樣子，可以利用迴圈技巧進行 N 次的隨機抽樣，每次取 n 個樣本並計算樣本平均數，共得 N 個樣本平均數, $\bar{x}_1, \bar{x}_2, \dots, \bar{x}_N$ ，最後畫出這 N 個平均數的直方圖。本題請先設定 $n = 30, N = 500$ 。

利用迴圈技巧解決這個問題有別於前面的範例，過程中每一次迴圈所產生的結果 $\bar{x}_k$ 必需被保留，才能在迴圈結束後，將所有計算所得的結果拿來畫直方圖。保留迴圈過程所產生的結果也是迴圈技巧的基本動作，下列程式可供參考。而執行結果如圖 4 所示。

```
clear
n=30;N=500;           設定參數值，可以並列
X=zeros(1,N);         設定一個 1xN 的零向量，準備置入 N 個樣本平均數
for i=1:N
    x=unifrnd(0,1,1,n); 產生樣本
    X(i)=mean(x);       計算樣本平均並依序置入 X 向量空間
end
hist(X)                直方圖
ecdf(X)                 Empirical CDF
```

程式第六行利用預先設定的零向量依序置入樣本平均數，等迴圈跑完，這些平均數仍存在向量 X 中，不會消失。在迴圈中執行向量置入的動作前，最好先將這個向量準備好，即第三行所為。就程式執行面而言，是先向記憶體要一個固定的區塊，將來置入數值時，速度會比較快。如果省略這個預設的動作，程式仍會執行，但是會慢很多，²尤其是迴圈數多的時候更是明顯。

²沒有事先向記憶體預定空間，每次儲存時， X 向量的大小會逐漸擴增，造成更多記憶體存取的動作(更換位置)，程式執行上會很明顯的感覺變慢。所以 MATLAB 建議先設定好固定大小的空間。

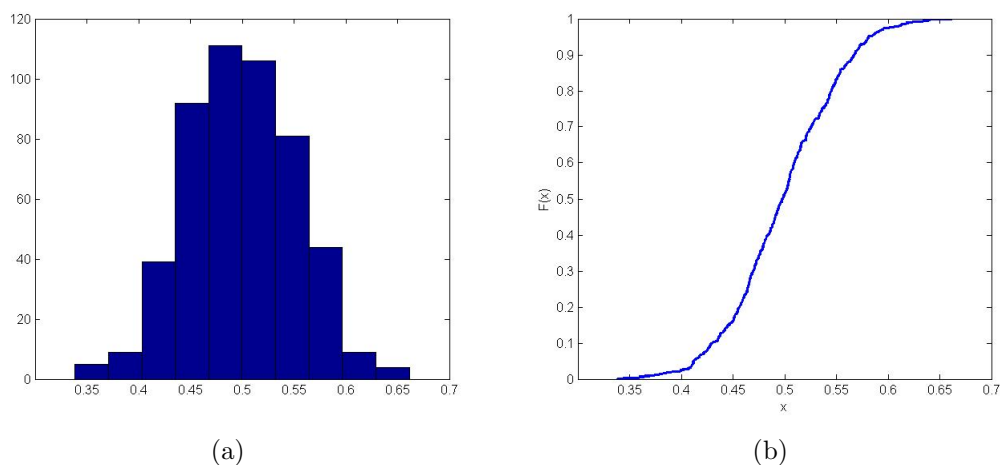


圖 4: (a) 直方圖 (b) Empirical CDF。

其實從 MATLAB 程式設計的角度來解決這個問題，並不需要用到迴圈技巧，簡單的矩陣運算便可以解決，請在作業完成這項工作。

範例 7: 以泰勒展開式表示指數函數 e^x ，寫成

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots + \frac{x^n}{n!} + \cdots$$

寫一支迴圈程式計算 e^2 。

類似泰勒展開式的函數有兩個特點，(一) 無限項次 (越後面的項次其值越小)，(二) 有規律的項次。迴圈技巧正適合解決這樣的問題。程式設計不可能執行無限項次的加法，必須以有限的迴圈數計算到最精確的數值。下列程式可供參考。

clear	
x=2;	
N=20;	迴圈數 (累加項次)
fval=0;	累加函數值
for i=0:N	
fval=fval + x^i/factorial(i);	進行累加
end	
fprintf('The exp(x) evaluated at x=%7.2f is %10.5f ',...	
x, fval)	結果呈現

上述程式設定迴圈數 $N=20$ ，這個作法並不精確，不過以目前學習到的程式技巧，暫時可用。正確的作法是設定更多的迴圈數，設法加入更多的項次，直到預設的精確值到達為止。這牽涉到終止迴圈的技巧，將於後面的單元說明。關於上述程式的結果，可以比較 MATLAB 提供的 $\exp(2)$ 。

上述程式最後一行利用列印指令 `fprintf` 將結果呈現出來，當指令太長時，如果不希望超出目前的畫面，可以強制斷行。只要在切斷處補上三個點「 $\cdots$ 」即可，後面的剩餘部分便可以移至下一行。

3 觀察

1. 迴圈技巧並非萬靈丹，也不一定是解決問題的首選。有些情形簡單的矩陣便可以輕鬆解決，譬如，計算 $\sum_{k=1}^n x_k y_k$ ，只需將 $x = [x_1 x_2 \cdots x_n]$ 向量乘上 $y = [y_1 y_2 \cdots y_n]$ ，即 xy' ，不必一看到 $\sum$ 只會想到迴圈。MATLAB 是矩陣運算的高手，寫 MATLAB 程式盡量從矩陣運算的角度來看會比較順手。

4 作業

1. 利用迴圈技巧設計一個如圖 3 的遊戲。
2. 模仿範例 4，畫出 T 分配的密度函數與自由度的變化圖。
3. 利用矩陣運算的方式計算範例 6 的 N 個平均數。

4. 改寫範例 7 的程式，儲存每個迴圈的累加結果在一個向量變數裡，最後列出該向量，藉此可以協助判斷累加項次 $N=20$ 是否合適。