



作者：張彰堂

Matlab 使用教學

目錄：

一、前言.....	1
二、開發環境.....	2
三、基本操作.....	5
1. 善用漁網.....	5
2. 變數型別、宣告、定義.....	6
3. 算數、邏輯運算.....	7
4. 程式流程控制.....	9
四、繪圖功能.....	12
五、讀取與顯示影像檔.....	13
六、結語.....	16
七、補充資料.....	16

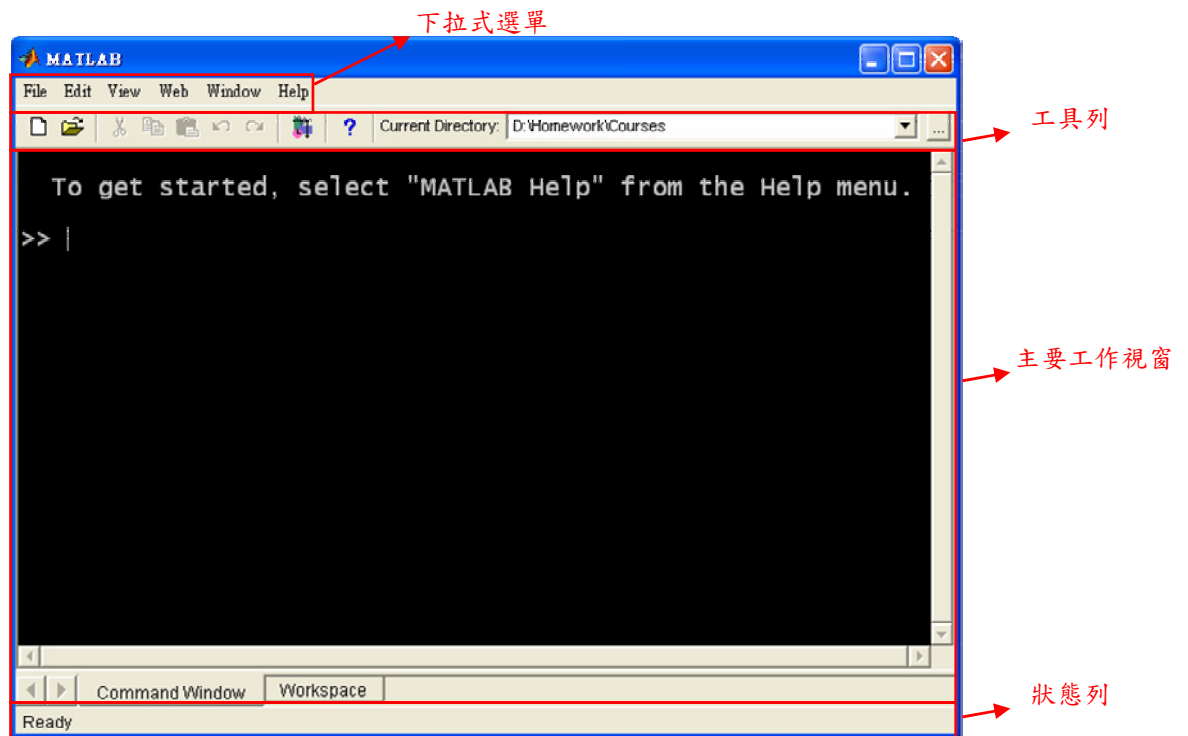
一、前言：

我們身處在目前這樣資訊科技爆炸的時代，早已不像古時候的科學家，僅僅靠著一枝筆、一張紙便足以解決問題。對於數目龐大、為數眾多與複雜的計算，只有仰賴一些工具來快速地達成我們的目標。毫無疑問的 Matlab 這套具有數值計算分析、系統模擬控制以及方便的繪圖能力等等集其一生的強大軟體，已經成為所有學科學、工程、尤其是電機資訊的同學們不可不學、不可不會、不可不用的工具。而這份文件的目的是，就是希望讓同學們很快的上手使用 Matlab。

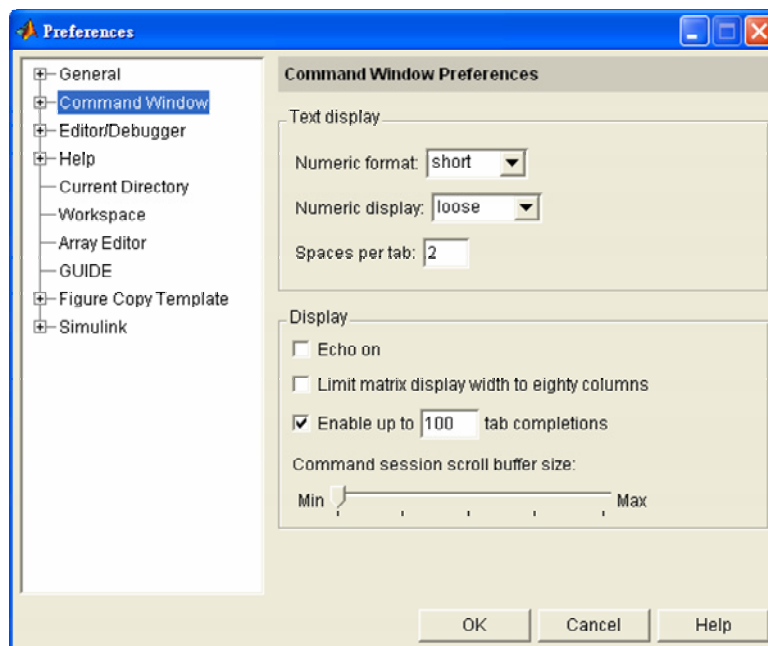
首先，我們假設同學們已經有一些程式語言的基礎（如 C 語言，的確 Matlab 的語法跟 C 語言相當地類似，並且相對於 C 語言而言，Matlab 是更高階的程式語言，若同學們對於 C 語言已經很熟悉的話，學習 Matlab 將更是得心應手）。教學大致分成幾個方向：Matlab 開發環境、基本語法、算數邏輯運算、讀取與顯示影像檔等等。主要目標還是要讓同學們能正確地將想法、演算法實作出來（且為了釐清觀念，避免混淆，將不會介紹 Matlab 其他的功能，或者是如何針對 Matlab 的一些特性實作更有效率的程式，對於這方面有興趣的同學，建議可以參閱坊間一些 Matlab 的書籍）。

二、開發環境：

在這裡，我們將以 Matlab 6.0 版的使用介面為各位作個簡單的介紹。倘若同學們使用的是 Matlab 5.x 版，或更之前的版本也沒有關係，也許介面外觀不太一樣，但您將一樣能找到其相對應的名稱與功能。以下使啟動後的 Matlab 外觀：



各位的外觀也許跟上圖不太一樣，在下拉式選單之中最重要的就是 File -> Preferences，如下圖：



在這之中，可以為自己的喜好作一些字型、顏色等等的設定。

在 Matlab 6.0 的主要工作視窗中，一共可包含了以下幾個子視窗，若欲開啟這些視窗，可以在下拉式選單中的 View -> 中勾選：

- ✓ Command Window：命令視窗
- ✓ Command History：命令歷史紀錄
- ✓ Current Directory：目前工作目錄
- ✓ Workspace：工作空間
- ✓ Launch Pad：快速啟用支援（分類說明、範例）
- ✓ Help：說明

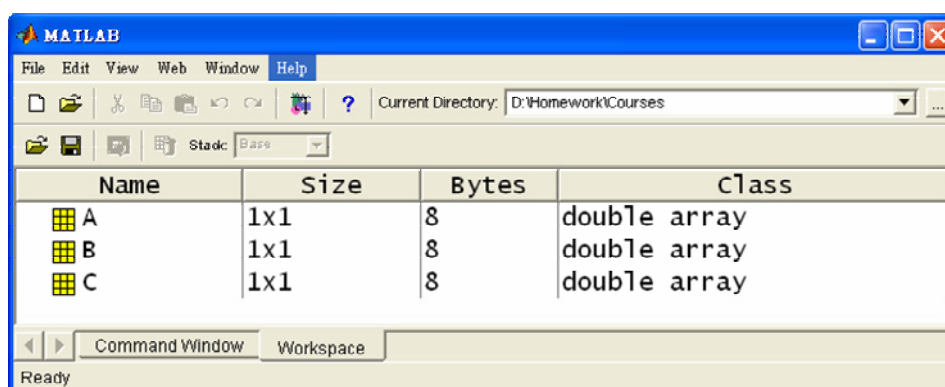
在圖中，只顯示出 Command Window（命令視窗）與 Workspace（工作空間），而這兩個也是最常使用的功能並會在以下作詳細的介紹：

Command Window 是 Matlab 主要的操作視窗，使用上有點類似 DOS 或者是 Unix 的 console，可以在裡面輸入想要執行的算式、函式（function）、甚至 DOS 的指令。程式執行的結果，也會顯示在 Command Window 裡。視窗中有一個 >> 的提示符號，同學們可以試著在後面輸入以下內容（紅色），看看其顯示的結果：

```
>> A = 1
A =
    1
>> B = 2
B =
    2
>> C = A + B
C =
    3
```

Matlab 便會幫你計算出 $C = 3$ ，是不是很容易呢？（至於變數的宣告，定義等等在之後會詳細介紹。）

Workspace 是 Matlab 儲存變數的空間，如果已經輸入以上的算式，Workspace 應該會顯示如下：

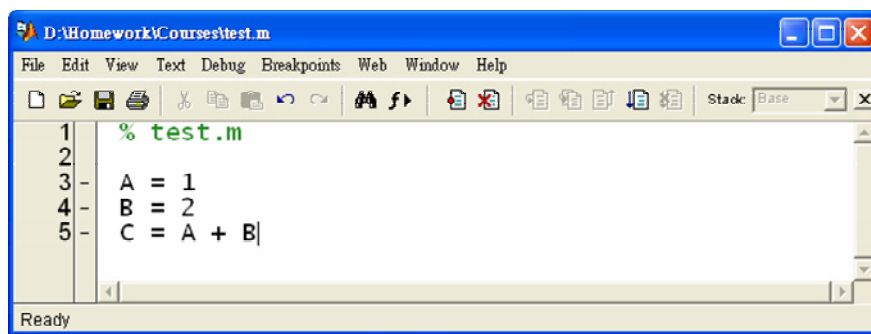


其中 Name 表示變數名稱，Size 表示維度，Bytes 為使用的記憶體，Class 是形別。

若將滑鼠移至某個變數上，點選後連按滑鼠左鍵兩次，便可以開啟 Array Editor（矩陣編輯器）並會顯示出該變數的值，供使用者瀏覽與編輯。

其餘的功能便只作簡單的說明，同學們可以自行使用看看。Command History（命令歷史紀錄）會記錄使用者於 Command Window 裡所鍵入的指令，可以快速瀏覽並點選重複欲執行的指令。Current Directory（目前工作目錄）有點像 Windows 的檔案總管，將目前所在的目錄內容顯示出來。Launch Pad 在這裡翻譯成快速啟用支援，同學們開啟後其實是一個樹狀目錄，裡面包含著 Matlab 的一些應用方向的使用說明與範例展示建議可以看看 Matlab 以及 Image Processing Toolbox 的說明，順便玩玩他的展示程式，相當有趣喔！最後 Help（說明），方便同學們查詢的介面，蠻容易也很常使用，就請同學們自行試試囉！

在下拉式選單的 File 中，可選擇 New 或 Open 一個 M-File（副檔名為 m 的檔案），或者直接從工具列最左邊的兩個圖示便是 New M-File 與 Open File。我們可以將欲執行的大量指令儲存成 M-File，之後只需在 Command Window 鍵入 M-File 的檔名，便會執行其內容。以下以同樣的例子說明如何使用 M-File。先點選 New M-File，之後會開啟一個 M-File 的編輯器如下，並鍵入以下內容，儲存成 test.m：



接著在 Command Window 下鍵入 test，是不是也得到同樣的結果呢？

編輯器中的（%）是 M-File 的註釋（Comment）字元，相當於 C++ 的（//）。值得注意的是，Matlab 並不像 C 可以分行，所以也沒有（/* */）這樣的註釋。若算式太長需分行時，則需在結尾加上連續三個句點（...）表示接續下一行。也因為 Matlab 不可以分行，故在敘述（statement）結尾不需加上分號（；），而（；）在 Matlab 則有另一個意義，加上（；）表示隱藏運算結果，同學們可以將上面的每個敘述加上（；），再執行一次 test，看看有什麼不同。

在使用 M-File 的時候，還有一個需注意的地方，就是 current directory，必須將其移至存放 M-File 的路徑，才可以執行該 M-File。有幾個方便的方法可以解決每次要更改目錄的困擾：第一，可以在下拉式選單的 File -> Set Path... 中將存放 M-File 的路徑加入；第二，可以將 Matlab 的捷徑中選內容，並將開始位置改為存放 M-File 的位置。若欲更改 current directory，可以點選工具列的（...），或者在命令視窗中使用 pwd（print working directory）顯示 current directory 與 cd（change directory）更改。

三、基本操作：

1. 善用漁網：

俗話說得好，釣魚給人吃，倒不如給他一根釣竿，教他如何釣魚。而我現在要為各位介紹的東西，不僅僅只是釣竿而已，甚至可以稱為漁網，那就是 Matlab 的 Help（查詢指令與線上查詢功能）。Matlab 功能強大的其中一個地方便是他擁有相當豐富的說明文件，可別因為他所佔的硬碟空間而省略安裝，他可以快速解決您的問題，您將會發現他有多方便！

Matlab 的 Help 大致可以分為二類共五個：

- Command Window 下：
 - help：指令用法查詢
 - lookfor：關鍵字搜尋
- 視窗或瀏覽器介面：
 - helpwin：視窗版的指令用法查詢與關鍵字搜尋
 - helpdesk：線上說明文件（html/pdf）
 - doc：特定函式的線上說明文件

以上之中，最常使用的便是 help，他可以快速查詢欲使用的指令或函式的說明，譬如欲知道 sin 怎麼使用即鍵入以下：

```
>> help sin

SIN    sine.
        SIN(X) is the sine of the elements of X.
Overloaded methods
        help sym/sin.m
```

當然也可以使用 help help 來查詢怎麼使用 help 指令。

lookfor 是關鍵字搜尋，假設您想使用 cosine 這個功能，卻不知到有哪些相關的函式可以使用，便可以輸入以下來作搜尋：

```
>> lookfor cosine
ACOS    Inverse cosine.
ACOSH   Inverse hyperbolic cosine.
COS     Cosine.
COSH    Hyperbolic cosine.
DCT2    Compute 2-D discrete cosine transform.
DCTMTX  Compute discrete cosine transform matrix.
DCTMTX2 Discrete cosine transform matrix.
IDCT2   Compute 2-D inverse discrete cosine transform.
DCT     Discrete cosine transform.
IDCT    Inverse discrete cosine transform.
COSINT  Cosine integral function.
ACOS    Symbolic inverse cosine.
ACOSH   Symbolic inverse hyperbolic cosine.
COS     Symbolic cosine function.
COSH    Symbolic hyperbolic cosine.
COSINT  Cosine integral function.
DCT     Discrete cosine transform.
IDCT    Inverse discrete cosine transform.
```

以上列出部分搜尋的結果，若想使用的是 inverse cosine，便可以進一步使用 help 查詢 ACOS 的使用方法。

helpwin 是視窗介面的 help 與 lookfor 功能，而 helpdesk 與 doc 使用來瀏覽 html/pdf 的說明文件，並在 Matlab 6.0 版之後整合在一起；同學們可以自行使用看看。

2. 變數型別、宣告、定義：

若您使用過 C 語言，便應該知道變數在使用之前都需事先宣告 (declaration)。Matlab 就不必這麼麻煩，在使用時自行產生適當的變數。並且 Matlab 預設資料型別為 double，並且所有的算數運算皆以 double 來運算，如此可以省去型別轉換的困擾（不過 Matlab 仍支援許多資料型別，在必要時可以作型別轉換，請參考 help datatypes）。以最前面使用的例子來看，執行 $C = A + B$ 時，並不知道 C 的資料型別該是什麼，不過執行完 $A + B$ 之後的資料型別為 double（預設為 double），於是便產生變數 C 為 double 以儲存 $A + B$ 的結果。

在變數的命名上，如同 C 語言一樣英文字母大小寫會被視為不同的變數，且變數名稱最長為 31 的字元長度，另外需注意的，Matlab 的變數並不可以以底線 (_) 字元作為開頭字母。

Matlab 最強大的功能其實在於能處理矩陣運算，以前面的例子來看其實 A、B、C 是以 1x1 double array 作為資料型別。我們可以使用中括號 ([]) 來定義我們的矩陣如： $A = [1 \ 2]$ ，便是一個 1x2 double array，也可以再加上分號 (;) 來定義二維陣列的各個 row 中的元素如： $B = [1 \ 2; 3 \ 4]$ 。接著，若要取用陣列中某個元素的值，可以使用小括號 (()) 如： $A(2)$ 表示 A 的第二個元素，或 $B(2, 1)$ 表示 B 的第二個 row 與第一個 column 的元素。也可以使用冒號 (:) 來存取某一些元素如： $B(1:3)$ 是第一個到第三個元素，與 $A(:)$ 表示 A 所有的元素。以下是應用的結果：

```
>> A = [1 2]
A =
     1     2
>> B = [1 2; 3 4]
B =
     1     2
     3     4
>> A(2)
ans =
     2
>> B(2, 1)
ans =
     3
```



```
>> B(1:3)
ans =
     1     3     2
```

```
>> A(:)
ans =
     1
     2
```

B 的元素

1	2
3	4

在記憶體中是先存 column 再存 row
所以，若以一維來存取則為 1, 3, 2, 4
因此，第一個到第三個元素為便是 1, 3, 2

若變數使用完，不再需要時可以使用 clear 指令來將該變數自 Workspace 中清除如下：

```
>> A = 1
A =
     1
```

```
>> clear A
>> A
??? Undefined function or variable 'A'.
```

已經不存在了

我們還可以使用 size 與 length 這兩個函式來取得變數的維度或長度如下：

```
>> A = [1 2];
>> B = [1 2; 3 4];
>> length(A)
```

```
ans =
     2
```

陣列長度為 2

```
>> size(B)
```

```
ans =
     2     2
```

矩陣維度為 2x2

3. 算數、邏輯運算：

Matlab 的算數運算包含以下幾種：

Arithmetic operators.

- plus - Plus (加法) +
- minus - Minus (減法) -
- mtimes - Matrix multiply (矩陣乘法) *
- times - Array multiply (元素乘法) .*
- mpower - Matrix power (冪次) ^
- power - Array power (元素冪次) .^
- mldivide - Backslash or left matrix divide (左除法) \
- mrdivide - Slash or right matrix divide (右除法) /
- ldivide - Left array divide (元素左除法) .\
- rdivide - Right array divide (元素右除法) ./

應用如下：

```
>> S = (1+2)*3/5*2^2
S =
    7.2000

>> AM = [1 2];
>> BM = [3 4];
>> CM = AM + BM - [1 1] - 2
CM =
     1     3

>> AM = [1 2];
>> BM = [3; 4];
>> CM = BM * AM
CM =
     3     6
     4     8

>> AM = [5 10];
>> BM = [1 2];
>> CM = AM ./ BM
CM =
     5     5

>> DM = AM .\ BM
DM =
    0.2000    0.2000
```

純量運算

矩陣加減，若加減一個純量則相當於每個元素作加減

矩陣乘法，BM 的 column 要等於 AM 的 row

AM 除以 BM 對應的個別元素

BM 除以 AM 對應的個別元素

矩陣的左除法與右除法初學者容易混淆，我建議同學們先不要使用，而規矩一點使用矩陣的 inverse 來處理可以避免錯誤的發生。

Matlab 的邏輯運算與 bitwise 運算是分開不同的，表示法也與 C 語言不竟相同（注意：（ \wedge ）是幕次算子而不是 xor，也沒有左移（ \ll ）與右移（ \gg ）而是 bitshift）。所有的邏輯運算與 bitwise 運算包含以下幾種：

Logical operators.

- and - Logical AND &
- or - Logical OR |
- not - Logical NOT ~
- xor - Logical EXCLUSIVE OR
- any - True if any element of vector is nonzero
- all - True if all elements of vector are nonzero

Bitwise operators.

- bitand - Bit-wise AND.
- bitcmp - Complement bits.

- bitor - Bit-wise OR.
- bitmax - Maximum floating point integer.
- bitxor - Bit-wise XOR.
- bitset - Set bit.
- bitget - Get bit.
- bitshift - Bit-wise shift.

使用方法蠻容易的，自行使用 help 察看。

4. 程式流程控制：

在介紹流程控制之前，要先為各位介紹關係運算子（relational operator）：

Relational operators.

- | | | |
|------|-------------------------|----|
| ■ eq | - Equal | == |
| ■ ne | - Not equal | ~= |
| ■ lt | - Less than | < |
| ■ gt | - Greater than | > |
| ■ le | - Less than or equal | <= |
| ■ ge | - Greater than or equal | >= |

其中與 C 語言不同的是 not equal 為（~=）而不是（!=）需特別注意。再與之前介紹過的邏輯運算子，便可以表示條件判斷式（boolean expression）。

程式的流程控制可以分為三個部分：條件判斷、迴圈、函式呼叫。

✧ 條件判斷：

Matlab 的條件判斷有二種：if-else 以及 switch-case。

if-else 之語法如下：

```

if 條件判斷 (boolean expression)
    敘述 (statement)
else
    敘述 (statement)
end

```

也可以使用多個條件判斷如下：

```

if 條件判斷 (boolean expression)
    敘述 (statement)
elseif 條件判斷 (boolean expression)
    敘述 (statement)
else
    敘述 (statement)
end

```

注意：if 最後要加上對應的 end，並且多重條件判斷之 elseif 需連在一起寫。

switch-case 的語法如下：

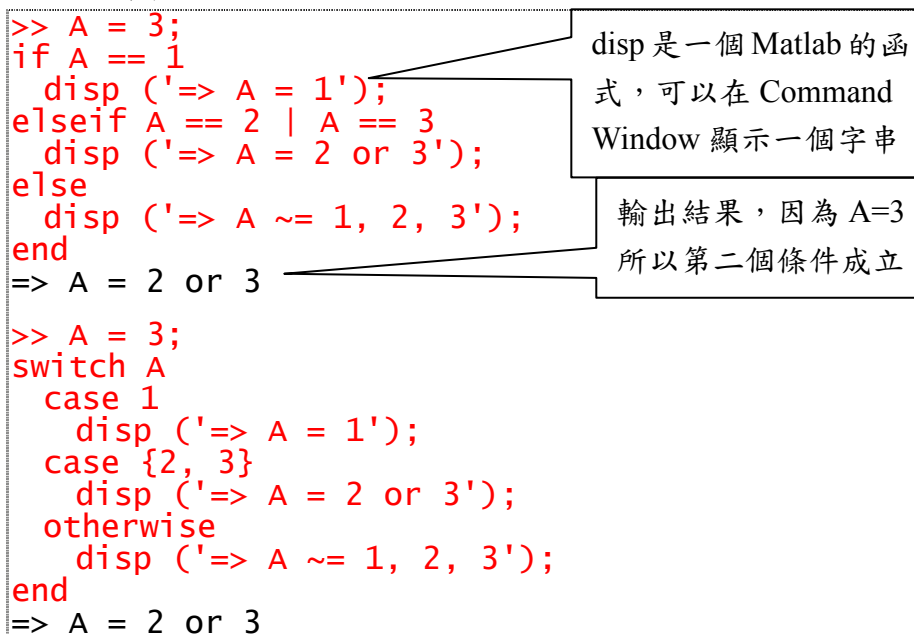
```

switch 變數
    case 數值集合
        敘述 (statement)
    case 數值集合
        敘述 (statement)
    otherwise
        敘述 (statement)
end

```

注意：switch 最後一樣要加上 end，並且與 C 語言不同的是，每個 case 結束無須加上 break，default 在 Matlab 中是使用 otherwise。

以下是一些簡單的例子：



```

>> A = 3;
if A == 1
    disp ('=> A = 1');
elseif A == 2 | A == 3
    disp ('=> A = 2 or 3');
else
    disp ('=> A ~= 1, 2, 3');
end
=> A = 2 or 3

```

disp 是一個 Matlab 的函式，可以在 Command Window 顯示一個字串

輸出結果，因為 A=3 所以第二個條件成立

```

>> A = 3;
switch A
    case 1
        disp ('=> A = 1');
    case {2, 3}
        disp ('=> A = 2 or 3');
    otherwise
        disp ('=> A ~= 1, 2, 3');
end
=> A = 2 or 3

```

◇ 迴圈：

Matlab 的迴圈也有二種：for 迴圈與 while 迴圈。

for 迴圈的語法如下：

```

for 變數 = 向量
    敘述 (statement)
end

```

在執行時，變數會依序等於向量中的元素值，並處理 for 迴圈的敘述。

while 迴圈的語法如下：

```

while 條件判斷 (boolean expression)
    敘述 (statement)
end

```

在執行時，只要條件成立，便會執行 while 迴圈中的敘述。

以下是簡單的例子：

```
>>
for i = 1:3
    A(i) = i;
end
A
A =
     1     2     3

>>
i = 1;
while i <= 3
    B(i) = i;
    i = i + 1;
end
B
B =
     1     2     3
```

i 會依序等於 1, 2, 3 並執行 A(i)=i 的動作

初始化 i

記得更新 i 的值，以免進入無窮迴圈

在 Matlab 的迴圈之中，也可以使用與 C 語言相同的 break 與 continue (6.0 版之後)，使用方法也跟 C 語言相同，故不多做介紹。

☆ 函式呼叫：

在 Matlab 中，我們也可以將時常使用的功能或者遞迴 (recursion) 編寫成一個函式，供需要時進行呼叫。Matlab 的函式必須為一個獨立的 M-File (最好檔名與函式名稱相同)，並在 M-File 開頭以 function 這個關鍵字宣告函式的名稱與輸入及輸出。

以下是一個簡單的例子，寫一個名為 MyMax 的函式用以比較兩個純量 a 與 b，並傳回較大的值 Max，與索引值 i。請先再 M-File 的編輯器輸入以下內容，並存成 MyMax.m：

```
function [Max, i] = MyMax(a, b);
% MyMax Compare a, b
% return the max value & index of them.

if a >= b
    Max = a; i = 1;
else
    Max = b; i = 2;
End
```

在 Command Window 下進行測試：

```
>> a = 1;
>> b = 2;
>> [Max, i] = MyMax(a, b)

Max =
     2

i =
     2
```

四、繪圖功能：

Matlab 的繪圖功能強大，這點絕對不是其他語言可以相比的。而在此我們僅僅只針對二維最基本的繪圖功能為大家介紹：

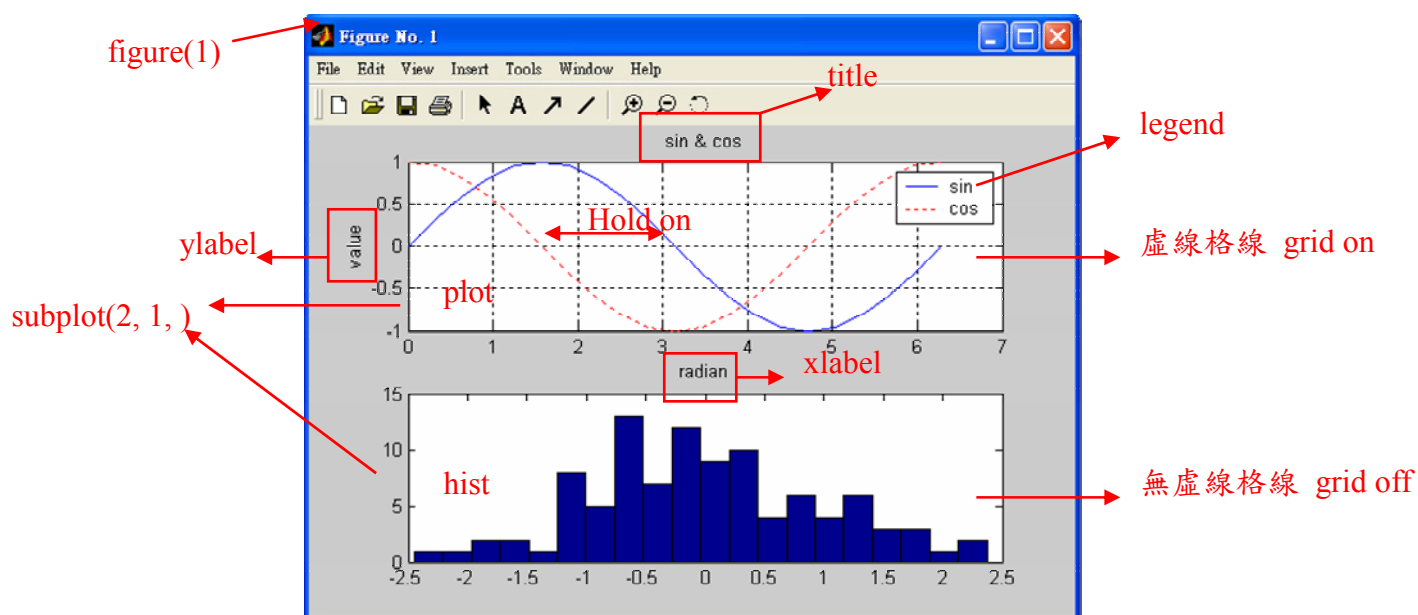


figure	圖表視窗
subplot	分割視窗
hold on/off	增添資料至原圖表/覆蓋原圖表
title	圖表標題
xlabel/ylabel	圖表水平標籤/垂直標籤
legend	圖表圖示說明
grid on/off	顯示格線/不顯示格線

繪圖指令，也簡單介紹最常使用的二個 plot 用來描繪 x-y 座標圖與 hist 用來描繪累計直方圖。以下是描繪上圖的程式碼及說明：

```

x = [0:0.1:2] * pi;    % 產生 0~2 每 0.1 為單位遞增之向量再乘以 pi
A = sin(x); B = cos(x);
C = randn(100, 1);    % 以常態分佈亂數產生 100x1 的向量
figure(1);             % 第一個圖表視窗
subplot(2, 1, 1);      % 分割視窗為 2x1，現在使用第一個
plot(x, A, 'b-');       % 描繪二維座標，b：藍色，-實線
hold on;               % 保留之前的圖
plot(x, B, 'r:');       % 新增，描繪二維座標，r：紅色，：虛線
title('sin & cos');     % 顯示標題
xlabel('radian');       % 顯示 x 座標的標籤
ylabel('value');       % 顯示 y 座標的標籤
legend('sin', 'cos');  % 顯示圖示說明
grid;                 % 顯示格線
subplot(2, 1, 2);      % 分割視窗為 2x1，現在使用第二個
hist(C, 20);          % 以 C 向量，分成 20 等分來描繪累積直方圖

```

五、讀取與顯示影像檔：

在介紹完 Matlab 的基本使用方法後，在這一節就要專門為各位介紹如何使用 Matlab 進行讀取與顯示一個影像檔。在說明之前當然需要先為各位介紹影像檔的基本知識與格式，這裡主要使用 Windows 的 BMP 檔與原始資料 RAW 檔。

一般完整的單色灰階影像是使用 8 bits 一共 256 個灰階（0~255）來表示，0 表示最深也就是黑色，255 表示最淺也就是白色。如以下所示：



而原始資料 RAW 檔，就是一個完全沒有檔案資訊，純粹影像資料的一種格式，例如：一張 176x144 灰階 RAW 檔的大小也就是 $176 \times 144 \times 8 \text{ bits} = 25344 \text{ bytes}$ 。（也由於沒有任何檔案資訊，對於一張 25344 bytes 的檔案有太多種寬高的組合，必須事先知道影像的寬與高，之後介紹的 BMP 檔可以解決這種問題。）以下是一個簡單的例子，如何讀取一個灰階的 RAW 檔（必須使用類似 C 語言的 fopen/fread/fclose 三個函式）：

```
fid = fopen('lena_mono.raw', 'r');
GY = fread(fid, [256 256], 'uint8');
fclose(fid);
```

第一行 fopen 開啟一個名為 lena_mono.raw 的檔案，屬性為 read，並傳回檔案的指標於 fid。第二行於 fid 這檔案讀取 256x256 的矩陣（假設影像檔的寬高為 256x256），每個元素大小為 8 bits 的 unsigned integer。第三行將檔案關閉。接著我們可以檢視一下 Workspace，看看是不是有一個 GY 的陣列，大小為 256x256（不過由於 Matlab 預設檔案型別為 double，因此已經自動轉為 double 了）。

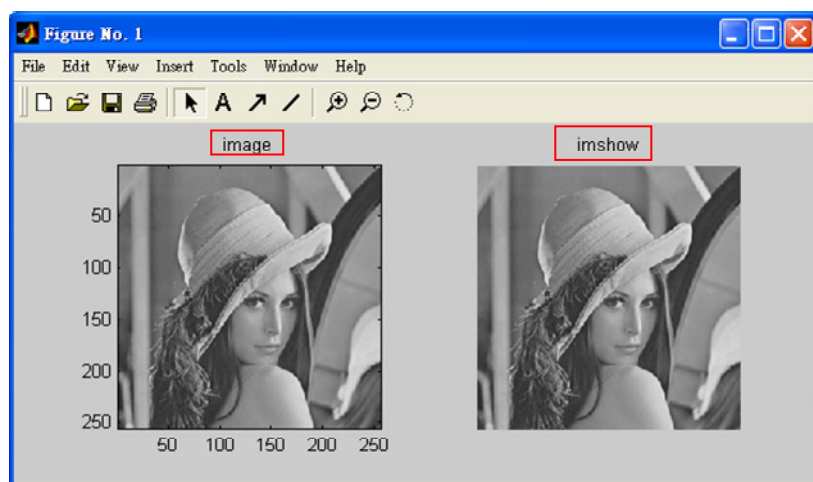
有了影像的資訊，接下來就是該如何顯示出來呢？這包含了一個很重要的觀念，就是 colormap（顏色對照表）。為什麼需要 colormap 呢？我們從檔案讀出來的只是一些 0~255 的值，然而這些值究竟該使用什麼顏色來顯示，就要依靠這個 colormap 了。所幸，Matlab 也已經幫我們設計一些常用的 colormap 相關函式，我們可以很輕易地產生我們所需的 colormap。並且 Matlab 也提供了二個函式供我們顯示二維的影像。以下是接續著上面的程式碼以顯示出灰階的影像：

```
GY = GY';
figure(1);
subplot(1, 2, 1);
image(GY);
colormap(gray(256));
axis image;

subplot(1, 2, 2);
imshow(GY, gray(256));
```

第一行先將我們的陣列做一次轉置（transpose），還記得我們前面提過 Matlab 的陣列是先儲存 column 再儲存 row 的嗎？然而我們的圖檔卻是水平的一橫行一橫行儲存起來，因此為了正確顯示他的方向，需先做一次轉置。再來我將視窗分成左右二個，分別使用 image 與 imshow 兩個函式顯示影像。image 函式直接輸入影像的陣列，colormap 是用來設定現在使用的 colormap，裡面的 gray(256) 表示產生 256 個灰階的 colormap，最後使用 axis image 來調整座標，使其以正確的長寬比顯示。imshow

是一個更快，更簡單的方法，他可以直接輸入影像陣列，接著輸入 colormap 便完成了。顯示結果如下：



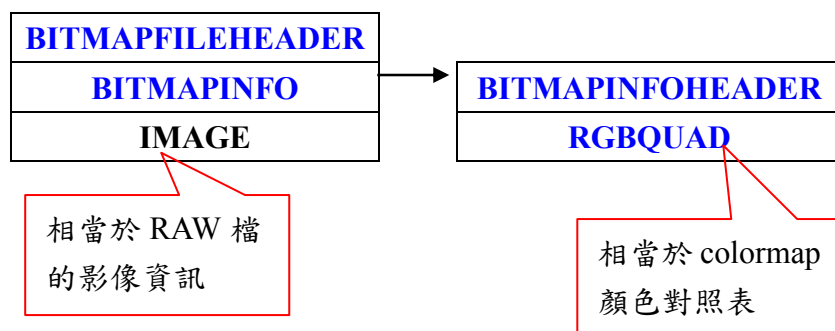
相信大家對於灰階圖檔的讀取與顯示都已經學會了。但是倘若所有的影像都只能使用灰階來顯示，那我們的人生就是黑白的了。因此接著要為各位介紹彩色的影像檔之讀取與顯示。

一個全彩的影像 (true color)，是使用 Red、Green、Blue 三種顏色所組成的。每一個顏色都是以 8 bits 來表示 (0~255)，所以也有 256 個色階。因此若要表示一個彩色的 pixel 需要 $8 \times 3 = 24$ bits。RGB 色階分別如下所示：



因此，若我們以向量 (R, G, B) 來表示，則紅色為 (255, 0, 0)，綠色為 (0, 255, 0)，藍色為 (0, 0, 255)，黑色為 (0, 0, 0)，白色為 (255, 255, 255)，紅色加綠色就變成黃色 (255, 255, 0)，紅色加藍色就變成紫色 (255, 0, 255) 這樣瞭解了嗎？

接著我們來看看如何讀取彩色的影像檔，由於原始 RAW 檔缺少檔案的資訊，因此非常不方便使用，接著就要為各位介紹常見的 BMP 檔。BMP 檔最前面包含有一小段 header (檔頭)，裡面含有一些影像的相關資訊，經由這些資訊我們可以知道影像的寬與高、灰階 (8 bits) 還是彩色 (24 bits) 等等。以下是 BMP 檔案結構示意圖：



BITMAPFILEHEADER、BITMAPINFO、BITMAPINFOHEADER 的 C 語言結構如下（節錄自 MSDN April, 2001）：

```
typedef struct tagBITMAPFILEHEADER
{
    WORD    bfType;           //固定為 BM
    DWORD   bfSize;           //檔案大小
    WORD    bfReserved1;      //保留為 0
    WORD    bfReserved2;      //保留為 0
    DWORD   bfOffBits;        //影像資訊的 bytes offset
} BITMAPFILEHEADER, *PBITMAPFILEHEADER;

typedef struct tagBITMAPINFO
{
    BITMAPINFOHEADER bmiHeader; //在下面
    RGBQUAD           bmiColors[1];
} BITMAPINFO, *PBITMAPINFO;

typedef struct tagBITMAPINFOHEADER
{
    DWORD   biSize;           //這個結構大小
    LONG    biWidth;          //影像寬
    LONG    biHeight;         //影像高
    WORD    biPlanes;         //
    WORD    biBitCount;       //灰階 8，全彩 24 bits
    DWORD   biCompression;    //
    DWORD   biSizeImage;      //影像大小
    LONG    biXPelsPerMeter;   //
    LONG    biYPelsPerMeter;   //
    DWORD   biClrUsed;        //
    DWORD   biClrImportant;    //
} BITMAPINFOHEADER, *PBITMAPINFOHEADER;
```

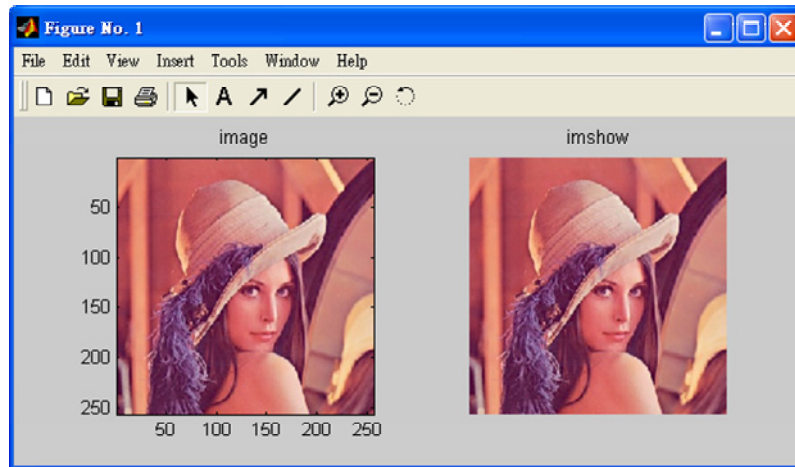
乍看之下似乎不太容易，不過 Matlab 已經提供我們方便的工具以獲得檔案資訊，便是 `imfinfo`，試試看輸入下指令：

```
>> imfinfo('lena_color.bmp')
```

Matlab 也已經提供 BMP 檔案的讀寫功能（`imread/imwrite`），以下直接用一個例子來告訴大家怎麼使用 Matlab 的函式，快速讀取與顯示 BMP 檔：

```
RGB = imread('lena_color.bmp');
figure(1);
subplot(1, 2, 1);
image(RGB);
axis image;
subplot(1, 2, 2);
imshow(RGB);
```

第一行 `imread`，直接輸入 BMP 檔名，就可以得到 RGB 的影像陣列，大小為高 x 寬 x 3 (R, G, B)，這個函式已經幫你做轉置的動作，因此不必再做轉置。顯示影像的方式跟灰階很類似，唯一比較不同的是 `colormap` 的部分不見了。我們再複習一下，所謂 `colormap` 是用來提供我們讀進來的數值 (0~255) 該使用什麼顏色來顯示，然而對於全彩的圖片已經使用了所有的色彩，因此不需要再設定顏色的對應。以下是輸出結果：



六、結語：

有了以上的技巧，相信同學們已經可以使用 Matlab 來開始嘗試寫作影像處理的方面的程式。若有什麼解說不清楚的地方，歡迎前來討論指教，謝謝。

七、補充資料（感謝劉昭勇提供）：

其他一些好用的 Matlab 指令，若對這些指令有興趣的人，可參考 Matlab 說明文件中，有其使用範例：

!	後可接一些 DOS 指令，如 <code>!copy</code>
A(:)	A 在此為變數名，將任何維度的變數，以 column vector 表示
find	Find indices and values of nonzero elements， <u>非常重要的指令</u>
end	Indicate last index of variable，如 <code>A(4:end)</code>
reshape	Reshape array
cell	Create cell array (可放任意資料型態的變數)
save/load	Save/Load workspace variables on disk

下列函式在 Image Toolbox 下，對處理二維資料特別好用的函式，下列兩個函式，若靈活應用，可將 for loop 用矩陣運算方式取而代之，在 Matlab 中，可大幅降低其運算速度及時間（但程式的可讀性也跟著降低，因此，不建議在作業中使用，以後需要時再用）

blkproc	Implement distinct block processing for an image
im2col	Rearrange image blocks into columns