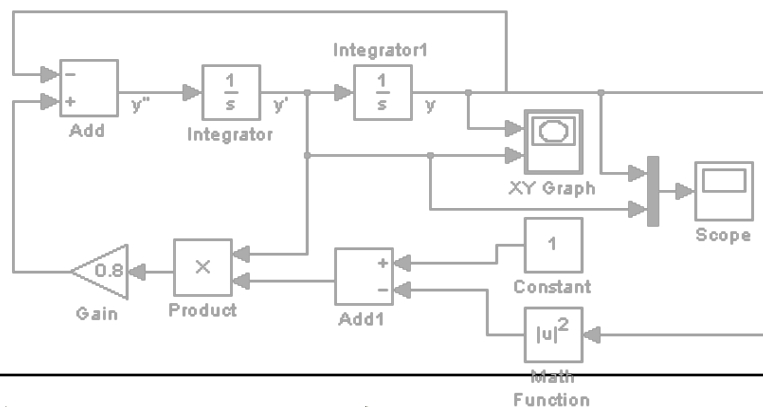


第七章

使用M檔案與函數



本章學習目標

認識M檔案

學習撰寫底稿與函數

學習偵錯的技巧

學習如何使用全域變數

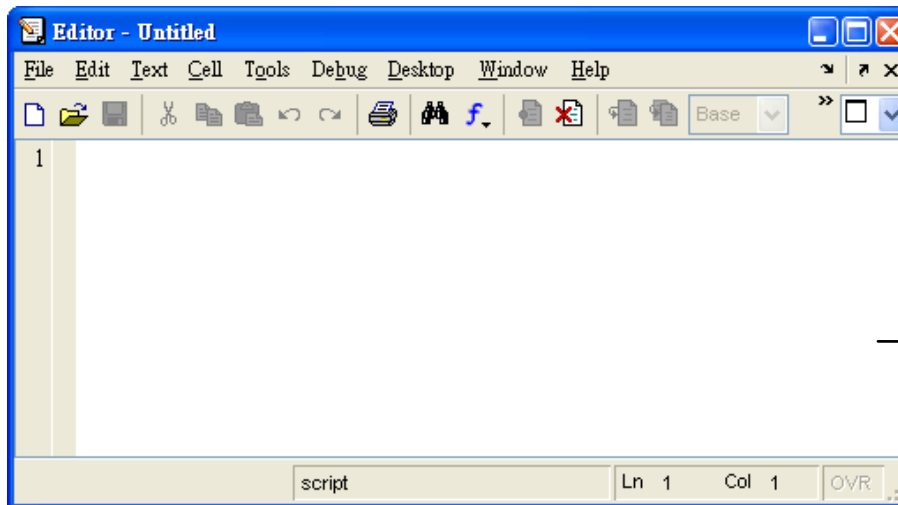
學習Matlab搜尋M檔案的方式



7.1 撰寫底稿

- 底稿（script）是由一系列Matlab的敘述所組成
- 底稿可方便編輯、除錯與執行程式碼
- 要開啟M檔案編輯器，可在指令視窗裡鍵入

>> edit



— M 檔案編輯區

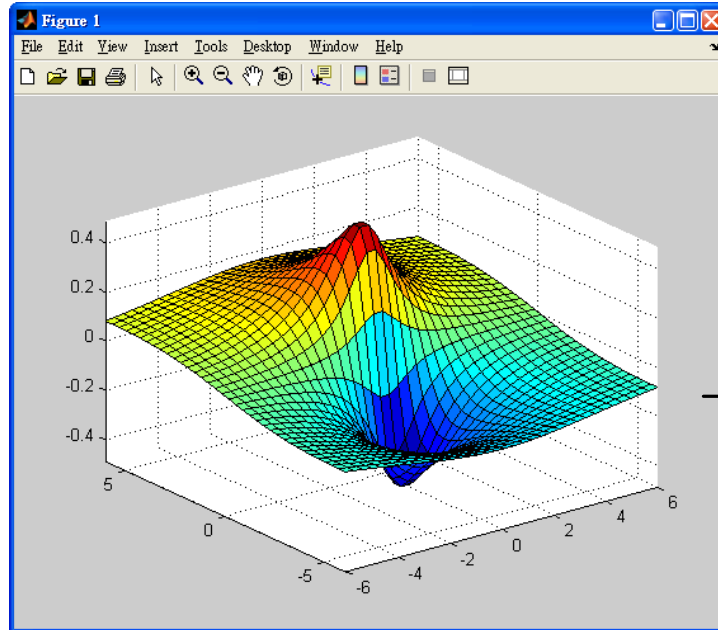
-
- 下面的程式碼鍵是簡單的底稿：

```
% script7_1.m, 底稿練習-繪出三維函數圖  
clear  
x=linspace(-6,6,36);  
y=linspace(-6,6,36);  
[xx yy]=meshgrid(x,y);  
zz=yy./(xx.^2+yy.^2+1);  
surf(xx,yy,zz); axis tight
```

- 鍵入並儲存好了之後，鍵入底稿的名稱

```
>> script7_1
```

即可執行這個底稿：



執行底稿 `script7_1`，即可繪出
 $f(x, y) = y / (x^2 + y^2 + 1)$ 的三
維函數圖

7.2 設計函數

- 函數（function）也是M檔案的一種
- 函數與底稿不同之處：
 1. 函數可傳入引數，也可以把運算結果傳回工作區，而底稿不行
 2. 在函數內使用的變數是區域變數，底稿是全域

7.2.1 函數的基本架構

`function` 輸出變數 = 函數名稱 (引數 1, 引數 2, ...)

— 函數定義列

%H1 列，此行可用來簡述函數的功用

— H1 列

%此區是函數的說明文字，可用來註解
%函數的語法、注意事項等

— 函數說明文字區

函數的主體

— 函數的主體

>> type linspace.m

```
function y = linspace(d1, d2, n)      —— 函數定義列
%Linspace Linearly spaced vector.    —— H1 列
%   Linspace(X1, X2) generates a row vector of 100 linearly
%   equally spaced points between X1 and X2.
%
%   Linspace(X1, X2, N) generates N points between X1 and X2.
%   For N < 2, Linspace returns X2.
%
%   Class support for inputs X1,X2:
%       float: double, single
%
%   See also LOGSPACE, :.
```

函數說明文字區

```
%   Copyright 1984-2004 The MathWorks, Inc.
%   $Revision: 5.12.4.1 $   $Date: 2004/07/05 17:01:20 $

if nargin == 2
    n = 100;
end

n = double(n);
y = [d1+(0:n-2)*(d2-d1)/(floor(n)-1) d2];
```

函數的主體

7.2.2 簡單的範例

- 函數func7_1可接收兩個引數，並傳回其加總：

```
function total=func7_1(x,y)
%FUNC7_1 sum of two numbers or vectors.
%FUNC7_1(X,Y) computes X+Y and returns the result.
%X and Y can be scalars or vectors.
%function's body starts here
total=x+y;
```

```
>> func7_1(3,5)
```

```
ans =
```

```
8
```

```
>> help func7_1
```

```
FUNC7_1 sum of two numbers or vectors.
```

```
FUNC7_1(X,Y) computes X+Y and returns the result.
```

```
X and Y can be scalars or vectors.
```


7.2.3 函數的引數與傳回值

- 從工作區接收的引數稱為輸入引數
- 輸出到工作區的引數稱為輸出引數，或稱為傳回值

```
function total = func7_1( x, y )
```

輸出引數（傳回值） 函數名稱 輸入引數

- 下表列出了幾種情況下，函數定義列的寫法：

表 7.2.2 函數定義列的幾種範例

函數定義列的格式	說 明
<code>function [x,y]=myfun(a)</code>	有一個輸入引數 a ，有兩個輸出引數 x 與 y
<code>function [x]=myfun(a)</code> <code>function x=myfun(a)</code>	有一個輸入引數 a ，有一個輸出引數 x
<code>function [x,y]=myfun()</code> <code>function [x,y]=myfun</code>	沒有輸入引數，但有兩個輸出引數 x 與 y
<code>function []=myfun(a)</code> <code>function myfun(a)</code>	沒有輸出引數，但有一個輸入引數 a

- 
-
- 有兩個傳回值的函數：

```
function [mn,mx]=func7_2(v)
mn=min(v);
mx=max(v);
```

```
>> [x,y]=func7_2([8 7 3 9 1])
```

```
x =
```

```
1
```

```
y =
```

```
9
```



○ 不需傳入引數的函數

```
function num=func7_3()  
num=length(primes(100));
```

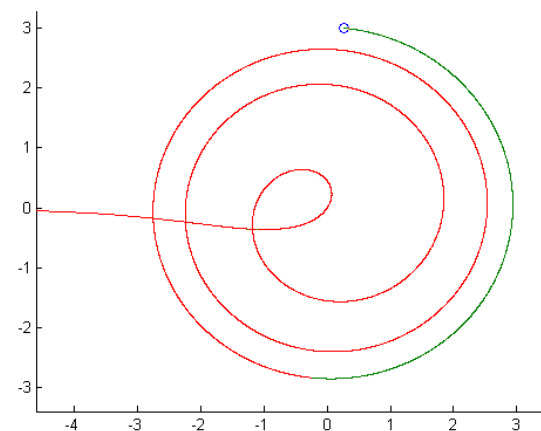
```
>> func7_3()  
ans =  
    25
```

```
>> func7_3  
ans =  
    25
```

○ 沒有傳回值的函數

```
function func7_4(n)
t=linspace(0.01,10*pi,n);
r=log(t);
comet(r.*cos(t),r.*sin(t));
```

```
>> func7_4(10000)
```



7.3 追蹤函數的執行與偵錯

7.3.1 在函數執行時列印訊息

- 要格式化的列印出某些訊息，可用 `fprintf` 函數：


表 7.3.1 格式化列印函數 `fprintf` 的語法

函數定義列的格式	說 明
<code>fprintf('str', e_1, e_2, \dots)</code>	依格式字串 <i>str</i> 所記載的格式碼，依序將運算式 e_1, e_2 填入 <i>str</i> 中列印出來。下面列出了格式字串裡常用的格式碼：
	<code>%c</code> ：列印字元
	<code>%s</code> ：列印字串
	<code>%md</code> ：以 m 個欄位的寬度列印整數
	<code>%m.nf</code> ：以 n 個小數位數，總共 m 個欄位的寬度列印數值
	<code>%m.ne</code> ：同上，但以指數型式來列印數值

○ fprintf函數所使用的特殊字元：

表 7.3.2 用於 fprintf 函數裡的特殊字元

特殊字元	說 明
\n	換行
\t	跳格
' '	印出單引號
\\	印出反斜線
%%	印出百分比符號



```
function func7_5(n)
if mod(n,2)==0
    fprintf('%d is even\n',n);
else
    fprintf('%d is odd\n',n);
end
```

```
>> func7_5(14)
14 is even
```

```
>> func7_5(63)
63 is odd
```


7.3.2 Matlab的M檔案偵錯環境

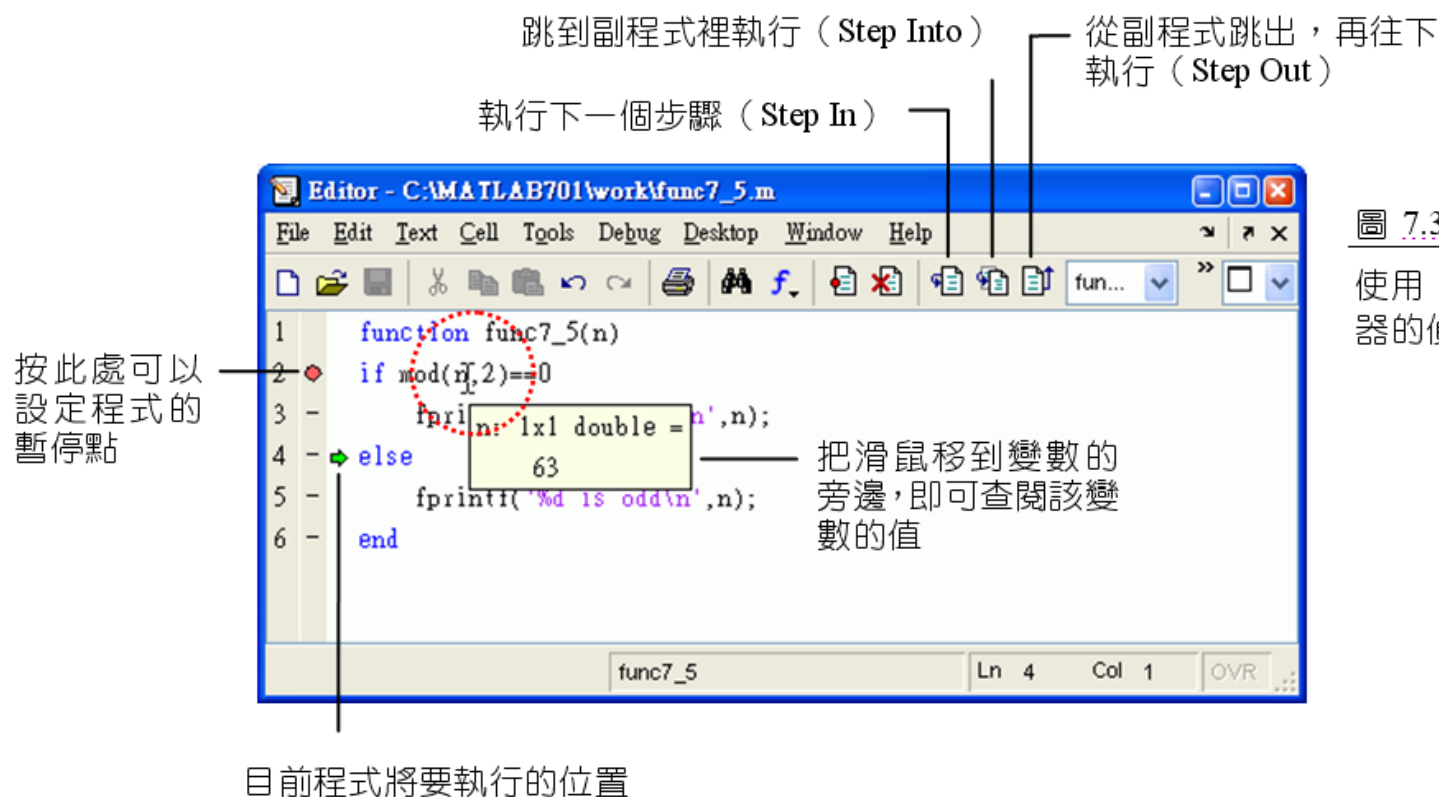


圖 7.3.1

使用 M 檔案編輯器的偵錯功能

7.4 函數的進階認識

7.4.1 指令類型的函數

- 如果函數`my_func(a, b)` 沒有任何的輸出引數：


1. 可用一般呼叫函數的方式來呼叫：

```
my_func(a,b);
```

2. 也可以採用類似指令的方式來呼叫它：

```
my_func a b ;
```

- 沒有輸出引數的函數也稱為 指令類型的函數



```
function func7_6(str)
fprintf('You input '%s''.\n',str)
```

```
>> func7_6('sss')
You input 'sss'.
```

```
>> func7_6 'sss'
You input 'sss'.
```

```
>> func7_6 sss
You input 'sss'.
```

7.4.2 函數輸入引數與輸出的個數


- 函數的輸入與輸出引數的個數可以不同：

<code>plot(y)</code>	% 只有一個輸入引數
<code>plot(x,y)</code>	% 有兩個輸入引數
<code>plot(x1,y1,x2,y2)</code>	% 有四個輸入引數
<code>zz=peaks;</code>	% 不需輸入引數
<code>[xx,yy,zz]=peaks(n)</code>	% 有一個輸入引數

-
- `nargin`與`nargout`二個變數，可查詢有幾個引數傳進來與傳出去：

表 7.4.1 `nargin` 與 `nargout` 變數

變數名稱	說 明
<code>nargin</code>	函數裡輸入引數的個數
<code>nargout</code>	函數裡傳回值的個數



```
function [x1,x2,x3]=func7_7(a1,a2)
fprintf('nargin = %d, ',nargin)
fprintf('nargout= %d\n',nargout)
x1=a1+a2;
x2=a1-a2;
x3=(a1+a2)/2;
```

```
>> [x,y,z]=func7_7(6,12)
nargin = 2, nargout= 3
x =
    18
y =
    -6
z =
     9
```


```
>> total=func7_7(6,12)
nargin = 2, nargout= 1
total =
    18
```

7.4.3 函數內變數的等級

- 在使用全域變數之前，必須利用global關鍵字宣告：

表 7.4.2 使用全域變數

語 法	說 明
<code>global var₁ var₂ ...</code>	宣告全域變數 var_1, var_2, \dots
<code>whos global</code>	查詢工作區內的全域變數
<code>clear global var₁ var₂ ...</code>	刪除全域變數 var_1, var_2, \dots



```
function func7_10(num)
global VAR;
VAR=VAR+num;
fprintf('在函數內，VAR=%g\n',VAR);
```

```
>> global VAR; VAR=10;
```

```
>> func7_10(5)
```

```
在函數內，VAR=15
```


```
>> VAR
```

```
VAR =
```

```
15
```


7.4.4 子函數與私有化目錄

- 同一個M檔案裡可以撰寫多個函數
- 一個M檔案只能有一個主函數，但可以有多個子函數
- 主函數可以呼叫子函數，子函數之間也可以相互呼叫
- 撰寫在M檔案的子函數，只能被同一個檔案內的函數呼叫。



```
function func7_11(v) % 主函數func7_11
subf(v);
fprintf('End of main function\n');
function subf(n) % 子函數subf
fprintf('sum(n)=%g\n',sum(n))
fprintf('prod(n)=%g\n',prod(n))
```

```
>> func7_11([1 2 3 4 5])
sum(n)=15
prod(n)=120
End of main function
```

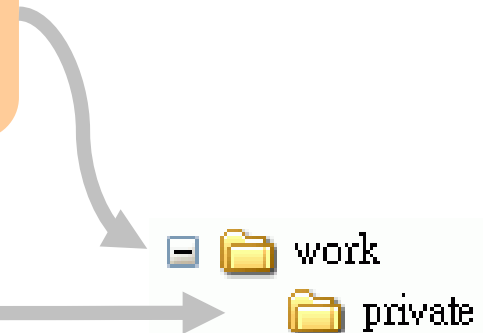
```
>> subf([1 2 3 4 5])
??? Undefined command/function 'subf'.
```


○ 私有化目錄：

可讓上層函數呼叫存放在`private`子資料夾裡的子函數

```
function func7_12(v) %主函數 func7_12
    subf(v);
    fprintf('End of func7_12\n');
```

```
function subf(n) % 子函數 subf
    fprintf('sum(n)=%g\n',sum(n))
    fprintf('prod(n)=%g\n',prod(n))
```



- 
-
- 在一個M檔案裡呼叫其它的函數時，Matlab呼叫的次序依序為
 1. 同一個M檔案內的子函數
 2. 若子函數不存在，則呼叫私有化目錄內的子函數
 3. 若私有化目錄內的子函數也不存在，則依搜尋路徑來找尋

7.4.5 保護程式碼—pcode

- pcode可保護程式碼，不讓外人查看。
- M檔案轉換成pcode之後，結果會是亂碼，但是還是可以執行，其執行方式與M檔案完全相同。

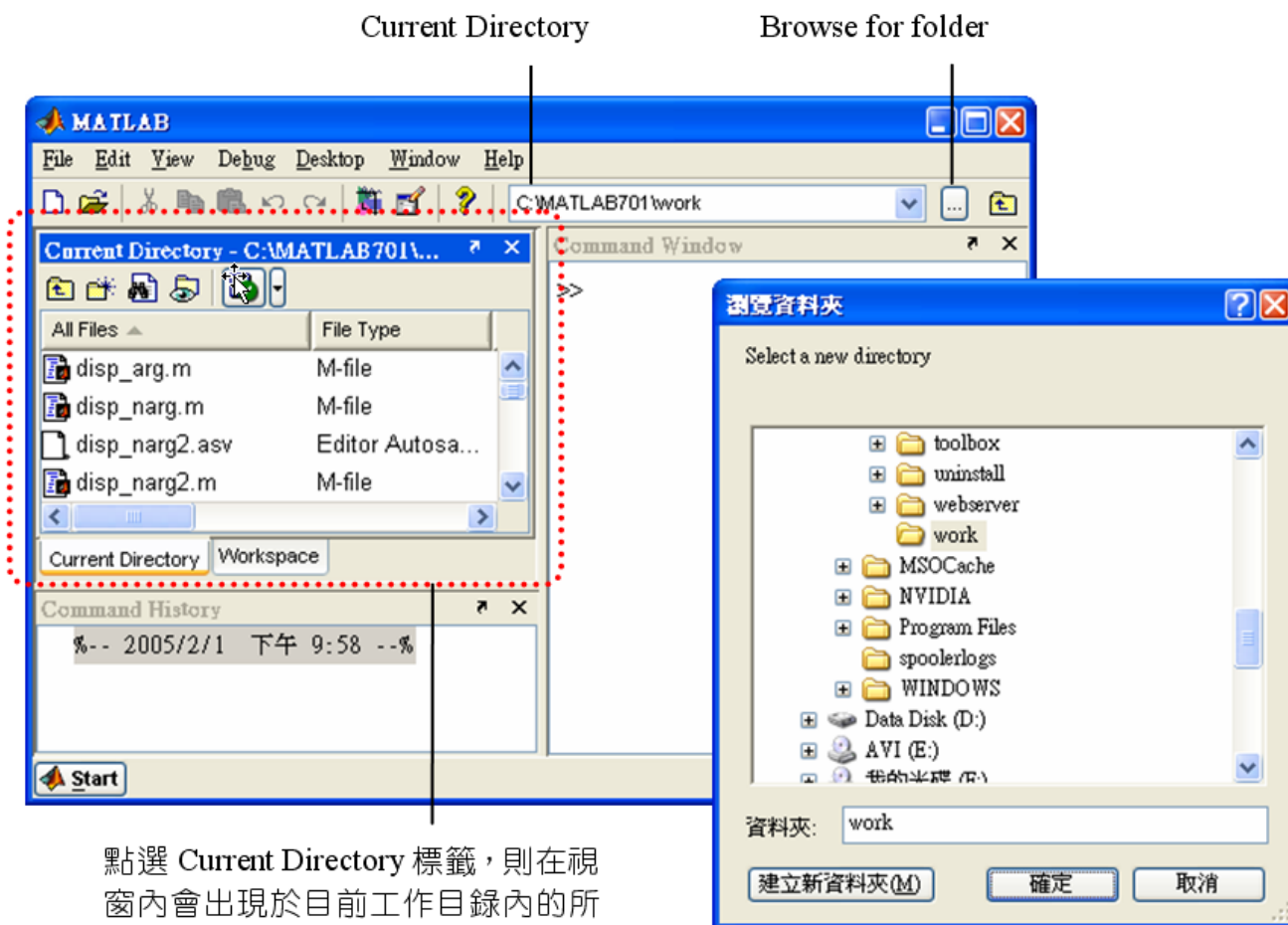
表 7.4.3 使用 pcode 函數

語 法	說 明
<code>pcode file_name.m</code>	將 M 檔案轉換成 pcode

```
>> pcode func7_4.m    % 將func7_4.m 轉換成pcode
```

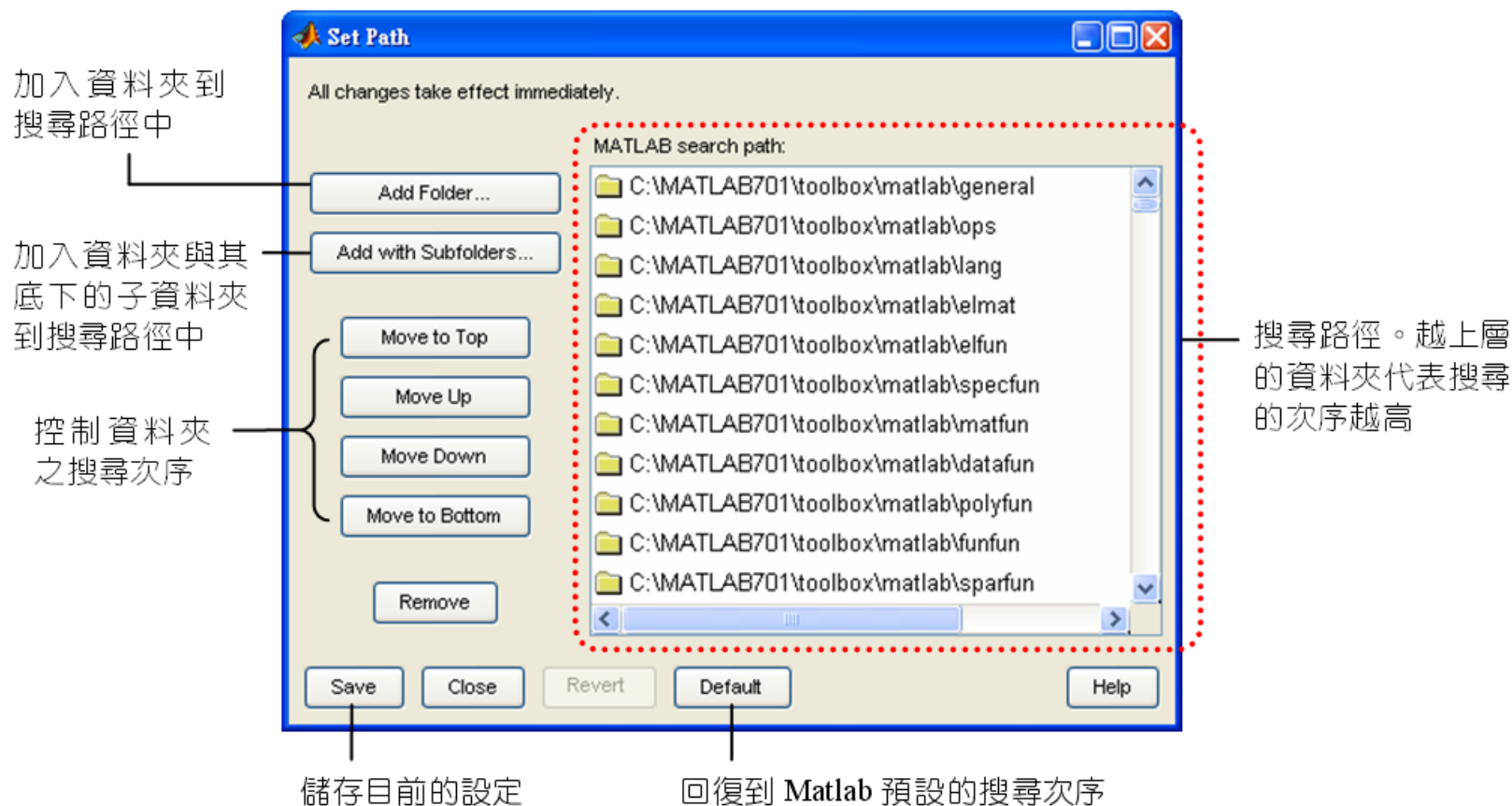
7.5 路徑的設定

7.5.1 設定目前工作目錄



7.5.2 設定Matlab搜尋的路徑

○ Set Path對話方塊：



7.6 匿名函數

- 匿名函數（anonymous functions）可以在Matlab的指令視窗裡直接定義一個函數，而不用把函數寫在M檔案裡：

表 7.6.1 匿名函數的定義

指 令	說 明
<code>fname=@(arg_list) expr</code>	定義匿名函數，函數名稱為 <i>fname</i> ，輸入引數為 <i>arg_list</i> ，函數的內容則定義在 <i>expr</i> 的位置


```
>> f=@(x) sin(2*x).*exp(-x/2)
```

```
f =
```

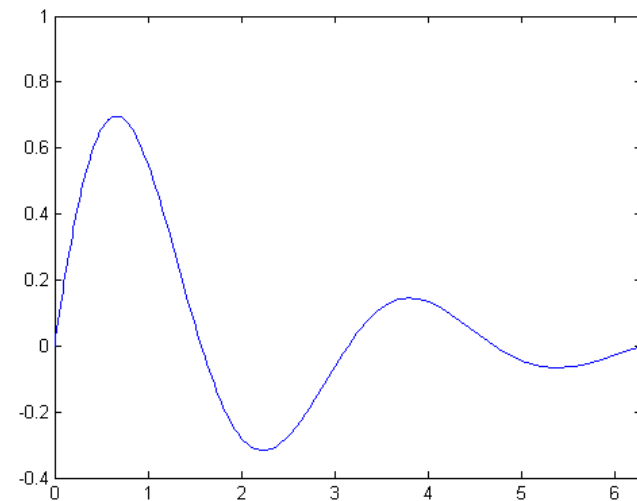
```
@(x) sin(2*x).*exp(-x/2)
```

```
>> fplot(f,[0,2*pi])
```

```
>> f(2.3)
```

```
ans =
```

```
0.0748
```





-The End-