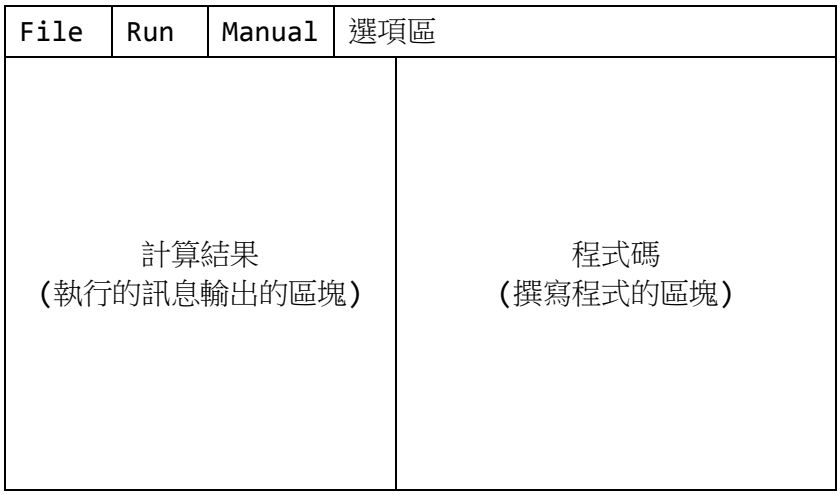
MiniMathLab Manual

廖建棋、劉浚、黃教荃

使用者介面分為三大部分：



選項區：

- 檔案：

  開啟：

  程式碼, 測試資料檔(讀取測試格式), 程式碼範例

  儲存：

  儲存程式碼, 儲存計算結果

- 執行程式(Ctrl + R)
- 開啟此說明手冊

程式語法：

註解：

以#開頭，後面為註解內容。

宣告變數：

格式：

變數名 = 值(, 變數名 = 值…)

or

變數名 = 變數名 =… = 變數名 =值(, 變數名 = 變數名 =…)

純量：

e.g.

```
a = -2.67 # a等於 -2.67
b = g = 2p # b等於g等於2π
c = 3e, d = 4! # c等於3e(自然對數底數)， d等於4階層
```

角度：

純量後以 a 結尾，代表是角度型態。

e.g.

```
a = 90a # a是 90°的角度
```

向量：

以”[“為開頭 ，以“,”做元素(純量)的區隔，最後以”]“結束。

e.g.

```
s = 2
vec = [1, s, 3] # vec是三維向量，裝了1, s純量, 3
```

矩陣：

和向量宣告相同，但每列以”|”做區隔，且元素可以為向量。行數由第一列的元素數量決定。

e.g.

```
vec = [1, 2, 3]
mat = [vec, 4 | 2, 4, 6, 8] # mat = 
```

$$\begin{bmatrix} 1 & 2 & 3 & 4 \\ 2 & 4 & 6 & 8 \end{bmatrix}$$

集合：

以”{“為開頭 ，以“,”做元素的區隔，最後以”}“結束。集合可以包含集合，但集合中所有元素必須是同型別(裝同一種東西)。

e.g.

```
vec = [6, 7]
set = {[1, 2, 3], [4, 5], vec} # 裝了三個向量(維度可不相同)的集合
```

變數指派：

語法同上，但值的型態需和變數一開始的型態一致，而維度大小可以不用一致。也可以由右而左連續指派。

e.g.

```
vec = [1, 2, 3]
```

```
vec3 = vec2 = vec # 用 vec 初始化 vec2, vec3

vec = vec2 * 2 # vec 等於兩倍的 vec2
```

多行指令：

直譯器是以每行為單位進行解譯，但如果一行的程式很長想寫成多行來表示呢? 例如在宣告矩陣時，往往想要以多列矩形的形式表達。這時，可以在每一行的的最後加上反斜線”\”，這告訴解譯器這行程式尚未完成，還要再讀下一行。

e.g.

```
mat = [1, 2, 3\
      |4, 5, 6\ # mat = [1, 2, 3| 4, 5, 6| 7, 8, 9]
      |7, 8, 9]
```

Expression：

Expression 形式如同一般程式語言一樣，可以有複雜的運算式和函數的呼叫。要注意的是基本運算子只支援+, -, *, /。但增加了向量的外積運算子” x ”(英文字母 x)，優先序較+, -高。使用時，需要在外積運算子左右兩側加入空白才能判別是外積運算，否則會當成是變數名的一部分。而向量的內積由”*”表示。另外，Expression 不能單獨存在，單獨存在是沒有意義的，需搭配指派或其他指令。如同 Python，指派不能在 Expression 當中發生。

e.g.

```
a = cos(angle([1, 0], [0, 1]) * 2) / 10 * [1, 2| 3, 4]

# a = [−0.1  −0.2
#      −0.3  −0.4]

b = asin(mag([1, 0, 0] x [0, 0, 1])) # b = 90°

c = 1 + 3 / 2 - 4 # c = -1.5
```

指令：

格式：

指令名 arg0(, arg1, arg2……)

指令名稱後至少接一個空格，可以接受一至多個參數。

print：

最常使用的指令，將 Expression 的結果輸出。

e.g.

```
print 1 + 3 / 2 – 4, asin(mag([1, 0, 0] x [0, 0, 1]))
```

輸出：

```
1 + 3 / 2 - 4 =
Scalar
  -1.5000000

asin(mag([1, 0, 0] x [0, 0, 1])) =
Angle
  90.0000000° (0.5000000π)
```

option：

可以調整浮點數的輸出格式，有三種選項可以調整：

w：顯示浮點數最少總寬度，若大於浮點數寬度，會在浮點數左側補上空白。

f：以正常浮點數格式表示，小數要顯示到小數點後第幾位。

e：以科學記號表示，小數要顯示到小數點後第幾位。

格式字母接著一個空白後給予值。值必須為正整數。f 或 e 值必須小於當前的w值才會生效。f和e兩種格式不能並存。設定一次後，之後的輸出皆會照此格式輸出，下一次執行也不必再設定一次。

e.g.

```
option w 10, f 2
```

輸出：

```
Set numeral width to 10
Set to fixed-point format, floating precision is 2
```

stg：

顯示目前有哪些變數，可依變數型態過濾顯示：

all：所有變數

scls：所有純量變數

agls：所有角度變數

vecs：所有向量變數

mats：所有矩陣變數

sets：所有集合變數

如果變數是經由測資格式所產生出來，那它就會一直存在在記憶體裡，除非使用 delete 指令，才能將變數刪除。這意味著，當程式一開始想要使用其變數名時，會因此而不能使用或者覆蓋其原本讀進來的值。

e.g.

```
s = 12

v1 = [1, 0], v2 = [1, 2, 3]

stg all, vecs
```

輸出：

```
all variables in the storage:
v2 = Vector dimension = 3
v1 = Vector dimension = 2
s = Scalar

vector variables in the storage:
v2 = Vector dimension = 3
v1 = Vector dimension = 2
```

delete：

刪除變數，使得後續程式可以再使用其變數名宣告變數。

e.g.

```
a = 12, b = 10

delete a, b #刪除 a, b

a = [1, 2]
```

集合常數：

所有都變數都會被分在各類型的集合裡，而這些集合稱為集合常數。分別為 scls, agls, vecs, mats, sets。這些變數就各自代表各類型的集合，所以不能當變數名使用，也不能被刪除。可以使用這些變數搭配具迭代功能的函數完成過濾等功能。

e.g.

```
s1 = 1, s2 = 2

print scls
```

輸出：

```
scls =
Set of scalars size = 2
{
   Scalar
      1.0000000

   Scalar
      2.0000000
}
```

函數表：

代號	型態
A	角度
S	純量
V	向量
M	矩陣
<T>S	裝 T 型態的集合
N	不回傳，產生額外訊息

類型	對應題號	回傳型	函數名稱	說明
純量	-	S	pow(S, S)	純量指數
	-	S	sqrt(S)	開平方
角度	-	A	angle(S)	將純量轉成角度(Degree)
	-	S	sin(A)	sin 函數
	-	S	cos(A)	cos 函數
	-	S	tan(A)	tan 函數
	-	A	asin(S)	arc-sin 函數
	-	A	acos(S)	arc-cos 函數
	-	A	atan(S)	arc-tan 函數
向量	-	V	to_vec(M)	把 1xN 或 Nx1 的矩陣轉成向量
	5	S	mag(V)	向量長
	6	V	normlzd(V)	單位向量
	8	S	compnt(V, V)	投影長
	9	V	proj(V, V)	投影
	10	S	tri_area(V, V)	兩向量圍成三角形面積
	11	N	is_pallel(V, V)	兩向量是否平行
	12	N	is_orth(V, V)	兩向量是否垂直
	13	A	angle(V, V)	兩向量夾角
	14	V	plane_norm(V, V)	兩向量組成的面上的法向量
	15	N	is_linear_ind(VS)	判斷向量集合裡的向量是否線性獨立
	16	VS	gs_orth_process(VS)	對基底向量集合做 Gram-Schmidt Orthogonalization Process
矩陣	-	M	to_mat_v(VS)	將向量集合直擺成矩陣
	-	M	to_mat_h(VS)	將向量集合橫擺成矩陣
	3	S	rank(M)	矩陣的秩
	4	M	transpose(M)	矩陣轉置
	-	M	pow(M, S)	矩陣指數，指數須為整數
	-	M	row_ech(M)	矩陣的列梯形形式
	-	M	solve_linear(M, V)	解線性系統，常數組為向量
	5	M	solve_linear(M, M)	解線性系統，常數組為矩陣

	6	S	det(M)	矩陣的行列式
	7	M	inverse(M)	反矩陣
	8	M	adj(M)	求伴矩陣
	9	MS	eigen(M)	求特徵向量和值，只限 3x3 以下的矩陣
	10	VS	power_eigen(M)	用 Power Method 求其中一組特徵向量和值
	11	M	least_square(M, M)	求近似函數
	12	MS	ul_decom(M)	上下三角矩陣分解，回傳包含這兩矩陣的集合
由索引取得，從 0 開始	-	S	get(V, S)	從向量中取得對某維的分量(0 是第一維)
	-	S	get(M, S, S)	從矩陣中取得某個元素
	-	T	get(<T>S, S)	從集合中取得某個元素
	-	<T>S	subset(<T>S, S, S)	從集合中取得指定範圍的子集合