**CH14**

**MATLAB運算元可分為五大類:**

1. 數學運算元

2. 關係運算元

3. 邏輯運算元

4. 位元運算元

5. 集合運算元

**數學運算元**

**數學運算元之優先順序:**

1. 轉置（.’）,冪次（.^）.共軛轉置（’）,矩陣冪次（^）

2.正號（+）,負號（-）

3.乘法（.\*）,元素右除（./）,元素左除（.\）,矩陣 乘法（\*）,矩陣右除（/）,

矩陣左除（\）

4.加法（+）,減法（-）

5.冒號（:）（例如： x = 1:2:5 等）

**%同一類的運算元均具有相同的優先度（Priority），因此在計算上，是由左至右依次完成** **%**

% Example1 結果

a = [1 2; 3 4]; %建立a矩陣 b =

b = a+1 %將a矩陣+1存入b 2 3

c = 1./a %元素右除存入c 4 5

c =

1.0000 0.5000

0.3333 0.2500

關係運算元

關係運算元 說明

== 等於

~= 不等於

< 小於

<= 小於或等於

> 大於

>= 大於或等於

重點:

MATLAB 的關係運算元總是以矩陣內元素對元素的方式作運算

c=a>b

若 a(i, j) > b(i, j)，則 c(i, j) 的值為 1(true),否則即為 0(false)

MATLAB 的關係運算元亦支援純量展開

% Example2 結果

a = [1 2; 3 4]; c=

c = a>2 0 0

1 1

% Example3 %求a矩陣中，滿足 a(i, j) >= b(i, j) 的元素

a = [10 20; 30 40]; 結果

b = [10 30; 20 40]; d=

c = a>=b; 10

d = a(c) % or d = a(a>=b) 30

40

find指令:可傳回非零元素索引,顯示一維索引或下標

% Example4 結果

x = [0 1 2; 0 0 3]; index =

index = find(x) 3

5 6

Tips: matlab中儲存矩陣是按照列儲存

x = [0(1) 1(3) 2(5);

0(2) 0(4) 3(6)]; %範例順序

% Example5 結果

x = magic(3); idx1 = idx2 =

[idx1, idx2] = find(x>5) 1 1

補充 x = magic(3) 3 2

x = 1 3

8 1 6 2 3

3 5 7 %idx1(列索引) idx2(行索引)

4 9 2

% Example6 %找出介於7和11的元素

x = magic(5); 結果

x(find(7<x & x<11)) ans=

10

8 9

邏輯運算元

邏輯運算元 說明

& AND

| OR

~ NOT

數值1為Ture,0為False

all為所有向量元素為「真」（非零），則 all 指令傳回 1，否則為 0。

any為任一向量元素為「真」（非零），則 any 指令傳回 1，否則為 0。

% Example7 結果

a = [0 1 2 3]; result1 =

result1 = all(a) 0

retult2 = any(a) result2 =

result3 = any(a<0) 1 result3 =

tips: all 及 any 指令可以接受矩陣輸入 0

此時他們會對每一個行向量進行運算

位元運算元

位元運算元 說明

bitand 位元對位元的 AND

bitcmp 每個位元的 NOT

bitor 位元對位元的 OR

bitmax 最大浮點整數值

bitxor 位元對位元的 XOR

bitset 設定某位元

bitget 取得某位元

bitshift 位元平移

% Example8 結果

bitor(10,4) %位元對位元的 OR ans =

14

tips : 10的二進制為1010

4 的二進制為0100

bitor會傳回1110的十進制為14

% Example9 結果

bitshift(18,1) %位元向左平移一個單位 ans =

36

tips : 18的二進制為10010

左平移一個單位為100100 %左平移一個單位相當於十進制\*2

bitshift會傳回100100的十進制為36

集合運算元

集合運算元 說明

union 聯集

intersect 交集

setdiff 差集

setxor 集合的xor

unique 傳回最小等效集合（每個元素只出現一次）

ismember 測試元素是否屬於某集合

% Example10

x = [1 2 3 4 5 6];

y = [1 1 5 5 9 9 9];

union\_result = union(x, y) % 聯集

intersect\_result = intersect(x, y) % 交集

setdiff\_result = setdiff(x, y) % 差集

setxor\_result = setxor(x, y) % XOR 運算

setuniq\_result = unique(y) % 取相異元素

ismember\_result = ismember(9, y) % 元素 9 是否屬於集合 y

結果

union\_result =

1 2 3 4 5 6 9

intersect\_result =

1 5

setdiff\_result =

2 3 4 6

setxor\_result =

2 3 4 6 9

setuniq\_result =

1 5 9

ismember\_result =

1