Ch.03 向量的建立與運算

矩陣是一組型態相同的純量排列成的一維資料結構。

IDL建立向量內容為零的函數(n為向量的長度)

函數(n為整數)	功能
BYTARR(n)	建立內容為零的短整數向量
FIXARR(n)	建立內容為零的整數向量
UINTARR(n)	建立內容為零的無號整數向量
LONGARR(n)	建立內容為零的長整數向量
ULONGARR(n)	建立內容為零的無號長整數向量
LONG64ARR(n)	建立內容為零的64位元長整數向量
ULONG64ARR(n)	建立內容為零的64位元無號長整數向量
FLTARR(n)	建立內容為零的浮點數向量
DOUBLEARR(n)	建立內容為零的雙精度浮點數向量
COMPLEXARR(n)	建立內容為零的複數向量
DCOMPLEXARR(n)	建立內容為零的雙精度複數向量
STRARR(n)	建立內容為零的字元向量

```
x0=bytarr(3)
help,x0 & print,x0
x=intarr(5)
help,x & print,x
y=fltarr(4) & y2=fltarr(4)
help,y,y2 & print,y,y2
y3=complex(y,y2)
help,y3 & print,y3
z=strarr(3)
help,z & print,z
end
```

```
% Compiled module: $MAIN$.
X0              BYTE      = Array[3]
  0   0   0
X               INT       = Array[5]
      0       0       0       0       0
Y               FLOAT     = Array[4]
Y2              FLOAT     = Array[4]
    0.000000    0.000000    0.000000    0.000000
    0.000000    0.000000    0.000000    0.000000
Y3              COMPLEX   = Array[4]
(    0.000000,    0.000000)(    0.000000,
0.000000)
(    0.000000,    0.000000)(    0.000000,
0.000000)
Z               STRING    = Array[3]

IDL>
```

建立向量內容均為特定值的函數(n為向量長度)

函數	功能
REPLICATE(value, n)	建立內容為value的向量，向量長度為n，value為任意資料型態
MAKE_ARRAY(n, VALUE=value)	建立內容為value的向量，向量長度為n，value為任意資料型態

```
x0=replicate(1,5)
help,x0 & print,x0
x1=make_array(4,VALUE=3)
help,x1 & print, x1
x2=replicate('ABC',3.1)
help,x2 & print,x2
x3=make_array(2,VALUE=0)
help,x3 & print,x3
Print,x3(0),x3(1)
end
```

```
% Compiled module: $MAIN$.
X0              INT       = Array[5]
     1       1       1       1       1
X1              FLOAT     = Array[4]
    3.10000     3.10000     3.10000
3.10000
X2              STRING    = Array[3]
ABC ABC ABC
X3              INT       = Array[2]
     0       0
IDL>
```

產生亂數向量的函數

函數	功能
Result = RANDOMU(seed, n)	產生亂數向量Result，包含n個均勻分布的元素。Seed是種子參數，可以不給定特定值，元素數值落在0和1間。
Result = RANDOMN(seed, n)	產生亂數向量Result，包含n個均勻分布的元素。Seed是種子參數，可以不給定特定值，元素數值呈現常態分布的浮點數。

```
x0=randomu(seed,5)
help,x0 & print,x0
x1=randomn(seed,4)
help,x1 & print,x1
end
```

```
% Compiled module: $MAIN$.
X0              FLOAT     = Array[5]
   0.837582    0.220759    0.996492   0.0850411    0.222375
X1              FLOAT     = Array[4]
  -0.483864    0.275686   -0.0846034    0.671800
IDL>
```

建立向量內容為下標的函數(n為向量的長度)

函數	功能
BINDGEN(n)	建立內容為下標的短整數向量
INDGEN(n)	建立內容為下標的整數向量
UINDGEN(n)	建立內容為下標的無號整數向量
LINDGEN(n)	建立內容為下標的長整數向量
ULINDGEN(n)	建立內容為下標的無號長整數向量
L64INDGEN(n)	建立內容為下標的64位元長整數向量
UL64INDGEN(n)	建立內容為下標的64位元無號長整數向量
FINDGEN(n)	建立內容為下標的浮點數向量
DINDGEN(n)	建立內容為下標的雙精度浮點數向量
CINDGEN(n)	建立內容為下標的複數向量
DCINDGEN(n)	建立內容為下標的雙精度複數向量
SINDGEN(n)	建立內容為下標的字元向量

向量的第1個下標是0, 第2個是1, … 第n個是n-1。

```
x0=bindgen(4)
help,x0 & print,x0
x1=indgen(4)
help,x1 & print,x1 & print,'x1[1]=',x1[1]
print,'x1[2-4]=',x1[1:*]
x2=findgen(4)
help,x2 & print,x2
x3=dcindgen(3)
help,x3 & print,x3
x4=sindgen(3)
help,x4 & print,x4
end
```

```
% Compiled module: $MAIN$.
X0              BYTE      = Array[4]
  0   1   2   3
X1              INT       = Array[4]
      0       1       2       3
x1[1]=       1
x1[2-4]=       1       2       3
X2              FLOAT     = Array[4]
    0.000000     1.00000     2.00000     3.00000
X3              DCOMPLEX  = Array[3]
(     0.00000000,     0.00000000)(      1.0000000,
0.00000000)
(      2.0000000,     0.00000000)
X4              STRING    = Array[3]
         0         1         2
IDL>
```

建立內容為不規則數值或字串的向量

函數	功能
Var = [a1,a2,a3,…,an]	建立內容為不規則數值或字串的向量，a1,a2,..,an為不規則數值或字串，n為向量的元素個數

```
x0=[5,3,8,4,9]
help,x0 & print,x0
x1=[5.1,3.8,9.3,4.2,8.4]
help,x1 & print,x1
print,'x1[1-3]=',x1[1:3]
end
```

```
% Compiled module: $MAIN$.
X0              INT       = Array[5]
     5       3       8       4       9
X1              FLOAT     = Array[5]
    5.10000     3.80000     9.30000
4.20000     8.40000
x1[1-3]=     3.80000     9.30000
4.20000
IDL>
```

轉換向量資料型態的函數(A為向量)

函數	功能
BYTE(A)	轉換A中所有元素為短整數
FIX(A)	轉換A中所有元素為整數
UINT(A)	轉換A中所有元素為無號整數
LONG(A)	轉換A中所有元素為長整數
ULONG(A)	轉換A中所有元素為無號長整數
LONG64(A)	轉換A中所有元素為64位元長整數
ULONG64(A)	轉換A中所有元素為64位元無號長整數
FLOAT(A)	轉換A中所有元素為浮點數
DOUBLE(A)	轉換A中所有元素為雙精度浮點數
COMPLEX(A)	轉換A中所有元素為複數的實部
DCOMPLEX(A)	轉換A中所有元素為雙精度複數的實部
STRING(A)	轉換A中所有元素為字元

```
x0=findgen(4)
Y0=fix(x0)
help,x0,y0 & print,x0,y0
x1=[5.1,3.2,6.7,2.8]
y1=complex(x1)
help,x1,y1 & print,x1,y1
end
```

```
% Compiled module: $MAIN$.
X0              FLOAT     = Array[4]
Y0              INT       = Array[4]
   0.000000     1.00000     2.00000
3.00000
     0      1      2      3
X1              FLOAT     = Array[4]
Y1              COMPLEX   = Array[4]
    5.10000     3.20000     6.70000
2.80000
(     5.10000,    0.000000)(     3.20000,
0.000000)
(     6.70000,    0.000000)(     2.80000,
0.000000)
IDL>
```

查詢向量相關資訊的函數

函數	功能
N_ELEMENTS(A)	求出向量A中元素的個數
FINITE(A)	判斷向量A中各個元素是否為有限

```
x0=[2,1.3,!values.f_nan,5.2,!values.f_infinity]
help,x0 & print,x0
print,finite(x0)
end
```

```
% Compiled module: $MAIN$.
X0              FLOAT     = Array[5]
     2.00000     1.30000         NaN
5.20000         Inf
  1   1   0   1   0
IDL>
```

向量下標操作符號

符號	說明
0	代表下標的開始
*	代表全部下標
:	宣告下標範圍或下標增加量, n1 : n2, 起始下標n1, 結束下標n2。 n1:n2:n3 起始下標n1，結束下標n2，下標增加量n3。

```
x0=[1.0,2.1,3.2,4.3,5.4]
print,x0
print,'x0[0]=',x0[0]
print,'x0[1:3]=',x0[1:3]
print,'x0[3:*]=',x0[3:*]
print,'x0[0:4:2]=',x0[0:4:2]
y2=fltarr(5)
print,'y2=',y2
y1=x0[1:3]
y2[2:4]=y1；資料依序存到下標為2,3,4
help,y1,y2 & print,y1,y2
y3=fltarr(5)
y3[1]=y1 ;從下標為1處開始存放資料
help,y3 & print,y3
end
```

```
% Compiled module: $MAIN$.
    1.00000     2.10000     3.20000     4.30000
5.40000
x0[0]=     1.00000
x0[1:3]=     2.10000     3.20000     4.30000
x0[3:*]=     4.30000     5.40000
x0[0:4:2]=     1.00000     3.20000     5.40000
y2=     0.000000     0.000000     0.000000
0.000000     0.000000
Y1              FLOAT     = Array[3]
Y2              FLOAT     = Array[5]
    2.10000     3.20000     4.30000
   0.000000     0.000000     2.10000     3.20000
4.30000
Y3              FLOAT     = Array[5]
   0.000000     2.10000     3.20000     4.30000
0.000000
IDL>
```

回傳下標的數學函數(A為向量)

函數	功能
Subscript = SORT(A)	排列A內的數值由小到大順序，Subscript記錄著數值的下標順序
Subscript = UNIQ(A)	去除A內重複的數值，Subscript記錄著數值的下標順序。 執行UNIQ函數時，須先把向量內的數值先以SORT函數由小到大排序。

```
a=[7,9,8,15,8]
b=sort(a)
print,a,b
print,a[b[0]],a[b[3]]
c=a[b] ;排序好的數值存到變數C
print,'c=',c
b1=uniq(c)
print,b1
print,'c[b1]=',c[b1]
end
```

```
% Compiled module: $MAIN$.
      7       9       8      15       8
      0       2       4       1       3
      7       9
c=       7       8       8       9      15
      0       2       3       4
c[b1]=       7       8       9      15
IDL>
```

向量與函數所使用的區隔符號

符號	說明
() 小括號	函數用小括號隔參數
[] 中括號	變數用中括號區隔下標

IDL的優先將名稱當作函數來處理，最好的方式是變數的名稱避免使用到函數相同的名稱。

```
fix=[4,5,6,7,8]
print,'fix(4)=',fix(4)
print,'fix[4]=',fix[4]
end
```

```
% Compiled module: $MAIN$.
fix(4)=      4
fix[4]=      8
IDL>
```

向量變換的函數

函數	功能
[A, B]	將向量A和B橫向併排，亦即擴充行
SHIFT(A,c)	平移向量A中元素的順序，c代表平移量，可正負值
REVERSE(A)	倒轉向量A中的元素順序

```
a=[1,2] & b=[3,4]
c=[a,b]
help,a,b,c & print,a,b,c
d=shift(c,1)
print,d,shift(c,-1)
e=reverse(c)
print,'e=',e
end
```

```
% Compiled module: $MAIN$.
A               INT       = Array[2]
B               INT       = Array[2]
C               INT       = Array[4]
      1      2
      3      4
      1      2      3      4
      4      1      2      3
      2      3      4      1
e=      4      3      2      1
IDL>
```

向量的數學運算(向量間的元素個數要相同)

指令	說明
A – B	將向量A中的元素減去向量B中相同位置的元素
A + B	將向量A中的元素加上向量B中相同位置的元素
A * B	將向量A中的元素乘以向量B中相同位置的元素
A + b	將向量A中的每一個元素加純量b
A / b	將向量A中的每一個元素除以純量b
A ^ b	將向量A中的每一個元素取純量b次方
A MOD b	將向量A中的每一個元素除以純量b後的餘數
SIN(A)、COS(A)、TAN(A)	將向量A中的每一個元素聯三角函數
EXP(A)	將向量A中的每一個元素取自然指數
ALOG(A)、ALOG10(A)	將向量A中的每一個元素取自然對數值、基底為10的對數值
ABS(A)	將向量A中的每一個元素取絕對值
SQRT(A)	將向量A中的每一個元素開根號

```
a=[4,5,6] & b=[3,2,1]
c1=b-a & c2=a+b
print,'c1=',c1
print,'c2=',c2
print,'a*b=',a*b
print,'a+4=',a+4
print,'a/5=',a/5
print,'a*3=',a*3
print,'a^2=',a^2
print,'a mod 3 =',a mod 3
end
```

```
% Compiled module: $MAIN$.
c1=      -1      -3      -5
c2=       7       7       7
a*b=      12      10       6
a+4=       8       9      10
a/5=       0       1       1
a*3=      12      15      18
a^2=      16      25      36
a mod 3 =       1       2       0
```

```
a=[4,5,6] & b=[3,2,1]
c1=b-a
x=[!pi/6,!pi/3]
y1=sin(x)
print,x,' sin(x)=',y1
y=[30,60]
print,y,' cos(y)=',cos(y*!dtor)
z1=asin(y1)
print,z1,'asin(y1)=',z1*!radeg
y2=exp(a) & z2=alog(y2)
print,'exp(a)=',y2 & print,'alog(y2)=',z2
print,'alog10(b)=',alog10(b)
print,c1,' abs(c1)=',abs(c1)
print,a,' sqrt(a)=',sqrt(a)
end
```

```
 0.523599      1.04720
sin(x)=
    0.500000     0.866025
     30      60
cos(y)=
    0.866025     0.500000
    0.523599      1.04720
asin(y1)=
     30.0000      60.0000
exp(a)=      54.5981      148.413      403.429
alog(y2)=      4.00000      5.00000      6.00000
alog10(b)=     0.477121     0.301030
0.000000
     -1      -3      -5
 abs(c1)=
      1       3       5
      4       5       6
 sqrt(a)=
     2.00000      2.23607      2.44949
IDL>
```

- IDL系統採取向量式的平行運算，亦即程式會被轉變成一組執行緒，互相獨立的執行緒即可分配到中央處理單元(CPU)的各個核心，以多工同時的方式來進行運算。
- 寫IDL程式時，應儘量避免使用迴圈，迴圈內各個指令不再是獨立個體，後一個指令必須等到前一個指令執行完成，才能繼續執行，無法做平行運算。