

Matricial e Vetorial

O IDL é uma linguagem baseada em matrizes sejam elas unidimensionais (vetores) ou bidimensionais.

Criação

Tipo	Função de Criação	Índice que gera a Função
<i>Byte</i>	<i>BYTARR</i>	BINDGEN
<i>Integer</i>	<i>INTARR</i>	INDGEN
<i>Unsigned integer</i>	<i>UINTARR</i>	UINDGEN
<i>Long integer</i>	<i>LONARR</i>	LINDGEN
<i>Unsigned long integer</i>	<i>ULONARR</i>	ULINDGEN
<i>64-bit integer</i>	<i>LONG64ARR</i>	L64INDGEN
<i>Unsigned 64-bit integer</i>	<i>ULONG64ARR</i>	UL64INDGEN
<i>Float</i>	<i>FLTARR</i>	FINDGEN
<i>Double</i>	<i>DBLARR</i>	DINDGEN
<i>Complex</i>	<i>COMPLEXARR</i>	CINDGEN
<i>Double complex</i>	<i>DCOMPLEXARR</i>	DCINDGEN
<i>String</i>	<i>STRARR</i>	SINDGEN
<i>Pointer</i>	<i>PTRARR</i>	
<i>Object</i>	<i>OBJARR</i>	
<i>Undefined</i>		
<i>Structure</i>	<i>REPLICATE</i>	

Vetores e matrizes com diferentes tipos de variáveis e dados podem ser iniciados com rotinas do IDL como as listadas na tabela acima.

```
IDL> vec = fltarr (15)
IDL> vec[5] = 4.56
IDL> vec[13]= 1234.333
```

Matrizes e vetores no IDL começam com 0; conseqüentemente o vetor *vec* está subscripto de 0 até 14.

Vetores e matrizes também podem ser criados atribuindo à variável certo número de valores passados por colchetes.

```
IDL> temp = [12, 82, 97, 23, 0, 78]
```

Esta instrução cria um vetor de seis inteiros. Uma matriz de múltipla dimensão pode ser criada colocando valores dentro de vários colchetes dentro de dois colchetes, observe:

```
IDL> id4x4 = [ [1, 0, 0, 0], [0, 1, 0, 0], [0, 0, 1, 0], [0, 0, 0, 1] ]
IDL> help, id4x4
ID4x4  INT      = Array [4, 4]
```

A variável *id4x4* tornou-se uma matriz 4 X 4 de inteiros. Considere a instrução abaixo:

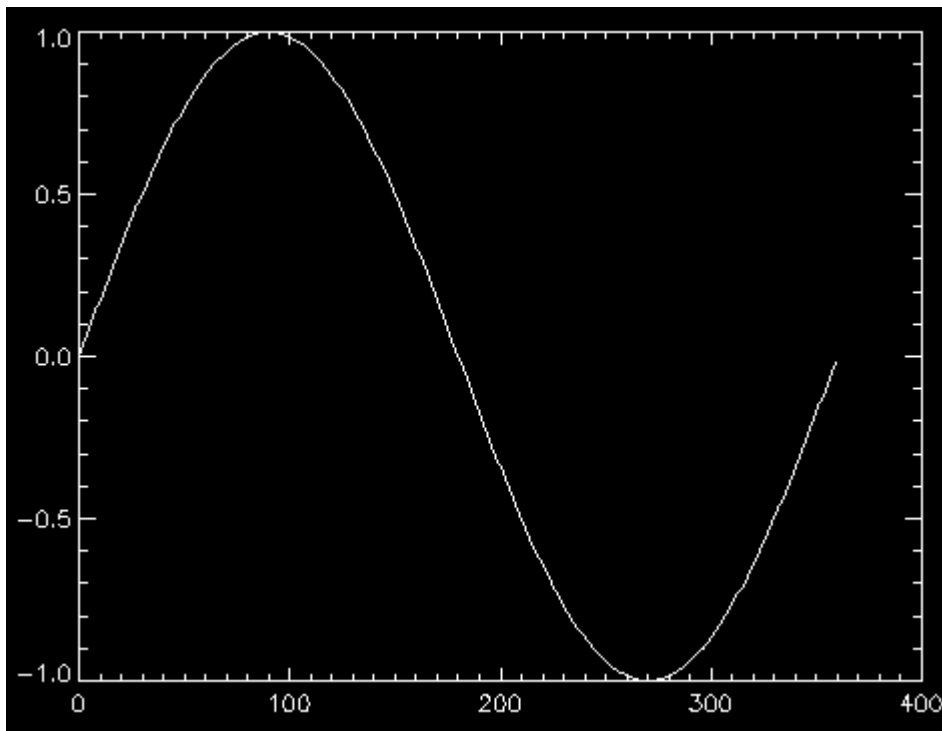
```
IDL> exemplo_arr = [3.4, 6.7D, 45U, 90L]
```

Esta instrução cria um vetor com diferentes tipos de variável. O IDL não apresenta problemas diante disso, ele executa uma conversão pelo tipo mais significativo. O tipo mais significativo nesta instrução é o *double* (6.7D). Todos os elementos do vetor são convertidos para *double*.

Operação de matrizes e vetores

Quando executamos uma operação com matrizes ou vetores no IDL, eles são executados em todos os elementos que estão dentro desta matriz ou vetor. Esta poderosa característica elimina a necessidade da utilização do laço para executar uma operação com cada um dos elementos.

```
IDL> x = findgen(360) * !dtr  
IDL> sincurve = sin(x)  
IDL> plot, sincurve
```



A primeira instrução multiplica os números 0.0, 1.0, 2.0, ...,359.0 pela variável de sistemas *!dtr* fazendo a transformação de grau para radiano constante, então a segunda instrução calcula o seno de todos os elementos resultados pela transformação de graus em radianos, e a terceira linha de comando como já vimos anteriormente nesse guia exibe, um gráfico em linha do calculo final da matriz. Isto beneficia o IDL, eliminando o tempo consumido para controlar os laços, que executam as operações em todos os elementos do vetor. Use a vantagem desta possibilidade o máximo que poder - é mais rápido e mais fácil de ler.

Matrizes Multidimensionais

As matrizes no IDL podem ter até oito dimensões. No caso da matriz bidimensional do IDL a subscrição é especificada normalmente como [coluna, linha].

```
IDL> multi = lindgen (4, 6)      ; quatro colunas, seis linhas
IDL> print, multi
```

```
  0      1      2      3
  4      5      6      7
  8      9     10     11
 12     13     14     15
 16     17     18     19
 20     21     22     23
```

O método de subscrição do IDL é por coluna. Internamente as matrizes são dispostas no formato de linha (os elementos do *multi* são numerados na ordem em que eles são armazenados na memória). A subscrição em uma matriz bidimensional pode ser feita utilizando a notação [coluna, linha].

```
IDL> print, multi [1, 0], multi [3, 5]
1      23
```

Todas as escalas de operadores podem ser utilizadas nas matrizes multidimensionais.

```
IDL> print, multi [* , 4]      ; a quinta linha
16     17     18     19
```

```
IDL> print, multi [2, *]      ; a terceira coluna
```

```
  2
  6
 10
 14
 18
 22
```

```
IDL> print, multi [2: 3, 1: 2] ; uma quadra – a 3ª e a 4ª coluna e a 2ª e 3ª linha
  6      7
 10     11
```

Multiplicação de Matrizes

O operador # calcula o produto de uma matriz, multiplicando as colunas da primeira matriz pelas linhas da segunda. A segunda matriz deve ter o mesmo número de colunas que a primeira tem de linhas. A matriz resultante terá o número de colunas da primeira matriz e o mesmo número de linhas da segunda matriz.

Por exemplo:

```
IDL> a = indgen(3,2)          ; matriz 3 X 2, na sintaxe do IDL.
IDL> print, a
  0      1      2
  3      4      5
IDL> b = indgen(2,3)          ; matriz 2 X 3.
IDL> print, b
```

```

      0  1
      2  3
      4  5
IDL> print, a # b
      3  4  5
      9 14 19
     15 24 33

```

Os cálculos são feitos desta forma:

$$\begin{array}{l}
 A_{0,0} \cdot B_{0,0} + A_{0,1} \cdot B_{1,0} \quad \left| \quad A_{1,0} \cdot B_{0,0} + A_{1,1} \cdot B_{1,0} \quad \left| \quad A_{2,0} \cdot B_{0,0} + A_{2,1} \cdot B_{1,0} \right. \\
 A_{0,0} \cdot B_{0,1} + A_{0,1} \cdot B_{1,1} \quad \left| \quad A_{1,0} \cdot B_{0,1} + A_{1,1} \cdot B_{1,1} \quad \left| \quad A_{2,0} \cdot B_{0,1} + A_{2,1} \cdot B_{1,1} \right. \\
 A_{0,0} \cdot B_{0,2} + A_{0,1} \cdot B_{1,2} \quad \left| \quad A_{1,0} \cdot B_{0,2} + A_{1,1} \cdot B_{1,2} \quad \left| \quad A_{2,0} \cdot B_{0,2} + A_{2,1} \cdot B_{1,2} \right.
 \end{array}$$

Usando os valores atuais:

$$\begin{array}{l}
 (0) \cdot (0) + (3) \cdot (1) \quad \left| \quad (1) \cdot (0) + (4) \cdot (1) \quad \left| \quad (2) \cdot (0) + (5) \cdot (1) \right. \\
 (0) \cdot (2) + (3) \cdot (3) \quad \left| \quad (1) \cdot (2) + (4) \cdot (3) \quad \left| \quad (2) \cdot (2) + (5) \cdot (3) \right. \\
 (0) \cdot (4) + (3) \cdot (5) \quad \left| \quad (1) \cdot (4) + (4) \cdot (5) \quad \left| \quad (2) \cdot (4) + (5) \cdot (5) \right.
 \end{array}$$

O operador `##` calcula o produto de uma matriz, multiplicando as linhas da primeira matriz pelas colunas da segunda. A segunda matriz deve ter o mesmo número de linhas que a primeira tem de colunas. A matriz resultante terá o número de linhas da primeira matriz e o mesmo número de colunas da segunda matriz.

Por exemplo:

```

IDL> print, a ## b
      10  13
      28  40

```

Os cálculos são feitos desta forma:

$$\begin{array}{l}
 A_{0,0} \cdot B_{0,0} + A_{1,0} \cdot B_{0,1} + A_{2,0} \cdot B_{0,2} \quad \left| \quad A_{0,0} \cdot B_{1,0} + A_{1,0} \cdot B_{1,1} + A_{2,0} \cdot B_{1,2} \right. \\
 A_{0,1} \cdot B_{0,0} + A_{1,1} \cdot B_{0,1} + A_{2,1} \cdot B_{0,2} \quad \left| \quad A_{0,1} \cdot B_{1,0} + A_{1,1} \cdot B_{1,1} + A_{2,1} \cdot B_{1,2} \right.
 \end{array}$$

Usando os valores atuais:

$$\begin{array}{l}
 (0) \cdot (0) + (1) \cdot (2) + (2) \cdot (4) \quad \left| \quad (0) \cdot (1) + (1) \cdot (3) + (2) \cdot (5) \right. \\
 (3) \cdot (0) + (4) \cdot (2) + (5) \cdot (4) \quad \left| \quad (3) \cdot (1) + (4) \cdot (3) + (5) \cdot (5) \right.
 \end{array}$$

A função *WHERE*

A função *WHERE* avalia uma expressão e retorna um índice unidimensional de cada elemento em um vetor para todas as expressões verdadeiras (ele retorna a posição, não os valores dos dados). Se a expressão for falsa, o valor de retorno é `-1`. Por exemplo, iremos alterar todos os elementos entre 0.4 e 0.5 de uma matriz para 1.0:

```

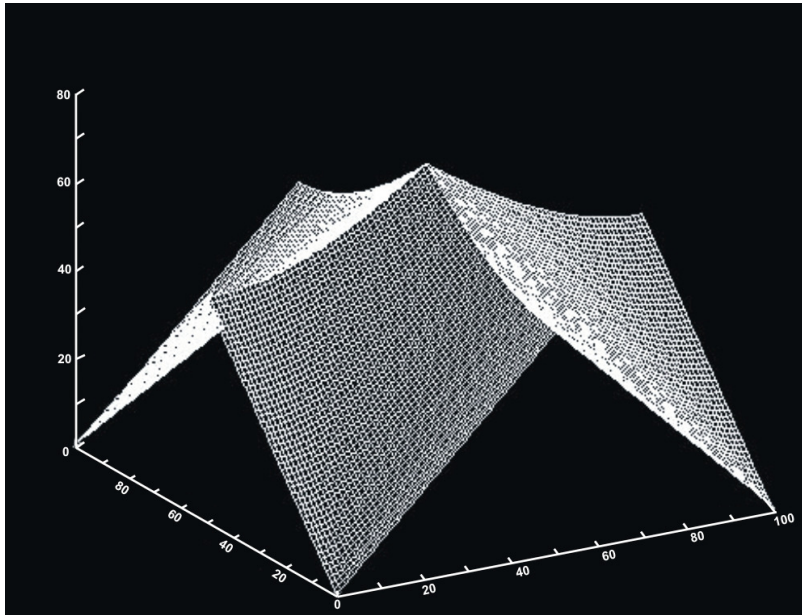
IDL> x = findgen (360) *!dtr
IDL> data = sin(x)
IDL> indices = where ( data gt 0.4 and data lt 0.5, count)
IDL> print, count
12
IDL> data [indices] = 1.0

```

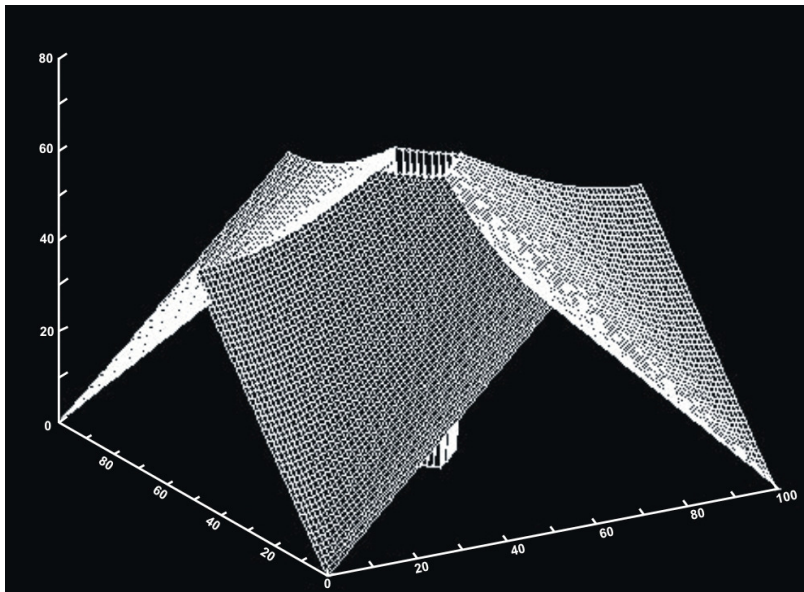
Observe que na terceira linha, onde se encontra a função *WHERE* há um argumento chamado *count*, este argumento cria uma variável com o nome que você decidir (no exemplo ela é chamada de *count*), esta variável irá retornar o número total de elementos encontradas pela função.

Uma matriz multidimensional também pode utilizar a função *WHERE*. Por exemplo,

```
IDL> d = dist (100)  
IDL> surface, d
```



```
IDL> multi_ind = where (d gt 65, count)  
IDL> print, count  
145  
IDL> d [multi_ind] = 0.0  
IDL> surface, d
```



Note que os comandos são praticamente iguais, aos de uma matriz unidimensional.