

Aide-memoire Fortran tableaux

Jean-Guy Caputo

1 Tableaux dynamiques

allocate (array, stat=err) permet de faire l'allocation mémoire du tableau. Si le mot clé stat= est spécifié, la variable *err* (de type integer) vaut 0 si l'allocation s'est déroulée sans problème et 1 sinon.

deallocate (array, stat=err) permet de libérer l'espace mémoire réservée à un tableau.

allocated(array) est une fonction intrinsèque renvoyant true ou false suivant que le tableau spécifié en argument est alloué ou non.

```
integer, dimension (:,:), allocatable :: a
integer :: n, m, err

read (*,*) n, m
if (.not. allocated(a)) then
  allocate(a(n,m),stat=err)
  if (err /= 0) then
    write(*,*) 'Erreur d''allocation dynamique du tableau a'
    stop
  endif
endif
...
deallocate(a)
```

2 Fonctions information sur des tableaux

lbound (A,dim) is a function which returns the lower dimension limit for the array A. If dim (the dimension) is not given as an argument, you get an integer vector, if dim is included, you get the integer value with exactly that lower dimension limit, for which you asked.

shape (SOURCE) is a function which returns the shape of an array SOURCE as an integer vector.

size (ARRAY, dim) is a function which returns the number of elements in an array ARRAY, if DIM is not given, and the number of elements in the relevant dimension if DIM is included.

product (array [,dim][,mask]) et **sum** (array [,dim][,mask]) retournent le produit / la somme des valeurs des éléments du tableau *array* (après un éventuel filtrage par *dim* et / ou *mask*).

Exemple :

product(A) retourne 720

sum(A,dim=1,A > 2) vaut (/ 4, 5, 9 /)

lbound (array [,dim]) et **ubound**(array [,dim]) retournent les bornes inférieures / supérieures de chacune des dimensions (ou seulement de la dimension *dim*) du tableau *array*.

Exemple :

integer, dimension(21:2, 45:49) :: tab

lbound(tab) vaut (/ 21, 45 /) et ubound(tab) vaut (/ 2, 49 /)

lbound(tab, dim=2) vaut 45 et ubound(tab, dim=1) vaut 2

all ALL(mask [,dim]) applique un masque de type logique sur les éléments du tableau et renvoie vrai si pour tous les éléments le résultat du masque est vrai. Si *dim* est précisé, la fonction travaille sur cet indice pour chaque valeur des autres dimensions.

any (mask [,dim]) fonctionne de la même façon que la fonction ALL à l'exception qu'elle renvoie vrai si l'un des résultats du masque est vrai.

count (mask [,dim]) comptabilise le nombre d'éléments pour lesquels le résultat du masque est vrai.

```
program tes
logical l
integer a(2,3), b(2,3)

c      l = all((/.false., .true., .true./))
write(*,*) all((/.false., .true., .true./))

c      l = any((/.false., .true., .true./))
write(*,*) any((/.false., .true., .true./))

a = 1
b = 1
b(2,2) = 2
b(2,3) = 2
write(*,*) a
write(*,*) b
write(*,*) all(a==b, 1)
```

```

write(*,*) all(a .eq. b, 2)
write(*,*) any(a .eq. b, 1)

write(*,*) any(a .eq. b, 2)
write(*,*) count(a /= b)

end

F
T
      1      1      1      1      1      1      1
      1      1      1      2      1      1      2
T F F
T F
T T T
T T
      2

```

minval (array [,dim][,mask]) et **maxval**(array [,dim][,mask]) retournent la plus petite / plus grande valeur du tableau *array* (après un éventuel filtrage par *dim* et / ou *mask*).

exemple

minval(A,dim=2,A > 2) vaut (/ 3, 4 /)

norme du max maxval(abs(A))

minloc (array [,dim][,mask]) et **maxloc** retournent l'indice de la plus grande ou de la plus petite valeur.

Attention : maxloc/minloc dans des sections de matrice, l'indice doit être corrigé et il faut mettre dim.

exemple recherche du pivot dans la colonne k sous diagonale de matrice D

ipiv = maxloc(D(k:n,k),1)+k-1

3 Initialisation de tableaux

```

integer,parameter::n=4
integer,dimension(n)::t1,t2,t3
integer i

```

```

t1=(/6,5,10,1/)
t2=(/ (i+1,i=1,n) /)
t3=(/ t2(1) t1(3) ,1,9  /)

```

4 Sections de tableaux

Section reguliere : elements forment une progression geometrique
 tab(ini : fin : pas)

exemple

```

integer,dimension(10) :: a=(/1,-2,3,-4,5,-6,7,-8,9,-10/)
integer :: i=3,j=7

a (:)
a(i:j)      (3, -4,5,-6,7)
a(i:j:i)    (3, -6)
a(i:)       (3,-4,5,-6,7,-8,9,-10)
a(:j)       (1,-2,3,-4,5,-6,7)
a(:,i)      (1,-4,7,-10)

```

Possibilite de sections non regulieres , adressage indirect

5 Operations entre tableaux

transpose (matrix), ou *matrix* est un tableau de dimension 2, retourne la transposée de matrix.

dot_product (a, b) retourne le produit scalaire de deux vecteurs, c'est à dire $a \cdot b$ si a et b sont de type entier ou réel, $\text{conjugué}(a) \cdot b$ si a et b sont complexes, et $\text{Somme}(a_i \cdot b_i)$ si a et b sont de type logique.

matmul (a, b) retourne le produit matriciel de a et b , sous réserve de compatibilité des dimensions. Soit a et b sont tous deux des matrices, soit a est un vecteur et b est une matrice, soit a est une matrice et b est un vecteur. a et b peuvent être de type quelconque.

norm2 (a) retourne la norme euclidienne du tableau a (vecteur ou matrice).

6 Action sur des tableaux

L'instruction **where** permet d'effectuer des opérations sur des éléments d'un tableau sélectionnés via un filtre de type logique.

```
real, dimension(10) :: a

where (a > 0.) a = sqrt(a)

where (a > 0.)
  a = log(a)
elsewhere
  a = 1.
end where
```

Pour plus d'informations

<https://gcc.gnu.org/onlinedocs/gfortran/Intrinsic-Procedures.html#Intrinsic-Procedures>