POLYTECH MONTPELLIER

Algorithmique

Application en MatLab

V. Berry
MAT-3 & MI-3

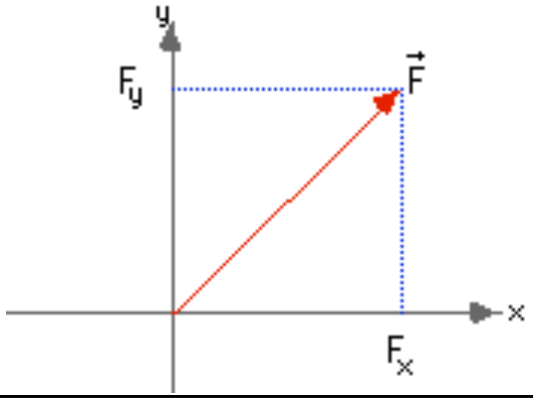
Vecteurs - Matrices - Répétitives

Cours 1-2

II - Variables & Types

Vecteurs

- Un vecteur représente une force, un déplacement, une vitesse, une accélération, ...
- Un vecteur dans le plan (ou dans un espace à plusieurs dimension) peut être exprimé par la liste de ses composantes (une pour chaque axe) :

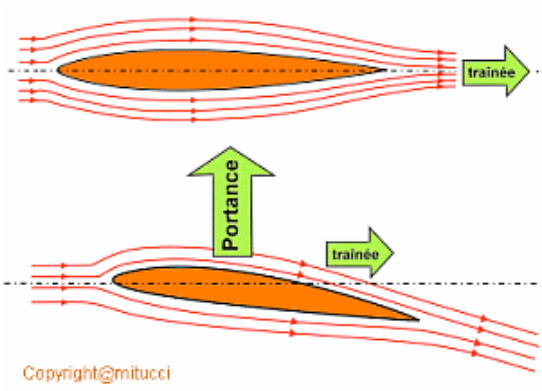


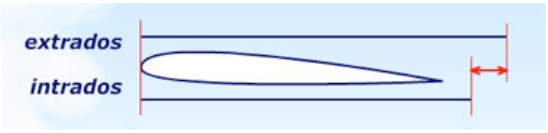
La plus simple représentation d'un vecteur ⇒ un tableau

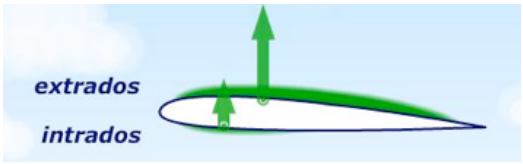
F = [*Fx Fy*]

Vecteurs

- Addition de vecteurs

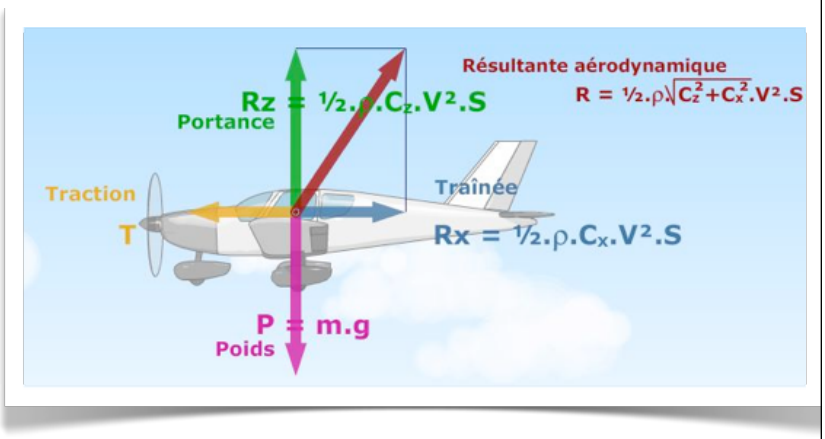






Portance = extrados+intrados

célèbre exemple issu de l'aéronautique



Résultante = Portance + Trainée

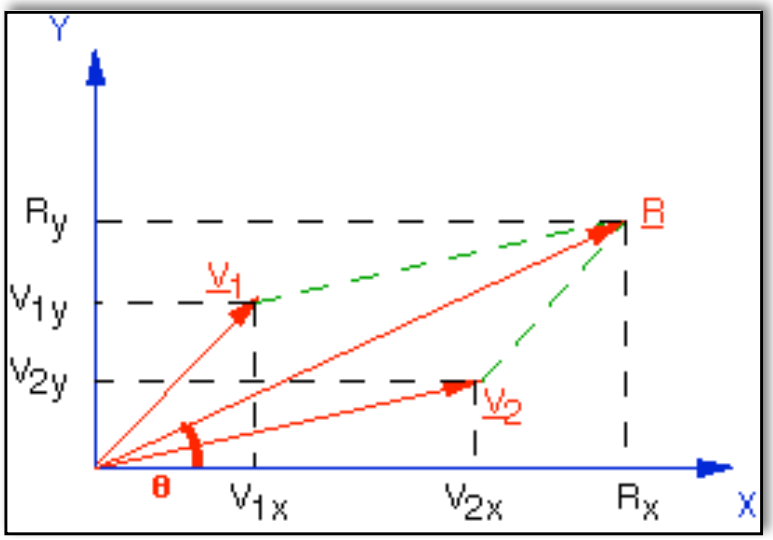
sources : http://www.aviation-fr.info et http://belum76.free.fr

Vecteurs

- Addition de vecteurs

$$Rx = V1x + V2x$$

$$Ry = V1y + V2y$$



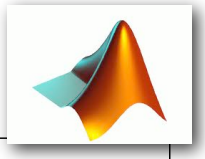
En matlab :

```
>> V1 = [6 2 ]
>> V2 = [2.3 5]
>> R = V1 + V2
R =
        8.30000  7.0000
```

module de R :

```
>> modR = sqrt(R(1)^2  + R(2)^2))
```

Types en Matlab



Vecteurs

Vecteurs ligne et colonne :

```
>>U = [1 2 3]  Entrée

U =
     1   2   3

>> V = [1   Entrée
  2       Entrée
  3]      Entrée

V =
     1
     2
     3
```

La différence est importante par exemple en multiplication de matrices

POLYTECH MONTPELLIER

Matrices

Types de variables

- Ensemble de valeurs stockées dans une structure à une ou plusieurs dimensions (variable de même nom + indice(s))

Exemple : *Usines (Ui) fournissant des chantiers (Cj)*

Besoins :

C1	C2	C3
10	9	5

Coût de livraison de chaque élément

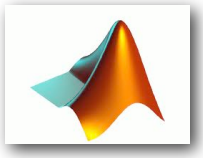
usines	chantiers C1	C2	C3
U1	3	12	10
U2	17	18	35
U3	7	10	24

Une solution potentielle :

	C1	C2	C3
U1	4	0	0
U2	6	6	0
U3	0	3	5

POLYTECH MONTPELLIER

Matrices en Matlab



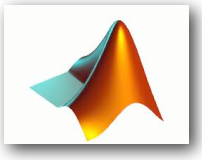
Comme pour les vecteurs, on utilise les crochets '[' et ']' pour définir le début et la fin de la matrice. Ainsi pour définir une variable M contenant la matrice on écrira :

```
>> M = [ 1 2 3
  11 12 13
  21 22 23 ]
```

crochets

```
M=
     1    2    3
    11   12   13
    21   22   23
```

Types en Matlab



Matrices

Point-virgule = séparateur de lignes

Autre possibilité :

```
>> M = [1,2,3 ; 11,12,13 ; 21,22,23]
```

ou bien, en remplaçant la virgule par des blancs :

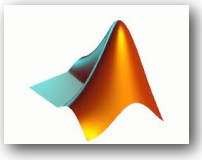
```
>> M = [1 2 3 ; 11 12 13 ; 21 22 23]
```

Ajouter une ligne à une matrice :

```
>> X = [5 6 7] ;
>> M = [M ; X] ;
```

Note : idem pour agrandir un vecteur :
V = [V x]

Types en Matlab



Matrices

accès aux éléments d'une matrice :

consultation

```
>> M = [1 2 3; 11 12 13; 21 22 23];
>> M(3,2)
ans = 22
```

modification

```
>> M(3,2)=32
M= 1   2   3
   11  12  13
   21  32  23
```

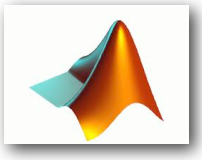
parenthèses

extraction d'une sous-matrice

```
>> M(1,2:2,3)
ans =  2   3
      12  13
```

POLYTECH MONTPELLIER

Types en Matlab



Matrices

Nombreuses autres opérations possibles sur les matrices :

inversion

```
>> inv(A)
```

transposée

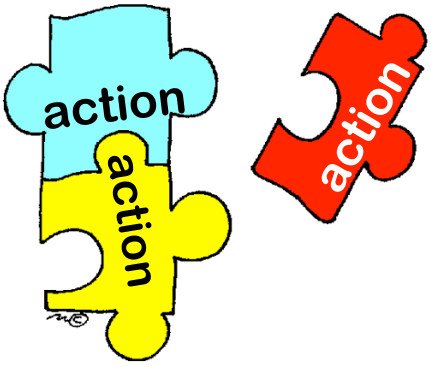
```
>> transp(A)  ;  A'   # au choix
```

déterminant

```
>> det(A)
```

composition, etc

Organisation d'un algorithme (suite)



Rappel :

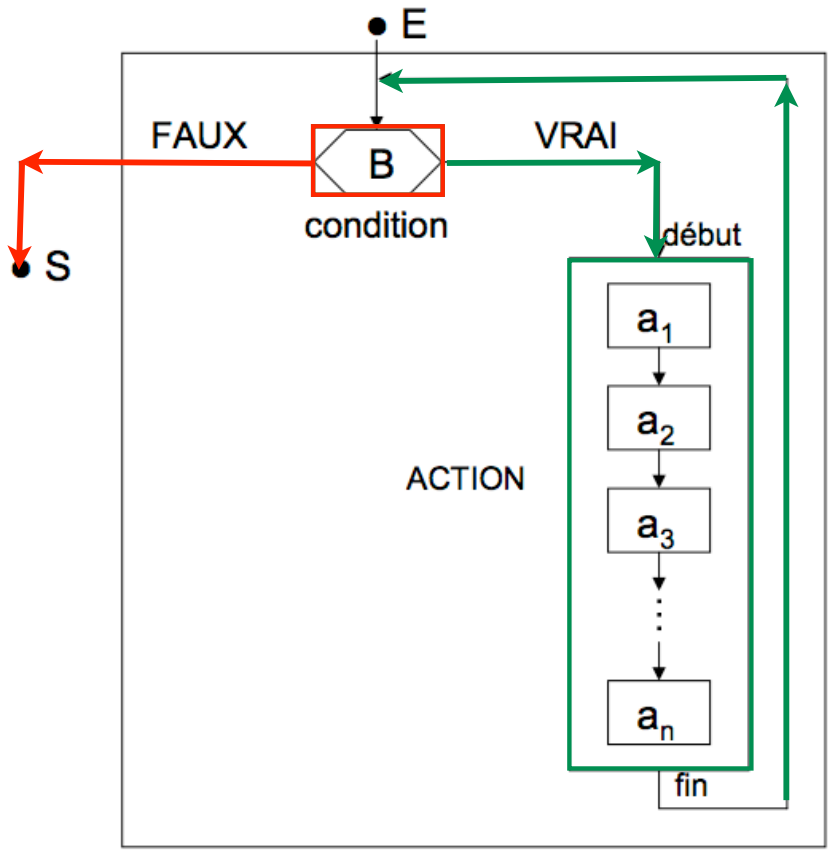
- Un algorithme est une suite d'actions manipulant des informations
- Ces actions peuvent être enchaînées de différentes façons.

POLYTECH MONTPELLIER

Structures d'instructions

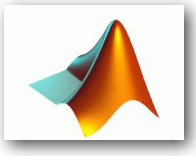
- séquences d'instructions
- conditionnelles
- répétitives (boucles)
  1. Le principe d'une répétitive
  2. Différents types de répétitives :
     - pour i de 1 à n-1 ...
     - tant que i < n ...
     - la vectorisation (Matlab)
- fonctions

Principe d'une répétitive



- On veut répéter une séquence d'actions tant qu'une condition est remplie

boucle « POUR » en Matlab



La boucle for

Quand on connait à l'avance le nombre de répétitions à effectuer

Répéter une suite d'actions **un nombre de fois fixé**

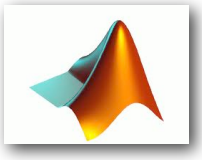
Syntaxe :

```
for i = borne_inf : borne_sup
    séquence d'instructions
end
```

mots clefs

«Boucle» pour les valeurs d'un *indice* (i), incrémenté à chaque itération («tour de boucle»), variant entre deux bornes

boucle « POUR » en Matlab



La boucle for

Répéter une suite d'action un nombre de fois fixé

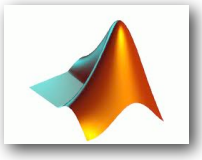
Syntaxe :

```
for i = borne_inf : borne_sup
    séquence d'instructions
end
```

Interprétation :

1. la variable i prend la valeur borne_inf
2. la séquence d'instructions est exécutée
3. i voit sa valeur augmentée de 1
4. les étapes 2 et 3 sont répétées *en boucle* jusqu'à ce que i **dépasse** la valeur borne_sup

boucle « POUR » en Matlab



Exemple :
usines à livrer
depuis chantiers

C coût de livraison

	C1	C2	C3
U1	3	12	10
U2	17	18	35
U3	7	10	24

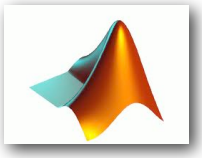
Q quantité à livrer

	C1	C2	C3
U1	8	6	3
U2	0	0	4
U3	2	1	0

Coût de la livraison de tous les chantiers ?

Question en amont = décider façon de livrer ayant un coût minimum, sachant contraintes

boucle « POUR » en Matlab



La boucle for

```
for i = borne_inf : borne_sup
   séquence d'instructions
end
```

Remarques :

- Si borne_inf > borne_sup, aucun tour de boucle n'est exécuté
- L'incrément de la variable de boucle est de 1 par défaut, mais on peut indiquer un autre incrément de la façon suivante :

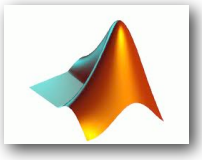
  borne_inf : pas : borne_sup

- Si l'on veut une progression non-régulière de l'indice, on peut uiliser une tableau d'indices :

  for i = tableau ; .... ; end

- la variable de boucle peut prendre des valeurs réelles
- **ne pas modifier la variable de boucle dans la séquence d'instructions au centre de la boucle**

boucle « tant que » en Matlab



La boucle while

Répéter une suite d'action un nombre de fois non-fixé à l'avance

Syntaxe :

```
while condition
    séquence d'instructions
end
```

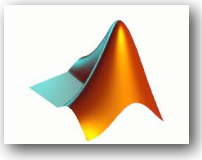
« corps » de la boucle

mots clefs

Interprétation :

1. condition est une expression logique, renvoyant vrai ou faux, évaluée avant d'effectuer chaque tour de boucle
2. si condition est vraie alors la séquence d'instructions est exécutée
3. le cycle 1 puis 2 se reproduit tant que la condition de continuation est vraie. Quand elle devient fausse, on exécute ensuite la première instruction suivant le mot-clef end.

boucle « tant que » en Matlab



La boucle while

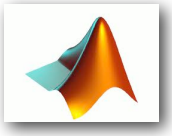
```
while condition
    séquence d'instructions
end
```

Remarques :

- la condition implique presque toujours des variables
- si une variable de boucle est utilisée, les instructions de la boucle doivent faire évoluer cette variable

```
while (epsilon < 0.1)
    instructions
    epsilon = espilon + 0.05 ;
end
```

boucle « tant que » en Matlab



Exemple : trouver le premier nombre dont la factorielle utilise plus de 100 chiffres :

```
n = 1;
while prod(1:n) < 1e100
    n = n + 1;
end
```

boucle « tant que » en Matlab

Pièges de la boucle «tant que»

- Oublier d'initialiser la variable de boucle avant la boucle

```
x = 10
while x>0
   .....
```

- Oublier de faire progresser cette variable dans la boucle

```
while x>0
   .....
   x = x - ...
end
```

- Donner une condition de boucle pas assez précise (la variable de boucle doit la vérifier à un moment donné)

Préférer
while x>0
à
while x~=0

boucle « tant que » en Matlab

Remarques sur la boucle « tant que » - suite

- certaines boucles « tant que » peuvent être représentées par des boucles « pour »
- ne pas le faire quand on ne connait pas à l'avance le nombre de tours de boucle (instr. break peu lisible)

Equivalence entre formes de boucles

Traduction de «tant que» en boucle «pour» : *recherche d'une valeur particulière dans un tableau de valeurs*

Algorithme Recherche(d T : tableau, d e : élément) : booléen
Données : un tableau T et un élément e
Résultat : vrai ssi e appartient au tableau

(tableau T non trié)

```
i=1; N = 10;
while (T(i)~=e) && (i<=N)
  i = i + 1;
end
return (i <= N);
```

quel est le soucis ?

traduction

? version avec un «pour» ?

quel est le soucis ?

source P. Janssen

Equivalence entre formes de boucles

Exemple 1 de traduction de «tant que» en boucle «pour» :
recherche d'une valeur particulière dans un tableau de valeurs

Version «tant que» équivalente à la version «pour» :

Algorithme Recherche(d T : tableau, d e : élément) : booléen
Données : un tableau T et un élément e
Résultat : vrai ssi e appartient au tableau

```
for i=1:N
    if T(i) == e
       return true;
    end
end
return false;
```

traduction correcte

POLYTECH MONTPELLIER

ANNEXE

Instructions répétitives

Imbrication de boucles

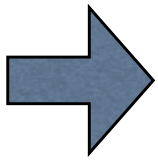
- il est parfois nécessaire de placer une boucle à l'intérieur d'une autre boucle
- la boucle intérieure est alors exécutée à chaque itération de la boucle extérieure
- ci-après, un exemple avec trois boucles imbriquées !

Imbrication de boucles

Exemple du test d'égalité de deux lignes d'une matrice :

- En Matlab, un système d'équations linéaires peut être codé dans une matrice
- chaque ligne représente une équation

$$
\begin{aligned}
3x + 2y - z + t + 10 &= 0 \\
x + z + 5 &= 0 \\
4y + t - 3 &= 0 \\
x + z + 5 &= 0
\end{aligned}
$$



M

$$
\begin{bmatrix}
3 & 2 & -1 & 1 & 10 \\
1 & 0 & 1 & 0 & +5 \\
0 & 4 & 0 & 1 & -3 \\
1 & 0 & 1 & 0 & +5
\end{bmatrix}
$$

- Avant de résoudre le système, on veut le simplifier si possible : savoir si des lignes sont identiques

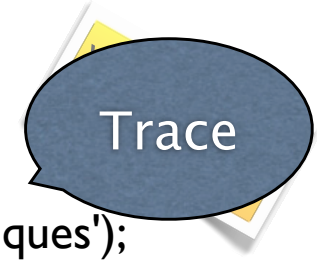
Imbrication de boucles

Exemple du test d'égalité de deux lignes d'une matrice :

```
trouve = false;  % on début on n'a pas trouvé deux lignes identiques
for lig1 = ??:??
   for lig2= ??:??
      % teste l'égalité des ligne1 et ligne2 de M :
      egales = true; % supposées égales
      for col = ??:??
         if ( M(lig1,col) ~= M(lig2,col))
            egales = false;  % lignes prouvées non égales
         end
      end
      if  (egales)  trouve = true;  end
   end
end
if ( trouve == true ) disp('M a deux lignes identiques');
else disp('M n''a que des lignes différentes'); end
```

M

1	1	1	1
1	0	1	0
1	1	0	1
1	0	1	0



POLYTECH MONTPELLIER

Imbrication de boucles

Exemple du test d'égalité de deux lignes d'une matrice :

M

1	1	1	1
1	0	1	0
1	1	0	1
1	0	1	0

Pouvez-vous proposer une version plus courte en tirant parti de la vectorisation ?

Quelques opérateurs Matlab

- Caractères spéciaux

```
() parenthèses
= affectation
, virgule : séparateur de valeurs
; point virgule : séparateur
d'instructions, de lignes dans les
matrices
% commentaire ou pour indiquer un
format
: deux-points : « jusqu'à », par
exemple 1:10
```

Opérateurs arithmétiques

```
+ addition
- soustraction
* multiplication
```

Quelques opérateurs Matlab

- Opérateurs relationnels

  `==` test d'égalité

  `~=` test de différence

  `>`

- Opérateurs logiques

  `&`

  `~`

POLYTECH MONTPELLIER

Liens

- Cours de F. Buffat à l'UFR de Méca de Lyon http://ufrmeca.univ-lyon1.fr/~buffat
- Cours de F. Nicol (Louvain-la-Neuve)
- http://www.phon.ucl.ac.uk/courses/spsci/matlab/
- Nombreux tutoriels sur le site de Matlab et sur Internet en général