

```
>> A=[1,3,5], B=[2,4,1], C=[1,1;1,2]
A =
    1     3     5
B =
    2     4     1
C =
    1     1
    1     2
>> [A,B],[A;B]
ans =
    1     3     5     2     4     1
    1     3     5
    2     4     1
```

3. Chaîne de caractère

La chaîne de caractères est un vecteur ligne.

Pour le créer une chaîne de caractères, on entre les caractères entre deux quotes (').

4. Matrice

Pour créer une matrice on suit la même démarche que les vecteurs, on sépare les colonnes par des espaces ou des virgules et on sépare les lignes par des points-virgules ou bien 'entrer'.

Créer les matrices suivantes :

$$A = \begin{bmatrix} 3 & 2 & 4 \\ 4 & 2 & 4 \\ 1 & 3 & 1 \end{bmatrix} \quad B = \begin{bmatrix} 4 & 12 & 24 \\ 40 & 2 & 5 \\ 4 & 53 & 31 \end{bmatrix} \quad C = \begin{bmatrix} A & B & B \\ B & A & B \\ B & B & A \end{bmatrix}$$

Activité 2 : Vecteurs usuels

1. $X1 = 0:2:10$ % vecteur ligne comportant les valeurs de 0 à 10 avec un pas = 2 : 0 2 4 6 ...10
2. $X2 = 1:0.5:4$ % ici le pas est 0,5, les valeurs seront : 1 1,5 2 2,5 4
3. $X3 = 10:-1:0$ % le pas peut prendre une valeur négative : 10 9 8 7 0
4. $X4 = 1:10$ % si on ne précise pas le pas, Matlab prend 1 comme valeur par défaut
5. $x = [1 \ 3 \ 5]$
 $y = [x \ 6 \ 8 \ 10]$ % y est un vecteur composé par la concaténation des composantes
% du vecteur x et les éléments 6, 8, 10
% concaténation horizontale : on utilise espace ou virgule
 $z = [x;1:3]$ % concaténation des vecteurs x et 1:3,
% concaténation verticale : on utilise entrer ou point-virgule
6. $\text{linspace}(a,b,n)$ % subdivision de l'intervalle [a b] en n points équidistances
7. $\text{linspace}(0,1,5)$ % subdivision de l'intervalle [0 1] en 5 points équidistances
8. $\text{linspace}(a,b)$ % si on précise pas n Matlab prend par défaut n=100
9. $\text{logspace}(a,b,n)$ % subdivision de l'intervalle $[10^a \ 10^b]$ en n points logarithmiquement équidistances
10. $\text{logspace}(a,b)$ % si on précise pas n Matlab prend par défaut n=50
11. $\text{length}(x)$ % longueur du vecteur x
12. $\text{size}(x)$ % taille de la matrice x
13. $\text{numel}(x)$ % nombre d'éléments de x (vecteur ou matrice)

Activité 3 : Matrices usuelles

1. $n=4 ; m=5 ;$
2. $\text{ones}(n)$ % matrice carrée d'ordre n, dont tous les éléments valent 1
3. $\text{ones}(n,m)$ % matrice de taille n X m, dont tous les éléments valent 1
4. $\text{ones}([n,m])$ % matrice de taille n X m, dont tous les éléments valent 1

En particulier :

5. $\text{ones}(1,n)$ % vecteur ligne de longueur n dont tous les éléments valent 1
6. $\text{ones}(m,1)$ % vecteur colonne de longueur m dont tous les éléments valent 1

De même pour :

7. $\text{zeros}(n)$ % matrice carrée d'ordre n dont tous les éléments valent 0
8. $\text{zeros}(n,m)$

9. zeros ([n,m])
10. zeros (1,n)
11. zeros (m,1)

Encore :

12. eye (n) % matrice unitaire d'ordre n
13. eye (n,m)
14. eye ([n,m])

Encore :

15. rand(n) % matrice carrée d'ordre n d'éléments entre 0 et 1 générés d'une manière aléatoire.
16. rand(n,m)
17. rand([n,m])
18. rand(1,n)
19. rand(m,1)

Encore :

20. randi(m,n) % matrice carrée d'ordre n d'éléments entiers entre 1 et m générés d'une manière aléatoire.
21. randi(m) % nombre entier générés d'une manière aléatoire entre 1 et m.
22. randi(m,a,b) % matrice de taille axb d'éléments entre 1 et m générés d'une manière aléatoire.

Encore :

23. randn(m,n) % matrice de taille mxn dont les éléments suivent la loi normale.
24. randn(m) % matrice crée d'ordre n dont les éléments suivent la loi normale.
25. randn(1,n) % vecteur ligne de taille n les éléments suivent la loi normale.

Encore :

26. magic(n) % matrice carrée d'ordre n dont les éléments sont entre 1 et n^2 et la somme de chaque ligne, colonne, diagonale, etc. égale $\frac{n(n^2+1)}{2}$

Activité 4 : Indexation de vecteurs et de matrices

27. x = [10 20 30 40 50 60 70 80 90]
 y = x(3) % **lecture** de l'élément 3 du vecteur x
 x(4) = 17 % **modification (écriture)** de l'élément 4 du vecteur x
 x(2 :5) % de l'élément 2 à l'élément 5 de x
 x(3:5)=[] % suppression des éléments de 3 à 5 de x
 % l'affectation du vecteur nul (ou vide) [] de la position 3 à 5 → la suppression
 x*y' % produit scalaire des vecteurs (**lignes de même taille**) x et y
 x'*y % produit scalaire des vecteurs (**colonnes de même taille**) x et y
 sum(x.*y) % produit scalaire des vecteurs (**de même taille**) x et y
28. sum(x) % calcul la somme des éléments de x
 prod(x) % calcul le produit des éléments de x
29. A = [1 2 3 ; 4 5 6 ; 7 8 9]
 y = A(3,2) % lecture de l'élément de la 3^{ème} ligne et la 2^{ème} colonne de A
 z = A(5) % une autre manière d'indexation des éléments de A
30. A(3,2) = 11 % écriture dans la matrice A
31. A(5) = 12 % écriture dans la matrice A

Activité 4 : Fonctions et Opérations sur les matrices

1. `M = ones(3,3)`
`x= M(5,5)` % accès en lecture d'une valeur qui n'existe pas → le système renvoie une erreur
`M(5,5) = 10` % accès en modification → le système complète la matrice par des 0

Tester encore :

2. `M = zeros(3)`
`M(5,5) = 0`
3. `M = ones(3,5)*77`
4. `M=rand(3,4)`
`M(:) = 77` % renvoie la matrice sous forme d'un vecteur ligne
5. `M = repmat(M,3,5)` % répéter la matrice M

Tester encore :

6. `M = 77`
`M = M(ones(3,5));`
 7. `M = ones(5)*77;`
 8. `M = zeros(5);`
`M(:) = 77;`
 9. `M = repmat(77,5,5);`
 10. `M = M( : , end:-1:1)` % inverser les colonnes de la matrice M
 11. `n = 5;`
`M = toeplitz([1 3 zeros(1,n-2)],[1 2 zeros(1,n-2)]);`
 12. `M = [1 2 ; 3 4];`
`M = kron(M,ones(2,3));`
 13. `M = [8 4 ; 1 7]`
`M = kron(M, eye(2,3))`
 14. `rot90(M)` % rotation de la matrice M
 15. `rot90(A,3)`
 16. `fliplr(M)` % Retourner la matrice de la gauche à la droite (inverser les colonnes)
 17. `flipud(M)` % Retourner la matrice haut en bas (inverser les lignes)
 18. `flipud(fliplr(M))`
 19. `fliplr(A)`
 20. `reshape(A,4,3)`
 21. `reshape(A,6,2)`
 22. `reshape(A,2,6)`
 23. `reshape(flipud(B),8,2)`
 24. `triu(B)` % renvoie la matrice triangulaire supérieure de la matrice B
 25. `tril(B)` % renvoie la matrice triangulaire inférieure de la matrice B
- tester encore :
26. `triu(B,-1)`
 27. `tril(A,2)`
 28. `diag(M)` % renvoie la diagonale de la matrice A
 29. `diag(rot90(B))`
 30. `rank(A)` % renvoie le rang de la matrice A

31. inv(A)	% renvoie l'inverse de la matrice A
32. det(A)	% renvoie le déterminant de la matrice A
33. trace(A)	% trace de A
34. poly(A)	% polynôme caractéristique de A
35. eig(A)	% valeurs propres de A
36. [U,D]=eig(A)	% calcul à la fois les vecteurs propres et les valeurs propres de A

Exercices

Exercice 1

Calculer le factoriel des de n dans les cas suivants :

$$n=10 \quad ; \quad n=20 \quad ; \quad n=100 \quad ; \quad n=500 \quad ; \quad n=1000$$

Indication : 1) utiliser la syntaxe « a:b » et la fonction prod() ; 2) chercher une fonction Matlab qui calcul directement le factoriel d'un nombre

Exercice 2:

Considérant les trois vecteurs u_1, u_2 et u_3 , et la matrice A ci-dessous

$$\vec{u}_1 = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}, \vec{u}_2 = \begin{pmatrix} -5 \\ 2 \\ 1 \end{pmatrix}, \vec{u}_3 = \begin{pmatrix} -1 \\ -3 \\ 7 \end{pmatrix}, A = \begin{pmatrix} 2 & 3 & 4 \\ 7 & 6 & 5 \\ 2 & 8 & 7 \end{pmatrix}.$$

- 1) Structures Matlab
 - a) Entrer ces données (les trois vecteurs et la matrice) sous Matlab.
 - b) Calculer $u_1 + 3u_2 - u_3/5$.
 - c) Calculer le produit scalaire entre les vecteurs : u_1 et u_2 .
 - d) Calculer le produit Au_1 .
- 2) Commandes Matlab ; Trouver les commandes Matlab permettant de :
 - a) Calculer $\|u_1\|_2$, $\|u_1\|_1$ et $\|u_1\|_\infty$ (indication : utiliser la fonction **norm**)
 - b) Déterminer les dimensions de la matrice A, en extraire le nombre de colonnes (indication : utiliser la fonction **size**)
 - c) Calculer le déterminant et l'inverse de A (indication : utiliser les fonctions **det** et **inv**).
- 3) Résolution de systèmes linéaires
Proposer deux méthodes permettant de résoudre le problème $A\vec{x} = \vec{u}_1$, et déterminer les commandes Matlab associées.