Travaux pratiques de MATLAB
TP N°2 : Vecteurs & Matrices

Atelier 1 : Vecteurs usuels

```
1.    x = [1 3 5]     % création d'un vecteur ligne dont les composantes sont 1, 2 et 3
2.    x = 0:1:10      % création d'un vecteur ligne dont les composantes sont 0, 1, 2, … 10
3.    x = 1:0.5:4     % nous pouvons préciser le pas, dans cet exemple le pas est 0,5
4.    x = 10:-1:0     % le pas peut prendre une valeur négative
5.    x = 1:10  % si on ne précise pas le pas, Matlab prend 1 comme valeur par défaut
6.  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 horizontale : on utilise entrer ou point-virgule
7.    linspace(a,b,n) % subdivision de l'intervalle [a b] en n points équidistances
8.    logspace(a,b,n) %subdivision de l'intervalle [10^a 10^b] en n points équidistances
9.    linspace(a,b)   % si on précise pas n Matlab prend par default n=100
10.   logspace(a,b)   % si on précise pas n Matlab prend par default n=50
NB : - logspace(a,b,n) = linspace(10^a,10^b,n)
11.
12.   y = sqrt(1:10)          b=sqrt(a)     □ a=b.*b
13.   sqrtm                        b=sqrtm(a)  □ a= b*b
14.   length(x)
15.   numel(x)
16.   size(X)
```

Atelier 2 : Matrices usuelles

```
1.    ones (n)    % matrice carrée d'ordre n, dont tous les éléments valent 1
2.    ones (n,m)    % matrice de taille n X m, dont tous les éléments valent 1
3.    ones ([n,m])  % matrice de taille n X m, dont tous les éléments valent 1
```

En particulier :

```
4.    ones(1,n) % vecteur ligne de longueur n dont tous les éléments valent 1
5.    ones(m,1) % vecteur colonne de longueur m dont tous les éléments valent 1
```

De même pour :

```
6.    zeros (n)     % matrice carrée d'ordre n dont tous les éléments valent 0
7.    zeros (n,m)
8.    zeros ([n,m])
9.    zeros (1,n)
10.   zeros (m,1)
```

Encore :

```
11.   eye (n)      % matrice unitaire d'ordre n
12.   eye (n,m)    % matrice de taille n X m, dont tous les éléments valent 0 et les
                   % éléments de la diagonale valent 1
13.   eye ([n,m])
```

Encore :

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

Encore :

```
19.   randi(m)     % matrice carrée d'ordre 1 dont les éléments sont des entiers
générés
                   % d'une manière aléatoire entre 0 et m : c'est donc un nombre entier
                   % généré d'une manière aléatoire entre 0 et m
20.   randi(m,n)   % matrice carrée d'ordre n dont les éléments sont des entiers
générés
                   % d'une manière aléatoire entre 0 et m.
21.   randi(m,a,b)  % matrice de taille aXb dont les éléments sont des entiers
générés
                     % d'une manière aléatoire entre 0 et m.
```

Encore :

```
22.   magic(n)  % matrice carrée d'ordre n dont les éléments sont entre 1 et n² et la
                % somme des chaque ligne, colonne, diagonale, etc. égale à n²
```

Atelier 3 : Indexation de vecteurs et de matrices

```
23. x = [1 3 51 3 51 3 51 3 5]
    y = x(3)     % lecture de l'élément 3 du vecteur x
    x(4) = 17    % modification (écriture) de l'élément 4 du vecteur x
24. A = [1 3 51 ; 3 51 3 ; 51 3 5]
    y = A(3,2)   % lecture de l'élément de la 3ème ligne et la 2ème colonne de A
    z = A(5)     % une autre manière d'indexation des éléments de A

25. A(3,2) = 11 % écriture dans la matrice A
26. A(5) = 12    % écriture dans la matrice A
```

Atelier 4 : Fonctions et Opérations sur les matrices

```
27. M = ones(3,5)
    M(5,5) = 0
28. M = zeros(3)
    M(5,5) = 0
29. M = ones(3,5)*77
30. rand(3,4)
    M(:) = 77
31. M = repmat(77,3,5);
```

```
32. M = 77;
    M = M(ones(3,5));
33. M = ones(5)*77;
34. M = zeros(5);
    M(:) = 77;
35. M = repmat(77,5,5);
36. M = M(:,end:-1:1)
37. M(3,5) = 1;
38. n = 5;
    M = toeplitz([1 3 zeros(1,n-2)],[1 2 zeros(1,n-2)]);
39. M = [1 2 ; 3 4];
    M = kron(M,ones(2,3));
40. M = [8 4 ; 1 7]
    M = kron(M, eye(2,3))
41. M(:,1:2)
42. rot90(M)
43. fliplr(M)
44. flipud(M)
45. triu(M)
46. tril(M)
47. diag(M)
48. rot90(B)
49. rot90(A,3)
50. fliplr(A)
51. flipud(fliplr(B))
52. reshape(A,4,3)
53. reshape(A,6,2)
54. reshape(A,2,6)
55. reshape(flipud(B),8,2)
56. triu(B)
57. triu(B,-1)
58. tril(A,2)
59. diag(rot90(B))
```

Atelier 5 :

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 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.