Université Moulay Ismaïl
Faculté des Sciences et Techniques
Errachidia

Département de Physique

Filière : Licence en science et technique :

Energies Renouvelables, Option : Technologie Solaires et éoliennes

Elément de Module : Calcul Scientifique (P522)

Polycopié des Travaux dirigés

Préparée par :
Pr. Sara TEIDJ

Année Universitaire : 2022-2023

Série 1

Exercice 1

Tester et comprendre les deux lignes suivantes

- $A=[1 \quad 2 \quad -1 \quad 1 \quad ; \quad -1 \quad 1 \quad 0 \quad 3]$
- find(A>0)

Exercice 2

On note u,v et w les vecteurs suivants : $u=[1 \quad -1 \quad 2]^t$, $v=[10 \quad -1 \quad 3]^t$, $w=[5 \quad -1 \quad 4]^t$

- Calculer : $a=3u$, $b=2u-v+5w$, $c=w-4v$.

Exercice 3

On note u et v les nombres complexes: $u=11-7i$, $v=-1+3i$.

Calculer les modules de u et de v , les produits $u\bar{v}+\bar{u}v$, la partie réelle et la partie imaginaire u^3+v^2

Exercice 4

On pose $A=\begin{bmatrix} 1 & -1 & 7 \\ -4 & 2 & 11 \\ 8 & 0 & 3 \end{bmatrix}$, $B=\begin{bmatrix} 3 & -2 & -1 \\ 7 & 8 & 6 \\ 5 & 1 & 3 \end{bmatrix}$

Que font les instructions suivantes ?

$3*A$; $A.*B$; $A./B$; $\cos(A)$; $\exp(B)$.

Exercice 5

1. Créer un vecteur u de 11 composantes contenant les nombres $-5,-4,...,4,5$

2. Créer un vecteur v de 1001 composantes contenant les nombres $-500,-499,-498,...,499,500$

3. Créer un vecteur w contenant 10 valeurs entre 0 et π séparées par un incrément constant.

Corrigé série 1

Corrigé de l'exercice 1

% On tape dans l'invite de Matlab

```
>> A=[1 2 -1 1; -1 1 0 3]

A =

    1    2   -1    1

   -1    1    0    3
```

% La commande précédente permet de créer une variable de type tableau (matrice) à deux

% lignes et quatre colonnes. La première ligne contient les termes spécifiés par la liste

% 1 2 -1 1 qui sont séparés par un blanc, alors que les termes de la deuxième ligne

% commencent après le point virgule le séparation des lignes et sont définis par la liste -1 1 0 3.

% On tape dans l'invite de Matlab

```
find(A>0)

ans =

    1

    3

    4

    7

    8
```

% La commande *find* permet de trouver les indices **des termes d'un tableau selon le critère**

% spécifié entre parenthèses. Ici le critère est A>0 qui signifie donc que l'on veut les termes

% du tableau A qui sont strictement positifs. La numérotation avec un seul indice adoptée par

% Matlab est telle que les termes sont numérotés par colonne de haut en bas et de gauche à

% droite. Le premier terme de A est donc 1, le deuxième est -1, le troisième est 2, …, le

% septième et 1 et le huitième est 3. La commande *find* produit comme résultat la liste

% 1 3 4 7 8 qui représente les indices **des termes (et non pas les termes) de la matrice A qui**

% satisfont le critère.

Corrigé de l'exercice 2

1.**% On tape dans l'invite de Matlab**

>> u=[1 -1 2].'

u =

1

-1

2

% Noter la présence du symbole .' en fin de l'instruction qui signifie le transposé selon la syntaxe de Matlab.

% On tape ensuite dans l'invite de Matlab

>> v=[10 -1 3].'

v =

10

-1

3

>> w=[5 -1 4].'

w =

5

-1

4

% Pour calculer 3u, on tape dans l'invite de Matlab

>>a= 3*u

ans =

3

-3

6

% On obtient donc le résultat qui s'écrit en notations mathématiques [3 -3 6]^t

% Pour calculer $2u-v+5w$ **, on tape dans l'invite de Matlab**

>>b= 2*u-v+5*w

ans =

17

-6

21

% Le résultat est donc $[17 \quad -6 \quad 2]^t$. Noter qu'il faut bien mettre * qui représente l'opération de multiplication et qu'on ne peut pas faire le calcul en utilisant la notation mathématique 2u-v+5w qui produirait un message d'erreur.

% Pour calculer $w-4v$, on tape dans l'invite de Matlab

>> c=w-4*v

c =

-35

3

-8

corrigé de l'exercice 3

% Pour affecter la valeur complexe u =11−7i , on tape dans l'invite de Matlab (avec la précaution signalée sur i)

>> u=11-7i

u =

11.0000 - 7.0000i

% Pour affecter la valeur complexe −1+3i à v , on tapera dans l'invite de Matlab l'instruction

% suivante

>> v=-1+3i

v =

-1.0000 + 3.0000i

1. % Les modules de u et de v se calculent par les commandes abs(u) et abs(v), où la fonction

% Ainsi, on obtient

>> abs(u)

ans =

13.0384

>> abs(v)

ans =

3.1623

% Le conjugué d'un nombre complexe se calcule dans Matlab avec la commande conj.

% Pour calculer l'expression $u\bar{v}+\bar{u}v$ **, on tape dans l'invite de Matlab**

>> u*conj(v)+conj(u)*v

ans =

-64

% Pour calculer les parties réelle et imaginaire de u^3+v^2**, on tape dans l'invite de Matlab**

% les commandes

>> real(u^3+v^2)

ans =

-294

% et

>> imag(u^3+v^2)

ans =

-2204

Corrigé de l'exercice 4

% Il s'agit d'opérations terme à terme, opérations dites de tableaux (array opérations). Ces

% opérations n'ont pas de sens algébrique propre comme c'est le cas de la multiplication des

% matrices. Les arguments des fonctions de Matlab peuvent être des variables de type tableau.

% Ainsi la fonction *cos* peut calculer en bloc tous les cosinus des termes du tableau qui lui

% est présenté comme argument. On peut de la sorte faire des opérations en bloc et éviter

% de distinguer les cas. Cela permet de gagner en vitesse de programmation. L'une des

% qualités de Matlab c'est justement ça: Matlab est bien Matrix Laboratoty.

% Les opérations terme à terme ne sont possibles que si les deux tableaux admettent

% les mêmes dimensions.

% La réponse à la question posée dans cet exercice est obtenue par les instructions suivantes :

>> A=[1 -1 7; -4 2 11; 8 0 3]

```
A =

     1    -1     7
    -4     2    11
     8     0     3

>> B=[3 -2 -1; 7 8 6; 5 1 3]

B =

     3    -2    -1
     7     8     6
     5     1     3

>> 3*A

ans =

     3    -3    21
   -12     6    33
    24     0     9

>> A.*B

ans =

     3     2    -7
   -28    16    66
    40     0     9

>> A./B

ans =

   0.333333333333333   0.500000000000000  -7.000000000000000
  -0.571428571428571   0.250000000000000   1.833333333333333
   1.600000000000000                   0   1.000000000000000
```

% On voit dans ce dernier résultat qu'en cas de besoin Matlab affiche le résultat avec le

% maximum de précision possible, c'est-à-dire la double précision.

```
>> cos(A)

ans =

    0.5403    0.5403    0.7539
   -0.6536   -0.4161    0.0044
   -0.1455    1.0000   -0.9900
```

%

```
>> exp(B)

ans =

  1.0e+03 *

    0.0201    0.0001    0.0004
    1.0966    2.9810    0.4034
    0.1484    0.0027    0.0201
```

% Ne pas oublier ici que tous les termes sont multipliés par 1000 comme l'indique la ligne

% en surbrillance jaune. 1000 en notation Matlab s'écrit 1.0 e+003, on peut aussi l'écrire

% tout simplement 1.e3 qui signifie la même chose que 10^3.

Corrigé de l'exercice 5

1.**% La commande à utiliser est *linspace*. On écrit alors dans l'invite de Matlab.**

```
>> u=linspace(-5,5,11)

u =

  -5   -4   -3   -2   -1    0    1    2    3    4    5
```

2.**% On écrit alors dans l'invite de Matlab.**

```
>> v =linspace(-500,500,1001) ;
```

3. **% Il suffit de taper**

```
>> w=linspace(0,pi,10)

 Columns 1 through 9

0    0.3491    0.6981    1.0472    1.3963    1.7453    2.0944    2.4435    2.7925

 Column 10

3.1416
```

Série 2

Exercice 1

- Ecrire un Fichiers script ou function appelé **polaire.m** qui permet de convertir les coordonnées cartésiennes d'un point en coordonnées polaires.
- Convertir les coordonnées cartésiennes suivants: (0,1)

Exercice 2

1. Ecrire un Fichiers script pour la fonction $f(x)=\frac{x^5-3}{\sqrt{x^2+1}}$

2. Tester la fonction sur quelques valeurs, par exemple $f(1)=-1.4142$, $f(0)=-3$.

3. Créer un tableau x d'abscisses de −5 à 5 et contenant 100 points.

4. Représenter la fonction f aux points xi en vert.

Exercice 3

Ecrire un script pour la fonction :$f(x) = e^{\sin(x)}$ sur l'intervalle $[-\pi/2, \pi/2]$.

1. Calculer $f(0), f(\pi/4), f(1)$.

2. Tracer la fonction $f(x)$ sur l'intervalle $[-\pi/2, \pi/2]$ en rouge, nommer les axes x et y et donner un titre à la courbe

3. Calculer le maximum de $f(x)$ sur l'intervalle $[-\pi/2, \pi/2]$

Exercice 4

Ecrire un script pour la fonction: $f(x) = e^{-x^2} \cos\left(\frac{\pi x}{2}\right)$ sur l'intervalle $[-1,1]$

1. Calculer $f(0)$, $f(1)$, $f(-1)$, $f(-5)$ et $f(5)$.

2. Tracer la fonction $f(x)$ sur l'intervalle $[-1,1]$ en rouge, nommer les axes x et y et donner un titre à la courbe, ajouter une grille à la courbe.

Corrigé de l'exercice 1

On peut proposer le Script suivant pour définir la fonction f :

```
function [rayon,theta]=polaire(x,y)

rayon=sqrt(x^2+y^2);

theta=atan(y/x);

end
```

En appelant notre fonction polaire pour calculer les deux coordonnées polaires d'un point défini par exemple par ses deux coordonnées cartésiennes (0,1), il convient de le faire sous la forme suivante:

```
>> [rayon,theta]=polaire(0,1)

rayon =    1

theta =   1.5708
```

Corrigé de l'exercice 2

1. **On peut proposer le Script suivant pour définir la fonction f :**

```
function [fdex]=fonction_f(x)

fdex=(x^5-3)/sqrt(x^2+1);

end
```

2. **On peut tester la fonction avec les commandes**

```
>> [fdex]=fonction_f(1)

fdex = -1.4142

>> [fdex]=fonction_f(0)

fdex =   -3
```

3. **Le tableau est crée par la commande suivante**

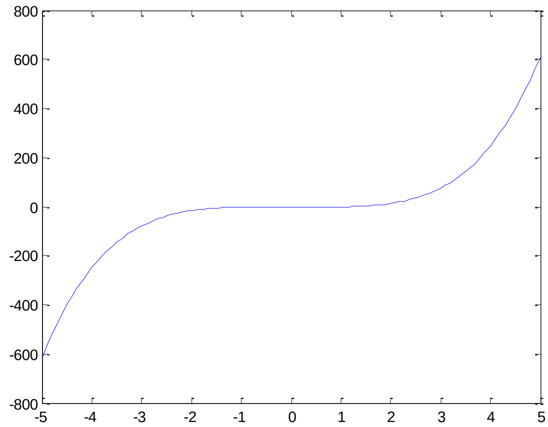
```
>>x=linspace(-5,5,100) ;
```

4. **Pour faire la représentation graphique de la fonction f aux points xi du tableau x créé dans la question précédente, Il suffit de transformer les opérations arithmétiques qui apparaissent dans le Script en opérations terme à terme qui ont la possibilité d'opérer sur des tableaux. Voici la transformation pertinente qu'il faut effectuer au niveau du Script qui définit la fonction f.**

```
function [fdex]=fonction_f(x)

fdex=(x.^5-3)./sqrt(x.^2+1);

end
```

Finalement le tracé de la fonction peut se faire à l'aide des commandes suivantes lesquelles de préférence devraient être écrites dans un script qui appelle la fonction_f et qui demande à *Matlab* de faire la représentation graphique (on pourra l'appeler trace_de_la_fonction):

```
%trace_de_la_fonction

clear all;
close all;
x=linspace(-5,5,100);
[y]=fonction_f(x);
plot(x,y)
```



Corrigé de l'exercice 3

---

1. Le Script définissant la fonction f est alors :

```
function [y]=fonction_ex3(x)

y=exp(sin(x));

end
```

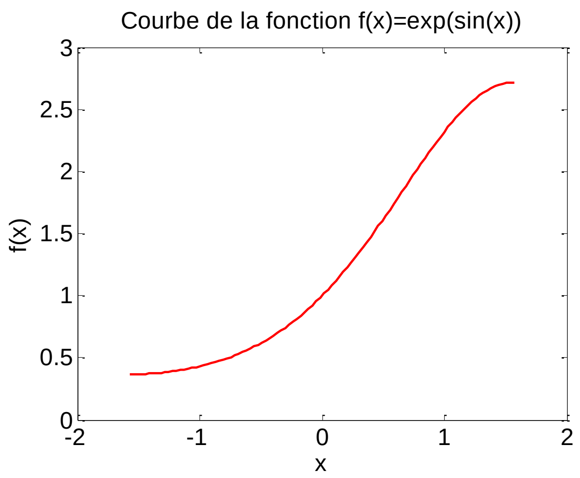
2. On peut tester la fonction avec les commandes

```
 >> [y]=fonction_ex3 (0)

y =

      1

>> [y]=fonction_ex3 (pi/4)

y =

    2.0281

>> [y]=fonction_ex3 (1)

y =

    2.3198
```

3. Le script suivant permet de tracer la courbe de la fonction f sur l'intervalle $[-\pi/2, \pi/2]$

```
clear all;

close all;

x=linspace(-pi/2,pi/2,100);
```

```
[y]=fonction_ex3(x);



plot(x,y,'r-')
xlabel('x')
ylabel('f(x)')
title('Courbe de la fonction f(x)=exp(sin(x))')
```



3. Pour calculer de manière approchée la valeur maximale prise par la fonction $f(x)$ sur l'intervalle $[-\pi/2, \pi/2]$, il suffit de taper la commande suivante :

```
>> max(y)
ans =
   2.7183
```

Corrigé de l'exercice 4

---

1. Le Script définissant la fonction f est alors :

```
function [y]=fonction_ex4(x)

y=exp(-x.^2).*cos(pi.*x/2);

end
```

2. On peut tester la fonction avec les commandes

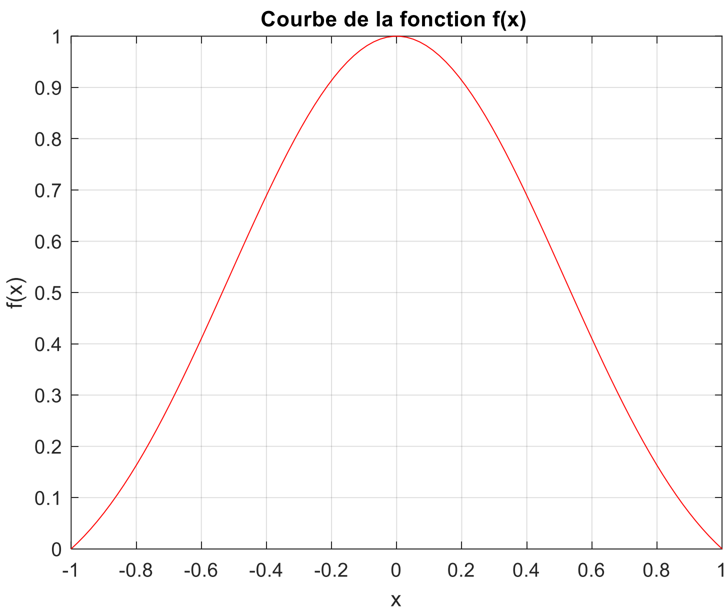
```
[y]=fonction_ex4(0)
y =      1
>> [y]=fonction_ex4(1)
y =    2.2526e-17
>> [y]=fonction_ex4(-1)
```

```
y =  2.2526e-17
```

```
>> [y]=fonction_ex4(-5)
y = 4.2520e-27
>> [y]=fonction_ex4(5)
y =   4.2520e-27
```

3. Pour tracer la courbe de la fonction f(x) sur l'intervalle [-1,1] , on peut utiliser le script suivant :

```
clear all;
close all;
x=linspace(-1,1,100);
[y]=fonction_ex4(x);
plot(x,y,'r')
grid on
xlabel('x')
ylabel('f(x)')
title('Courbe de la fonction f(x) ')
```



Série 3

Exercice 1

On veut écrire un programme qui calcule la racine carrée d'un nombre entré par l'utilisateur et qui affiche le résultat dans une phrase. Le résultat devra être le suivant :

Entrez un nombre: 23 (23 est entré par l'utilisateur)

La racine carrée de 23 est 4.7958 (sortie à l'écran)

Indication: On peut convertir les nombres en chaîne de caractères en utilisant la fonction, num2str.

Exercice 2

1. Ecrire un Script pour la fonction $g(x)=\begin{cases} 0 & x<-1 \\ 1/(e^{-15(x+3/4)}+1) & -1\leq x<-1/2 \\ 1 & -1/2\leq x<3 \\ e^{-15(x-7/2)^2} & 3\leq x<4 \\ 0 & x\geq 4 \end{cases}$

sur l'intervalle $[-2, 5]$

2. Tester la fonction sur quelques valeurs, par exemple $g(-1)=0.0230$, $g(3)=0.0235$.

3. Créer un tableau d'abscisses x de −2 à 5 par pas de 0.01.

4. Représenter la fonction g aux points xi.

Exercice 3

Tracer sur le domaine plan $[-2,2]\times[-3,3]$ la surface répondant à l'équation cartésienne suivante avec la commande ***surf*** et ***mesh.***

$z=e^{-x^2-y^2}$.

Corrigé série 3

Corrigé de l'exercice 1

%La solution qui correspond à la syntaxe exacte est donc :

```
clear all

close all

a = input('Entrez un nombre: ');

b = sqrt(a);

str = ['La racine carrée de ' num2str(a) ' est ' num2str(b)];

disp(str)
```

Corrigé de l'exercice 2

1. **La fonction $x \mapsto g(x)$ est définie par morceaux. Son expression change en fonction de l'intervalle considéré. En choisissant de l'étudier sur l'intervalle [−2, 5], il y a 5 intervalles différents qui partitionnent le domaine d'étude. Sur chacun d'eux la fonction admet une expression différente. On peut aussi définir cette fonction directement à l'aide des fonctions indicatrices des sous intervalles du domaine d'étude. Le Script suivant qui porte le nom *fonction_1.m* permet de définir explicitement la fonction $x \mapsto g(x)$ sur le domaine [−2, 5].**

```
function [y]=fonction_1(x)

c1=x>=-2 & x<-1;

c2=x>=-1 & x<-1/2;

c3=x>=-1/2 & x<3;

c4=x>=3 & x<4;

c5=x>=4 & x<=5;

y=c1*0+c2*(1/(exp(-15*(x+3/4))+1))+c3*1+c4*exp(-15*(x-
7/2)^2)+c5*0;

end
```

2. >> [y]=fonction_1(-1)

```
y =

   0.0230

>> [y]=fonction_1(3)
```

y=

  0.0235

3. >> x=-2:0.01:5;

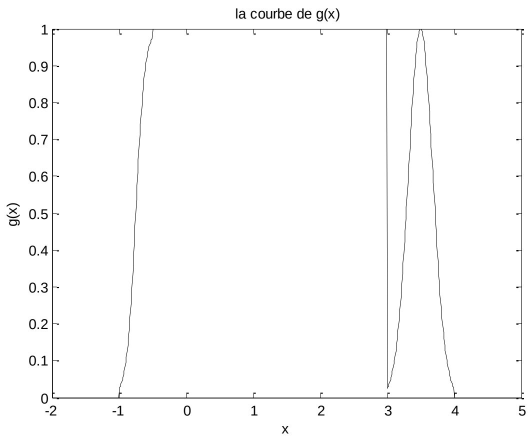
4. **Pour représenter la courbe de la fonction $x \mapsto g(x)$ sur l'intervalle d'étude, il faut donner au Script qui définit cette fonction la possibilité de calculer l'image d'un tableau. Notons à cet effet que les tests logiques définissant les variables logiques $c1$ à $c5$ s'appliquent sans aucun problème à des tableaux. D'où l'adaptation suivante du M-file fonction_1.m**

```
function [y]=fonction_1(x)

c1=x>=-2 & x<-1;

c2=x>=-1 & x<-1/2;

c3=x>=-1/2 & x<3;

c4=x>=3 & x<4;

c5=x>=4 & x<=5;

y=c1.*0+c2.*(1./(exp(-15*(x+3/4))+1))+c3.*1+c4.*exp(-15*(x-
7/2).^2)+c5.*0;

end
```

Il suffit maintenant de créer un script qui appelle la fonction fonction_1 c'est-à-dire g, en permettant de tracer son graphe à l'aide de tableau x.

```
clear all;

close all;

x=-2:0.01:5;

y=fonction_1(x);

plot(x,y,'k')

xlabel('x')

ylabel('g(x)')

title(' la courbe de g(x)')
```

On obtient alors le graphe suivant:



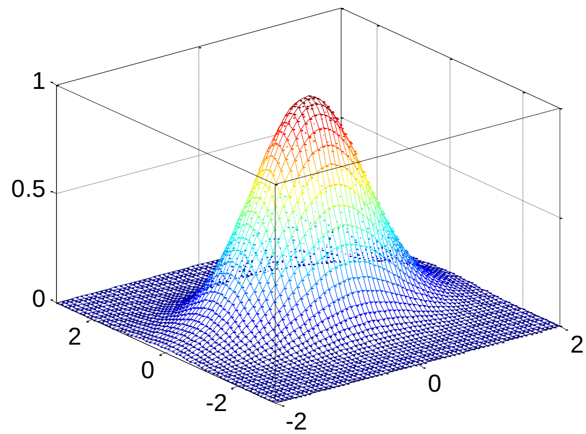
Corrigé de l'exercice 3

Pour tracer la surface $(x,y) \mapsto z = e^{-x^2-y^2}$, il suffit de créer deux tableaux: l'un des abscisses xi sur le domaine $[-2,2]$, l'autre des ordonnées yj sur le domaine $[-3,3]$ et de calculer ensuite la cote zij correspondant à chaque pair (xi, yj) avant de faire la représentation graphique de la surface par la commande *surf* ou *mesh*.

L'ensemble de ces opérations sont regroupées dans le script suivant :

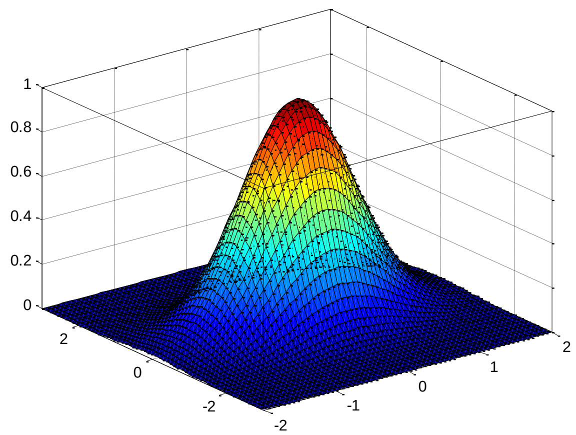
```
clear all;
close all;
N=60;
x=(linspace(-2,2,N))'*ones(1,N);
y=ones(N,1)*linspace(-3,3,N);
z=exp(-x.^2-y.^2);
mesh(x,y,z)
axis([-2 2 -3 3 0 1]);
box;
```

On obtient alors le graphe suivant :



En substituant la commande *surf* à la commande *mesh* , le résultat devient

Noter bien ici l'astuce qui permet de transformer le tableau des abscisses en une matrice contenant toutes les abscisses des points qui discrétisent le domaine $[-2,2]\times[-3,3]$**. Cette matrice admet des lignes identiques. Une autre transformation permet de récupérer une matrice des ordonnées avec cette fois-ci les colonnes qui sont identiques. Le calcul de z peut alors être effectué en bloc à l'aide de l'expression** exp(-x.^2-y.^2) **constante par ligne de la matrice des x et constante par colonne de la matrice des y.**



Série 4

Exercice 1

Traduire en langage Matlab l'algorithme PGCD suivant :

a,b entiers positifs

tant-que $a \neq b$ faire

si $a > b$ alors $a \leftarrow a - b$

sinon $b \leftarrow b - a$

fin tant-que

PGCD$\leftarrow$a

Exercice 2

Ecrire la fonction **vec2col.m** qui transforme tout vecteur (ligne ou colonne) passé en argument en vecteur colonne.

Un traitement d'erreur testera que la variable passée à la fonction est bien un vecteur et non une matrice (utiliser la commande size).

Exercice 3

Ecrire un script qui permet selon le choix de transformer les cordonnées cartésiennes (x,y,z) :

- en coordonnés cylindriques (r,θ,z)
- en coordonnées sphériques (ρ,θ,φ).

Corrigé série 4

Corrigé de l'exercice 1

Pour traduire en langage Matlab l'algorithme PGCD considéré dans cet exercice, nous avons besoin des structures de contrôle, en l'occurrence la boucle *while* et le test *if-else-end*. Nous proposons le M-file suivant pour le programme traduisant cet algorithme:

```
clear all;
close all;
a=input('Entrer l''entier positif  a   ');
b=input('Entrer l''entier positif  b   ');
a0=a;
b0=b;
while a~=b
    if a>b
        a=a-b;
    else
        b=b-a;
    end
end
str=['Le plus grand diviseur commun de  ' num2str(a0) '  et  '
num2str(b0) '  est  ' num2str(a)];
disp(str);
```

***On a fait appel aux variables auxiliaires a0 et b0 pour stocker les valeurs initiales de a et b qui sont modifiées dans la boucle *while* et ne gardent pas donc leurs valeurs d'entrées à la sortie de cette boucle. Elles sont en fait écrasées dans les lignes 9 et 11.**

```
Entrer l'entier positif  a   8
Entrer l'entier positif  b   6
Le plus grand diviseur commun de  8  et  6  est  2
```

Corrigé de l'exercice 2

On propose le M-file de type fonction suivant :

```
function [y]=vec2col(x)
n=size(x);
n1=n(1);
n2=n(2);
if n1==1
    y=x.';
elseif n2==1
    y=x;
else
    erreur=['la variable entrée n''est pas un vecteur ligne ou colonne'];
    display(erreur)
end
```

On peut le tester avec les commandes

```
>> vec2col([1 1 1])

ans =

   1

   1

   1

>> vec2col([1; 1; 1])

ans =

   1

   1

   1
```

>> vec2col([1 1; 1 2])

erreur =

la variable entrée n'est pas un vecteur ligne ou colonne

Corrigé de l'exercice 3

Un exemple de script répondant à l'énoncé est donné dans la suite. Remarquer que l'on sort les coordonnées cylindriques selon l'ordre : (r,θ,z) et les coordonnées sphériques selon l'ordre (r,θ,φ).

```
function [a,b,c]=transformation(x,y,z,choix)
switch choix
    case 'cylindrique'
        a=sqrt(x^2+y^2);
        b=atan(y/x);
        c=z;
    case 'sphérique'
        a=sqrt(x^2+y^2+z^2);
        b=atan(y/x);
        c=atan(sqrt(x^2+y^2)/z);
    otherwise
        display('Vous n''avez pas préciser la nature de la
tansformation')
end
end
```

Exemples d'utilisation :

>> [a,b,c]=transformation(1,1,1,'cylindrique')

a =

1.4142

b =

0.7854

```
c = 1
```

%

```
>> [a,b,c]=transformation(2,3,4,'sphérique')

a =

   5.3852

b =

   0.9828

c =

   0.7336
```

Avec *Matlab* vous avez quatre commandes qui vous permettent de faire la transformation des coordonnées cartésiennes en coordonnées cylindriques ou bien des coordonnées cartésiennes en coordonnées sphériques, ainsi que leurs inverses.

```
[theta,rho,z]=cart2pol(x,y,z) % transforamtion catésiennes en cylindriques

[x,y,z]=pol2cart(theta,rho,z) % transforamtion cylindriques en cartésiennes

[theta,phi,r]=cart2pol(x,y,z) % transforamtion catésiennes en sphériques

[x,y,z]=sph2cart(theta,phi,r) % transforamtion sphériques en catésiennes
```

Attention:

*** Ici tous les angles sont calculés en radians (*Matlab* utilise cette unité par défaut).**

*** *Matlab* sort les coordonnées cylindriques selon l'ordre (θ, r, z) et les coordonnées sphériques selon l'ordre (θ, φ, r).**