



Université Abdelmalek Essaadi
École Nationale des Sciences Appliquées Al Hoceima



Cours d' *Informatique 3*: **MATLAB**

MATLAB POUR L'INGÉNIEUR

AP-2

Chapitre 6

Les Macros

Pr. Amina GHADBAN

Amina GHADBAN

Programmation : Macro (1/12)

➔ Fonctions ou macros : *function*

- Les fonctions sont des enchaînements de commandes Matlab, elles sont regroupées sous un nom de fonction permettant de commander leur exécution.
- Les macros peuvent contenir un groupe de commandes destiné à être exécuté plusieurs fois au cours du calcul avec éventuellement des valeurs de paramètres différents.
- La fonction peut aussi être chargée de réaliser un calcul avec un certain algorithme.
- Une macro est un script :
 - qui reçoit des arguments en entrée.
 - qui renvoie des résultats.

Programmation : Macro (2/12)

➔ Fonctions ou macros : *function*

- Pour des programmes longs et compliqués, il est souhaitable de les découper en fonctions, correspondant à des étapes afin d'améliorer la lisibilité et la compréhension de l'algorithme.
- Les fonctions sont écrites et enregistrées dans un fichier avec une extension '**.m**' portant le nom de la fonction.
- Les macros peuvent être appelées une fois définies, comme toutes les autres fonctions qui existent dans la boîte à outils de Matlab.

Programmation : Macro (3/12)

➔ Fonctions ou macros : *function*

- La syntaxe de définition d'une fonction externe est :

function *arguments de sortie* = *Nom de la fonction*(*arguments d'entrée*)

function [y_1, \dots, y_m] = *nom*(x_1, \dots, x_n)

Avec:

'**nom**' est le nom de la fonction (on sauvegarde '**nom.m**').

' x_1, \dots, x_n ' les arguments d'entrée.

' y_1, \dots, y_m ' les arguments de sortie.

Programmation : Macro (4/12)

Fonctions ou macros : *function*

- Le nom du fichier contenant la fonction porte obligatoirement le nom de cette dernière.

- Les fonctions peuvent être appelées à partir de :
 - * éditeur de commandes
 - * une autre fonction
 - * un script

Programmation : Macro (5/12)



Fonctions ou macros : *function*

- Exemple :

```
function y=fact(n)
    y=prod(1:n); %calcul de n!
```

Une fois sauvegardé sous le nom 'fact.m', sur l'espace de commande on tape :

```
>> fact(3)
```

```
ans =
```

```
6
```

```
>> fact(5)
```

```
ans =
```

```
120
```

Programmation : Macro (6/12)

➔ Fonctions ou macros : *function*

■ Exemple:

* Coordonnées polaires:

function [r,theta] = polaire(x,y)

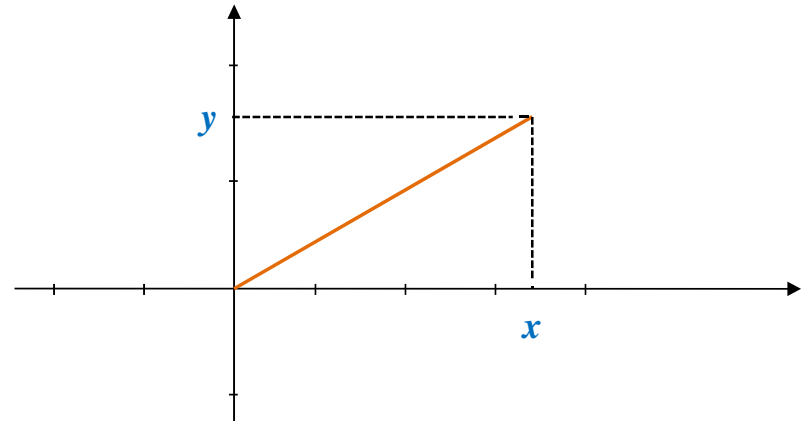
$r = \text{sqrt}(x^2+y^2);$

$\theta = \text{atan}(y/x);$

Dans l'éditeur de commande, on tape:

[r,t]=polaire(3,4) ,

[r,t]=polaire(2.5,-1) , ...



Programmation : Macro

(7/12)



Fonctions ou macros : *function*



Exemple:

* **Coordonnées polaires :**

Dans l'éditeur de commande, on tape:

```
[r,theta] =  
polaire(3,4)
```

```
r =
```

```
5
```

```
theta =
```

```
0.9273
```

```
[r,theta] =  
polaire(2,7)
```

```
r =
```

```
7.2801
```

```
theta =
```

```
1.2925
```


Programmation : Macro (8/12)

➔ Fonctions ou macros : *function*

■ Exemple :

function [S,V] = SurfVolCylindre(r,h)

S = pi*2*r*h;

V = pi*r^2*h;

Exécution:

[S1,V1] = SurfVolCylindre(3,5)

S1 =

94.2478

V1 =

141.3717

[S2,V2] =
SurfVolCylindre(10,8)

S2 =

502.6548

V2

2.5133e+003

Programmation : Macro (9/12)

➔ Fonctions ou macros :
function

■ Exemple : 1^{er} script : *Cal2Joule.m*

```
function Cal = Cal2Joule(j)
```

```
Cal = 4.18*j; %1 Calorie = 4.18 joules
```

```
%2ème script ConversionEnergie.m
```

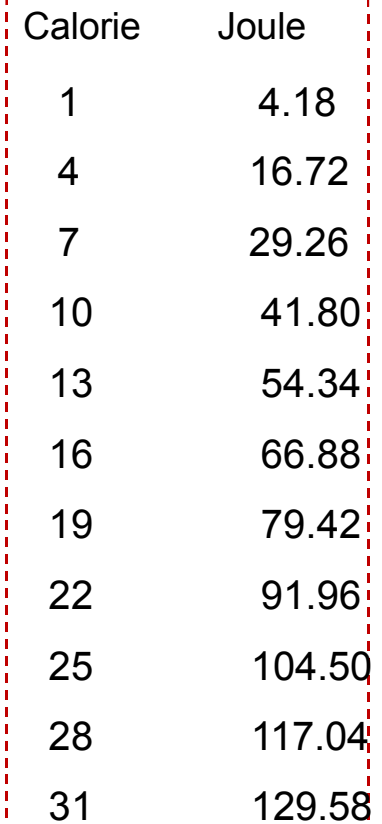
```
disp(['Calorie',blanks(10),'Joule']);
```

```
for j = 1 : 3 : 31
```

```
    fprintf('%4d %17.2f\n',j,Cal2Joule(j));
```

```
end
```

```
%Après exécution du 2ème script dans l'espace de commande
```



Calorie	Joule
1	4.18
4	16.72
7	29.26
10	41.80
13	54.34
16	66.88
19	79.42
22	91.96
25	104.50
28	117.04
31	129.58

Programmation : Macro (10/12)

➔ Fonctions ou macros : *function*

```
function y = f2(x)
if (x < 0)
    y = sin(2*x);
elseif ((x >= 0) & (x < 3))
    y = exp(-x)*sin(2*x);
else (x >= 3)
    y = sqrt(sin(2*x)+1);
end
```

```
>> f2(-5)
```

```
ans =
    0.544
     0
```

```
>> f2(-1)
```

```
ans =
 -0.9093
```

```
>> f2(0)
```

```
ans =
     0
```

```
>> f2(1)
```

```
ans =
    0.334
     5
```

```
>> f2(1.7)
```

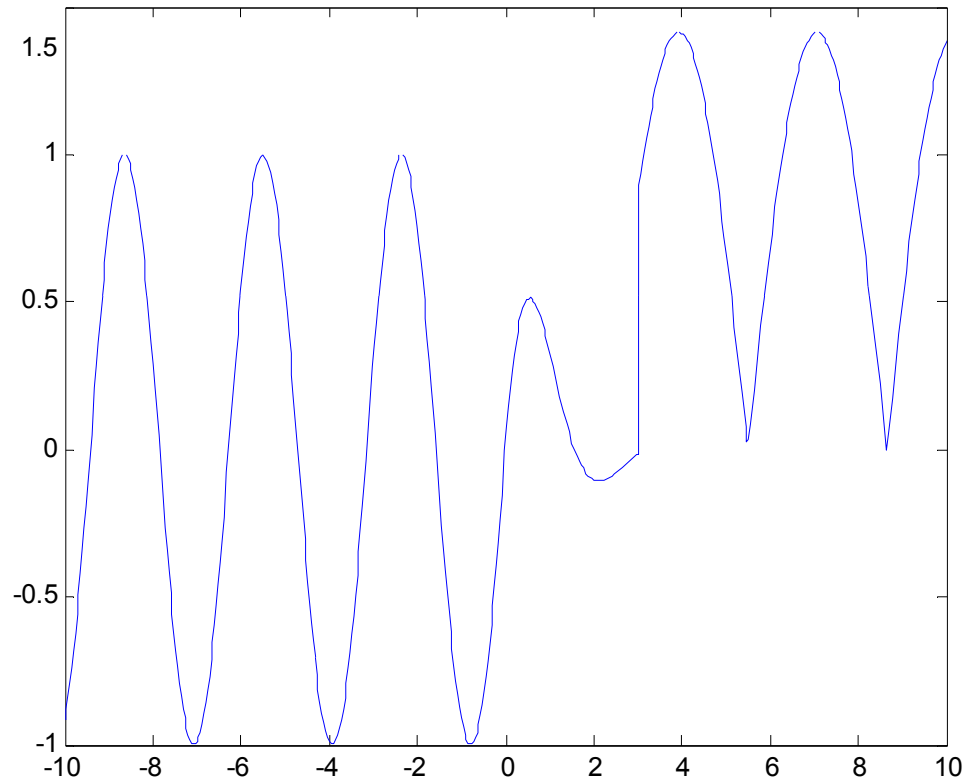
```
ans =
 -0.0467
```

```
>> f2(5)
```

```
ans =
    0.6753
```

➔ Fonctions ou macros :
function

```
>> fplot('f2',[-10 10]);
```





Cours d' *Informatique 3*: **MATLAB**

MATLAB POUR L'INGÉNIEUR

AP-2

Fin de la partie 2

Pr. *Amina GHADBAN*