MATLAB

Essential MATLAB for Scientists

Chap 4 : Functions

Common functions

⇨Plot a graph

plot(x,y,color), hist(x)

⇨Generate a vector or a matrix

rand(1,10), linspace(1,5,10)

⇨Mathematic calculation

sqrt(4), sum(x), log(20)

⇨Conversion of numerical and string data

num2str(10), str2num('10')

⇨ Load data

data=xlsread('mytable.xlsx')

Read data from excel

Create a table on excel with 3 columns:

Column A could be values from 1 to 60

Column B could be values from column A * 5

Column C could be strings (Data1, Data2, Data3 etc…)

In the command window:
data=xlsread('mytable.xlsx')

Read data from excel

In the command window:
Error using xlsread
Unable to open file 'values.xlsx'.
File 'C:\Users\delphine.syvilay\Documents\MATLAB\values.xlsx' not found.

In the command window:
cd('C:\Users\delphine.syvilay\Desktop')
data=xlsread('mytable.xlsx');
size(data) *You will get only numerical values*

[data,txt]=xlsread('mytable.xlsx'); *You will get numerical values and string*
size(txt)
txt(1,1)

Mathematic function

$f(x)=ax+b$ Affine function

$f(x)=ax^2+bx+c$ Quadratic function

$f(x)=\cos(x)$ Cosinus function

…

In matlab, it is the same, you can create your own function with calculations that you want to operate

Create your own function

- ⇨When you don’t want to repeat the same command lines
- ⇨When you don’t want to clear the values of your variables if you changed a parameter
- ⇨When your program is too long

Create your own function

⇨Syntax:

Output variables
Will be generated from your calculations
You can add as many as you want, if they are generated during operations

Input variables
Will be used for calculations
You can add as many as you want, if they are used during operations

```
function [x,y]=myfunct(param1,param2)
        x=
        y=

end
```

calculation that you want to operate

Create your own function

⇨Example:

%% This function is going to generate an affine function with 2 parameters: param1 and param2. x vector will be generated thanks to param3 and y vector will be calculated.

```
function [x,y]=myfunct(param1,param2,param3)
        x=1:param3
        y=param1*x+param2
End
```

Write these command lines in a new script, and save it as .m file

!!!The function name and the file name MUST be the same

Create your own function

Command window

```
myfunct(2,3,5)
```

You will get x vector

```
[x,y]= myfunct(2,3,5)
```

You will get x vector and y vector

Script file

```
%% This function is going to generate an affine function with 2 parameters: param1 and param2. x vector will be generated thanks to param3 and y vector will be calculated.

function [x,y]=myfunct(param1,param2,param3)
        x=1:param3;
        y=param1*x+param2;
end
```

Create your own function

Command window

myfunct(2,3,5)

You will get x vector

[x,y]= myfunct(2,3,5)

You will get x vector and y vector

[z,w]= myfunct(4,5,6)

Try with other output variable names like z and w

You will still keep x,y values and you will generate new vectors (z and w)

Script file

```
%% This function is going to generate an affine function with 2 parameters: param1 and param2. x vector will be generated thanks to param3 and y vector will be calculated.

function [x,y]=myfunct(param1,param2,param3)
        x=1:param3;
        y=param1*x+param2;
end
```

Create your own function

Command window

```
param1=2;
param2=3;
param3=5;
[x,y]= myfunct(param1,param2,param3)
```

It is another way to use the function with the names of your input variables

```
A=4;
B=5;
T=6;
[z,w]= myfunct(A,B,T)
```

You can see that you are not obliged to use the same name for input variables than those used to create your own function

Script file

```
%% This function is going to generate an affine function with 2 parameters: param1 and param2. x vector will be generated thanks to param3 and y vector will be calculated.

function [x,y]=myfunct(param1,param2,param3)
        x=1:param3;
        y=param1*x+param2;
end
```

Create your own function

Command window

```
param1=2;
param2=3;
param3=5;
[x,y]= myfunct(param1,param2,param3)

A=4;
B=5;
T=6;
[z,w]= myfunct(A,B,T)

figure;
plot(x,y,'r');
hold on
plot(z,w,'b');
```

If you forget hold on, it will overwrite your first plot

Script file

```
%% This function is going to generate an affine function with 2 parameters: param1 and param2. x vector will be generated thanks to param3 and y vector will be calculated.

function [x,y]=myfunct(param1,param2,param3)
        x=1:param3;
        y=param1*x+param2;
end
```

Create your own function

Command window

```
param1=2;
param2=3;
param3=5;
[x,y]=
myfunct(param1,param2,param3,[0 0 1])
hold on
A=4;
B=5;
T=6;
[z,w]= myfunct(A,B,T,[0 1 1])
```

Script file

```
%% This function is going to generate an affine function with 2 parameters: param1 and param2. x vector will be generated thanks to param3 and y vector will be calculated.

function [x,y]=myfunct(param1,param2,param3, c)
        x=1:param3;
        y=param1*x+param2;
plot(x,y,'color',c)
end
```

Change the script, add a new parameter

Function inside for loop

Command window

```
param1=[1:1:20];
param2=[-1:0.5:8.5];
param3=50;
cm = parula(length(param1));

for i=1:length(param1)
[x{i},y{i}]=myfunct(param1(i),param2(i),p
aram3)
hold on
plot(x{i},y{i},'color',cm(i,:));
end
```

Script file

```
%% This function is going to generate an affine
function with 2 parameters: param1 and param2. x
vector will be generated thanks to param3 and y
vector will be calculated.

function [x,y]=myfunct(param1,param2,param3)
        x=1:param3;
        y=param1*x+param2;
end
```

Example: Calculate the area and the perimeter of a circle

Create a function to calculate the area and the perimeter of a circle where the radius is an input.

On a matlab script:
-The user is asked to enter the radius (m).
Use the function created.

-Change the matlab script to ask to the user 5 times.

-Change the function by adding a condition (if the radius is less than 1m, display « Radius is too small ».

Example: 2 equations with 2 unknowns

$$\begin{cases} -2x+3y=8 \\ 4x+10y=16 \end{cases}$$

Create a function which can help you to find the solutions if you don't know how to resolve it.

No hint: **20/20**

1st hint: **17/20**

2nd hint: **14/20**

3rd hint: **11/20**

Example: 2 equations with 2 unknowns

$$\begin{cases} -2x+3y=8 \\ 4x+10y=16 \end{cases}$$

Create a function which can help you to find the solutions if you don't know how to resolve it.

1st hint:
Create x vector [-5 5]
Create y vector [-5 5]

$$\begin{cases} a1\,x+b1\,y=c1 \\ a2\,x+b2\,y=c2 \end{cases}$$

Example: 2 equations with 2 unknowns

$$\begin{cases} -2x+3y=8 \\ 4x+10y=16 \end{cases}$$

Create a function which can help you to find the solutions if you don't know how to resolve it.

1st hint:
Create x vector [-5 5]
Create y vector [-5 5]

2nd hint:

Use for loop
Use if decision

$$\begin{cases} a1\,x+b1\,y=c1 \\ a2\,x+b2\,y=c2 \end{cases}$$

Example: 2 equations with 2 unknowns

$$\begin{cases} -2x+3y=8 \\ 4x+10y=16 \end{cases}$$

Create a function which can help you to find the solutions if you don't know how to resolve it.

1st hint:
Create x vector [-5 5]
Create y vector [-5 5]

2nd hint:

Use for loop
Use if decision

3rd hint:

Use **2** for loop

$$\begin{cases} a1\,x+b1\,y=c1 \\ a2\,x+b2\,y=c2 \end{cases}$$

Example: 2 equations with 2 unknowns

Command window

```
x=[-5:1:5];
y=[-5:1:5];
a1=-2;b1=3;c1=8;
a2=4;b2=10;c2=16;
[xsolution,ysolution]=c
alculate_solutions(a1,
b1,c1,a2,b2,c2,x,y)
```

Script file

```
function[xsolution,ysolution]=calculate_solutions(a1,b1,c1,a2,b2,c2,
x,y)
for i=1:length(x)
        for j=1:length(y)
                res_1st_eq=a1*x(i)+b1*y(j);
                res_2nd_eq=a2*x(i)+b2*y(j);
                if res_1st_eq==c1 && res_2nd_eq==c2
                        xsolution=x(i);
                        ysolution=y(j);
                end
        end
end
end
```