

# Initiation à la programmation avec Octave

Alexandre Mayer

Laboratoire de Physique du Solide

Université de Namur

Rue de Bruxelles 61, B-5000 Namur (Belgique)

[alexandre.mayer@unamur.be](mailto:alexandre.mayer@unamur.be)

---

<http://perso.fundp.ac.be/~amayer>

# Table des matières

- ◆ Pourquoi Octave ?
- ◆ Installation
- ◆ Utilisation élémentaire
- ◆ Programmes
- ◆ Opérations élémentaires de lecture et d'écriture
- ◆ Structures de contrôle
- ◆ Types de variables

# Table des matières

- ◆ Tableaux
- ◆ Fichiers
- ◆ Procédures
- ◆ Algèbre linéaire
- ◆ Interpolations
- ◆ Intégration
- ◆ Optimisation
- ◆ Systèmes d'équations différentielles

# Table des matières

- ◆ Transformées de Fourier
- ◆ Solutions d'équations non linéaires
- ◆ Graphiques (réalisations plus avancées)
- ◆ Thèmes divers et avancés

# Pourquoi Octave ?

- ◆ Octave est un logiciel libre et équivalent à Matlab.
- ◆ Matlab est de plus en plus utilisé dans le monde scientifique.
- ◆ Ce langage est moins contraignant que le Fortran 90.
- ◆ Les codes sont efficaces grâce à l'utilisation de librairies précompilées.
- ◆ On peut faire facilement tout ce dont on a besoin (graphiques, résolution de systèmes, etc).
- ◆ C'est un bon langage pour apprendre la programmation.

# Installation

<http://www.gnu.org/software/octave/index.html>

The screenshot shows a Firefox browser window with the URL [www.gnu.org/software/octave/index.html](http://www.gnu.org/software/octave/index.html) in the address bar. The page content is the GNU Octave homepage. At the top, there's a navigation menu with links for Home, About, Download, Support, and Get Involved. Below that is a 'Donate' section with a form for amount and payment method selection (Pay with credit card or Pay with PayPal). A note below the form states: "Following the Continue link will take you to a Free Software Foundation page for payment processing." The main content area features a title "GNU Octave" with a logo, two screenshots of the software interface showing code and plots, and a detailed description of what GNU Octave is. There are also news sections for OctConf 2013, SOCIS mentorship, and a recent hiring announcement.

**GNU Octave**

GNU Octave is a high-level interpreted language, primarily intended for numerical computations. It provides capabilities for the numerical solution of linear and nonlinear problems, and for performing other numerical experiments. It also provides extensive graphics capabilities for data visualization and manipulation. Octave is normally used through its interactive command line interface, but it can also be used to write non-interactive programs. The Octave language is quite similar to Matlab so that most programs are easily portable.

Octave is distributed under the terms of the [GNU General Public License](#).

**News**

**January 28, 2013**  
OctConf 2013 will take place in Milano, Italy during June 24-26, 2013. All Octave users, developers, and enthusiasts are welcome to attend!

**August 1, 2012**  
ESA approved two students to work for Octave under the SOCIS mentorship programme! Congratulations to Wendy Liu and Andrius Sutis who will be working on finishing the Agora website and low-level I/O respectively.

**July 18, 2012**  
Octave has been accepted into SOCIS. We're now hiring for one position!

# Download GNU Octave



GNU Octave 3.6.2 was released May 31, 2012.

## GNU/Linux systems

Executable versions of Octave for GNU/Linux systems are provided by the individual distributions. Distributions known to package Octave include: [Debian](#), [Fedora](#), [Gentoo](#), and [SuSE](#). These packages are created by volunteers. The delay between an Octave source release and the availability of a package for a particular GNU/Linux distribution varies. The Octave project has no control over that process.

## BSD systems

Executable versions of Octave for BSD systems are provided by the individual distributions. Both [FreeBSD](#) and [OpenBSD](#) have Octave packages. These packages are created by volunteers. The delay between an Octave source release and the availability of a package for a particular GNU/Linux distribution varies. The Octave project has no control over that process.

## OS X

The Wiki has some instructions for [installing Octave on OS X systems](#).

## Windows

### Cygwin

There is an [Octave package for Cygwin](#).

### MinGW

The Wiki has some instructions for [installing Octave on Windows systems](#).

## Sources

The latest released version of Octave is always available from <ftp://ftp.gnu.org/gnu/octave>.

If you are interested in working with the latest version of the Octave sources, check out the [resources for developers](#).

[Home](#)

[About](#)

[Download](#)

[Support](#)

[Get Involved](#)

### Donate

Your donations help to fund continuing maintenance tasks, development of new features and the organization of Octave conferences.

Amount (USD)

\$ 50.00

Pay with credit card

Pay with PayPal

[Continue...](#)

Following the Continue link will take you to a Free Software Foundation page for payment processing.

<http://perso.fundp.ac.be/~amayer/cours/Octave>

The screenshot shows a Firefox browser window with the title bar "Initiation à la programmation avec Octave". The address bar displays the URL "perso.fundp.ac.be/~amayer/cours/Octave/". The page content is titled "Initiation à la programmation avec Octave" by Alexandre Mayer. It includes sections for "Notes de cours", "Installation sous Windows", and "Références", along with links to documentation and installation instructions. A small cartoon image of Snoopy sitting on his doghouse is visible at the bottom.

**Initiation à la programmation avec Octave**

Alexandre Mayer

**Notes de cours**

[Initiation à la programmation avec Octave](#)

Pour une information complète sur Octave, vous pouvez également consulter la [documentation](#).

**Installation sous Windows**

[Instructions pour installer Octave sous Windows](#)

**Références**

- Le site de [GNU Octave](#).
- Le site du système d'exploitation [UBUNTU](#).



# Utilisation élémentaire

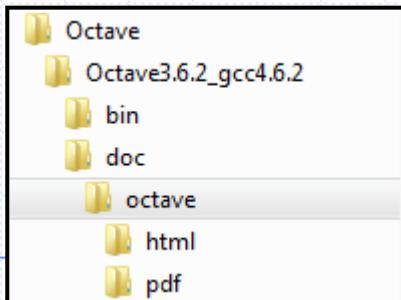
## Chapitre 1



→

Octave

GNU Octave, version 3.6.2  
Copyright (C) 2012 John W. Eaton and others.  
This is free software; see the source code for copying conditions.  
There is ABSOLUTELY NO WARRANTY; not even for MERCHANTABILITY or  
FITNESS FOR A PARTICULAR PURPOSE. For details, type 'warranty'.  
  
Octave was configured for "i686-pc-mingw32".  
  
Additional information about Octave is available at <http://www.octave.org>.  
  
Please contribute if you find this software useful.  
For more information, visit <http://www.octave.org/help-wanted.html>  
  
Read <http://www.octave.org/bugs.html> to learn how to submit bug reports.  
  
For information about changes from previous versions, type 'news'.  
  
warning: gmsh does not seem to be present some functionalities will be disabled  
warning: dx does not seem to be present some functionalities will be disabled  
warning: function C:\Octave\Octave3.6.2\_gcc4.6.2\share\octave\packages\statistics-1.1.3\fstat.m shadows a core library function  
octave:1> \_



Nom	Modifié le
liboctave.html	9/06/2012 19:30
octave.html	9/06/2012 19:29
OctaveFAQ.html	9/06/2012 19:27

Nom	Modifié le	Type	Taille
liboctave.pdf	31/05/2012 17:42	Adobe Acrobat D...	284 Ko
octave.pdf	6/06/2012 22:56	Adobe Acrobat D...	3.922 Ko
OctaveFAQ.pdf	31/05/2012 17:33	Adobe Acrobat D...	232 Ko
refcard-a4.pdf	31/05/2012 17:42	Adobe Acrobat D...	128 Ko
refcard-legal.pdf	31/05/2012 17:42	Adobe Acrobat D...	128 Ko
refcard-letter.pdf	31/05/2012 17:42	Adobe Acrobat D...	128 Ko

octave.pdf - Adobe Reader

Fichier Édition Affichage Fenêtre Aide

Signets

- Preface
- A Brief Introduction to Octave
- Getting Started
- Data Types
- Numeric Data Types
- Strings
- Data Containers
- Variables
- Expressions
- Evaluation
- Statements
- Functions and Scripts
- Errors and Warnings
- Debugging
- Input and Output
- Plotting
- Matrix Manipulation
- Arithmetic
- Linear Algebra
- Vectorization and Faster Code Execution
- Nonlinear Equations
- Diagonal and Permutation Matrices
- Sparse Matrices
- Numerical Integration

**GNU Octave**

A high-level interactive language for numerical computations  
Edition 3 for Octave version 3.6.2  
February 2011

Free Your Numbers

octave.pdf - Adobe Reader

Fichier Edition Affichage Fenêtre Aide

Signets

- + Preface
- + A Brief Introduction to Octave
- + Getting Started
- + Data Types
- + Numeric Data Types
- + Strings
- + Data Containers
- + Variables
- + Expressions
- + Evaluation
- + Statements
- + Functions and Scripts
- + Errors and Warnings
- + Debugging
- + Input and Output
- + Plotting
- + Matrix Manipulation
- + Arithmetic
- + Linear Algebra
- + Vectorization and Faster Code Execution
- + Nonlinear Equations
- + Diagonal and Permutation Matrices
- + Sparse Matrices
- + Numerical Integration

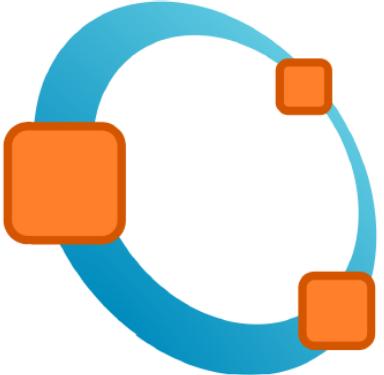
1 / 796 141% 100% 125% 150% 175% 200% 250% 300% 350% 400% 450% 500% 550% 600% 650% 700% 750% 800% 850% 900% 950% 1000% 1050% 1100% 1150% 1200% 1250% 1300% 1350% 1400% 1450% 1500% 1550% 1600% 1650% 1700% 1750% 1800% 1850% 1900% 1950% 2000% 2050% 2100% 2150% 2200% 2250% 2300% 2350% 2400% 2450% 2500% 2550% 2600% 2650% 2700% 2750% 2800% 2850% 2900% 2950% 3000% 3050% 3100% 3150% 3200% 3250% 3300% 3350% 3400% 3450% 3500% 3550% 3600% 3650% 3700% 3750% 3800% 3850% 3900% 3950% 4000% 4050% 4100% 4150% 4200% 4250% 4300% 4350% 4400% 4450% 4500% 4550% 4600% 4650% 4700% 4750% 4800% 4850% 4900% 4950% 5000% 5050% 5100% 5150% 5200% 5250% 5300% 5350% 5400% 5450% 5500% 5550% 5600% 5650% 5700% 5750% 5800% 5850% 5900% 5950% 6000% 6050% 6100% 6150% 6200% 6250% 6300% 6350% 6400% 6450% 6500% 6550% 6600% 6650% 6700% 6750% 6800% 6850% 6900% 6950% 7000% 7050% 7100% 7150% 7200% 7250% 7300% 7350% 7400% 7450% 7500% 7550% 7600% 7650% 7700% 7750% 7800% 7850% 7900% 7950% 8000%

Outils | Commentaire

# GNU Octave

---

A high-level interactive language for numerical computations  
Edition 3 for Octave version 3.6.2  
February 2011



Free Your Numbers

# help

```
Octave
octave:2> help plot
'plot' is a function from the file C:\Octave\Octave3.6.2_gcc4.6.2\share\octave\3.6.2\m\plot\plot.m

-- Function File: plot (Y)
-- Function File: plot (X, Y)
-- Function File: plot (X, Y, PROPERTY, VALUE, ...)
-- Function File: plot (X, Y, FMT)
-- Function File: plot (H, ...)
-- Function File: H = plot (...)
    Produce two-dimensional plots.

Many different combinations of arguments are possible. The
simplest form is

    plot (Y)

where the argument is taken as the set of Y coordinates and the X
coordinates are taken to be the indices of the elements starting
with 1.

To save a plot, in one of several image formats such as PostScript
or PNG, use the 'print' command.

If more than one argument is given, they are interpreted as

    plot (Y, PROPERTY, VALUE, ...)

or

    plot (X, Y, PROPERTY, VALUE, ...)

or

    plot (X, Y, FMT, ...)

and so on. Any number of argument sets may appear. The X and Y
values are interpreted as follows:

* If a single data argument is supplied, it is taken as the set
  of Y coordinates and the X coordinates are taken to be the
  indices of the elements, starting with 1.

* If the X is a vector and Y is a matrix, then the columns (or
  rows) of Y are plotted versus X. (using whichever
  combination matches, with columns tried first.)

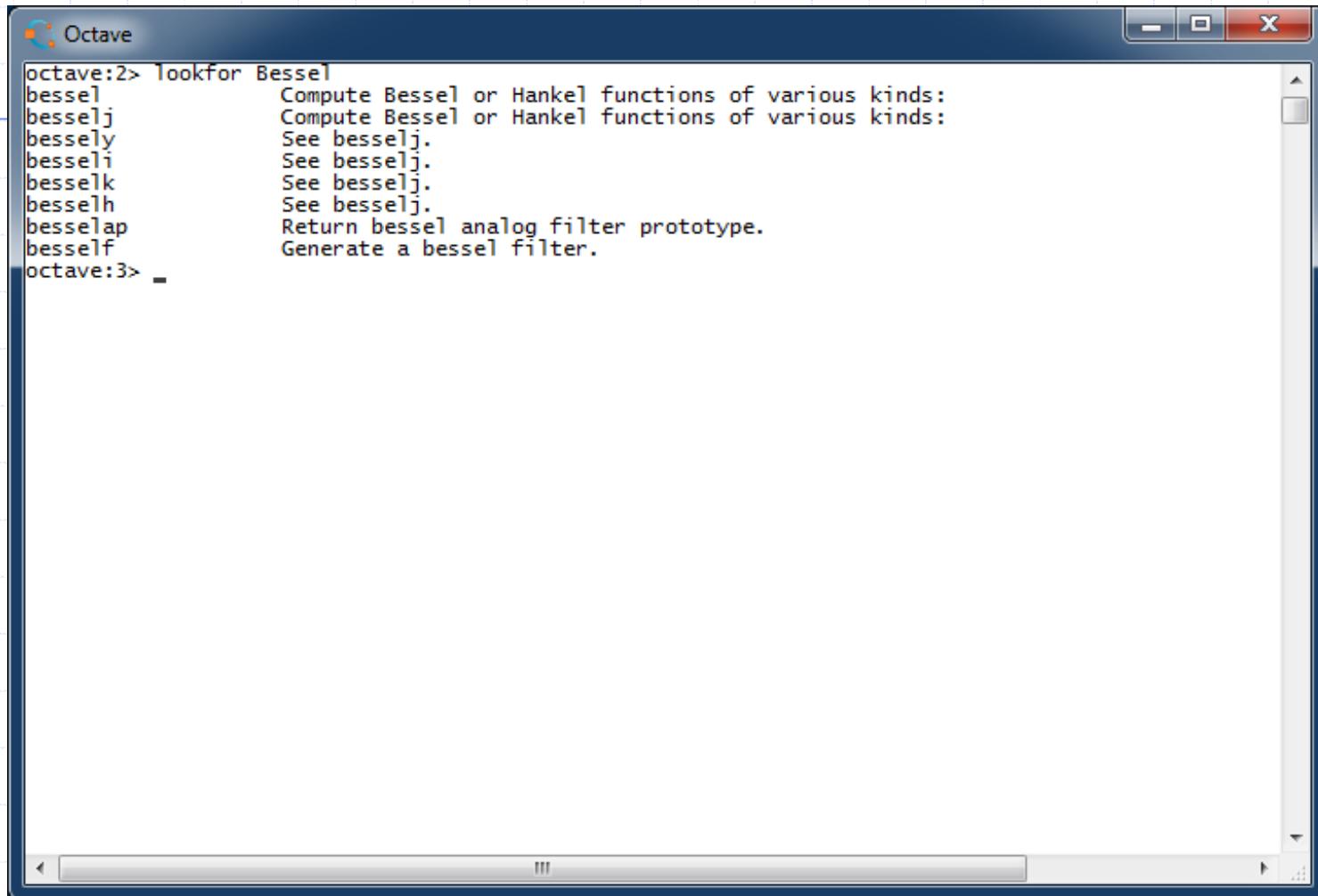
* If the X is a matrix and Y is a vector, Y is plotted versus
  the columns (or rows) of X. (using whichever combination
  matches, with columns tried first.)

* If both arguments are vectors, the elements of Y are plotted
  versus the elements of X.

* If both arguments are matrices, the columns of Y are plotted
  versus the columns of X. In this case, both matrices must
  have the same number of rows and columns and no attempt is
  made to transpose the arguments to make the number of rows
  match.

If both arguments are scalars, a single point is plotted.
```

# lookfor



The screenshot shows a terminal window titled "Octave" with the following command and output:

```
octave:2> lookfor Bessel
bessel          Compute Bessel or Hankel functions of various kinds:
besselj         Compute Bessel or Hankel functions of various kinds:
bessely         See besselj.
besseli         See besselj.
besselk         See besselj.
besselh         See besselj.
besselap        Return bessel analog filter prototype.
besself         Generate a bessel filter.
octave:3> _
```

# Commandes de base

clc	effacer l'écran
clear	effacer toutes les variables
whos	afficher toutes les variables
exist	vérifier l'existence d'un objet
format short	affichage avec 5 chiffres significatifs
format short e	format scientifique
format long	affichage avec 15 chiffres significatifs
format long e	format scientifique
exit	sortir d'Octave

# Affectation

```
x=2  
x=2;
```

% commentaire  
% affectation sans affichage

Il n'est pas nécessaire de déclarer les variables qu'on utilise.

Octave fait la différence entre les majuscules et les minuscules. Une variable appelée « X » est différente d'une variable appelée « x ».

Opérations élémentaires : + - \* / ^

Pour continuer une instruction sur la ligne suivante, on termine la ligne en cours par « ... ».

# Affectation

`x=2;`

 `x=x+2;`

 `x=x+2;`

`...`

% x est égal à 2

% x est égal à 4

% x est égal à 6

Le signe « = » doit s'interpréter en informatique comme « prend la valeur de ». Octave commence par évaluer l'expression située à droite du signe =. Le résultat obtenu est alors placé dans la variable située à gauche du signe =.

Dans une expression telle que « `x=x+2` », on utilise la valeur de x et on lui ajoute 2. Le résultat obtenu remplace ensuite la valeur de x.

## Octave

```
octave:3> x=2
x = 2
octave:4> y=3;
octave:5> whos
Variables in the current scope:
```

Attr	Name	Size	Bytes	Class
====	====	=====	=====	=====
	x	1x1	8	double
	y	1x1	8	double

```
Total is 2 elements using 16 bytes
```

```
octave:6> x
x = 2
octave:7> y
y = 3
octave:8> x*y
ans = 6
octave:9> -
```

# Constantes

pi

3.1416

e

2.7183

i et j

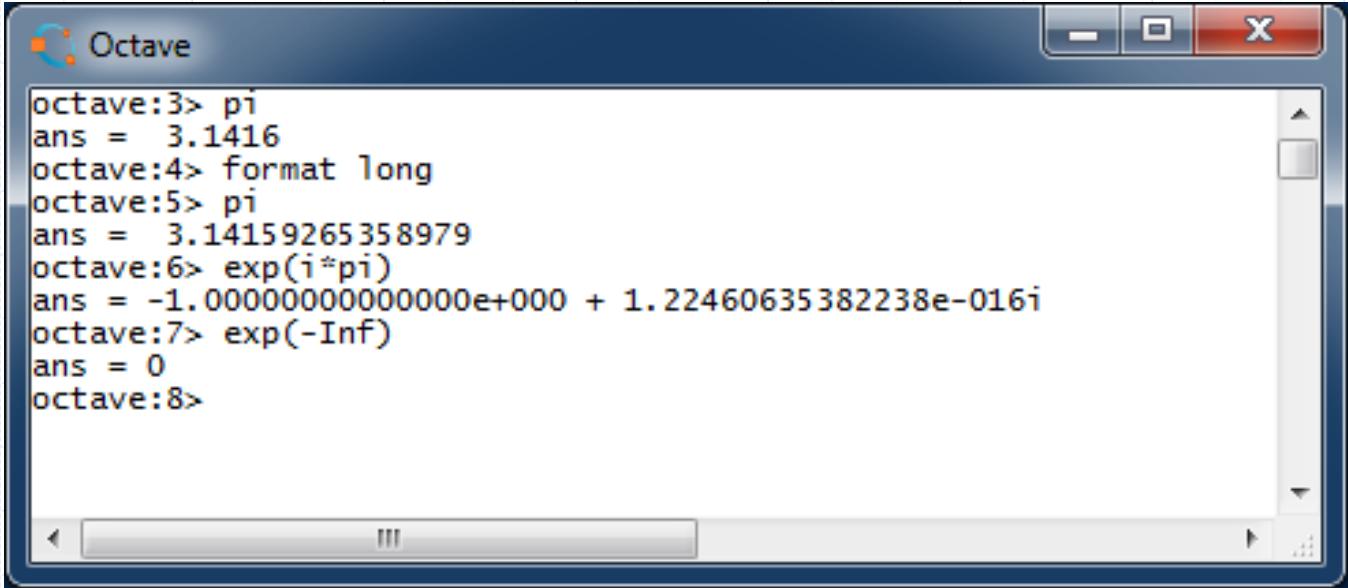
racine carrée de -1

Inf

infini

eps

epsilon machine ( $2^{-52}$ )



The image shows a screenshot of an Octave terminal window titled "Octave". The window contains the following command-line session:

```
octave:3> pi
ans = 3.1416
octave:4> format long
octave:5> pi
ans = 3.14159265358979
octave:6> exp(i*pi)
ans = -1.000000000000000e+000 + 1.22460635382238e-016i
octave:7> exp(-Inf)
ans = 0
octave:8>
```

# Nombres complexes

On crée un nombre complexe en faisant  $z=2+i*3$ .

On peut également faire  $z=\text{complex}(2,3)$ .

`conj(z)`

% complexe conjugué de z

`real(z)`

% partie réelle de z

`imag(z)`

% partie imaginaire de z

`abs(z)`

% module de z

`arg(z)`

% argument de z

$$z = \text{abs}(z) * e^{i * \text{arg}(z)}$$

### Octave

```
octave:2> z=2+3*i
z = 2 + 3i
octave:3> conj(z)
ans = 2 - 3i
octave:4> real(z)
ans = 2
octave:5> imag(z)
ans = 3
octave:6> abs(z)
ans = 3.6056
octave:7> arg(z)
ans = 0.98279
octave:8> abs(z)*exp(i*arg(z))
ans = 2.0000 + 3.0000i
octave:9> -
```

# Vecteurs et matrices

v=[ 1 ; 2 ; 3 ]

% vecteur (colonne)

M=[ 1 2 3 ; 4 5 6 ; 7 8 9 ]

% matrice

v'

% transposé de v

M\*v

% produit matrice-vecteur

v'\*v

% produit scalaire

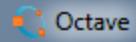
v\*v'

% produit tensoriel

v.\*v

% produit élément par élément

Voir aussi : dot, cross



```
octave:3> v=[ 1 ; 2 ; 3 ]
```

```
v =
```

```
1  
2  
3
```

```
octave:4> M=[ 1 2 3 ; 4 5 6 ; 7 8 9 ]
```

```
M =
```

```
1 2 3  
4 5 6  
7 8 9
```

```
octave:5> v'
```

```
ans =
```

```
1 2 3
```

```
octave:6> M'
```

```
ans =
```

```
1 4 7  
2 5 8  
3 6 9
```

```
octave:7> M*v
```

```
ans =
```

```
14  
32  
50
```

```
octave:8> v'*v
```

```
ans = 14
```

```
octave:9> v*v'
```

```
ans =
```

```
1 2 3  
2 4 6  
3 6 9
```

```
octave:10> v.*v
```

```
ans =
```

```
1  
4  
9
```

```
octave:11>
```

# Vecteurs et matrices

v = [ 1 ; 2 ; 3 ]

% vecteur (colonne)

M = [ 1 2 3 ; 4 5 6 ; 7 8 9 ]

% matrice

v(1)

% élément  $v_1$

M(2,3)

% élément  $M_{2,3}$

# Graphiques 1-D

```
1.  
x=0:.1:10;
```

% construction de x

```
y=sin(x).*exp(-x/2);
```

% construction de y

```
plot(x,y)
```

% représentation graphique

```
xlabel("x")
```

Matlab utilise des  
« ' » au lieu de « " »

```
ylabel("y=y(x)")
```

```
title("Representation de y=y(x)")
```

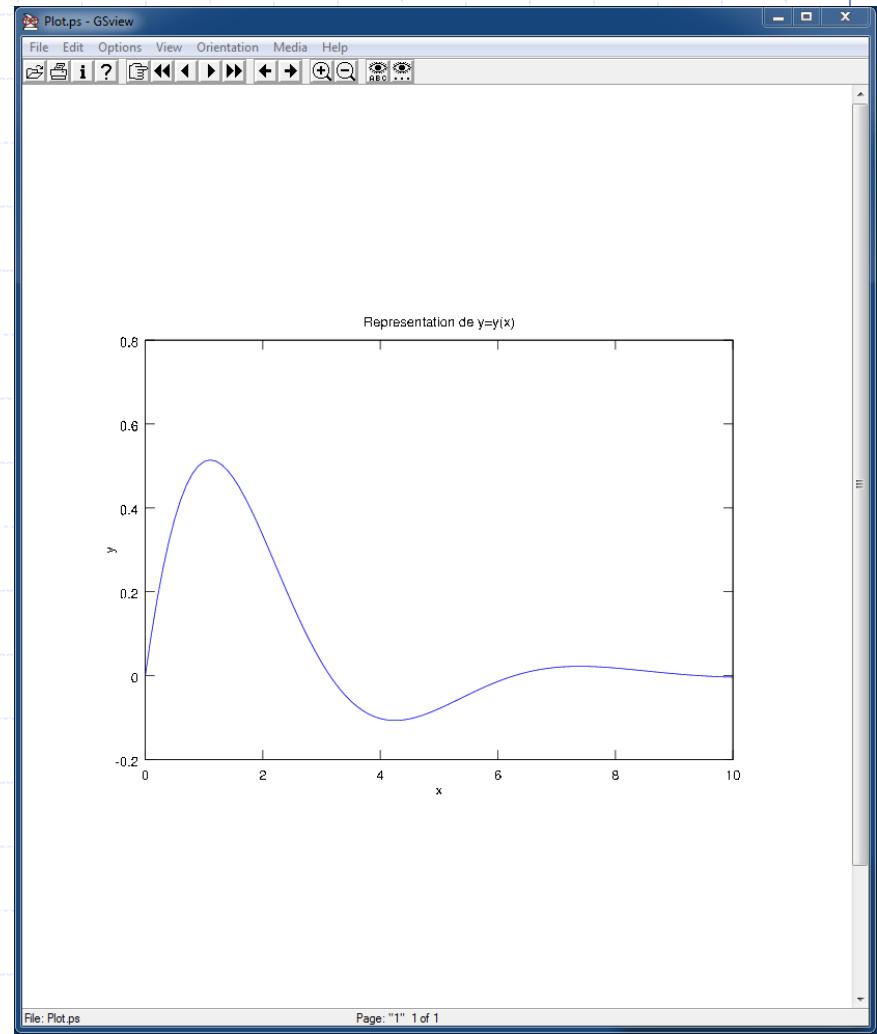
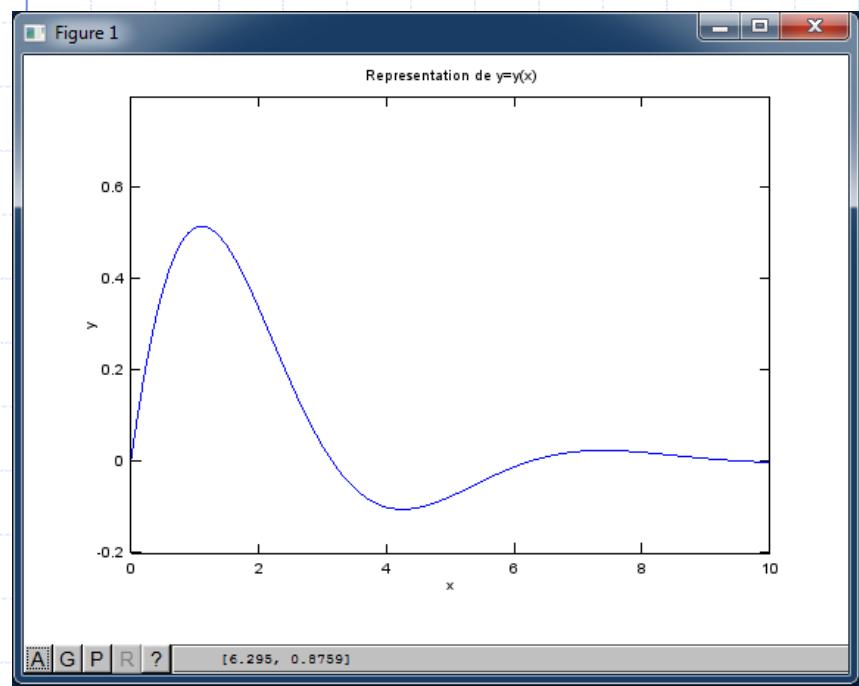
```
print Plot;
```

% produit un fichier Plot.ps

Octave

```
octave:3> x=0:.1:10;
octave:4> y=sin(x).^exp(-x/2);
octave:5> plot(x,y)
octave:6> xlabel("x")
octave:7> ylabel("y")
octave:8> title("Representation de y=y(x)")
octave:9> print Plot
```

C:\Users\amayer>del C:\Users\amayer\AppData\Local\Temp\oct-2.ps  
octave:10> -



# Graphiques 2-D

x=-5:.1:5;

% construction de x

y=-5:.1:5;

% construction de y

[X,Y]=meshgrid(x,y);

% X et Y sont des matrices  
% reprenant les valeurs de  
% x et y pour chaque point  
% de la grille

Z=exp(-(X.^2+Y.^2)/2);

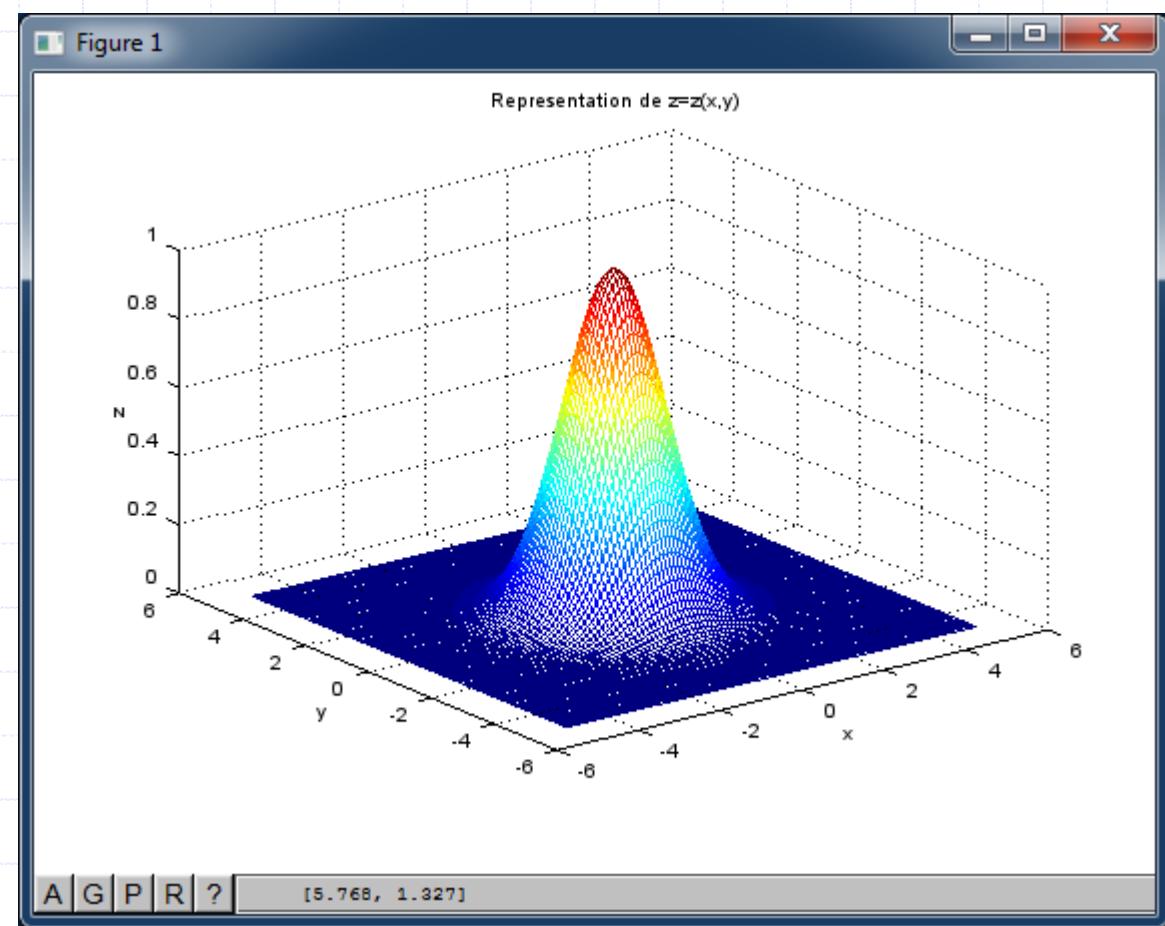
% construction de Z

% Z est une matrice

mesh(X,Y,Z)

Octave

```
octave:2> x=-5:.1:5;
octave:3> y=-5:.1:5;
octave:4> [X,Y]=meshgrid(x,y);
octave:5> Z=exp(-(X.^2+Y.^2)/2);
octave:6> mesh(X,Y,Z)
octave:7> xlabel("x")
octave:8> ylabel("y")
octave:9> zlabel("z")
octave:10> title("Representation de z=z(x,y)")
octave:11>
```

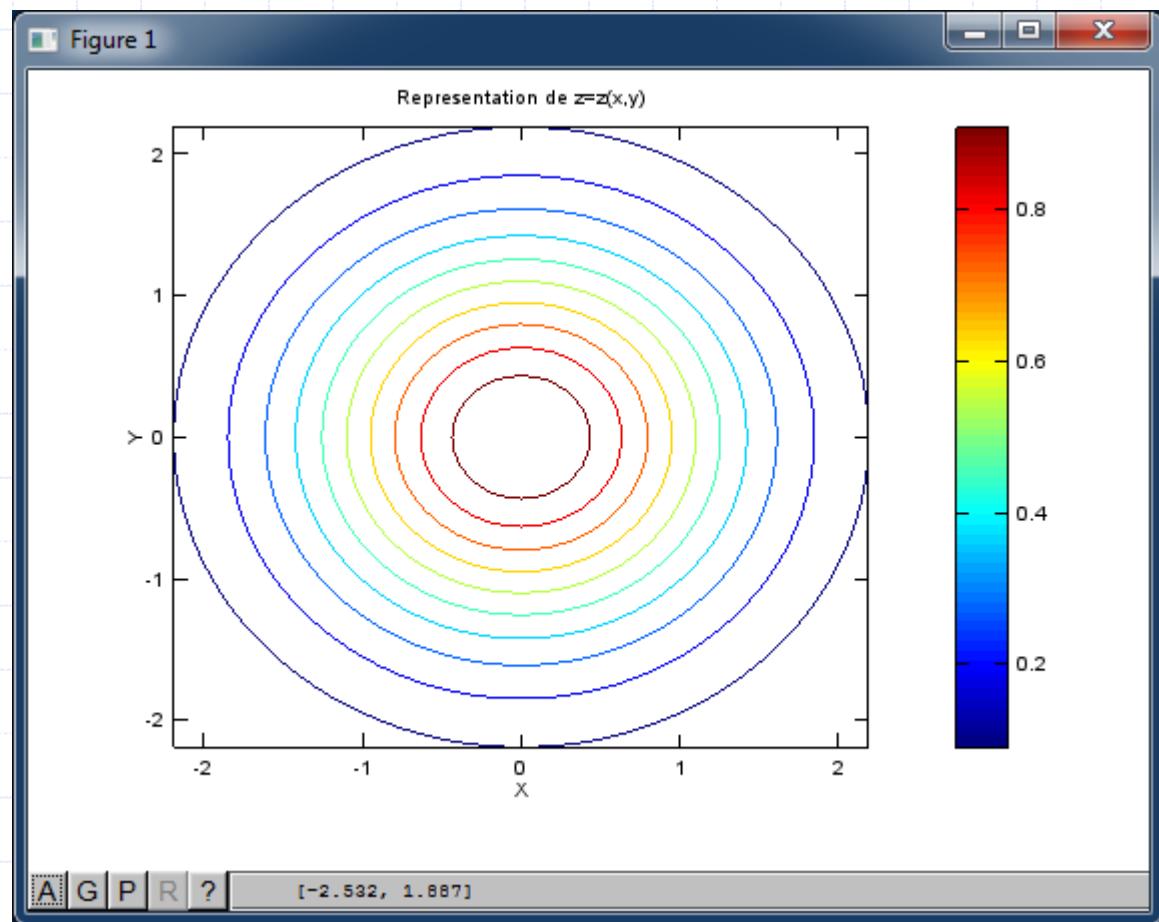


Octave

```
octave:2> x=-5:.1:5;
octave:3> y=-5:.1:5;
octave:4> [X,Y]=meshgrid(x,y);
octave:5> Z=exp(-(X.^2+Y.^2)/2);
octave:6> contour(x,y,Z,10)
octave:7> xlabel("X")
octave:8> ylabel("Y")
octave:9> title("Representation de z=z(x,y)")
octave:10> colorbar
octave:11> _
```

contour(x,y,Z,10)

nombre de  
courbes de niveau

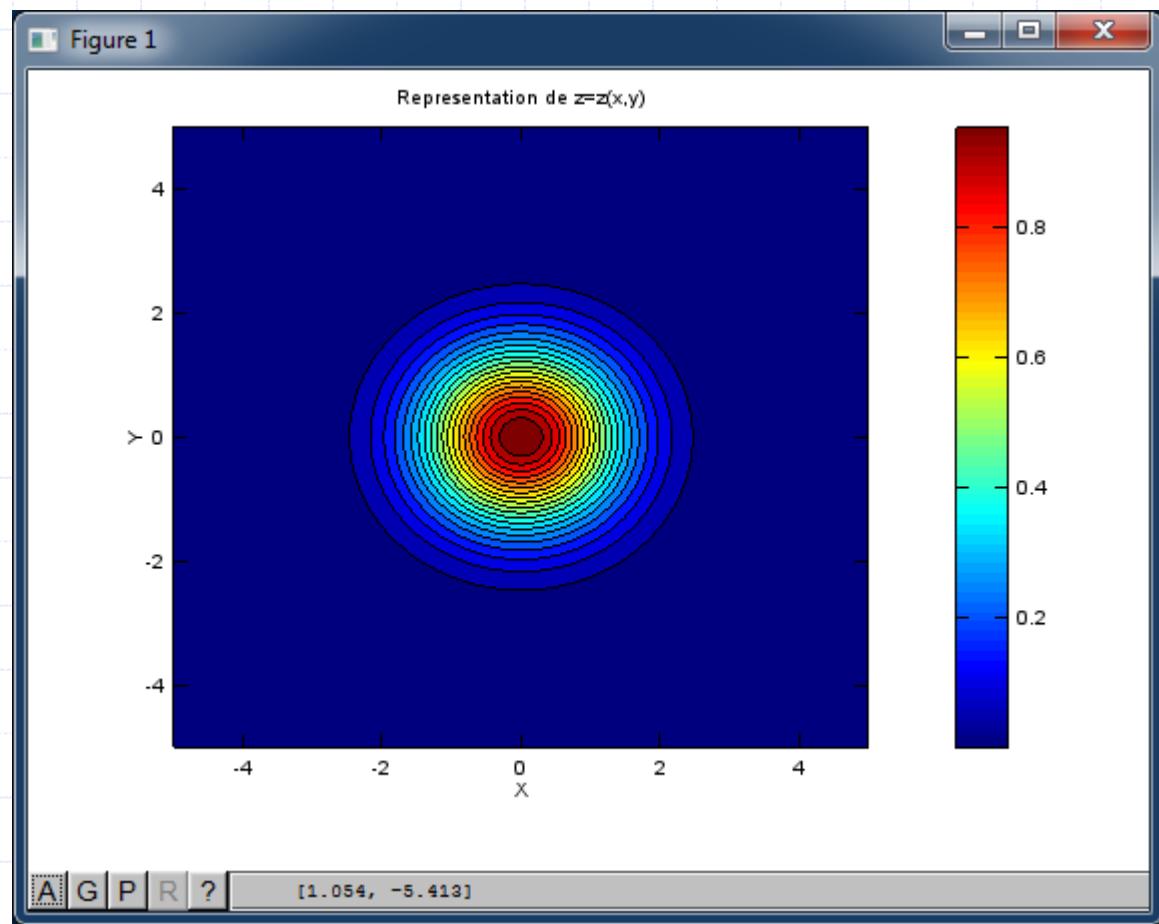


Octave

```
octave:2> x=-5:.1:5;
octave:3> y=-5:.1:5;
octave:4> [X,Y]=meshgrid(x,y);
octave:5> Z=exp(-(X.^2+Y.^2)/2);
octave:6> contourf(x,y,Z,20)
octave:7> xlabel("X")
octave:8> ylabel("Y")
octave:9> title("Representation de z=z(x,y)")
octave:10> colorbar
octave:11>
```

contourf(x,y,Z,20)

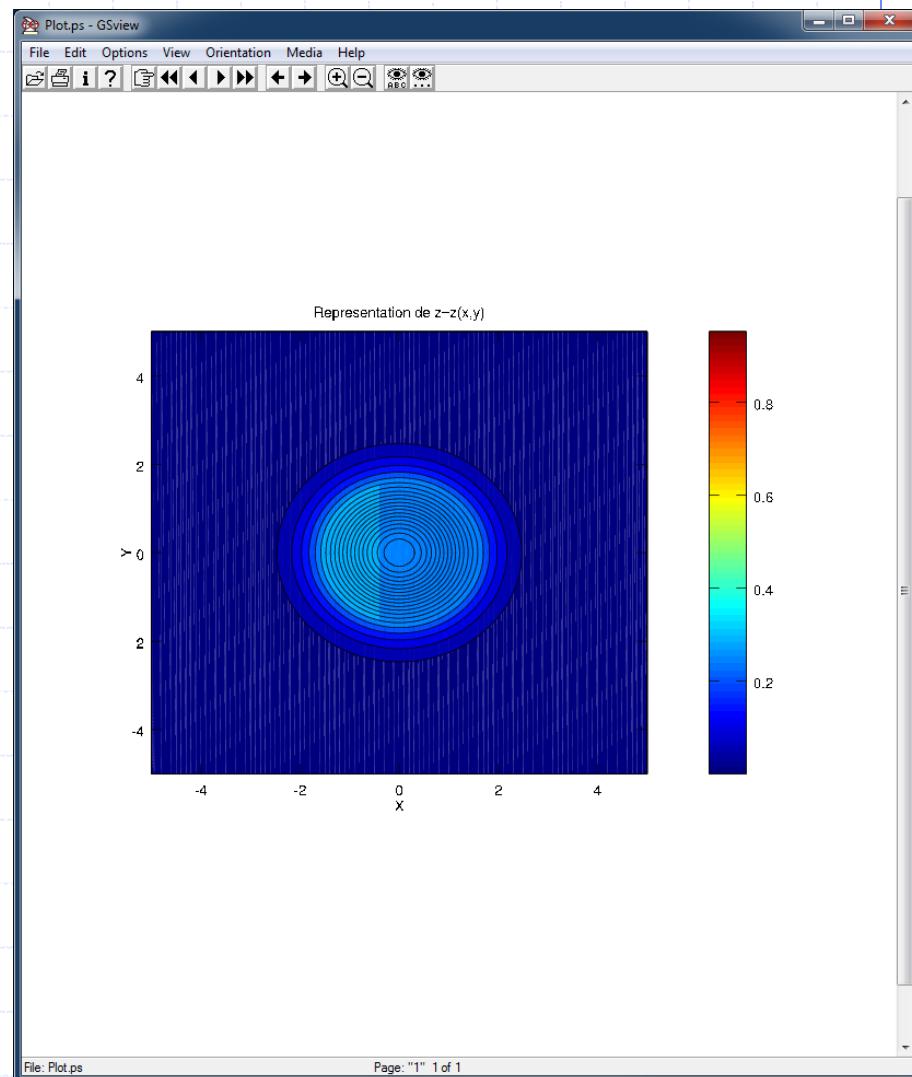
nombre de  
courbes de niveau



Octave

```
octave:2> cd d:  
octave:3> cd Octave  
octave:4> x=-5:.1:5;  
octave:5> y=-5:.1:5;  
octave:6> [X,Y]=meshgrid(x,y);  
octave:7> Z=exp(-(X.^2+Y.^2)/2);  
octave:8> contourf(X,Y,Z,20)  
octave:9> xlabel("X")  
octave:10> ylabel("Y")  
octave:11> title("Representation de z=z(x,y)")  
octave:12> colorbar  
octave:13> print Plot
```

print Plot  
print -dpsc Plot.ps



# Programmes

## Chapitre 2

# Qu'est-ce qu'un programme ?

On peut éditer un fichier « Main.m » qui contient les instructions à exécuter.

L'éditeur peut être Notepad++, JEdit, emacs, etc.

On exécute le programme principal en lançant « Main » dans l'environnement d'Octave.

Notez bien la distinction à faire entre le nom du programme (« Main.m ») et l'instruction (« Main ») qui permet de le lancer.

# Comment se déplacer dans une structure de fichiers ?

Une fois dans Octave, tapez « cd d: » pour aller sur le disque D:.

Tapez « cd Octave » pour aller dans le répertoire D:\Octave.

Tapez « cd .. » pour revenir en arrière.

Tapez « pwd » pour savoir où vous êtes.

Tapez « dir » (Windows) ou « ls » (Linux) pour voir la liste des fichiers.

Tapez « edit Main.m » pour éditer votre programme.

Tapez « Main » pour lancer votre programme.

Octave

```
octave:2> cd d:  
octave:3> cd Octave  
octave:4> cd 01-Programme  
octave:5> pwd  
ans = d:\Octave\01-Programme  
octave:6> dir  
. .. Main.m  
octave:7> edit Main.m
```

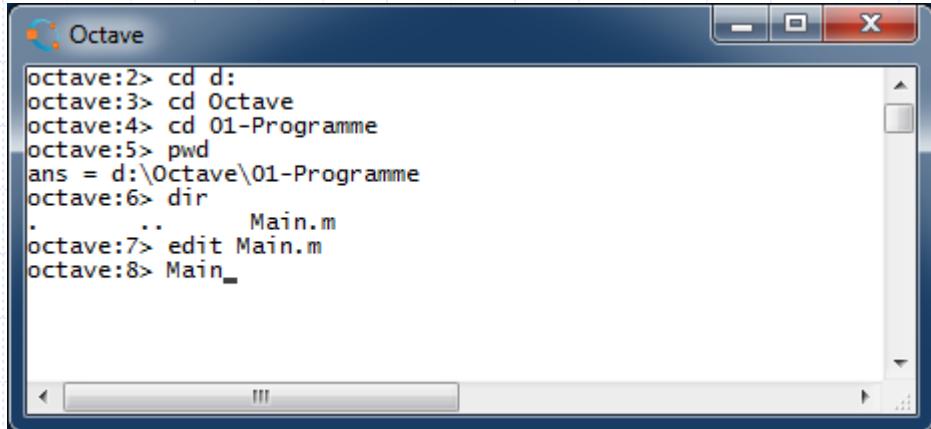
d:\Octave\01-Programme\Main.m - Notepad++

The screenshot shows a Notepad++ window titled "d:\Octave\01-Programme\Main.m - Notepad++". The window has a standard Windows-style title bar with icons for minimize, maximize, and close. Below the title bar is a menu bar with File, Edit, Search, View, Encoding, Language, Settings, Macro, Run, Plugins, Window, and ?.

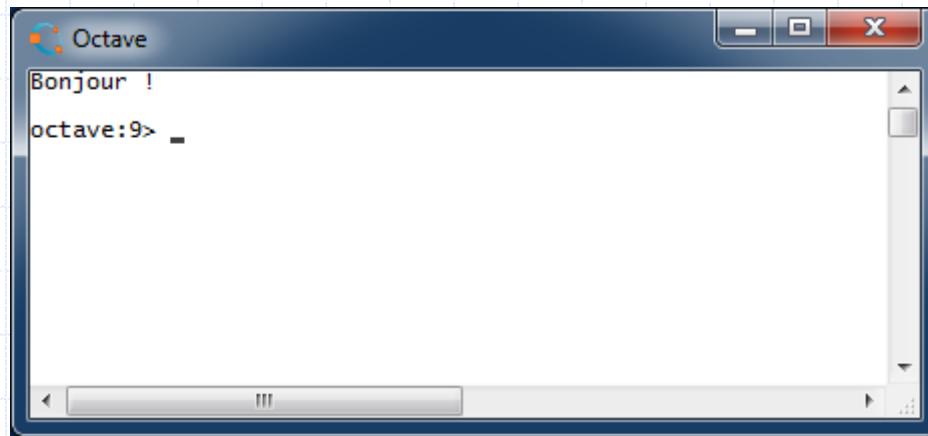
The main text area contains the following MATLAB/Octave code:

```
1 clear ; clc ;  
2  
3 display("Bonjour !");  
4  
5  
6
```

At the bottom of the Notepad++ window, there is a status bar displaying the following information: MATrix LABoratory, length : 44, lines : 6, Ln:1 Col:1 Sel:0|0, Dos\Windows, ANSI as UTF-8, and INS.



```
Octave
octave:2> cd d:
octave:3> cd Octave
octave:4> cd 01-Programme
octave:5> pwd
ans = d:\Octave\01-Programme
octave:6> dir
. .. Main.m
octave:7> edit Main.m
octave:8> Main.m
```



```
Octave
Bonjour !
octave:9>
```

On va considérer dorénavant que toutes les instructions sont écrites au sein d'un programme.

## Instructions utiles pour gérer l'exécution d'un programme :

pause

% pause jusqu'à ce qu'une touche soit  
% enfoncée

pause (nsec)

% pause pendant nsec secondes

break

% sortir du programme

# Opérations élémentaires de lecture et d'écriture

## Chapitre 3

# Lecture au clavier/écriture à l'écran

```
display("Bonjour !");
```

```
n=input("Entrez la valeur de n : ");
```

```
fprintf("La valeur de n est %d \n", n);
```

```
fprintf("%s %d \n", "La valeur de n est ", n);
```

Matlab utilise des  
« ' » au lieu de « " »

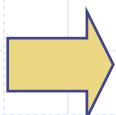
D:\Octave\02-Lecture-Ecriture\Main.m - Notepad++

```
File Edit Search View Encoding Language Settings Macro Run Plugins Window ? X
Main.m
1 clear ; clc ;
2
3 display("Bonjour !");
4
5 n=input("Entrez la valeur de n : ");
6
7 fprintf("La valeur de n est %d \n", n);
8
9 fprintf("%s %d \n", "La valeur de n est", n);
10
```

length: Ln:10 Col:1 Sel:0|0 Dos\Windows ANSI as UTF-8 INS

Octave

```
octave:2> cd d:
octave:3> cd Octave
octave:4> cd 02-Lecture-Ecriture
octave:5> dir
.          ..      Main.m
octave:6> Main_
```



Octave

```
Bonjour !
Entrez la valeur de n : 5
La valeur de n est 5
La valeur de n est 5
octave:7> _
```

# Formats

%d

entier signé

%f

réel (format fixe)

%e

réel (format scientifique)

%g

réel (choix automatique entre %f et %e)

%s

chaine de caractères

%c

caractère

# Formats

%8d

entier représenté sur 8 caractères

%6.3f

réel (format fixe) représenté  
sur 6 caractères dont 3 après la virgule

%10.3e

réel (format scientifique) représenté  
sur 10 caractères dont 3 après la virgule

\r

carriage return

\n

nouvelle ligne

\t

tab

\b

backspace

\\\

\'

Utilisez %-24.17G pour  
afficher toutes les  
décimales d'un nombre réel.

Utilisez \r\n pour les fins  
de ligne d'un fichier.

D:\Octave\03-Formats\Main.m - Notepad++

File Edit Search View Encoding Language Settings Macro Run Plugins Window ? X

Main.m Main.m

```
1 clear ; clc ;
2
3 display("Bonjour !");
4
5 i = 10 ;
6
7 fprintf("La valeur de i est %d\n\n", i);
8
9 x = 100./3.;
10
11 fprintf("La valeur de x est %f\n", x);
12 fprintf("La valeur de x est %6.3f\n", x);
13 fprintf("La valeur de x est %e\n", x);
14 fprintf("La valeur de x est %10.3e\n\n", x);
15
16 ville="Kinshasa";
17 fprintf("Le nom de la ville est %s\n\n", ville);
18
19 display("Au revoir.");
20
```

length: Ln:20 Col:1 Sel:0|0 Dos\Windows ANSI as UTF-8 INS

Octave

```
octave:2> cd d:  
octave:3> cd Octave  
octave:4> cd 03-Formats  
octave:5> dir  
. . . Main.m  
octave:6> Main.m
```

Octave

```
Bonjour !  
La valeur de i est 10  
La valeur de x est 33.333333  
La valeur de x est 33.333  
La valeur de x est 3.333333e+001  
La valeur de x est 3.333e+001  
Le nom de la ville est Kinshasa  
Au revoir.  
octave:7> _
```



# Structures de contrôle

Chapitre 4

# Boucles for

```
for i = 1:5
```

...

i=1, 2, 3, 4, 5

```
end
```

```
for i=1:2:5
```

...

i=1, 3, 5

```
end
```

# Boucles for

```
for i = 5:-1:1
```

...

```
end
```

```
i=5, 4, 3, 2, 1
```

L'instruction « break » permet de sortir  
prématièrement de la boucle.

L'instruction « continue » permet de passer  
à l'itération suivante.

D:\Octave\04-For\Main.m - Notepad++

File Edit Search View Encoding Language Settings Macro Run Plugins Window ? X

Main.m

```
1 clear ; clc ;
2
3 display("Bonjour !");
4
5 for i = -5:5
6   fprintf("La valeur de i est %d\n", i);
7 end
8
9 display("\nEt maintenant ...");
10
11 for i = -5:5
12   if (i==0)
13     continue;
14   end
15   fprintf("La valeur de i est %d et 1/i = %f\n", i, 1/i);
16   if (i==3)
17     break;
18   end
19 end
20
21 display("\nAu revoir.");
22
```

length : 303 lir Ln:1 Col:1 Sel:0|0 Dos\Windows ANSI as UTF-8 INS

Octave

```
octave:2> cd d:  
octave:3> cd Octave  
octave:4> cd 04-For  
octave:5> dir  
. . . Main.m  
octave:6> Main.m
```



Octave

```
Bonjour !  
  
La valeur de i est -5  
La valeur de i est -4  
La valeur de i est -3  
La valeur de i est -2  
La valeur de i est -1  
La valeur de i est 0  
La valeur de i est 1  
La valeur de i est 2  
La valeur de i est 3  
La valeur de i est 4  
La valeur de i est 5  
  
Et maintenant ...  
  
La valeur de i est -5 et 1/i = -0.200000  
La valeur de i est -4 et 1/i = -0.250000  
La valeur de i est -3 et 1/i = -0.333333  
La valeur de i est -2 et 1/i = -0.500000  
La valeur de i est -1 et 1/i = -1.000000  
La valeur de i est 1 et 1/i = 1.000000  
La valeur de i est 2 et 1/i = 0.500000  
La valeur de i est 3 et 1/i = 0.333333  
  
Au revoir.  
octave:7>
```

# Structures while

while (condition)

...

end

Les instructions contenues dans cette structure while sont exécutées tant que la condition est vérifiée.

D:\Octave\05-While\Main.m - Notepad++



```
File Edit Search View Encoding Language Settings Macro Run Plugins Window ? X
Main.m
1 clear ; clc ;
2
3 display("Bonjour !");
4
5 x=0;
6 while (x<1|x>10)
7   x=input("Entrez un x compris entre 1 et 10 : ");
8 end
9
10 fprintf("\nMerci ! Vous avez choisi x = %f\n\n",x);
11
12 display("Au revoir.");
13
```

length: Ln:13 Col:1 Sel:0|0 Dos\Windows ANSI as UTF-8 INS

Octave

```
Bonjour !
Entrez un x compris entre 1 et 10 : 20
Entrez un x compris entre 1 et 10 : -4
Entrez un x compris entre 1 et 10 : 6
Merci ! Vous avez choisi x = 6.000000
Au revoir.

octave:6>
```

# Structures if

if (condition)

...

% instructions exécutées

% si la condition est vérifiée

end

```
if (x>0)
    display("x est strictement positif");
end
```

# Structures if

if (condition)

...

% instructions exécutées

% si la condition est vérifiée

else

...

% instructions exécutées sinon

end

```
if (x>0)
    display("x est strictement positif");
else
    display("x est négatif");
end
```

# Structures if

if (condition1)

...

% instructions exécutées

% si la condition 1 est vérifiée

elseif (condition2)

...

% instructions exécutées

% si la condition 2 est vérifiée

else

...

% instructions exécutées sinon

end

D:\Octave\06-If\Main.m - Notepad++

File Edit Search View Encoding Language Settings Macro Run Plugins Window ? X

Main.m

```
1 clear ; clc ;
2
3 display("Bonjour !");
4
5 x=input("Entrez une valeur pour x : ");
6
7 if (x>0)
8   display("x est strictement positif");
9 elseif (x<0)
10  display("x est strictement negatif");
11 else
12   display("x est null");
13 end
14
15 display("Au revoir.");
```

length: 2 Ln:15 Col:23 Sel:0|0 Dos\Windows ANSI as UTF-8 INS



Octave

Bonjour !

Entrez une valeur pour x : -4  
x est strictement negatif

Au revoir.

octave:6>

# Opérateurs de comparaison

<, <=, >, >=, ==, ~=

# Opérateurs logiques

&	et
	ou
~	not
xor	ou exclusif

```
if (x~=y & x+y>0)  
    ...  
end
```

Voir aussi : `isequal` pour la comparaison de tableaux ou de chaînes de caractères.

# Structures switch

switch expression

case {valeur1}

...

% instructions à exécuter

% si expression=valeur1

case {valeur2}

...

% instructions à exécuter

% si expression=valeur2

otherwise

...

% instructions à exécuter sinon

end

D:\Octave\07-Switch\Main.m - Notepad++

File Edit Search View Encoding Language Settings Macro Run Plugins Window ? X

Main.m

```
1 clear ; clc ;
2
3 display("Bonjour !");
4
5 ville=input("Entrez le nom d'une ville : ");
6
7 switch ville
8 case {"Kinshasa" "Bukavu"}
9   display("Cette ville est en R.D.C.");
10 case {"Namur"}
11   display("Cette ville est en Belgique.");
12 otherwise
13   display("Je ne connais pas cette ville.");
14 end
15
16 display("Au revoir.");
17
```

length:3 Ln:17 Col:1 Sel:0|0 Dos\Windows ANSI as UTF-8 INS



Octave

Bonjour !

Entrez le nom d'une ville : "Kinshasa"

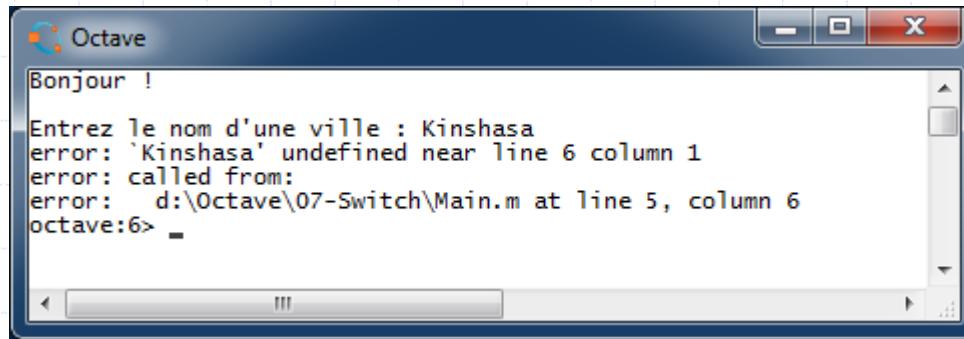
Cette ville est en R.D.C.

Au revoir.

octave:6>

# Structures try/catch

Notez bien que ...



La solution pour gérer les erreurs consiste à utiliser une structure try/catch :

try

...

% instructions à essayer

catch

...

% instructions en cas d'erreur

end

D:\Octave\08-Try-Catch\Main.m - Notepad++

File Edit Search View Encoding Language Settings Macro Run Plugins Window ? X

Main.m

```
1 clear ; clc ;
2
3 display("Bonjour !");
4
5 try
6     ville=input("Entrez le nom d'une ville : ");
7 catch
8     display("Vous devez utiliser des guillemets.");
9     return; ←
10 end
11
12 switch ville
13 case {"Kinshasa" "Bukavu"}
14     display("Cette ville est en R.D.C.");
15 case {"Namur"}
16     display("Cette ville est en Belgique.");
17 otherwise
18     display("Je ne connais pas cette ville.");
19 end
20
21 display("Au revoir.");
```

length:4 Ln:21 Col:23 Sel:0|0 Dos\Windows

« return » permet de sortir du programme

Octave

Bonjour !

Entrez le nom d'une ville : Kinshasa

Vous devez utiliser des guillemets.

octave:6>



# Types de variables

Chapitre 5

# Types de variables

- ◆ Variables entières, réelles, complexes
- ◆ Chaînes de caractères
- ◆ Variables logiques
- ◆ Variables structurées

La représentation interne d'une variable s'adapte en fonction du contenu. On ne doit rien déclarer.

Il est utile de connaître les instructions de conversion et la manière de manipuler ces variables.

# Variables entières, réelles, complexes

n=2

% entier

x=1.6

% réel

x=1.6E9

$1.6 \times 10^9$

z=1.+2.\*i

% complexe

1+2.i

# Conversion réels - entiers

x=1.6

% réel

i=floor(x) → 1

% entier

i=ceil(x) → 2

i=round(x) → 2

x=-1.6

% réel

i=floor(x) → -2

% entier

i=ceil(x) → -1

i=round(x) → -2

i=2

% entier

x=i

% réel

# Conversion complexes - réels

`z=1.+2.*i`

% complexe

`x=real(z)`

% partie réelle

`y=imag(z)`

% partie imaginaire

`r=abs(z)`

% valeur absolue

`x=1.`

% réel

`y=2.`

`z=complex(x,y)` % complexe

# Chaînes de caractères

a="Bonjour";

% chaîne de caractères

b=" a tous";

phrase=[a b];

% concaténation

phrase(1) → "B"

On manipule les éléments d'une chaîne de caractères comme pour un tableau.

D:\Octave\09-Chaines>Main.m - Notepad++

```
File Edit Search View Encoding Language Settings Macro Run Plugins Window ? X
Main.m
1 clear ; clc ;
2
3 a="Bonjour";
4 b=" a tous !";
5
6 phrase=[a b];
7
8 fprintf('%s\n\n',phrase);
9
10 for i = 1:length(phrase)
11   fprintf('%c\n',phrase(i));
12 end
13
14 display("\nAu revoir.");
15
```

length: Ln:14 Col:25 Sel:0|0 Dos\Win

« length » donne la longueur



Octave

```
Bonjour a tous !
B
o
n
j
o
u
r
a
t
o
u
s
!
Au revoir.
octave:6> _
```

# Conversion chaînes de caractères-entiers/réels

i=2

% entier

string=int2str(i) → "2"

x=1.6

% réel

string=num2str(x) → "1.6"

string="1.6"

% chaîne

x=str2num(string) → 1.6

# Ecrire dans une chaîne de caractères

```
string=sprintf("a = %f",a);
```

L'utilisation de « sprintf » est similaire à celle de « fprintf ».

# Variables logiques

Une variable logique prend la valeur  
« true » ou « false ».

```
output=false;  
if (output)  
    display("Bonjour !");  
end
```

# Variables structurées

```
Carbon.symbol="C";
```

```
Carbon.Z=6;
```

```
Carbon.A=12.0107;
```

```
fprintf("La masse atomique du carbone  
est %g.\n", Carbon.A);
```

# Tableaux

## Chapitre 6

# Vecteurs et matrices

v=[ 1 ; 2 ; 3 ]

% vecteur colonne

v(1) → élément  $v_1 \rightarrow 1$

$$v = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$$

M=[ 1 2 3 ; 4 5 6 ; 7 8 9 ] % matrice

$$M = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

M(2,3) → élément  $M_{2,3} \rightarrow 6$

# Vecteurs et matrices

v=[ 1 2 3 ]

% vecteur ligne

v(1) → élément  $v_1 \rightarrow 1$

$$v = (1 \quad 2 \quad 3)$$

v=[ 1, 2, 3 ]

% vecteur ligne

v(1) → élément  $v_1 \rightarrow 1$

$$v = (1 \quad 2 \quad 3)$$

(autre syntaxe pour le même résultat)

# Modes de construction

v=[ 1, 2, 3 ] % constructeur

$$v = \begin{pmatrix} 1 & 2 & 3 \end{pmatrix}$$

v=1:3 % itérateur

for i=1:3 % boucle sur chaque élément

v(i)=...

end

# Modes de construction

v=zeros(3,1) % matrice 3x1 de 0

$$v = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

v=ones(1,3) % matrice 1x3 de 1

$$v = \begin{pmatrix} 1 & 1 & 1 \end{pmatrix}$$

M=eye(3,3) % matrice identité

$$M = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

M=diag([1,2,3]) % matrice diagonale

# Modes de construction

M=rand(3,3) % matrice avec des nombres  
% aléatoires  
% (distribution uniforme)

M=randn(3,3) % matrice avec des nombres  
% aléatoires  
% (distribution normale)

Voir aussi : x=rand pour générer  
un simple nombre aléatoire.

# Modes de construction

`x=linspace(a,b,100)` % x construit comme un  
% vecteur ligne à 100 éléments  
% répartis de manière uniforme  
% entre a et b

`x=logspace(a,b,100)` % x construit comme un  
% vecteur ligne à 100 éléments  
% répartis de manière logarithmique  
% entre a et b

# Combinaisons

v=[1, 2]

$$v = \begin{pmatrix} 1 & 2 \end{pmatrix}$$

v=[v, 3]

$$v = \begin{pmatrix} 1 & 2 & 3 \end{pmatrix}$$

# Sections

1.

$M = [ 1 \ 2 \ 3 ; 4 \ 5 \ 6 ; 7 \ 8 \ 9 ]$  % matrice

$$M = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

$M(2,3) \rightarrow$  élément  $M_{2,3} \rightarrow 6$

$M(:,2) \rightarrow$  éléments de la colonne 2

$$M(:, 2) = \begin{pmatrix} 2 \\ 5 \\ 8 \end{pmatrix}$$

$M(2,:) \rightarrow$  éléments de la ligne 2

$M(1:2,1:2) \rightarrow$  bloc 2x2

$$M(2, :) = ( 4 \ 5 \ 6 )$$

$$M(1 : 2, 1 : 2) = \begin{pmatrix} 1 & 2 \\ 4 & 5 \end{pmatrix}$$

# Sections

4.

M=[ 1 2 3 ; 4 5 6 ; 7 8 9 ] % matrice

M(1,1) → 1

$$M = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

M([1 3],1) →  $M([1 \ 3], 1) = \begin{pmatrix} 1 \\ 7 \end{pmatrix}$

M([1 3],[2 1 3]) →  $M([1 \ 3], [2 \ 1 \ 3]) = \begin{pmatrix} 2 & 1 & 3 \\ 8 & 7 & 9 \end{pmatrix}$

A=[ eye(3) ones(3,2) ; zeros(2,3) rand(2) ]

# Opérations sur des vecteurs

v=[ 1, 2, 3 ] % constructeur

$$v = \begin{pmatrix} 1 & 2 & 3 \end{pmatrix}$$

length(v) → 3

min(v) → 1

max(v) → 3

sum(v) → 6

prod(v) → 6

Voir aussi :  
mean, median, var, std, sort  
conv, xcorr, cov, corrcoef

# Opérations sur des matrices

A=[1 2 ; 3 4 ]

$$A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$$

size(A) → ( 2 2 )

size(A,1) → 2

selon indice n° 1

A(end) → 4 (dernier élément)

A(end-1) → 2 (avant-dernier élément)  
(rangement colonne par colonne)

# Opérations sur des matrices

$$A = [1 \ 2 ; 3 \ 4]$$

$$B = [1 \ 3 ; 3 \ 9]$$

$$A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$$

$$B = \begin{pmatrix} 1 & 3 \\ 3 & 9 \end{pmatrix}$$

$A * B \rightarrow$  produit matriciel entre A et B

$$A * B = \begin{pmatrix} 7 & 21 \\ 15 & 45 \end{pmatrix}$$

$A_{\cdot} * B \rightarrow$  produit élément par élément

$$A_{\cdot} * B = \begin{pmatrix} 1 & 6 \\ 9 & 36 \end{pmatrix}$$

# Opérations sur des matrices

$$A = [1 \ 2 ; 3 \ 4]$$

$$A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$$

$A'$  → transposée de A

$\text{trace}(A)$  → trace de A

$\text{rank}(A)$  → rang de A

$\det(A)$  → déterminant de A

$\text{inv}(A)$  → inverse de A

$\text{tril}(A)$  → partie triangulaire inférieure de A

$\text{triu}(A)$  → partie triangulaire supérieure de A

# Opérations sur des matrices

A=[1 2 ; 3 4 ]

$$A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$$

norm(A) → norme-2 de A

norm(A,1) → norme-1 de A

norm(A,inf) → norme- $\infty$  de A

cond(A) → condition de A en norme-2

cond(A,1) → condition de A en norme-1

cond(A,inf) → condition de A en norme- $\infty$

# Opérations sur des matrices

A=[1 2 ; 3 4 ]

$$A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$$

lambda=eig(A)

→ valeurs propres de A

[V,lambda]=eig(A)

→ valeurs propres et

vecteurs propres de A

[U,S,V]=svd(A)

→ décomposition SVD de A

help svd

Voir aussi : lu, qr, orth

# Opérations sur des matrices

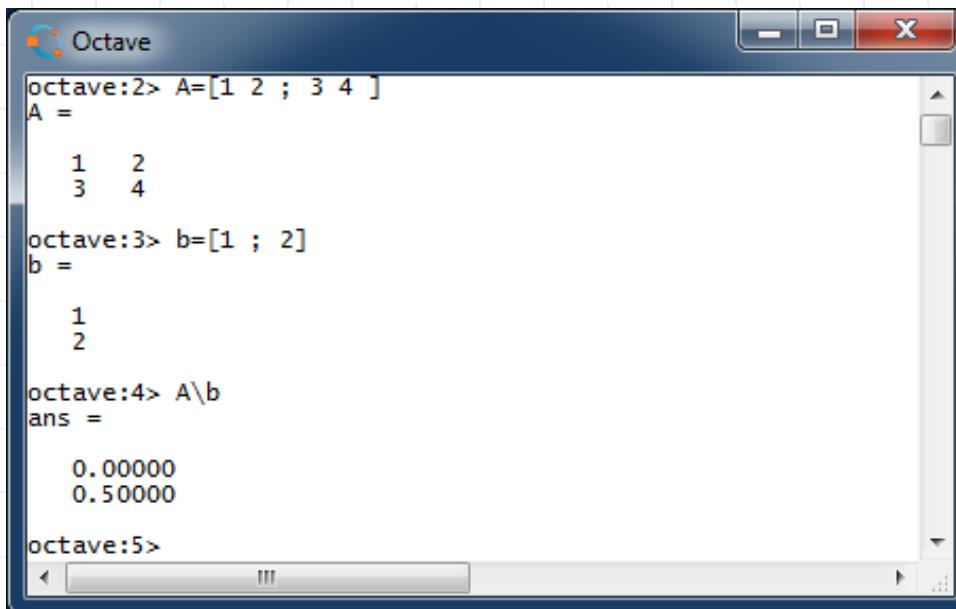
$A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$

$b = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$

$$A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$$

$$b = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

$A \setminus b \rightarrow$  solution  $x$  de  $A x = b$



The screenshot shows the Octave command-line interface. The user has entered the following commands:

```
octave:2> A=[1 2 ; 3 4]
A =
  1  2
  3  4

octave:3> b=[1 ; 2]
b =
  1
  2

octave:4> A\b
ans =
  0.00000
  0.50000

octave:5>
```

Voir aussi :  
linsolve, cgs, pcg

# Fichiers

## Chapitre 7

# save/load

save fichier.dat % enregistrer toutes les

% variables dans

% fichier.dat

save fichier.dat x % enregistrer x dans

% fichier.dat

save fichier.dat x y % enregistrer x et y

% dans fichier.dat

save("fichier.dat","x","y")

% autre

% syntaxe

# save/load

load fichier.dat

% lire le contenu de  
% fichier.dat s'il a été  
% créé avec save

help save  
help load

Octave

```
octave:2> cd d:  
octave:3> cd Octave  
octave:4> cd 10-save  
octave:5> x=2  
x = 2  
octave:6> save fichier.dat x  
octave:7> dir  
. .. fichier.dat  
octave:8> clear  
octave:9> x  
error: `x' undefined near line 9 column 1  
octave:9> load fichier.dat  
octave:10> x  
x = 2  
octave:11>
```

fichier.dat - Bloc-notes

```
Fichier Edition Format Affichage ?  
# Created by Octave 3.6.2, Tue Mar 05 17:00:33 2013 Paris, Madrid <unknown@lps-10-02>  
# name: x  
# type: scalar  
2
```

# Ecriture d'un fichier

On veut écrire.

```
fid=fopen("fichier.dat","w");      ← ouverture  
fprintf(fid,"x = %f \r\n",x);      ← écriture  
fclose(fid);                      ← fermeture
```

↑  
fid fait référence au  
numéro du fichier.

\r\n pour retour à la ligne  
dans un fichier

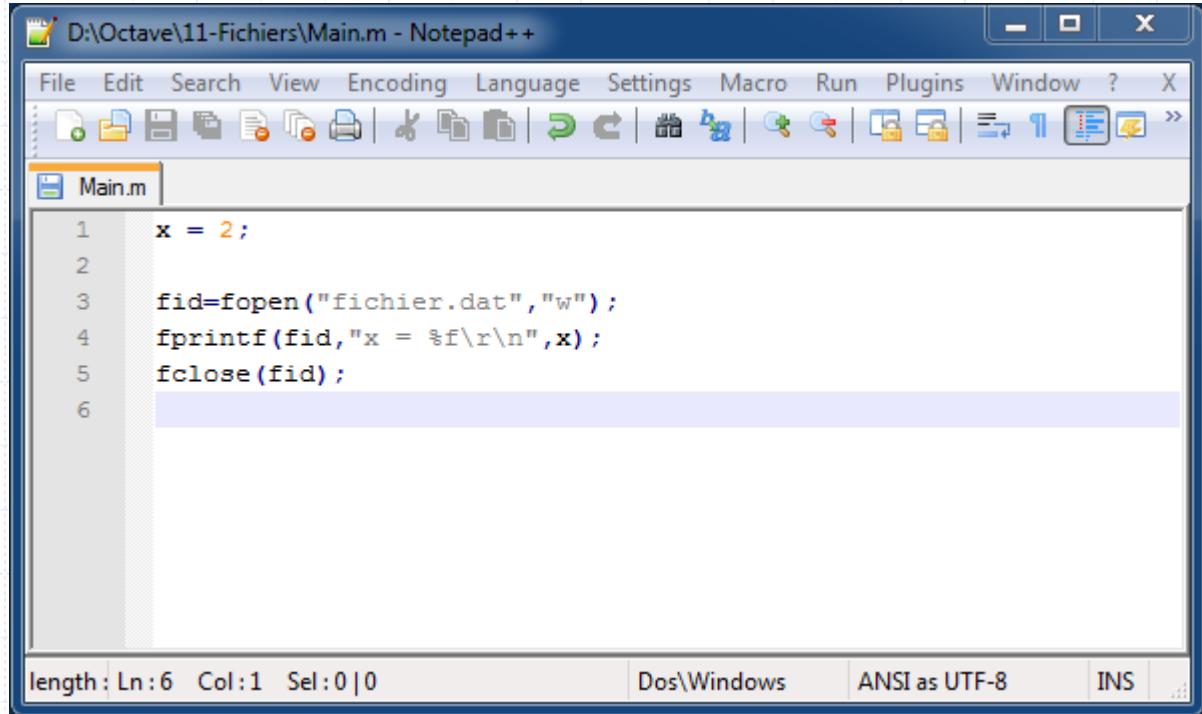
D:\Octave\11-Fichiers>Main.m - Notepad++

File Edit Search View Encoding Language Settings Macro Run Plugins Window ? X

Main.m

```
1 x = 2;
2
3 fid=fopen("fichier.dat","w");
4 fprintf(fid,"x = %f\r\n",x);
5 fclose(fid);
6
```

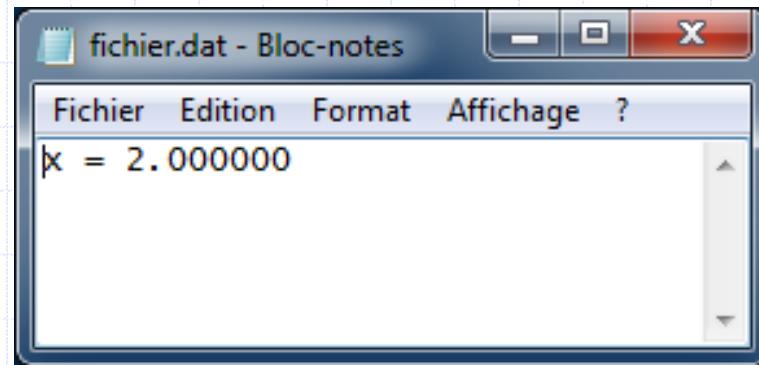
length : Ln :6 Col :1 Sel :0|0 Dos\Windows ANSI as UTF-8 INS



fichier.dat - Bloc-notes

Fichier Edition Format Affichage ?

x = 2.000000



# Lecture d'un fichier

On veut lire.

```
fid=fopen("fichier.dat","r");    ← ouverture  
line=fgetl(fid);← lire une ligne (le résultat est une  
fclose(fid);                chaîne de caractères)    ← fermeture
```

fid fait référence au  
numéro du fichier.

x=str2num(string) permet  
de convertir le contenu d'une  
chaîne de caractères « string »  
en nombre réel « x »

D:\Octave\11-Fichiers>Main.m - Notepad++

```
File Edit Search View Encoding Language Settings Macro Run Plugins Window ? X
Main.m
1 fid=fopen("fichier.dat","r");
2 line=fgetl(fid);
3 x=str2num(line(4:length(line)));
4 fclose(fid);
5
6 printf("x = %f\n",x);
7
```

length : Ln :1 Col :1 Sel :0 |0 Dos\Windows ANSI as UTF-8 INS

fichier.dat - Bloc-notes

```
Fichier Edition Format Affichage ?
x = 2.000000
```

Octave

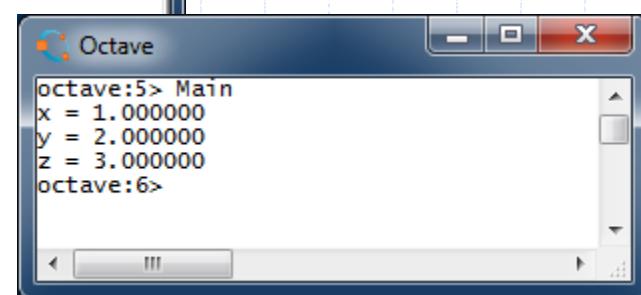
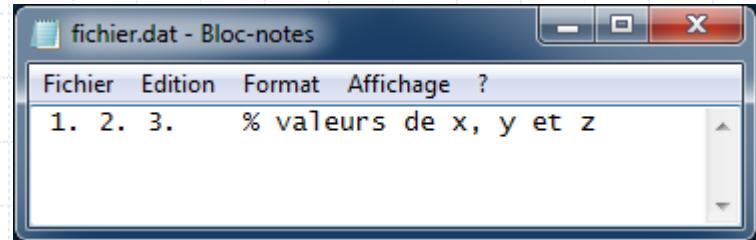
```
octave:5> Main
x = 2.000000
octave:6> -
```

# textscan

D:\Octave\12-textscan\Main.m - Notepad++

```
File Edit Search View Encoding Language Settings Macro Run Plugins Window ? X
Main.m
1 fid=fopen("fichier.dat","r");
2 data=textscan(fid,'%f %f %f',1);
3 x=data{1};
4 y=data{2};
5 z=data{3};
6 fclose(fid);
7
8 printf("x = %f\n",x);
9 printf("y = %f\n",y);
10 printf("z = %f\n",z);
11
```

length: Ln:11 Col:1 Sel:0 | 0 Dos\Windows ANSI as UTF-8 INS



numéro du fichier

data=textscan(fid,'%f %f %f',1);

format de la ligne à lire

nombre de lignes à lire

fichier.dat - Bloc-notes

```
Fichier Edition Format Affichage ?  
5 % nombre de points  
1. 2. 3. % valeurs de x, y et z pour le point 1  
5. 6. 4. % valeurs de x, y et z pour le point 2  
6. 3. 7. % valeurs de x, y et z pour le point 3  
2. 6. 2. % valeurs de x, y et z pour le point 4  
1. 1. 3. % valeurs de x, y et z pour le point 5
```

D:\Octave\13-textscan\Main.m - Notepad++

```
File Edit Search View Encoding Language Settings Macro Run Plugins Window ? X  
Main.m  
1 fid=fopen("fichier.dat","r");  
2 data=textscan(fid,'%f',1); n=data{1};  
3 for i=1:n  
4 data=textscan(fid,'%f %f %f',1);  
5 x(i)=data{1};  
6 y(i)=data{2};  
7 z(i)=data{3};  
8 end  
9 fclose(fid);  
10  
11 printf("n = %d\n",n);  
12 for i=1:n  
13 printf(' (x,y,z) = (%f,%f,%f)\n',x(i),y(i),z(i));  
14 end  
15
```

length : Ln :15 Col:1 Sel:0 | 0 Dos\Windows ANSI as UTF-8

```
data=textscan(fid,'%f %f %f',1, ...  
'commentStyle','%', ...  
'delimiter','\n');
```

Octave

```
octave:5> Main  
n = 5  
(x,y,z) = (1.000000,2.000000,3.000000)  
(x,y,z) = (5.000000,6.000000,4.000000)  
(x,y,z) = (6.000000,3.000000,7.000000)  
(x,y,z) = (2.000000,6.000000,2.000000)  
(x,y,z) = (1.000000,1.000000,3.000000)  
octave:6>
```

# Procédures

Chapitre 8

Le programme principal « Main.m » peut faire appel à des procédures pour l'exécution d'une tâche particulière.

Ces procédures peuvent se charger par exemple du calcul d'une fonction  $f(x)$ .

Les procédures doivent être reprises dans des fichiers séparés. Le nom de ces fichiers doit correspondre au nom de la procédure.

Par exemple, pour une fonction  $f(x)$ , le fichier s'appellera f.m.

# Exemple

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

```
if (x~=0.)  
    y = sin(x)/x;  
else  
    y = 1.;
```

```
end
```

```
return;
```

d:\Octave\14-Procedures\Main.m - Notepad++

File Edit Search View Encoding Language Settings Macro Run Plugins Window ? X

Main.m

```
1 clear ; clc;
2
3 x=input("Entrez x : ");
4 printf("\nLa valeur de f(x) est : %f\n", f(x));
5
6 x=-20:.1:20;
7 for i=1:length(x)
8   y(i)=f(x(i));
9 end
10 plot(x,y)
11 xlabel("X")
12 ylabel("Y")
13 title("Representation de y=f(x)")
14 grid
15
16 print -dpSC Plot.ps;
17
```

length: Ln:17 Col:1 Sel:0|0

D:\Octave\14-Procedures\f.m - Notepad++

File Edit Search View Encoding Language Settings Macro Run Plugins Window ? X

f.m

```
1 function [y] = f (x)
2
3 if (x~=0.)
4   y = sin(x)/x;
5 else
6   y = 1.;
7 end
8
9 return;
```

length: Ln:1 Col:1 Sel:0|0 Dos\Windows ANSI as UTF-8 INS

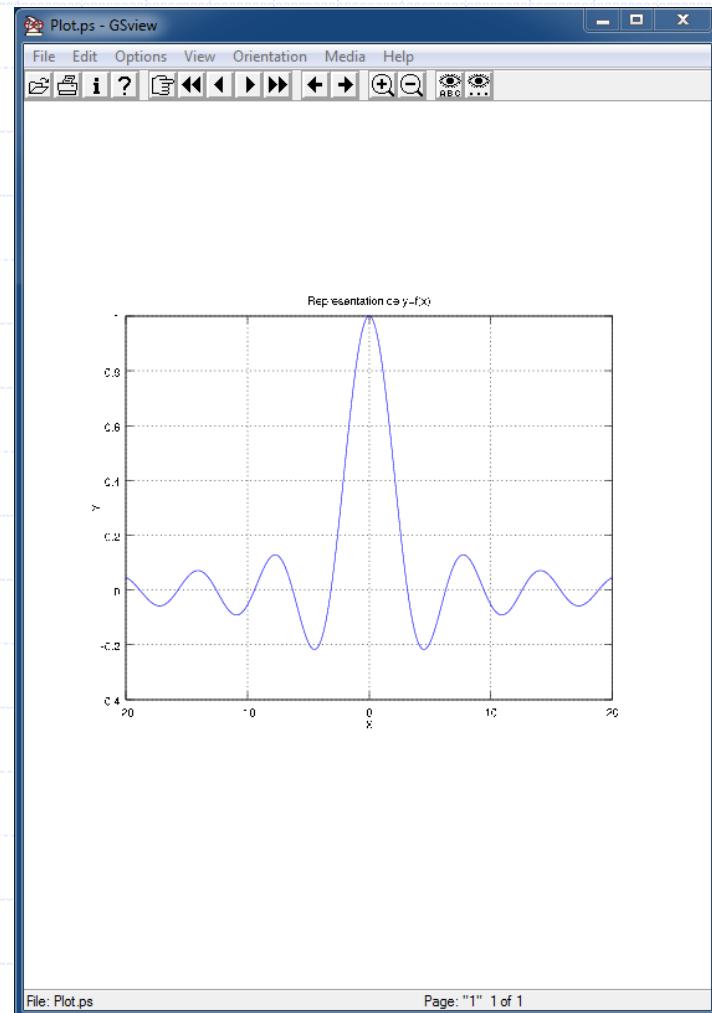
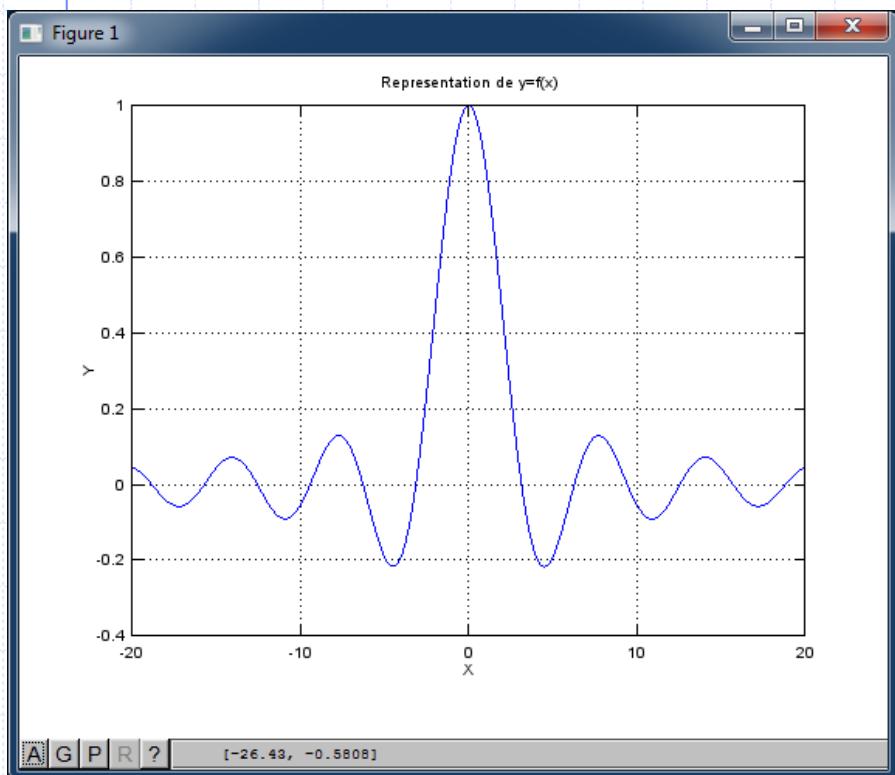
Octave

```
octave:2> cd d:  
octave:3> cd Octave  
octave:4> cd 14-Procedures  
octave:5> dir  
. . . Main.m f.m  
octave:6> Main
```



Octave

```
Entrez x : 5.  
La valeur de f(x) est : -0.191785
```



# Fonctions à plusieurs sorties

```
function [x,y] = f (n)
```

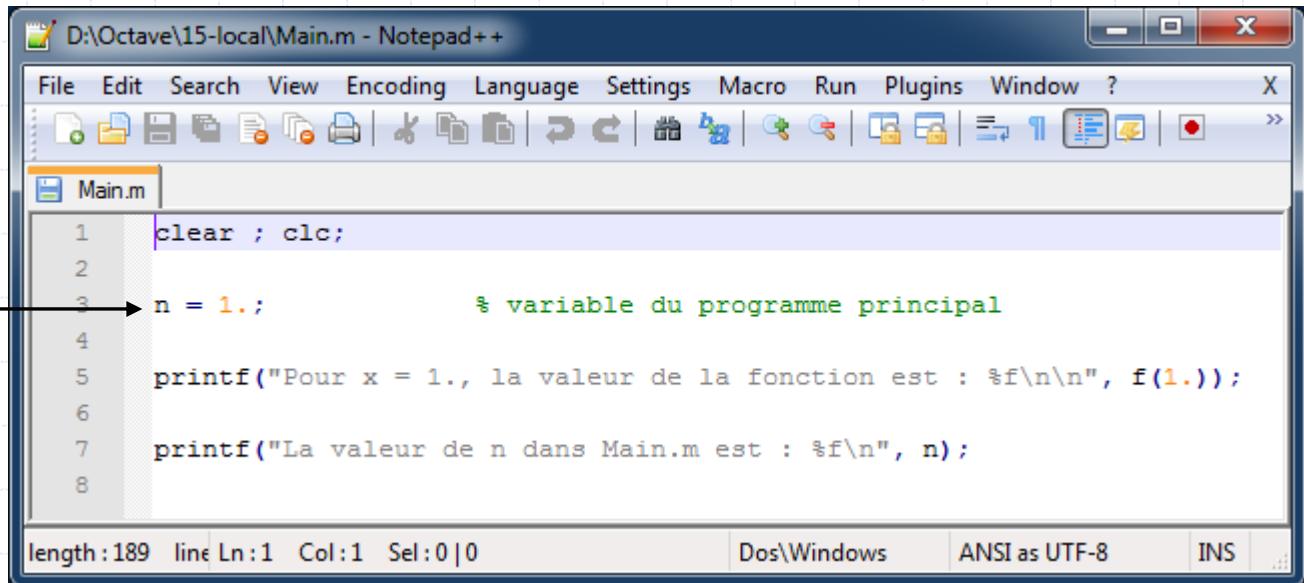
```
x=2.^n;  
y=1.+4.*n;
```

```
return;
```

nargin	% nombre d'arguments d'entrée
nargout	% nombre d'arguments de sortie

# Variables locales

Ce « n » est une variable du programme principal.



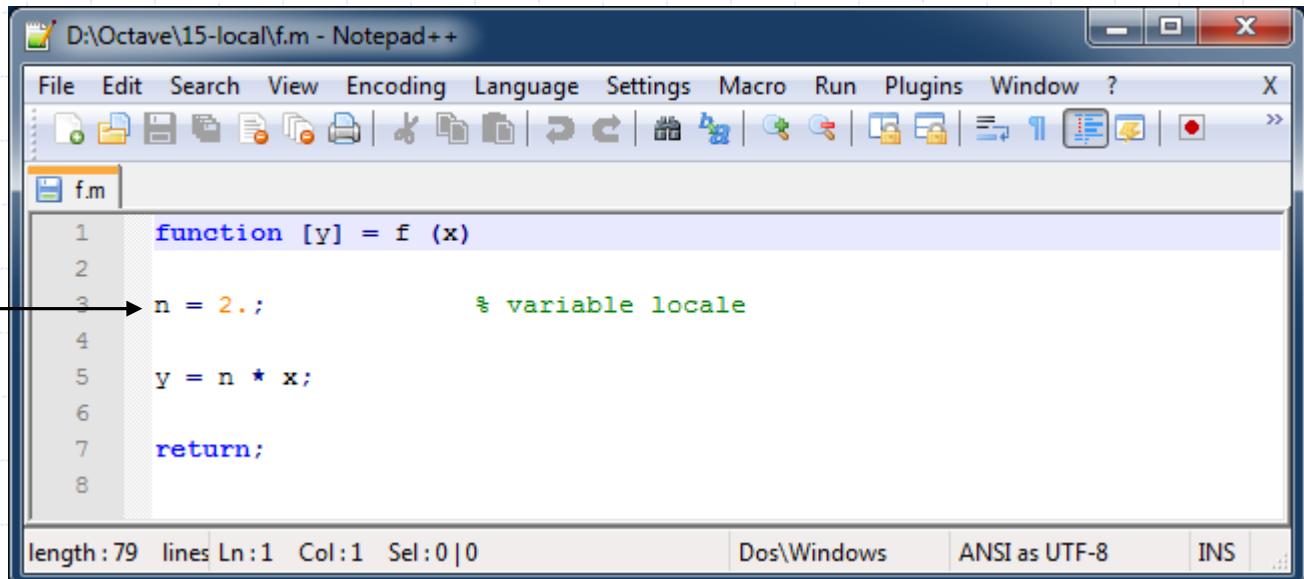
D:\Octave\15-local\Main.m - Notepad++

```
File Edit Search View Encoding Language Settings Macro Run Plugins Window ? X
Main.m
1 clear ; clc;
2
3 → n = 1.; % variable du programme principal
4
5 printf("Pour x = 1., la valeur de la fonction est : %f\n\n", f(1.));
6
7 printf("La valeur de n dans Main.m est : %f\n", n);
8
```

length : 189 line Ln :1 Col :1 Sel :0 | 0 Dos\Windows ANSI as UTF-8 INS

This screenshot shows the Octave script file Main.m in Notepad++. The variable n is defined on line 3 as 1., which is annotated with a callout arrow from the text box above. The script also contains printf statements to output the value of the function f(1.) and the value of n itself.

Ce « n » est une variable de la fonction.

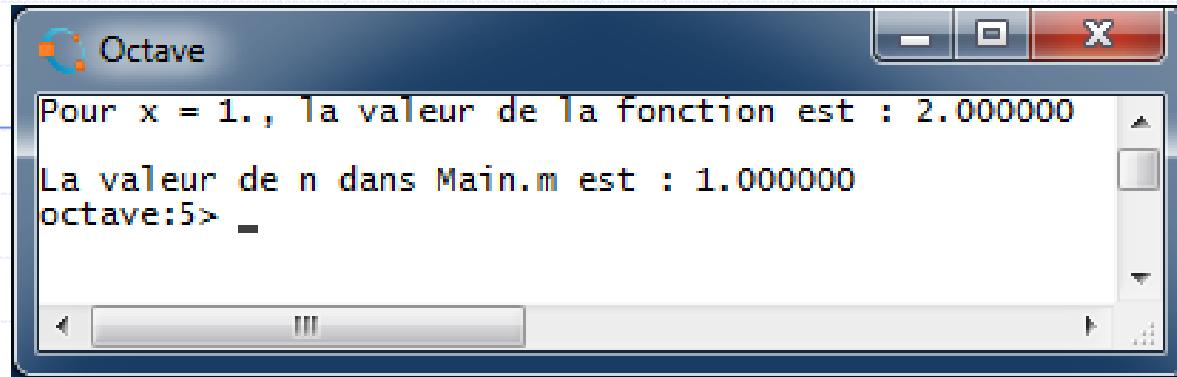


D:\Octave\15-local\f.m - Notepad++

```
File Edit Search View Encoding Language Settings Macro Run Plugins Window ? X
f.m
1 function [y] = f (x)
2
3 → n = 2.; % variable locale
4
5 y = n * x;
6
7 return;
8
```

length : 79 lines Ln :1 Col :1 Sel :0 | 0 Dos\Windows ANSI as UTF-8 INS

This screenshot shows the Octave function file f.m in Notepad++. The variable n is defined on line 3 as 2., which is annotated with a callout arrow from the text box above. The function takes an input x and returns the value of n multiplied by x.



C'est le « n » défini dans la fonction qui a été utilisé.

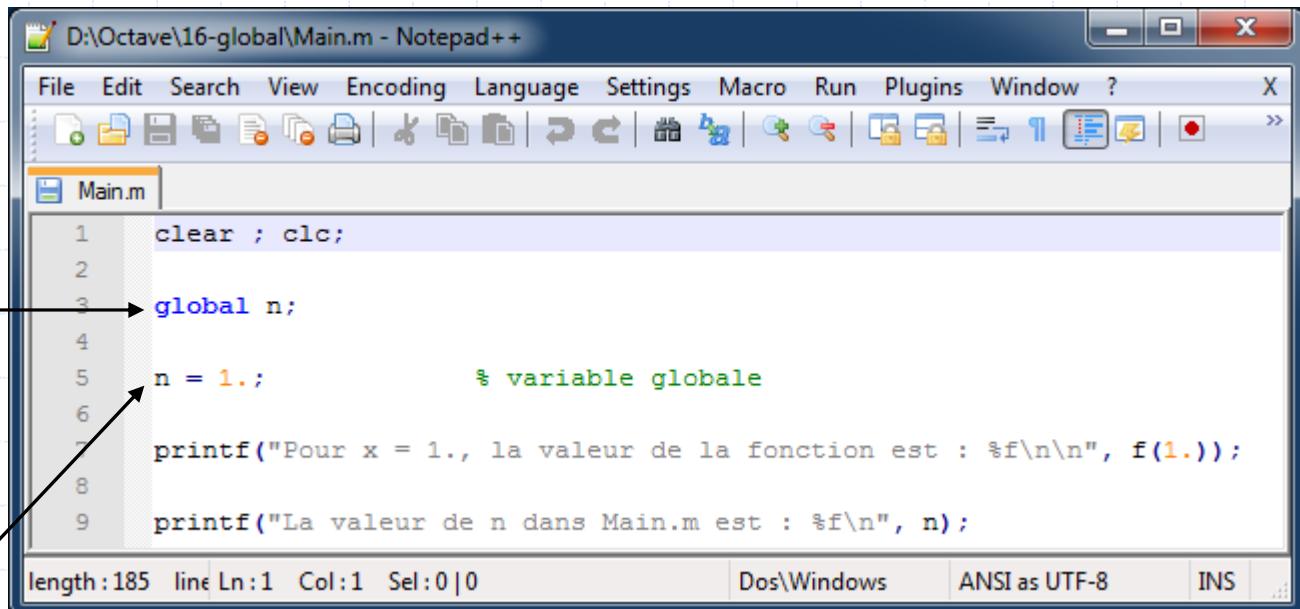
Il s'agit d'une « variable locale ». Cette valeur de « n=2. » n'est connue en effet qu'à l'intérieur de la fonction.

La variable « n » utilisée dans la fonction est différente de la variable « n » utilisée dans le programme principal. Cette dernière a conservé sa valeur de « n=1. ».

# Variables globales

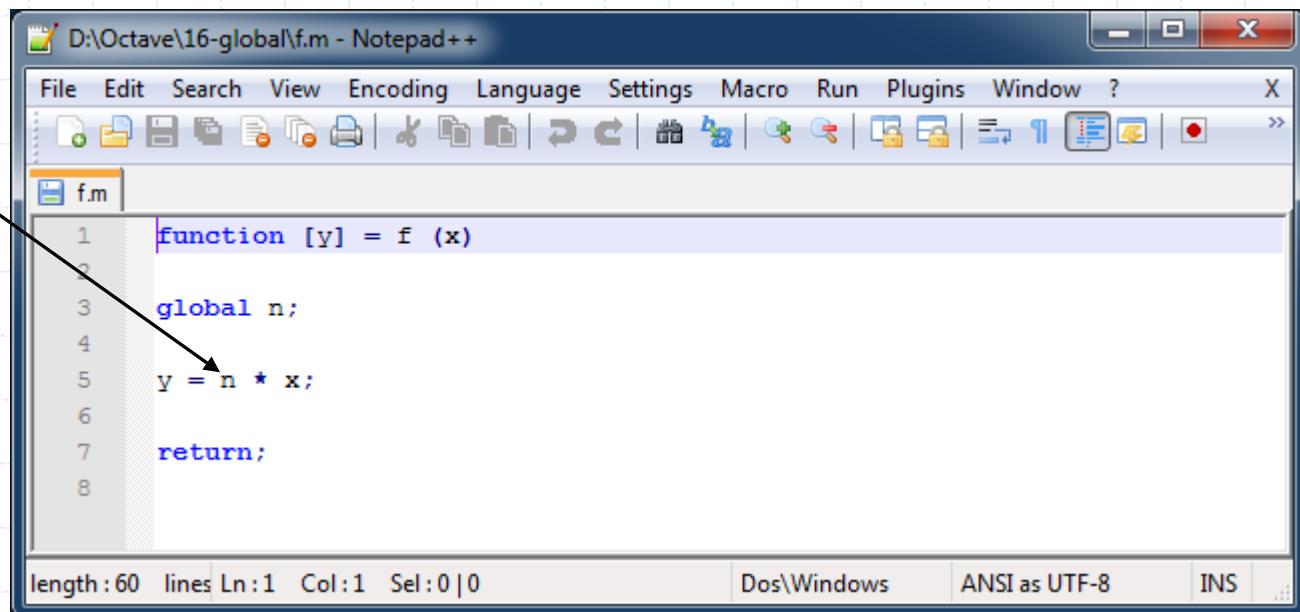
Instruction  
« global »  
à utiliser.

Ce « n » est  
une variable  
globale.



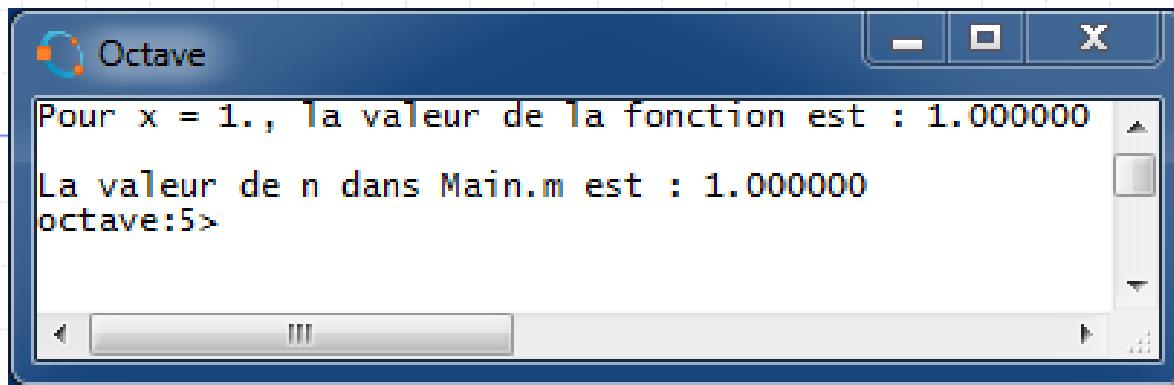
```
D:\Octave\16-global\Main.m - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
Main.m
1 clear ; clc;
2
3 → global n;
4
5 n = 1.; % variable globale
6
7 printf("Pour x = 1., la valeur de la fonction est : %f\n\n", f(1.));
8
9 printf("La valeur de n dans Main.m est : %f\n", n);

length : 185 line Ln : 1 Col : 1 Sel : 0 | 0 Dos\Windows ANSI as UTF-8 INS
```



```
D:\Octave\16-global\f.m - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
f.m
1 function [y] = f (x)
2
3 global n;
4
5 y = n * x;
6
7 return;

length : 60 lines Ln : 1 Col : 1 Sel : 0 | 0 Dos\Windows ANSI as UTF-8 INS
```



Grâce à l'instruction « global n » qui est reprise dans le programme principal et la fonction, « n » est ici une « variable globale ».

Ce « n » reçoit une valeur dans le programme principal. Cette valeur est connue dans la fonction.

# Algèbre linéaire

## Chapitre 9

Comment calculer le produit matriciel entre

$$A = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 4 & 8 \\ 1 & 3 & 9 & 27 \\ 1 & 4 & 16 & 64 \end{pmatrix} \text{ et } B = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 4 & 7 & 11 \\ 3 & 7 & 14 & 25 \\ 4 & 11 & 25 & 50 \end{pmatrix}$$

D:\Octave\18-produit>Main.m - Notepad++

File Edit Search View Encoding Language Settings Macro Run Plugins Window ? X

Main.m

```
1 clear ; clc
2
3 A = [ 1, 1, 1, 1 ; ...
4     1, 2, 4, 8 ; ...
5     1, 3, 9, 27 ; ...
6     1, 4, 16, 64 ]
7
8 B = [ 1, 2, 3, 4 ; ...
9     2, 4, 7, 11 ; ...
10    3, 7, 14, 25 ; ...
11    4, 11, 25, 50 ]
12
13 AB=A★B
14
```

length: Ln:1 Col:1 Sel:0|0 Dos\Windows ANSI as UTF-8 INS

Octave

```
A =
1 1 1 1
1 2 4 8
1 3 9 27
1 4 16 64

B =
1 2 3 4
2 4 7 11
3 7 14 25
4 11 25 50

AB =
10 24 49 90
49 126 273 526
142 374 825 1612
313 834 1855 3648

octave:5> -
```

Comment résoudre le système d'équations linéaires :

$$\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 4 & 8 \\ 1 & 3 & 9 & 27 \\ 1 & 4 & 16 & 64 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} -2 \\ 0 \\ 12 \\ 40 \end{pmatrix}$$

D:\Octave\18-systeme\_lineaire\Main.m - Notepad++

File Edit Search View Encoding Language Settings Macro Run Plugins Window ? X

Main.m

```
1 clear ; clc
2
3 A = [ 1, 1, 1, 1 ; ...
4   1, 2, 4, 8 ; ...
5   1, 3, 9, 27 ; ...
6   1, 4, 16, 64 ]
7
8 b = [ -2 ; 0 ; 12 ; 40 ]
9
10 x=A\b
```

length: Ln:1 Col:1 Sel:0|0 Dos\Windows ANSI as UTF-8 INS

Octave

```
A =
1 1 1 1
1 2 4 8
1 3 9 27
1 4 16 64

b =
-2
0
12
40

x =
0
-2
-1
1

octave:5>
```

Comment calculer les valeurs propres et les vecteurs propres de la matrice

$$\begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & 2 & 1 & 0 \\ 0 & 1 & 2 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}$$

D:\Octave\19-système\_propre\Main.m - Notepad++

File Edit Search View Encoding Language Settings Macro Run Plugins Window ? X

Main.m

```
1 clear ; clc
2
3 A = [ 1, 1, 0, 0 ; ...
4      1, 2, 1, 0 ; ...
5      0, 1, 2, 1 ; ...
6      0, 0, 1, 2 ]
7
8 [V,lambda]=eig(A);
9
10 for i=1:4
11   fprintf('Valeur propre : %f\n', lambda(i,i));
12   fprintf('Vecteur propre : (%f,%f,%f,%f)\n\n', V(1:4,i));
13 end
14
```

length:1 Ln:1 Col:1 Sel:0|0 Dos\Windows ANSI as UTF-8 INS

Octave

```
A =
1 1 0 0
1 2 1 0
0 1 2 1
0 0 1 2

Valeur propre : 0.120615
Vecteur propre : (0.656539,-0.577350,0.428525,-0.228013)

Valeur propre : 1.000000
Vecteur propre : (0.577350,-0.000000,-0.577350,0.577350)

Valeur propre : 2.347296
Vecteur propre : (-0.428525,-0.577350,0.228013,0.656539)

Valeur propre : 3.532089
Vecteur propre : (0.228013,0.577350,0.656539,0.428525)

octave:6> -
```

# Interpolations

Chapitre 10

# interp1

La fonction interp1 permet d'interpoler un ensemble de points. Supposons que l'on dispose d'un ensemble de points avec des coordonnées données par X et Y.

La fonction qui interpolate ces points peut être évaluée pour une valeur de x quelconque par

```
y=interp1(X,Y,x)
```

La méthode d'interpolation par défaut est l'interpolation linéaire. Pour une interpolation par la méthode spline, vous pouvez faire :

```
y=interp1(X,Y,x,'spline')
```

x et y peuvent désigner une liste de points

Voir aussi : interp2, interp3, interpn, spline

D:\Octave\21-interp1>Main.m - Notepad++

File Edit Search View Encoding Language Settings Macro Run Plugins Window ?

Main.m

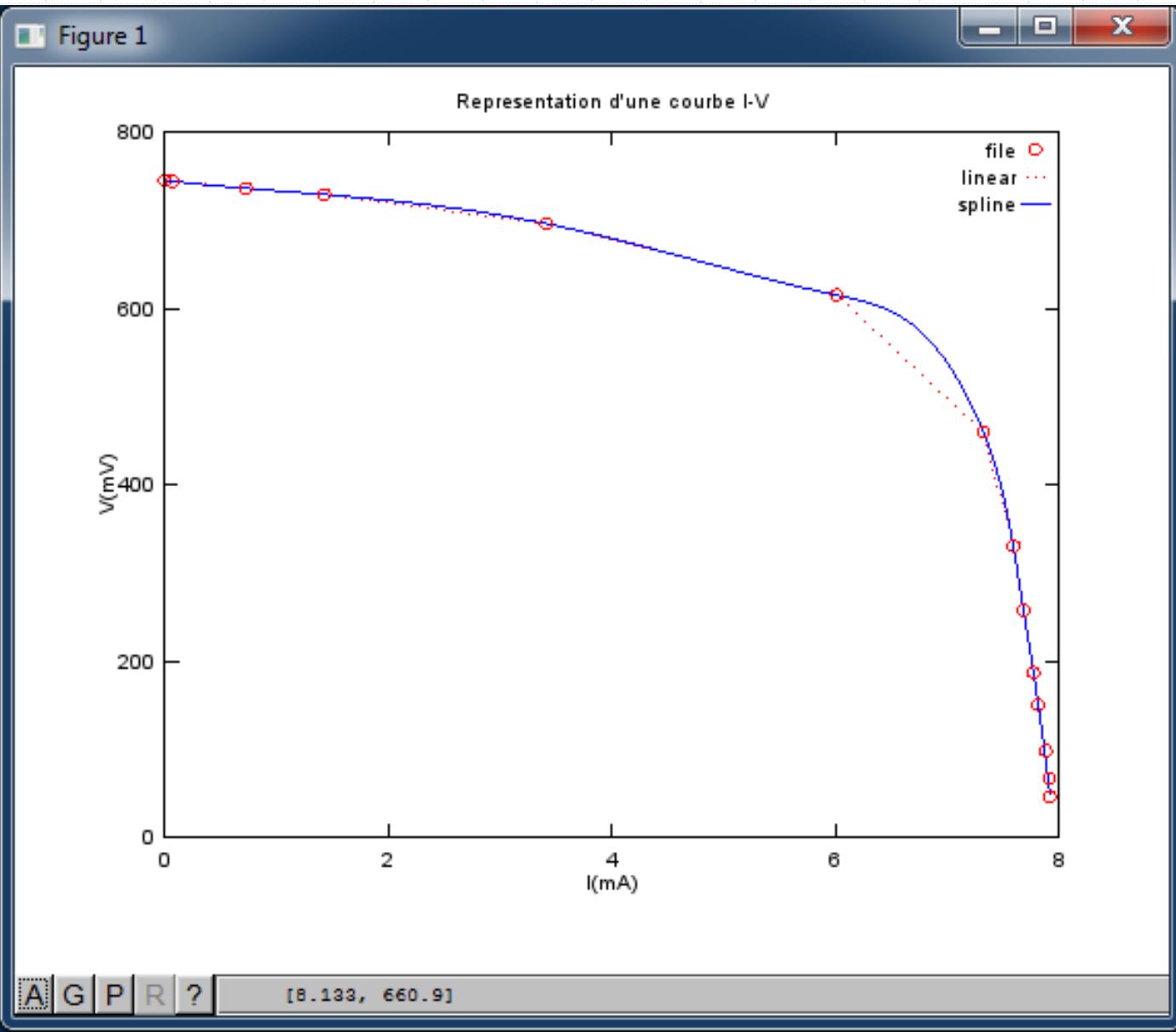
```
1 clear ; clc;
2
3 fid=fopen("cell.dat","r");           % lecture du fichier
4 data=textscan(fid,'%f',1) ; n=data{1};
5 for i=1:n
6     data=textscan(fid,'%f %f',1);
7     X(i)=data{1};                   % coordonnées X et Y des
8     Y(i)=data{2};                   % points à interpoler
9 end
10 xlabel=fgetl(fid);
11 ylabel=fgetl(fid);
12 Title =fgetl(fid);
13 fclose(fid);
14
15 x=linspace(min(X),max(X),100);    % grille fine
16 y_linear=interp1(X,Y,x);          % interpolation linéaire
17 y_spline=interp1(X,Y,x,'spline'); % interpolation par spline
18
19 figure;
20 plot(X,Y,'ro')                  % points du fichier
21 hold
22 plot(x,y_linear,'r:')           % interpolation linéaire
23 plot(x,y_spline,'b-')           % interpolation par spline
24 legend('file','linear','spline')
25 xlabel(Xlabel)
26 ylabel(Ylabel)
27 title>Title
28 print -dpsc Interpolation.ps
29
```

length:758 lines:2 Ln:1 Col:1 Sel:0 | 0 Dos\Windows ANSI as UTF-8 INS

cell.dat - Bloc-notes

Fichier Edition Format Affichage ?

14	0.00	745.
	0.07	744.
	0.73	736.
	1.43	729.
	3.41	696.
	6.01	615.
	7.32	460.
	7.59	331.
	7.68	258.
	7.77	187.
	7.81	151.
	7.88	98.6
	7.91	67.6
	7.92	46.4
I(mA)		
V(mV)		
Representation d'une courbe I-V		



# Intégration

## Chapitre 11

Comment calculer une intégrale telle que

$$\int_a^b f(x)dx$$

Exemple :

$$\int_0^\pi \sin(x)dx = 2$$

# quad(@f,a,b)

```
D:\Octave\21-quad\Main.m - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Window ? X
Main.m
1 clear ; clc;
2
3 a = 0.;
4 b = 3.1415926535897932;
5
6 integrale=quad(@f,a,b)
7
length: Ln:1 Col:1 Sel:0 | 0 Dos\Windows ANSI as UTF-8 INS
```

@f fait référence à une fonction  $f(x)$ , qui doit être spécifiée dans un fichier f.m

```
Octave
integrale = 2
octave:5>
```

help quad

```
D:\Octave\21-quad\f.m - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Window ? X
f.m
1 function [y]=f(x)
2
3 y = sin(x);
4
length: Ln:1 Col:1 Sel:0 | 0 Dos\Windows ANSI as UTF-8 INS
```

# Optimisation

## Chapitre 12

## Quelques fonctions utiles :

`x=fminbnd(@f,x1,x2)`

% cherche le minimum de  $f(x)$

% entre  $x_1$  et  $x_2$

`x=fminunc(@f,x0)`

% cherche le minimum de  $f(x)$

% en partant de  $x_0$ .  $x$  et  $x_0$

% peuvent être des vecteurs

% alternative (moins bonne)

% à fminunc

`x=fminsearch(@f,x0)`

@f fait référence à une fonction  $f(x)$ , qui doit être spécifiée dans un fichier f.m

Voir aussi : linprog, quadprog et fmincon pour des minimisations sous contrainte.

Pour spécifier des options :

```
options=optimset('TolX',1.E-6,'TolFun',1.E-8,'MaxFunEvals',5000);
```

% TolX=tolérance sur les composantes de x

% TolFun=tolérance sur les valeurs de la fonction

% MaxFunEvals=nb max d'évaluations de la fonction

```
x=fminunc(@f,x0,options) % cherche le minimum de f(x)
```

% en partant de x0 et en

% tenant compte des options

help fminunc

help optimset

D:\Octave\22-fminunc\Main.m - Notepad++

File Edit Search View Encoding Language Settings Macro Run Plugins Window ? X

Main.m

```
1 clear ; clc ;
2
3 x0 = [0.,0.]; % point de départ
4
5 options=optimset('TolX',1.E-6,'TolFun',1.E-8,'MaxFunEvals',5000);
6 % TolX = tolérance sur les valeurs de x
7 % TolFun = tolérance sur les valeurs de la fonction
8 % MaxFunEvals = nb max d'évaluations de la fonction
9
10 x = fminunc (@f, x0, options); % recherche du minimum de f
11
12 fprintf('Le minimum de la fonction f correspond a x_1 = %f\n',x(1));
13 fprintf(' et x_2 = %f\n',x(2));
```

MATRIX L length: 515 lines: 14 Ln:14 Col:1 Sel:0|0 Dos\Windows ANSI as UTF-8 INS

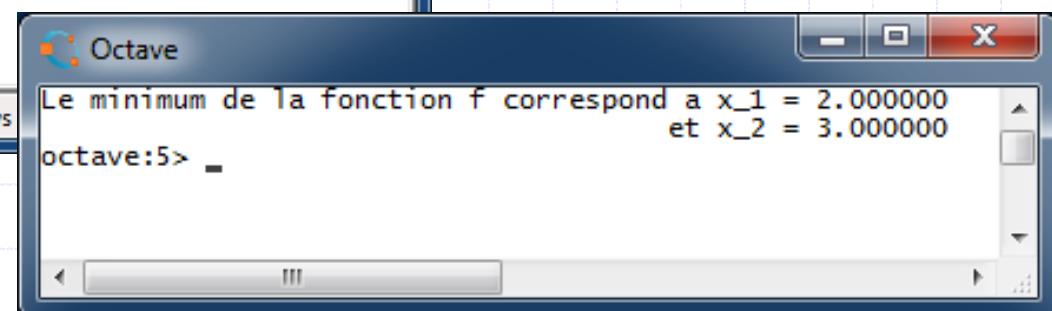
D:\Octave\22-fminunc\f.m - Notepad++

File Edit Search View Encoding Language Settings Macro Run Plugins Window ? X

f.m

```
1 function [result] = f(x)
2
3 result = (x(1)-2.)**2+(x(2)-3.)**2;
4
```

length: Ln:1 Col:1 Sel:0|0 Dos\Windows



# Systèmes d'équations différentielles

## Chapitre 13

Comment résoudre le système d'équations différentielles

$$\frac{dY}{dt} = f(t, Y)$$

où  $Y(t)$  est un vecteur à  $n$  composantes et  $f(t, Y)$  est une fonction donnée ? On prend  $Y_0 = Y(t_{\text{start}})$  comme condition initiale. On souhaite déterminer la valeur de  $Y(t)$  pour  $t = t_{\text{end}}$ .

# ode45

T contient en sortie les valeurs du temps t

Y contient en sortie les valeurs correspondantes de Y(t)

$[T, Y] = \text{ode45}(@f, \text{tspan}, Y_0)$

$@f$  fait référence à une fonction  $f(t, Y)$  qui doit être programmée dans un fichier f.m. Pour des valeurs d'entrée du temps t et de Y(t), elle doit fournir les valeurs de  $dY/dt$ .

Y<sub>0</sub> spécifie les conditions initiales, càd Y(tstart).

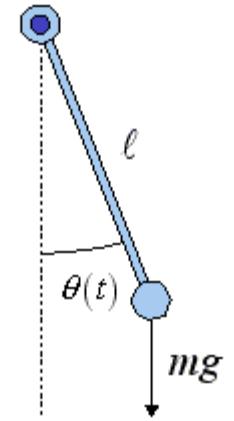
tspan=[tstart,tend] est un vecteur à 2 composantes qui spécifie les valeurs de départ et d'arrivée pour le temps t.

Considérons un pendule de longueur  $l$  et de masse  $m$ .

Appelons  $\theta(t)$  l'angle entre ce pendule et la verticale. On peut montrer qu'en l'absence de frottement l'angle  $\theta(t)$  satisfait l'équation

$$\frac{d^2\theta(t)}{dt^2} = -\frac{g}{l} \sin[\theta(t)].$$

On souhaite étudier le mouvement de ce pendule sur une période de 12 s, sachant que le pendule démarre avec une valeur initiale de  $\theta(0) = \pi/2$ . On prendra  $l=5$  m et  $m=2$  kg.



On peut mettre l'équation différentielle

$$\frac{d^2\theta(t)}{dt^2} = -\frac{g}{l} \sin[\theta(t)].$$

sous la forme

$$\frac{dy_1}{dt} = y_2,$$

$$\frac{dy_2}{dt} = -\frac{g}{l} \sin y_1,$$

où  $y_1(t)$  correspond à  $\theta(t)$  et  $y_2(t)$  à  $\frac{d\theta(t)}{dt}$ .

La condition initiale pour  $Y(t)$  est donnée par  $Y_0 = \begin{pmatrix} \pi/2 \\ 0 \end{pmatrix}$ .

D:\Octave\23-ode45\Main.m - Notepad++

File Edit Search View Encoding Language Settings Macro Run Plugins Window ? X

Main.m

```
1 clear ; clc ;
2
3 global g l
4
5 g = 9.81; % m/s^2
6 l = 5.; % m
7
8 tstart = 0.; % s
9 tend = 12.; % s
10 options = odeset('RelTol',1.E-6,'AbsTol',1.E-6,'InitialStep',1.E-3,'MaxStep',0.05);
11 [T,Y] = ode45 (@f,[tstart,tend],[pi/2,0.],options);
12
13 plot(T(:,1),Y(:,1))
14 xlabel("t (s)")
15 ylabel("theta (radian)")
16 title("Time-evolution of the pendulum angle")
17 print -dpssc Theta.ps
18
```

M length:379 lines:18 Ln:1 Col:1 Sel:

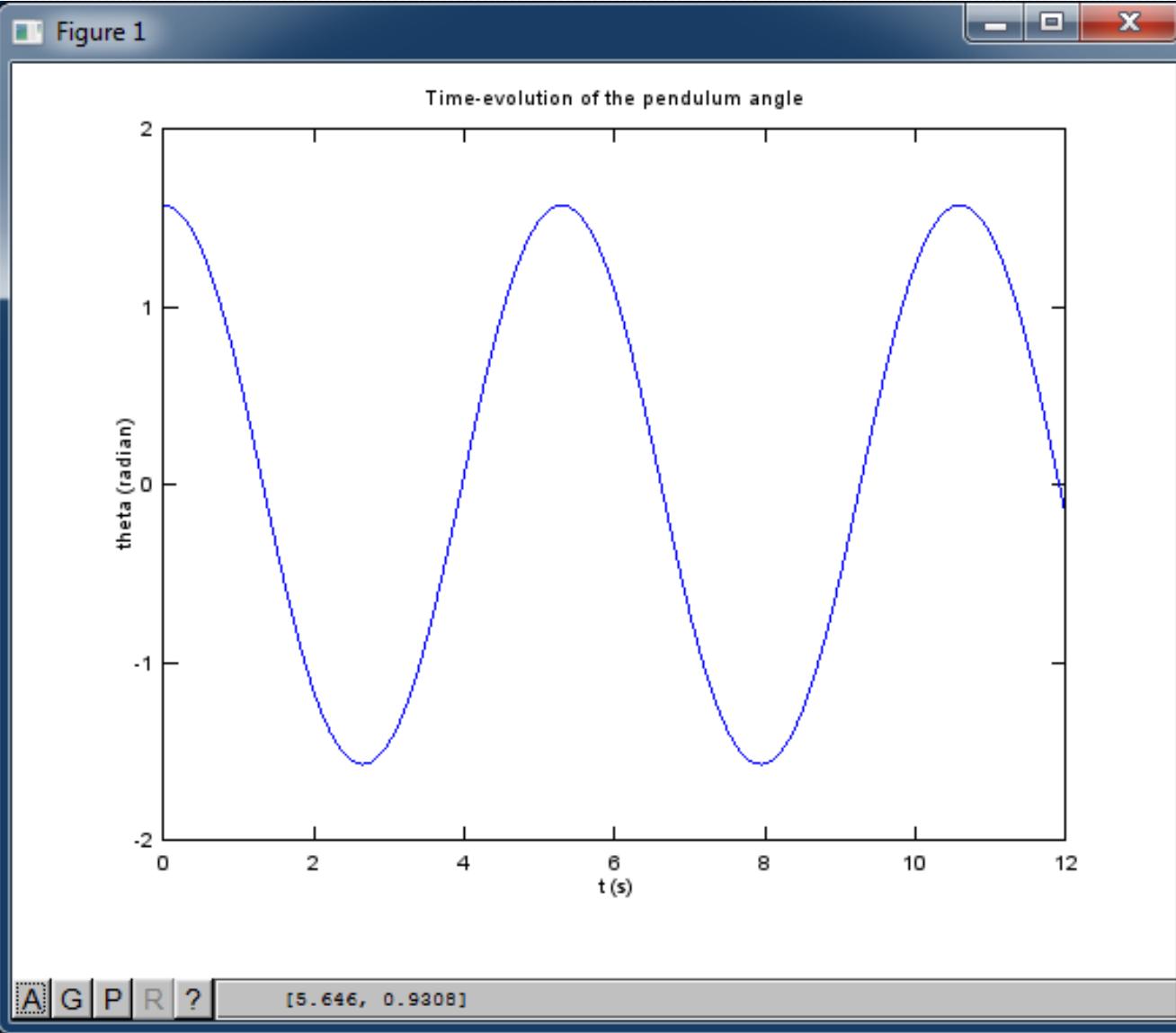
D:\Octave\23-ode45\f.m - Notepad++

File Edit Search View Encoding Language Settings Macro Run Plugins Window ? X

f.m

```
1 function [dYdt] = f (t,Y)
2
3 global g l
4
5 dYdt(1) = Y(2);
6 dYdt(2) = -(g/l)*sin(Y(1));
7
```

length: Ln:1 Col:1 Sel:0|0 Dos\Windows ANSI as UTF-8 INS





FR EN

# Convode

Introduction

Menu

e.d.o.

e.d.p.

s.e.d.o.

s.e.d.p.

pfaff

Vous souhaitez résoudre:

- Une équation différentielle ordinaire (E.D.O.)
- Une équation aux dérivées partielles (E.D.P.)
- Un système d'équations différentielles ordinaires (S.E.D.O.)
- Un système d'équations aux dérivées partielles (S.E.D.P.)
- Une équation pour la dérivée totale (Pfaff)

Poursuivre

Annuler



Responsable

Réalisé avec le logiciel [Webware](#), Septembre 2004

# Transformées de Fourier

Chapitre 14

Soit un signal  $x(t)$  que l'on a échantillonné selon des valeurs  $x_k = x(k \cdot \frac{T}{n})$  pour  $k = 0, 1, \dots, n - 1$ .  $T$  désigne la durée de l'échantillonage et  $n$  le nombre de points ( $n$  doit être une puissance de 2).

L'algorithme de "Fast Fourier Transform" fournit les coefficients de Fourier  $X_l = X(l \cdot \frac{1}{T})$  de ce signal échantillonné, pour  $l = 0, 1, \dots, n - 1$ .

Les coefficients de Fourier  $X_l$  sont périodiques ( $X_{l+n} = X_l$ ). Cette situation reflète la périodicité de la transformée de Fourier  $X(f)$  du signal échantillonné ( $X(f) = X(f + \frac{n}{T})$ ).

Il est indispensable de suivre un cours sur les transformées de Fourier rapides pour utiliser les outils de ce chapitre correctement.

```
X = fft(x)    % Fast Fourier Transform
x = ifft(X)   % Inverse Fast Fourier Transform
```

Soit donc un signal échantillonné selon  $x_k = x(k \cdot \frac{T}{n})$ , où  $k = 0, 1, \dots, n - 1$ .

Fast Fourier Transform :

$$X_l = \sum_{k=0}^{n-1} x_k e^{-i2\pi \frac{kl}{n}}$$

pour  $l = 0, 1, \dots, n - 1$ .

Inverse Fast Fourier Transform :

$$x_k = \frac{1}{n} \sum_{l=0}^{n-1} X_l e^{i2\pi \frac{kl}{n}}$$

pour  $k = 0, 1, \dots, n - 1$ .

Les coefficients  $X_l$  correspondent aux valeurs de la transformée de Fourier  $X(f)$  du signal  $x(t)$ , avec  $X_l = X(l \cdot \frac{1}{T})$ .

Le facteur  $1/n$  n'apparaît pas où on le voudrait.

```

X = fftshift(fft(x)) % Fast Fourier Transform with
                      % shift of the frequencies
x = ifft(ifftshift(X)) % Inverse Fast Fourier Transform
                      % from shifted frequencies

```

La fonction `fftshift` a pour effet de translater les coefficients  $X_l$  de sorte qu'ils correspondent à l'intervalle de fréquences  $[-\frac{n}{2T}, \frac{n}{2T}]$  plutôt que  $[0, \frac{n}{T}]$ .

Les coefficients  $X_l$  correspondent alors aux valeurs de la transformée de Fourier  $X(f)$  pour des fréquences  $f$  données par  $f = -\frac{n}{2T} + l \cdot \frac{1}{T}$ .

Si `fftshift` a été utilisé avec `fft` pour la transformation directe, il faut utiliser `ifftshift` avec `ifft` pour la transformation inverse.

Comment calculer la transformée de Fourier d'un signal

$$x(t) = 1 \times \cos(2\pi \cdot 5 \cdot t) + 2 \times \sin(2\pi \cdot 10 \cdot t)$$

qui serait échantillonné sur un intervalle de temps  $T$  de 8 secondes par  $n=256$  valeurs.

D:\Octave\22-fft\Main.m - Notepad++

File Edit Search View Encoding Language Settings Macro Run Plugins Window ?

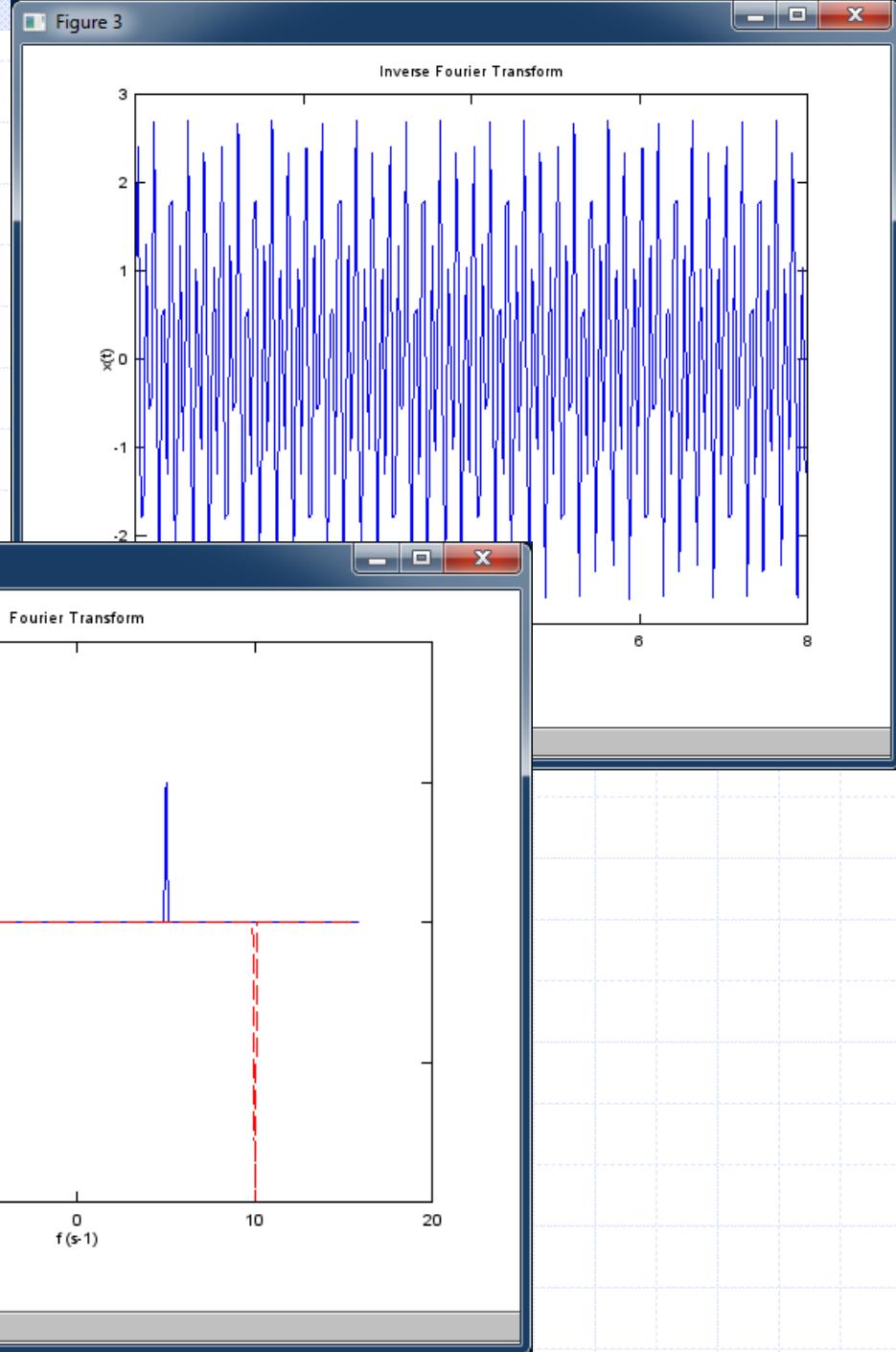
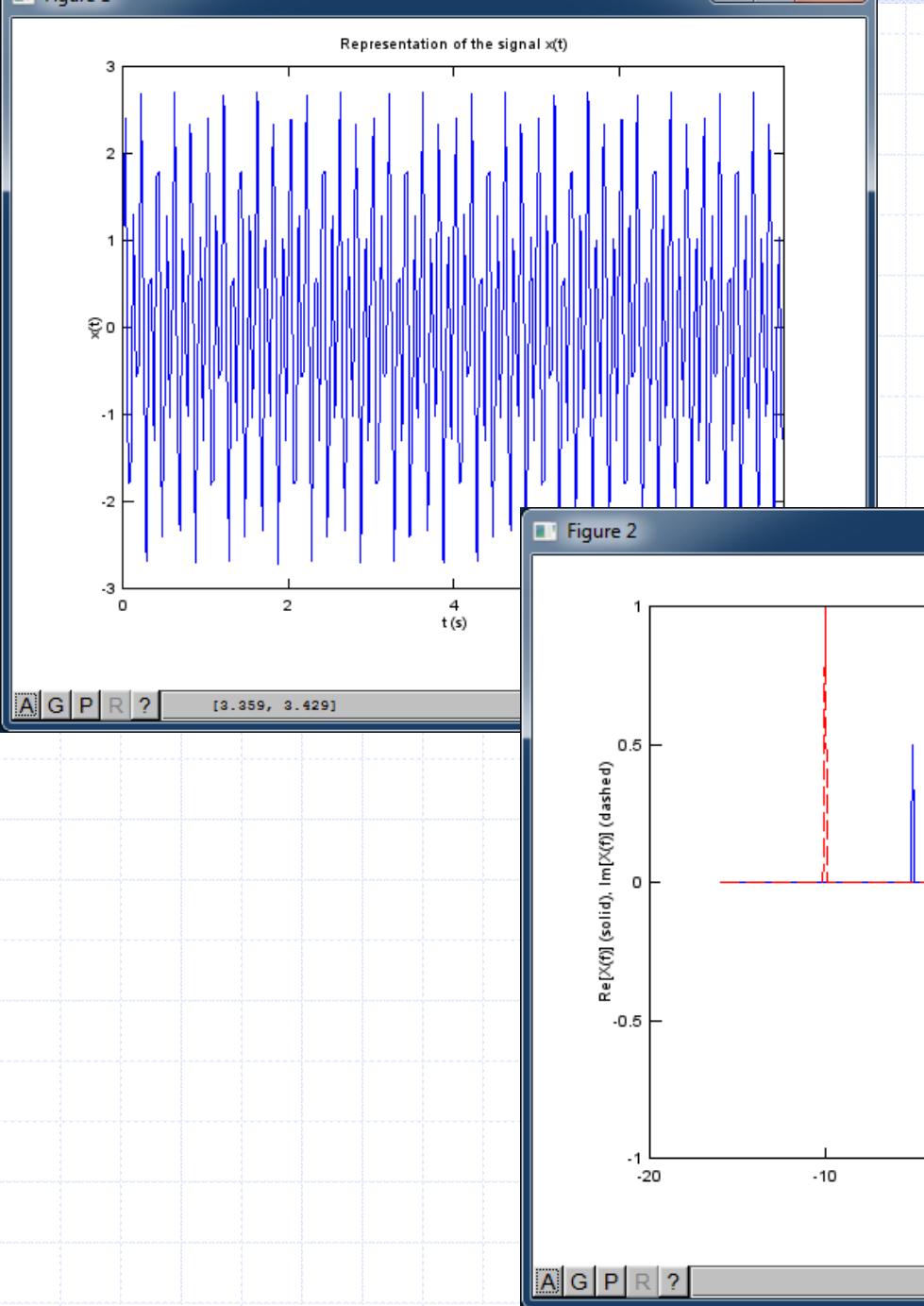
Main.m

```
1 clear ; clc;
2
3 T = 8.; % durée de l'échantillonage (s)
4 n = 256; % nombre de points
5
6 for k=1:n % valeurs du signal x(t)
7 t(k) = (k-1)*T/n;
8 x(k) = 1.*cos(2*pi*5.*t(k))+2.*sin(2.*pi*10.*t(k));
9 end
10 figure
11 plot(t,x)
12 xlabel("t (s)")
13 ylabel("x(t)")
14 title("Representation of the signal x(t)")
15 print -dpsc Signal.ps
16
17 X=fftshift(fft(x))/n; % Fourier Transform (shifted in frequency)
18 for l=1:n
19 f(l)=-n/(2*T)+(l-1)/T; % shifted frequencies
20 end
21 figure
22 plot(f,real(X), 'b')
23 hold
24 plot(f,imag(X), 'r--')
25 xlabel("f (s-1)")
26 ylabel("Re[X(f)] (solid), Im[X(f)] (dashed)")
27 title("Fourier Transform")
28 print -dpsc FourierTransform.ps
29
30 x=ifft(ifftshift(X))*n; % Inverse Fourier Transform
31 figure
32 plot(t,x)
33 xlabel("t (s)")
34 ylabel("x(t)")
35 title("Inverse Fourier Transform")
36 print -dpsc InverseFourierTransform.ps
```

length:836 line: Ln:1 Col:1 Sel:0|0 Dos\Windows ANSI as UTF-8 INS

Octave n'accepte pas des vecteurs x ou X dont les indices iraient de 0 à n-1. Ces indices doivent aller de 1 à n.

Figure 1



## Fonction utile pour l'utilisation de FFT :

p=nextpow2(n)

% plus petit p tel que  $2^p \geq n$

## Filtres utiles en traitement du signal :

w=hanning(n)

% fenêtre de Hanning de longueur n

w=hamming(n)

% fenêtre de Hamming

w=blackman(n)

% fenêtre de Blackman

w=chebwin(n,r)

% fenêtre de Chebyshev

w=boxcar(n)

% fenêtre rectangulaire

w=bartlett(n)

% fenêtre triangulaire de Bartlett

w=kaiser(n,beta)

% fenêtre de Kaiser

# Solutions d'équations non linéaires

## Chapitre 15

# fzero et fsolve

Soit  $f(x)$  une fonction portant sur un scalaire  $x$ .

Pour trouver une racine de cette fonction  $f(x)$  entre des points  $a$  et  $b$ , on peut faire

```
x=fzero(@f,[a,b])
```

où  $@f$  fait référence à la fonction dont on cherche la racine. Cette fonction doit être spécifiée dans un fichier f.m.

Pour une fonction portant sur un vecteur  $X$ , on doit faire

```
X=fsolve(@f,X0)
```

où  $X_0$  désigne le point de départ.

D:\Octave\25-fsolve\Main.m - Notepad++

File Edit Search View Encoding Language Settings Macro Run Plugins Window ? X

Main.m

```
1 clear ; clc;
2
3 X=0:.1:10; Y=f(X); % listes de points à représenter
4
5 x=fzero(@f,[2,4]); % recherche d'une racine pour x entre 2 et 4
6
7 fprintf('La fonction f(x) est nulle pour x=%f\n\n',x);
8
9 figure;
10 plot(X,Y) % liste de points
11 hold
12 plot(x,0,'ro') % solution trouvée
13 grid
14 xlabel("X")
15 ylabel("Y")
16 title("Representation of y=y(x) ")
17 print -dppsc Plot.ps
18
```

length:37 Ln:1 Col:1 Sel:0|0 Dos\Windows ANSI as UTF-8 INS

D:\Octave\25-fsolve\f.m - Notepad++

File Edit Search View Encoding Language Settings Macro Run Plugins Window ? X

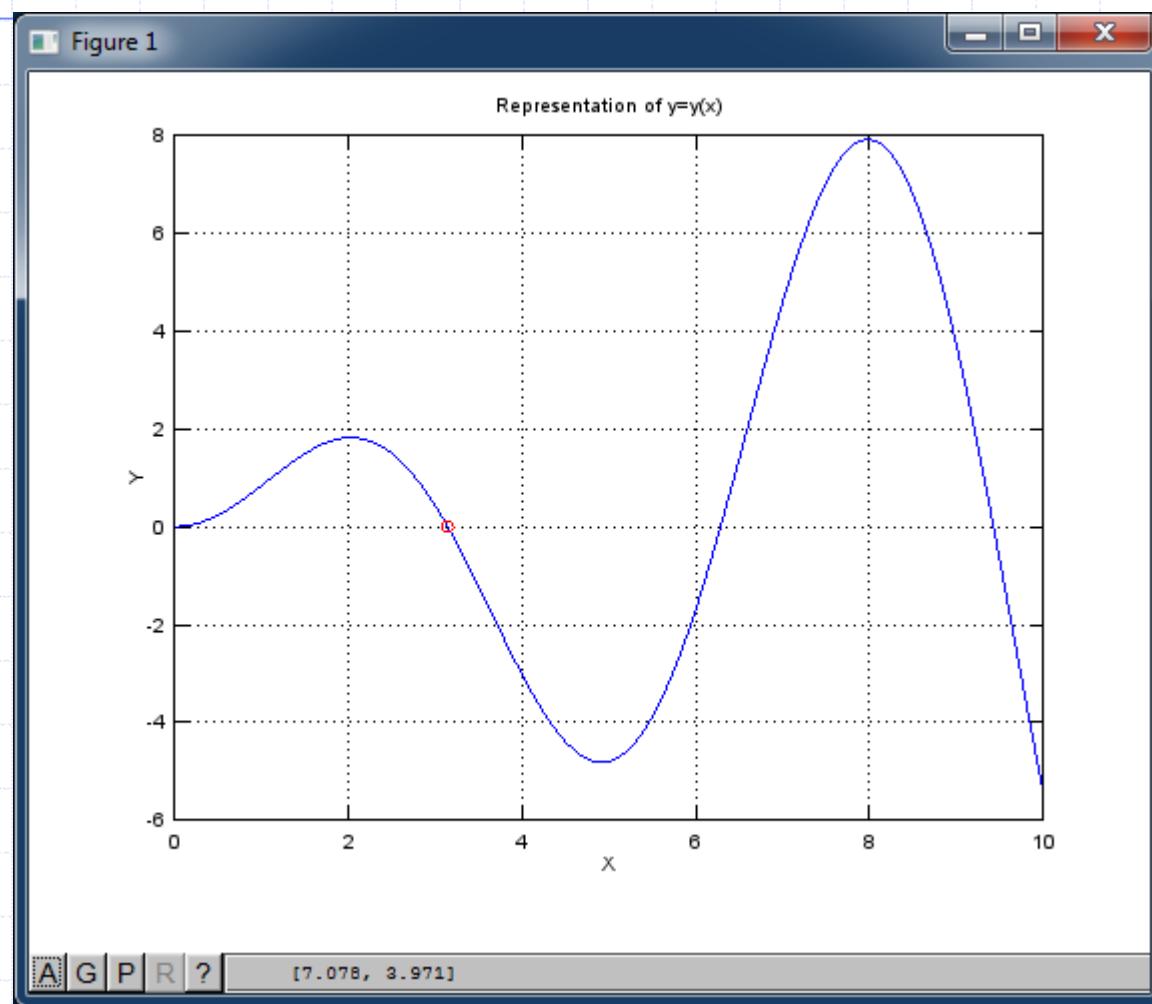
f.m

```
1 function y = f(x)
2
3 y = x .* sin(x);
4
```

length: Ln:1 Col:1 Sel:0|0 Dos\Windows ANSI as UTF-8 INS

Octave

```
La fonction f(x) est nulle pour x=3.141593
```



# Graphiques (réalisations plus avancées)

Chapitre 16

Il existe différentes manières de représenter graphiquement des points dont les coordonnées cartésiennes sont contenues dans des vecteurs  $x$  et  $y$  :

plot ( $x,y$ )	% échelle linéaire pour $x$ et $y$
semilogx ( $x,y$ )	% échelle logarithmique pour $x$
semilogy ( $x,y$ )	% échelle logarithmique pour $y$
loglog ( $x,y$ )	% échelle logarithmique pour $x$ et $y$

Si les points sont donnés en coordonnées polaires, on peut faire :

polar ( $\theta,r$ )	% utilisation de coordonnées polaires
----------------------	---------------------------------------

où  $\theta$  et  $r$  sont des vecteurs qui contiennent les coordonnées polaires des points à représenter.

Pour des applications en statistique, on peut encore faire :

bar (x,y)

% représentation sous forme de  
% rectangles

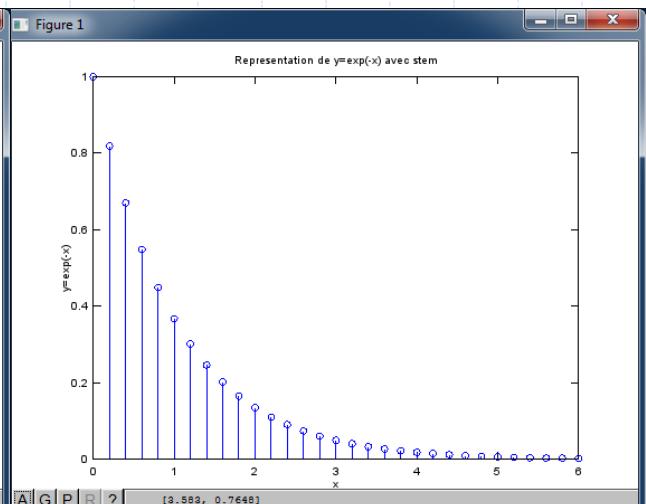
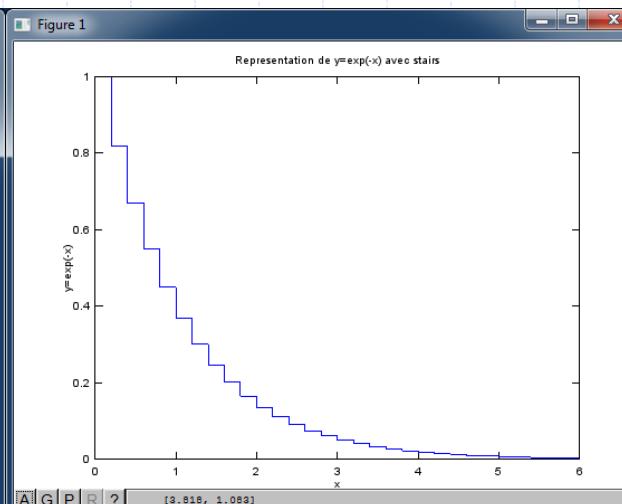
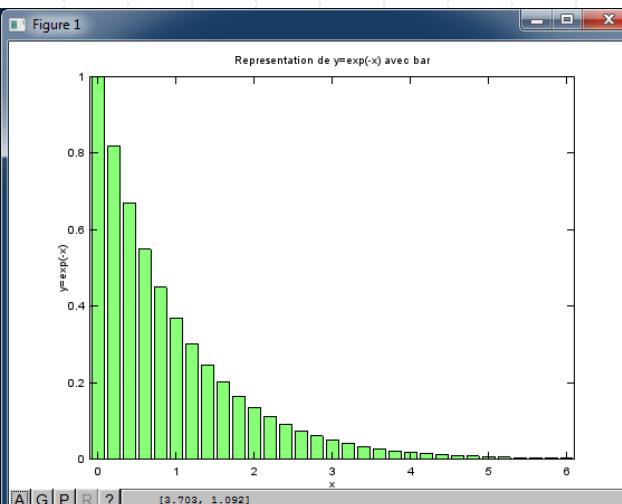
stairs (x,y)

% représentation sous forme de  
% cage d'escaliers

stem (x,y)

% représentation en bâtonnets

Voir aussi : hist pour la représentation d'histogrammes



bar (x,y)

stairs (x,y)

stem (x,y)

Pour l'ouverture de plusieurs fenêtre graphiques et la réalisation de plusieurs instructions plot dans la même fenêtre, les instructions suivantes sont utiles :

figure	% ouverture d'une nouvelle fenêtre % graphique
hold	% permettre aux prochaines instructions % plot de s'afficher sur le même % graphique (sinon effacement)
hold off	% annuler l'instruction hold
clf	% effacer la fenêtre
close	% fermeture de la fenêtre en cours % (cette action n'est pas obligatoire)

La présentation d'un graphique peut être améliorée par l'utilisation des commandes suivantes :

grid

% afficher une grille

axis ([xmin,xmax,ymin,ymax])

% fixer l'extension des axes

legend("legende 1","legende 2") % légende pour chq courbe

text (x,y,"texte");

% afficher « texte » au point (x,y)

Pour changer la couleur et le style des lignes :

plot (x,y,'b-')

% ligne solide de couleur bleue

'r'=red, 'g'=green, 'b'=blue, 'c'=cyan, 'm'=magenta,  
'y'=yellow, 'k'=black, 'w'=white, 'i'=invisible  
'-'=solid,'--'=dashed, '-.'=dot-dashed, ':'=dotted  
's'=squares, 'd'=star, 'p'=pentagon, 'h'=hexagon,  
'+', 'o', '\*', '.', 'x', '^', '>', '<'

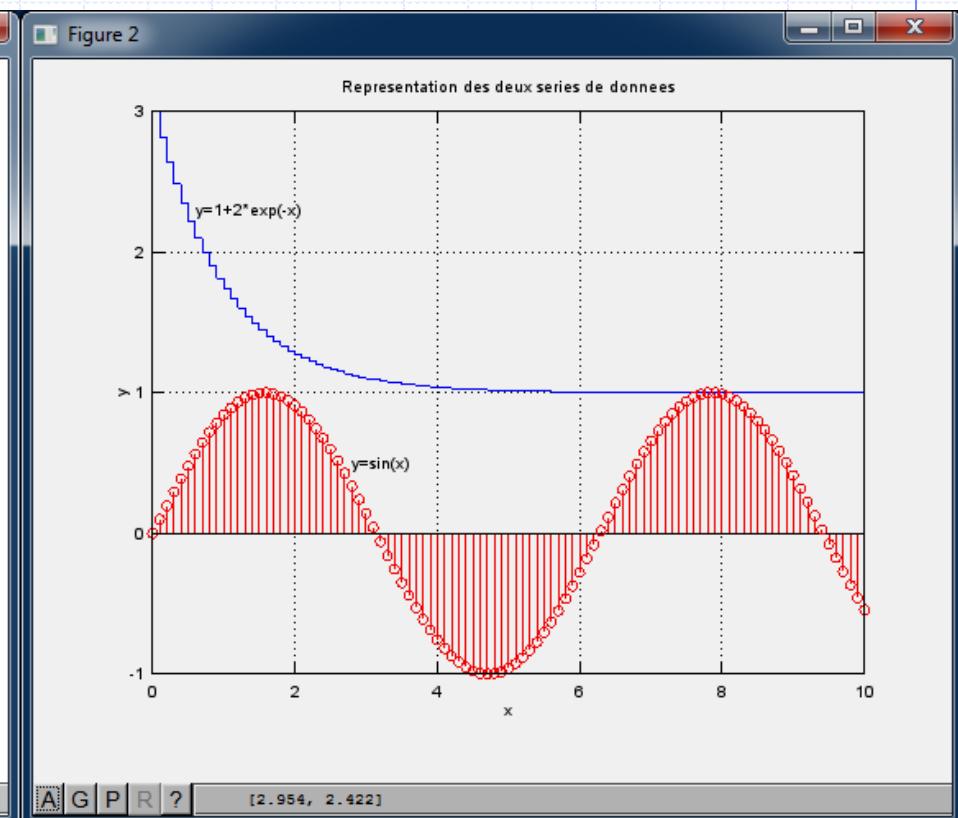
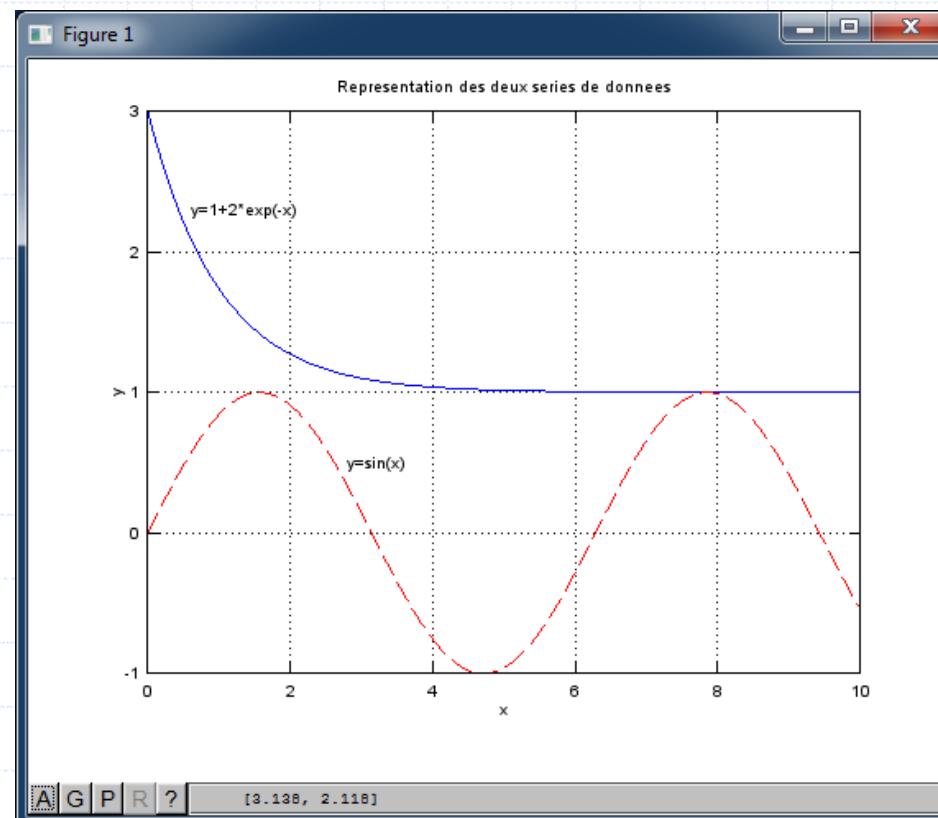
## Exemple :

```
figure  
plot (x1,y1,'b-') % représentation de (x1,y1)  
hold  
plot (x2,y2,'r--') % représentation de (x2,y2)  
xlabel("x")  
ylabel("y")  
title("Representation des deux series de donnees")  
print -dpsc Plot.ps % sortie PostScript  
close
```

D:\Octave\25-graphiques\Main.m - Notepad++

```
File Edit Search View Encoding Language Settings Macro Run Plugins Window ? X
Main.m
1 clear ; clc ;
2
3 x1 = 0:.1:10;
4 y1 = 1+2*exp(-x1);
5 x2 = x1;
6 y2 = sin(x2);
7
8 figure
9 plot(x1,y1,'b-')
10 text(0.6,2.3,"y=1+2*exp (-x )")
11 hold
12 plot(x2,y2,'r--')
13 text(2.8,0.5,"y=sin(x )")
14 grid
15 xlabel("x")
16 ylabel("y")
17 title("Representation des deux series de donnees")
18 print -dpssc Plot1.ps;
19
20 figure
21 stairs(x1,y1,'b')
22 text(0.6,2.3,"y=1+2*exp (-x )")
23 hold
24 stem(x2,y2,'r--')
25 text(2.8,0.5,"y=sin(x )")
26 grid
27 xlabel("x")
28 ylabel("y")
29 title("Representation des deux series de donnees")
30 print -dpssc Plot2.ps;
```

length: Ln:1 Col:1 Sel:0 | 0 Dos\Windows ANSI as UTF-8 INS



# subplot

On peut partitionner la fenêtre en Nx lignes et Ny colonnes grâce à l'instruction :

`subplot (Nx,Ny,id)`

Le troisième argument spécifie la portion qui est active. Ces portions sont énumérées de gauche à droite et de haut en bas.

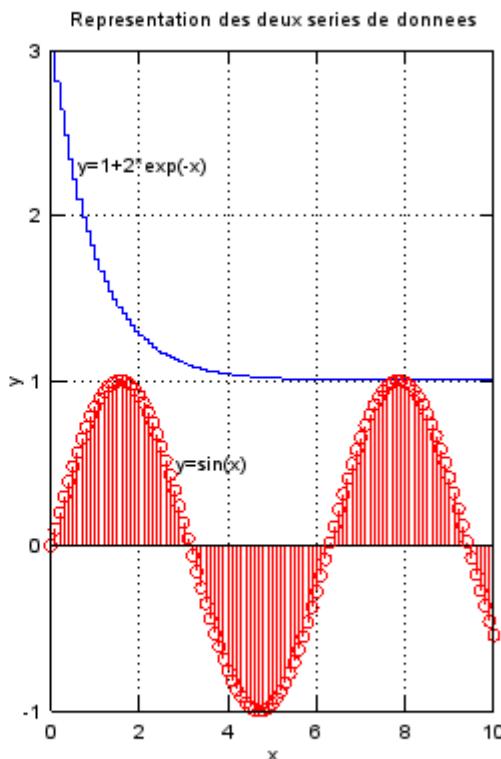
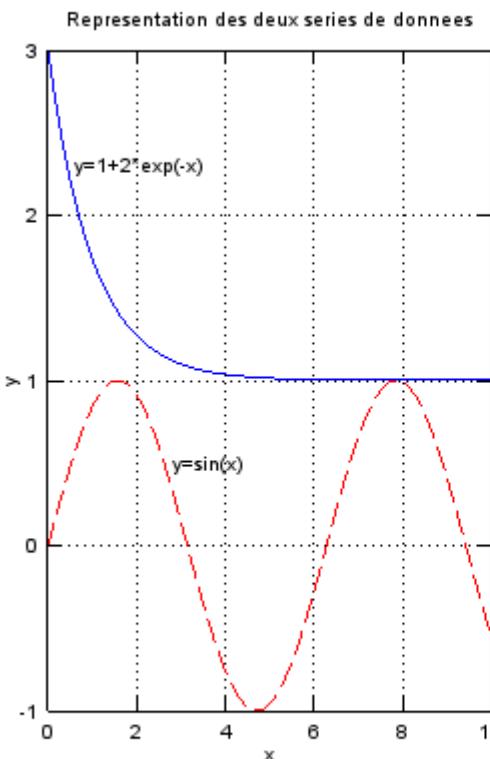
Les instructions plot s'exécutent dans la portion active. On peut changer la portion active par une nouvelle instruction subplot.

D:\Octave\26-subplot\Main.m - Notepad++

```
File Edit Search View Encoding Language Settings Macro Run Plugins Window ? X
Main.m
1 clear ; clc ;
2
3 x1 = 0:.1:10;
4 y1 = 1+2*exp(-x1);
5 x2 = x1;
6 y2 = sin(x2);
7
8 figure
9 subplot(1,2,1)      % partitionnement à 1 ligne et 2 colonnes
10 % avec portion 1 rendue active
11 plot(x1,y1,'b-')
12 text(0.6,2.3,"y=1+2*exp (-x )")
13 hold
14 plot(x2,y2,'r--')
15 text(2.8,0.5,"y=sin(x )")
16 grid
17 xlabel("x")
18 ylabel("y")
19 title("Representation des deux series de donnees")
20 subplot(1,2,2)      % partitionnement à 1 ligne et 2 colonnes
21 % avec portion 2 rendue active
22 stairs(x1,y1,'b')
23 text(0.6,2.3,"y=1+2*exp (-x )")
24 hold
25 stem(x2,y2,'r--')
26 text(2.8,0.5,"y=sin(x )")
27 grid
28 xlabel("x")
29 ylabel("y")
30 title("Representation des deux series de donnees")
31 print -dpsc Plot.ps;
```

length:670 lines:: Ln:1 Col:1 Sel:0|0 Dos\Windows ANSI as UTF-8 INS

Figure 1



A

G

P

R

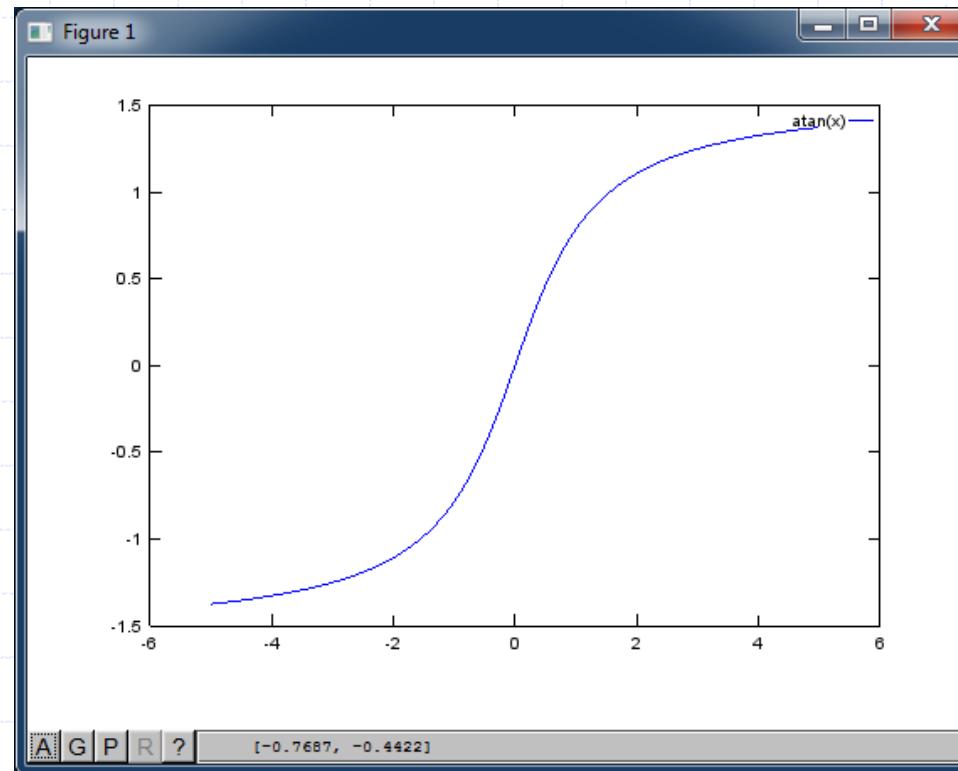
?

[2.773, 3.461]

# Représentation simplifiée pour des fonctions réelles

fplot("atan(x"),[-5,5])

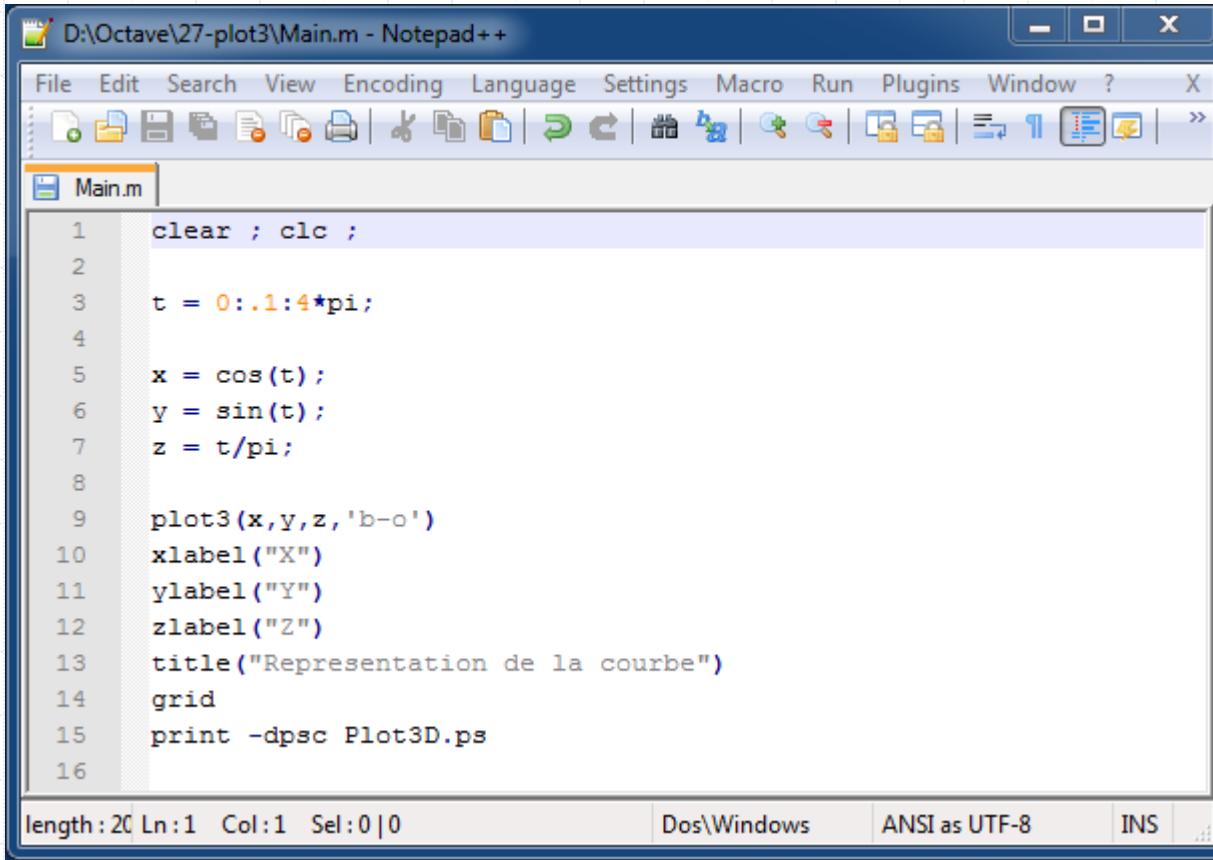
% représentation graphique de  
%  $y = \text{atan}(x)$  pour  $x$  compris  
% entre -5 et +5.



# Courbes en 3-D

plot3 (x,y,z)

% représentation en 3-D des  
% points de coordonnées x, y, z

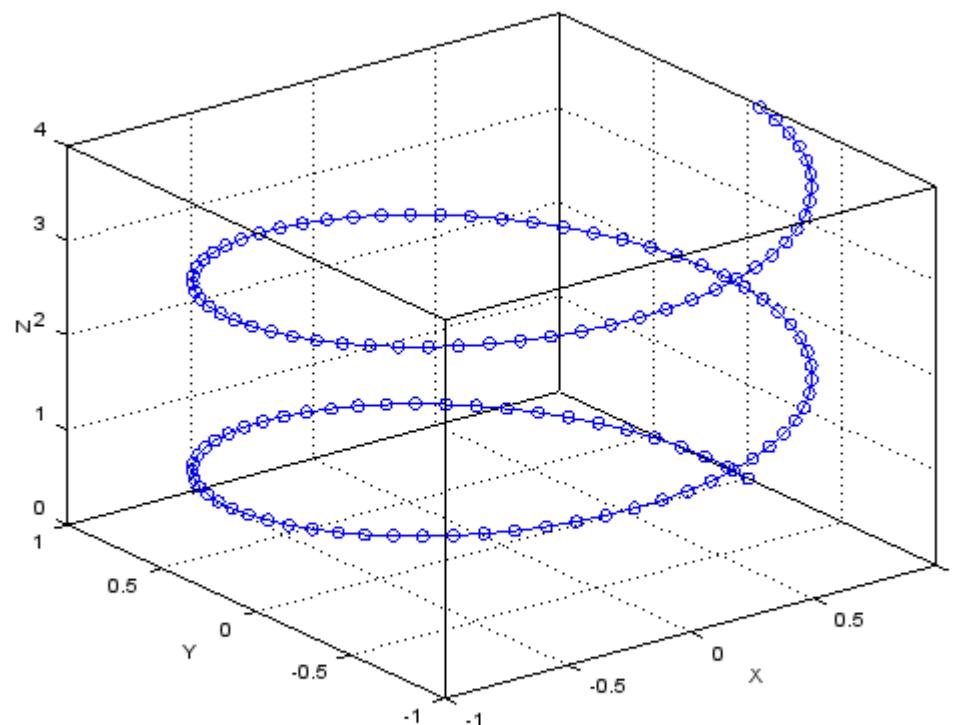


```
D:\Octave\27-plot3\Main.m - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
Main.m
1 clear ; clc ;
2
3 t = 0:.1:4*pi;
4
5 x = cos(t);
6 y = sin(t);
7 z = t/pi;
8
9 plot3(x,y,z,'b-o')
10 xlabel("X")
11 ylabel("Y")
12 zlabel("Z")
13 title("Representation de la courbe")
14 grid
15 print -dpsc Plot3D.ps
16
```

length:20 Ln:1 Col:1 Sel:0|0 Dos\Windows ANSI as UTF-8 INS

Figure 1

Representation de la courbe



A

G

P

R

?

[-0.9508, 1.72]

# Thèmes divers et avancés

## Chapitre 17

# polyfit

La fonction polyfit permet d'établir l'ajustement polynomial d'un ensemble de points :

$$P = \text{polyfit}(X, Y, n)$$

fournit ainsi les coefficients du polynôme de degré  $n$  qui passe au mieux (dans le sens des moindres carrés) à travers le nuage de points décrits par  $X$  et  $Y$ .

Ce polynôme est alors donné par

$$P(1)x^n + P(2)x^{n-1} + \dots + P(n+1)$$

La valeur de ce polynôme, pour une valeur donnée de  $x$ , peut être calculée par  $y = \text{polyval}(P, x)$ .

# eval

Octave permet de faire ce qui suit :

`eval("y=sin(x)")`

Le contenu de la chaîne de caractères est interprété et cette instruction est ainsi équivalente à

`y=sin(x)`

# feval

On peut encore faire :

```
fon="sin";
y=feval(fon,x)
```

La chaîne de caractères contient le nom de la fonction à évaluer et le résultat sera  $y=\sin(x)$ .

On peut faire référence à une fonction externe  $f(x)$  en faisant :

```
y=feval(@f,x)
```

Cette fonction externe doit être spécifiée dans un fichier f.m.

# @ et inline

On peut définir des fonctions « en ligne » :

```
fun=@(x)sin(x);
```

ou

```
fun=inline("sin(x)","x");
```

et utiliser ensuite fun comme on utiliserait f ou @f pour une fonction externe :

```
x=fminbnd(fun,pi,2*pi);
x=fzero(fun,[0.9*pi,1.1*pi]);
y=fun(pi/2);
y=feval(fun,pi/2);
```

# tic toc

Pour mesurer le temps d'exécution d'un programme, il suffit d'utiliser les instructions :

```
tic ;
```

...

```
toc
```

On peut aussi faire :

```
tstart = tic;
```

...

```
dt = toc(tstart);
```

si on veut utiliser l'intervalle de temps dt.

Il s'agit ici d'un « temps horloge ».

D:\Octave\17-tictoc\Main.m - Notepad++

File Edit Search View Encoding Language Settings Macro Run Plugins Window ? X

Main.m

```
1 clear ; clc;
2
3 tic;
4
5 somme=0.;
6 for i = 1:10000
7     somme=somme+1./i^2;
8 end
9 fprintf('La valeur de somme est : %f\n\n',somme);
10
11 toc
12
```

length: Ln:1 Col:1 Sel:0|0 Dos\Windows ANSI as UTF-8 INS

Octave

```
La valeur de somme est : 1.644834
Elapsed time is 0.0860089 seconds.
octave:5> _
```

# cputime

Pour mesurer un intervalle de « temps cpu » on peut faire :

```
tstart = cputime;
```

...

```
dt = cputime-tstart;
```

Il s'agit ici du *temps de calcul* consacré à Octave.

Le « temps cpu » est en général différent du « temps horloge » donné par tic/toc par le fait que plusieurs processeurs peuvent éventuellement participer à l'exécution de vos instructions et par le fait que ces processeurs se consacrent en général à plusieurs tâches en même temps.

# addpath

Pour ajouter le répertoire d:\Octave\Library dans le PATH :

```
addpath("d:/Octave/Library")
```

On peut indiquer de cette manière où se trouvent les procédures utilitaires utilisées dans un projet particulier. On évite ainsi de devoir les recopier dans le répertoire où se trouve le programme qui les utilise.

Cette manière de travailler facilite la mise à jour de ces procédures utilitaires (leur mise à jour s'applique automatiquement à chaque projet qui fait simplement référence au répertoire où elles se trouvent).

Voir aussi : path, rmpath

# Download

Vous pouvez retrouver ce document sur

<http://perso.fundp.ac.be/~amayer/cours/Octave>