

3. Week

Introduction to MATLAB

Short MATLAB History

The MathWorks, Inc.

Founded in 1984 by Cleve Moler and Jack Little

What is MATLAB

MATLAB(Matrix laboratory) is an interactive software system. It integrates mathematical computing, visualization, and a powerful language to provide a flexible environment for technical computing. Typical uses include

- ☐ Math and computation
- ☐ Algorithm development
- ☐ Data acquisition
- ☐ Modeling, simulation, and prototyping
- ☐ Data analysis, exploration, and visualization
- ☐ Scientific and engineering graphics
- ☐ Application development, including graphical user interface building

Software Development Philosophy

- Matrix-based numeric computation
MATrix LABoratory
- High-level programming language
Programming data type specification
not required & no pointers –
- Superb graphics provide excellent data
visualization
- Toolboxes provide application specific
functionality

Starting MATLAB

Windows

double-click the MATLAB shortcut icon on your Windows desktop.

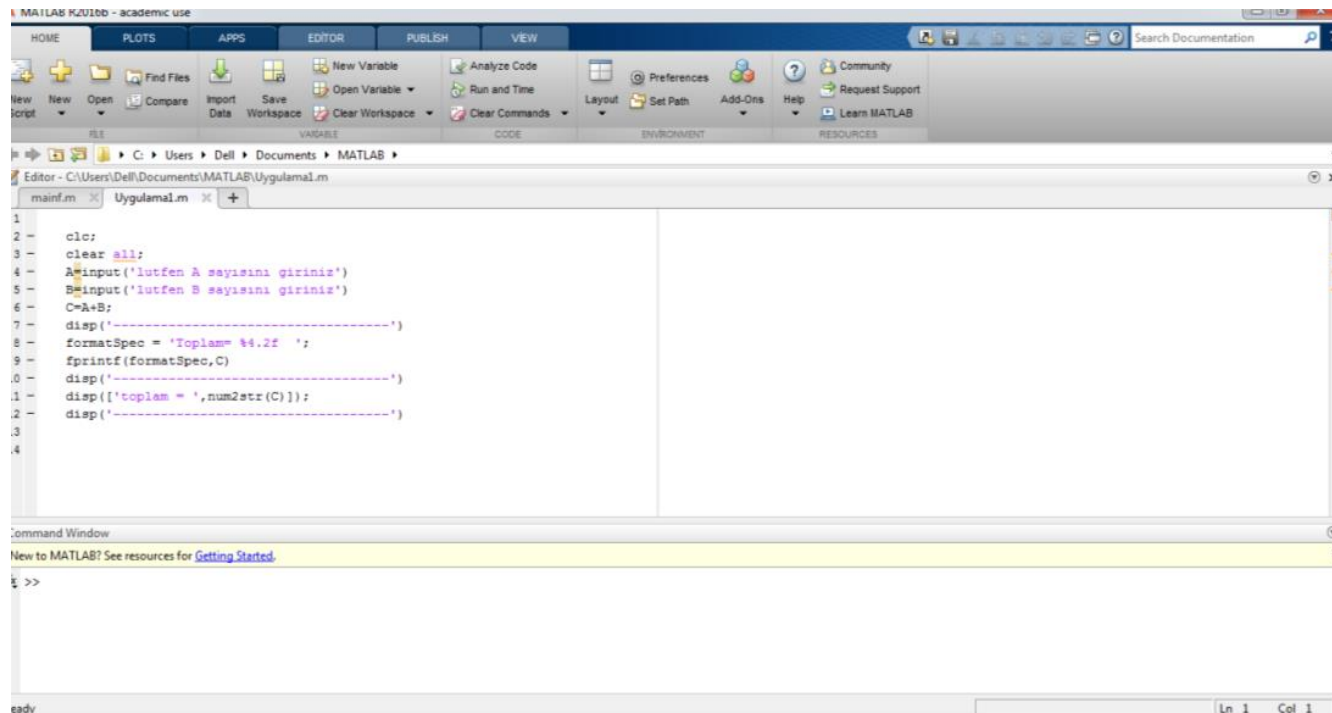
UNIX

type `matlab` at the operating system prompt.

After starting MATLAB, the MATLAB desktop opens.

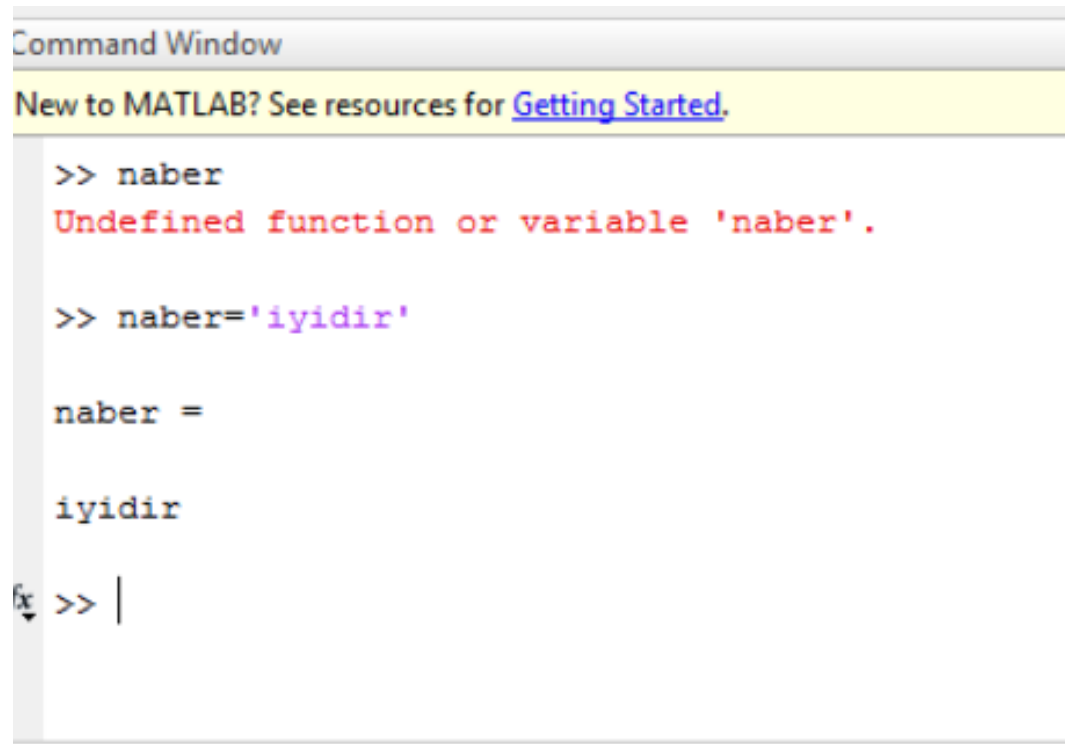
How to get started

- when the computer has started go through the following steps in the different menus
- you will see a little icon for Matlab, double click on that
- within about 30 seconds Matlab will open
- you should now see a screen like the picture below



MATLAB Desktop

At the bottom you have the **Command Window** – this is where you type commands and usually the answers (or error messages) appear here too

A screenshot of the MATLAB Command Window. The window has a title bar that says "Command Window". Below the title bar is a yellow banner with the text "New to MATLAB? See resources for [Getting Started](#)." The main area of the window shows the following text: ">> naber" followed by a red error message "Undefined function or variable 'naber'." Then, ">> naber='iyidir'" is entered, followed by the output "naber =" and "iyidir" on the next line. At the bottom, there is a prompt ">> |" with a cursor. On the left side of the window, there is a vertical toolbar with icons for various functions like save, load, and help.

```
Command Window
New to MATLAB? See resources for Getting Started.

>> naber
Undefined function or variable 'naber'.

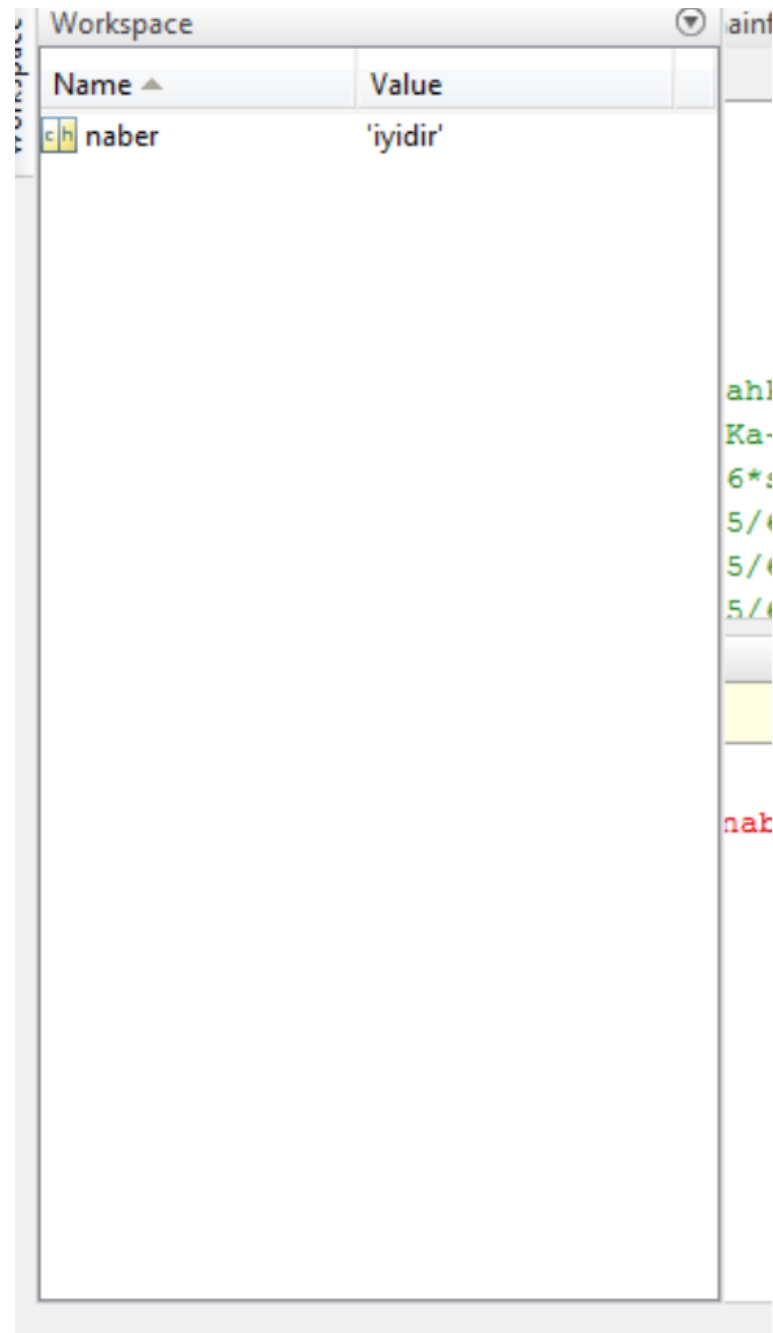
>> naber='iyidir'

naber =

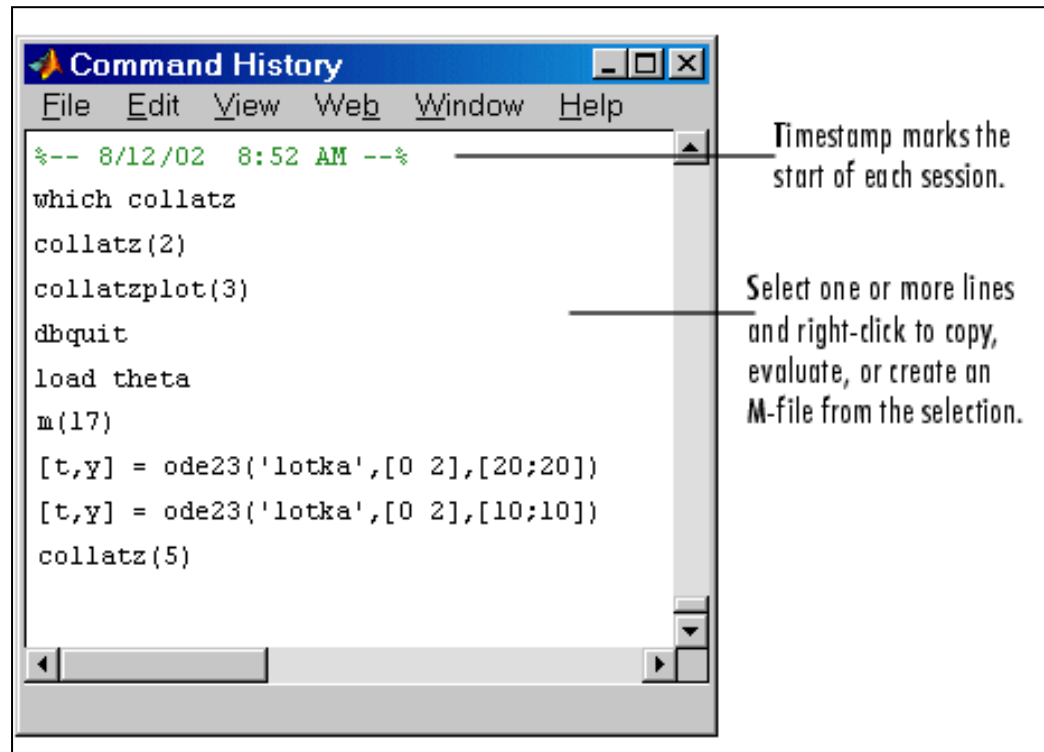
iyidir

fx >> |
```


On the top right you have the **Workspace window** – if you define new quantities (called variables) their names should be listed here.

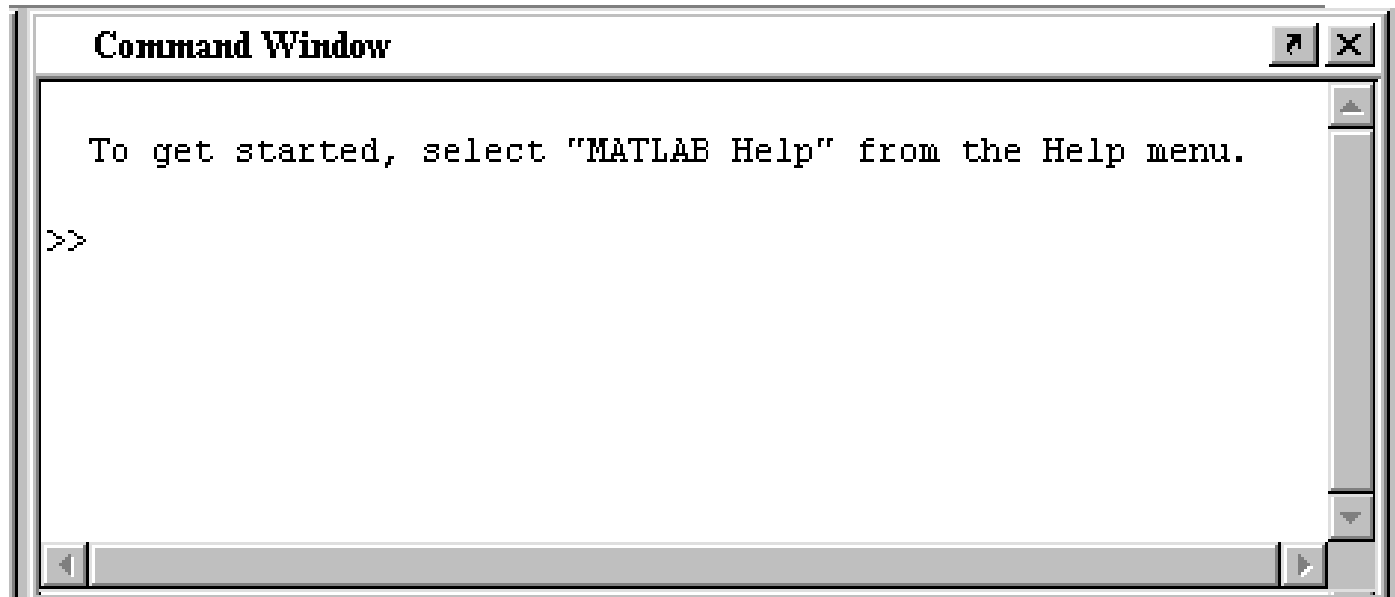


On the bottom you have **Command History Window** – this is where past commands are remembered. If you want to re-run a previous command or to edit it you can drag it from this window to the command window to re-run it.



The Matlab prompt is >>

In the text that follows, any line that starts with two greater than **signs (>>)** is used to denote the **Matlab command line**. **This is where you enter your commands.** Look at the Command Window and you will see the cursor flickering after the >> prompt. This means that Matlab is waiting for further instructions.



All commands typed at the prompt are
executed as soon as you press enter

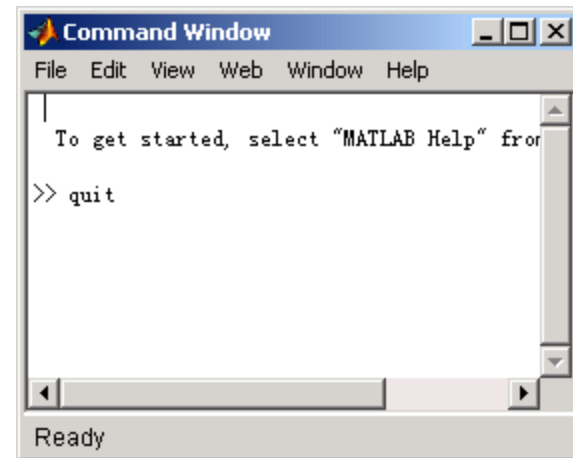
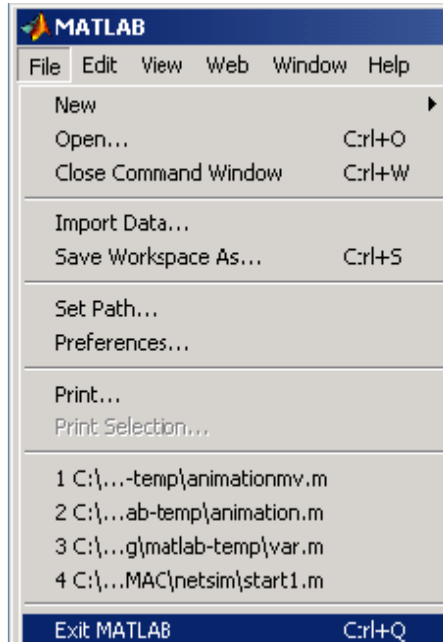
```
>> (2*3)+1/250
```

```
ans =
```

```
6.0040
```

Quitting MATLAB

select Exit MATLAB from the File menu in the desktop, or type quit in the Command Window.



Variables and the Workspace

Defining Numbers and Characters

- ❑ To define a scalar we may simply use the equal sign. $a=3$
- ❑ To define a vector we use the equal sign and square brackets. $v=[3\ 5\ 7]$
- ❑ Matrix definition follows the same form as vector definition. The matrix rows are separated by semicolons, or by returns.
 $m1=[1\ 2\ 3; 4\ 5\ 6]$
or $m2=[1\ 2\ 3$
 $4\ 5\ 6]$

❑ To define character variables we use the equal sign and single quotes. `c='abc'`

Variable names

Matlab distinguishes between lower case and capital letters - so **A and a are different objects for Matlab**. The normal convention is to use lower case names except for global variables. Only the first 19 characters in a variable name are important.

Defining a Vector

Matlab is a software package that makes it easier for you to enter **matrices and vectors**, and manipulate them.

□ Almost all of Matlab's basic commands revolve around the use of vectors. A vector is defined by placing a sequence of numbers within square brackets:

```
>> v = [3 1]
```

```
v =
```

```
3      1
```

This creates a row vector which has the label "v". The first entry in the vector is a 3 and the second entry is a 1. Note that Matlab printed out a copy of the vector after you hit the enter key.

❑ If you do not want to print out the result **put a semi-colon** at the end of the line:

```
>> v = [3 1];
```

```
>>
```

❑ If you want to view the vector just type its label:

```
>> v =
```

```
3 1
```

❑ You can define a vector of any size in this manner:

```
>> v = [3 1 7 -21 5 6]
```

```
v =
```

```
3 1 7 -21 5 6
```

❑ Notice, though, that this always creates a row vector. If you want to create a column vector you need to take the transpose of a row vector. The transpose is defined using an apostrophe ('):

```
>> v = [3 1 7 -21]'
```

```
v =
```

```
3
```

```
1
```

```
7
```

```
-21
```

❑ A common task is to create a large vector with numbers that fit a **repetitive pattern**. Matlab can define a set of numbers with a common increment using colons. For example, to define a vector whose first entry is 1, the second entry is 2, the third is 3, up to 8 you enter the following:

```
>> v = [1:8]
```

```
v =
```

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

- ❑ If you wish to use an **increment other than one** that you have to define the start number, the value of the increment, and the last number. For example, to define a vector that starts with 2 and ends in 4 with steps of .25 you enter the following:

```
>> v = [2:.25:4]
```

```
v =
```

```
Columns 1 through 7
```

```
2.0000 2.2500 2.5000 2.7500 3.0000 3.2500 3.5000
```

```
Columns 8 through 9
```

```
3.7500 4.0000
```

Accessing elements within a vector

- ❑ You can view individual entries in this vector. For example to view the first entry just type in the following:

```
>> v(1)
```

```
ans =
```

```
2
```

This command prints out entry 1 in the vector. Also notice that **a new variable** called **ans** has been created. Any time you perform an action that does not include an assignment Matlab will put the label **ans** on the result.

❑ To simplify the creation of large vectors, you can define a vector by specifying the first entry, an increment, and the last entry. Matlab will automatically figure out how many entries you need and their values. For example, to create a vector whose entries are 0, 2, 4, 6, and 8, you can type in the following line:

```
>> 0:2:8
```

```
ans =
```

```
0 2 4 6 8
```

Matlab also keeps track of the last result. In the previous example, a variable **"ans"** is created. To look at the transpose of the previous result, enter the following:

```
>> ans'
```

```
ans =
```

```
0
```

```
2
```

```
4
```

```
6
```

```
8
```

To be able to keep track of the vectors you create, you can give them names. For example, a row vector v can be created

```
>> v = [0:2:8]
```

```
v =
```

```
    0  2  4  6  8
```

```
>> v
```

```
v =
```

```
    0  2  4  6  8
```

```
>> v;
```

```
>> v'
```

```
ans =
```

```
0
```

```
2
```

```
4
```

```
6
```

```
8
```

Note that in the previous example, if you end the line with a semi-colon, the result is not displayed. This will come in handy later when you want to use Matlab to work with very large systems of equations. Matlab will allow you to look at specific parts of the vector. If you want to only look at the first three entries in a vector you can use the same notation you used to create the vector:

```
>> v(1:3)
```

```
ans =
```

```
0 2 4
```

```
>> v(1:2:4)'
```

```
ans =
```

```
0 4
```

Examining Numbers and Characters

who

This command lists the names of all currently defined variables.

whos

This command lists the names and the properties of all currently defined variables.

what

This command lists the files on current directory.

size

Size (name) returns the size of the variable (name).

clear

Erases the values of all defined variables.

Workspace Commands

save

Saves all of the defined variables into a workspace called matlab.mat.

load

Loads the workspace matlab.mat.

help

Lists all of the available operators and functions.

help (cmd)

Lists the help file for (cmd).

Mathematical Functions

Simple arithmetic with Matlab

- + Is the addition operator.
- Is the subtraction operator.
- * Performs multiplication.
- / Performs right division. ($b/a = b * \text{Inv}(a)$)
- \ Performs left division. ($b \backslash a = \text{Inv}(b) * a$)
- ^ Performs exponentiation (power).

Where necessary brackets should be used to make the order in which things are evaluated completely clear.

Simple arithmetic with Matlab

Enter the following commands to learn the basic way that formulas can be evaluated in Matlab (press enter after you have typed in the command. Record the answer beside the command.

```
>> 3 + 5 - 7
```

The result of this calculation is stored in the temporary variable called **ans**. It is changed when you do another calculation.

Simple arithmetic with Matlab

If you want to save the answer from a calculation you can give it a name e.g.

```
>> t = 3 + 5 - 7
```

This line creates a variable `t` to save the answer in. If you want to find out what `t` is use

```
>> t
```

Equally you can use `t` in a formula

```
>> 2*t + 2
```

The order of operations and putting brackets in can matter

In the following calculation does Matlab do the * or the ^ first

>> 3^2*4

One way to do the calculation is that powers are done before multiplication.

This $3^2*4 = 9*4 = 36$

because Matlab does 3^2 first and then multiplies the answer by 4 – generally powers are done first, the multiplication and division and finally additions and subtractions.

One way to force Matlab (or any software) to do calculations in a defined order is to use brackets.

Compare the following commands and make sure that you understand what is happening and why you get the answer

```
>>>> (3-4)/(4-2)
```

```
>> (3-4)/4 -2
```

The rules for evaluating expressions involving numbers are

- things inside brackets are done first
- within a bracket or expression generally operations closest to a value are done first

Extended arithmetic (IEEE)

You need to understand the following section to interpret the output from some calculations – particularly when we make accidental errors.

What does Matlab produce if you ask it to find:

>> 1/0

>> -1/0

>> 0/0

>> 1/Inf

The symbols **Inf** (meaning infinity) and **NaN (not a well defined number)** are part of a special standard for computer arithmetic. This is the IEEE international standard for extended arithmetic.

Typically you get **Inf** from $1/0$ and **NaN** from $0/0$ type calculations.

FORMAT set output format

All computations in MATLAB are done in double precision. FORMAT may be used to switch between different output display formats as follows:

FORMAT Default. Same as SHORT. [3.1416](#)

FORMAT SHORT Scaled fixed point format with 5 digits. [3.1416](#)

FORMAT LONG Scaled fixed point format with 15 digits. [3.141592653589793](#)

FORMAT SHORT E Floating point format with 5 digits. [3.1416e+000](#)

FORMAT LONG E Floating point format with 15 digits. [3.141592653589793e+000](#)

FORMAT SHORT G Best of fixed or floating point format with 5 digits. [3.1416](#)

FORMAT LONG G Best of fixed or floating point format with 15 digits. [3.14159265358979](#)

FORMAT + The symbols +, - and blank are printed for positive, negative and zero elements.

Imaginary parts are ignored. [+, -, blank](#)

FORMAT BANK Fixed format for dollars and cents. [3.14](#)

FORMAT RAT Approximation by ratio of small integers [355/113](#)

Matlab works to 15 significant figures but usually only shows 5 digits

Type **pi** and you will normally get 3.1416

You can change to the **long format** in which all 15 figures are displayed

```
>> pi
```

```
>> format long
```

```
>> pi
```

An easier and more flexible format is scientific notation with 5 significant figures

```
>> format short e
```

```
>> pi
```

Mostly we work in the short format.

to go back to short format

```
>> format short
```