

Why Matlab?

Matlab is an interactive, high-level, user-friendly programming and visualization environment. It allows much faster programs development in comparison with the traditional low-level compiled languages like Fortran or C.

The trade-off here is the execution speed. The Matlab code may run 10 times slower than the corresponding Fortran/C code. But modern computers are fast!

Matlab as a calculator

Typing `matlab` from the Unix prompt will start the program. Simple calculations can be performed interactively:

```
>> 7+(19-6)*8^2/4-3    % simple example
ans =
    212
```

Matlab performs here basic arithmetic operations `+` `-` `*` `/` and exponentiation `^` on integers following the same precedence/order of evaluation rules as in Fortran/C languages: 1) parentheses, 2) exponentiation, 3) multiplication/division, 4) addition/subtraction;

`%` a comment, Matlab ignores everything to the right of this symbol; `ans` is a built-in default variable name to which the result is assigned. We can refer to this name in the next statement:

```
>> (ans-12)/2
ans =
    100
```

It would be better to assign the result/expression to the variable of your own choosing:

```
>> sun=47+13
sun =
    60
```

Typing **sun** at the prompt shows that the variable **sun** has the value 60:

```
>> sun
sun =
    60
```

Variable names can be up to 63 characters long, start with a letter, and contain only letters, numbers or the underscore “_” character.

*Variable names are case sensitive: **sun** and **Sun** are different!*

Multiple commands can appear on the same line, provided they are separated by a comma - if you want to see the result of the previous command - or a semicolon if you want to suppress the display:

```
>> pc=24; mac=13, computers=sun+pc+mac
mac =
    13
computers =
    97
```

This also shows that Matlab can evaluate expressions involving previously defined variables. However, each variable must be assigned a value before it is used in calculations. For example,

```
>> unix_workstations=sun+alpha
??? Undefined function or variable 'alpha'.
```

All variables in the workspace can be listed with the command

```
>> who
```

```
Your variables are:
```

```
ans          mac          sun
computers    pc
```

To get information about **who** (or any other Matlab command), type

```
>> help who
```

```
WHO      List current variables.
WHO lists the variables in the current work-
space.
WHOS lists more information about each variable.
...etc.
```

clear command removes *all* variables from the workspace;
clear some_variable removes **some_variable** from the workspace.

In the examples above we manipulated with integer numbers. To perform basic operations on real numbers (rational & irrational), Matlab uses familiar from the Fortran course double precision *floating-point* arithmetic. As we will see later discussing the floating point arithmetic, Matlab computations use about 16 decimal digits. However, unlike in Fortran or C, we do not need to declare a type (integer, real) of our variables.

Consider the example: compute the volume of a sphere with a radius of 0.69 cm ($V = 4\pi r^3/3$).

First, in Matlab π is **pi**. Typing

```
>> r=.69; V=4*pi*r^3/3
V =
    1.3761
```

displays the result using 5 digits (default format). If more digits are needed, we can use **format** command, which controls how numbers appear on the screen. Read help on **format**, which is rather informative:

```
>> help format
```

```
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.
```

```
FORMAT SHORT    Scaled fixed point format with 5 digits.
```

```
FORMAT LONG     Scaled fixed point format with 15 digits.
```

```
FORMAT SHORT E Floating point format with 5 digits.
```

```
FORMAT LONG E  Floating point format with 15 digits.
```

```
.....etc.
```

Thus, for example

```
>> format long e
```

```
>> V
```

```
V =
    1.376055281384172e+000
```

outputs more digits; **e** here stands for “exponent”, e.g.,

```
>> 254.178
```

```
ans =
```

```
    2.541780000000000e+002
```

$254.178 = 2.54178 \times 10^2$. To return to the default mode, type

```
>> format short
```

Note, that **format** does not affect computations.

Matlab also knows complex numbers. Letters **i** and **j** can be used as imaginary part, unless redefined:

```
>> (1+2i)*(1-2i)
```

```
ans =  
5
```

```
>> (1+2i)/(1-2i)
```

```
ans =  
-0.6000 + 0.8000i
```

If variable **i** was assigned and used as a real number,

```
>> i=34
```

```
i =  
34
```

to redefine it to be a $\sqrt{-1}$, enter the command **clear i,j** or:

```
>> i=sqrt(-1)
```

Here **sqrt(x)** is a built-in Matlab mathematical function; many other elementary and special functions are available: **exp(x)**, **log(x)**, **log10(x)**, **sin(x)**, **tan(x)**, etc. To find the proper Matlab “spelling” of a function name, **lookfor** command is useful, for example:

```
>> lookfor cosecant
```

ACSC Inverse cosecant.
ACSCH Inverse hyperbolic cosecant.
CSC Cosecant.
CSCH Hyperbolic cosecant.

lookfor searches Matlab system files for the string **cosecant**, and finds **csc(x)**. **lookfor** can take some time to complete, type Ctrl-c to interrupt the search.

EXERCISES

1. Evaluate in Matlab and display results using different formats

a)
$$\frac{4\sqrt{5} - 2\sqrt{6}}{(\sqrt[4]{3} + \sqrt[4]{2})(\sqrt[4]{3} - \sqrt[4]{2})}$$

b)
$$3 + \frac{1}{7 + \frac{1}{15 + \frac{1}{1 + \frac{1}{293}}}}$$

c)
$$\sin\left(\arcsin\frac{3}{5} + \arcsin\frac{8}{17}\right) - \frac{77}{85}$$

2. Explain the difference between **exp(pi/3i)**, **exp(pi/3*i)**, **exp(pi/3/i)**.

3. Check the formulas $e^{ix} = \cos x + i \sin x$ and $\sin(ix) = i \sinh x$ for $x = \frac{\pi}{6}, \frac{2\pi}{3}$.

4. Compute magnitude and angle of xy and x/y , where $x = \frac{1}{2} + i\frac{\sqrt{3}}{2}$,

and $y = \frac{1}{\sqrt{2}}(1 + i)$. (explore **help** on **abs** and **angle** commands)