Chapter One

Introduction

The tutorials are independent of the rest of the document. The primarily objective is to help you learn quickly the first steps. The emphasis here is ''learning by doing". Therefore, the best way to learn is by trying it yourself. Working through the examples will give you a feel for the way that MATLAB operates. In this introduction we will describe how MATLAB handles simple numerical expressions and mathematical formulas. The name MATLAB stands for MATrix LABoratory. MATLAB was written originally to provide easy access to matrix software developed by the LINPACK (linear system package) and EISPACK (Eigen system package) projects.

Basic features

As we mentioned earlier, the following tutorial lessons are designed to get you started quickly in MATLAB. The lessons are intended to make you familiar with the basics of MATLAB. We urge you to complete the exercises given at the end of each lesson.

A minimum MATLAB session

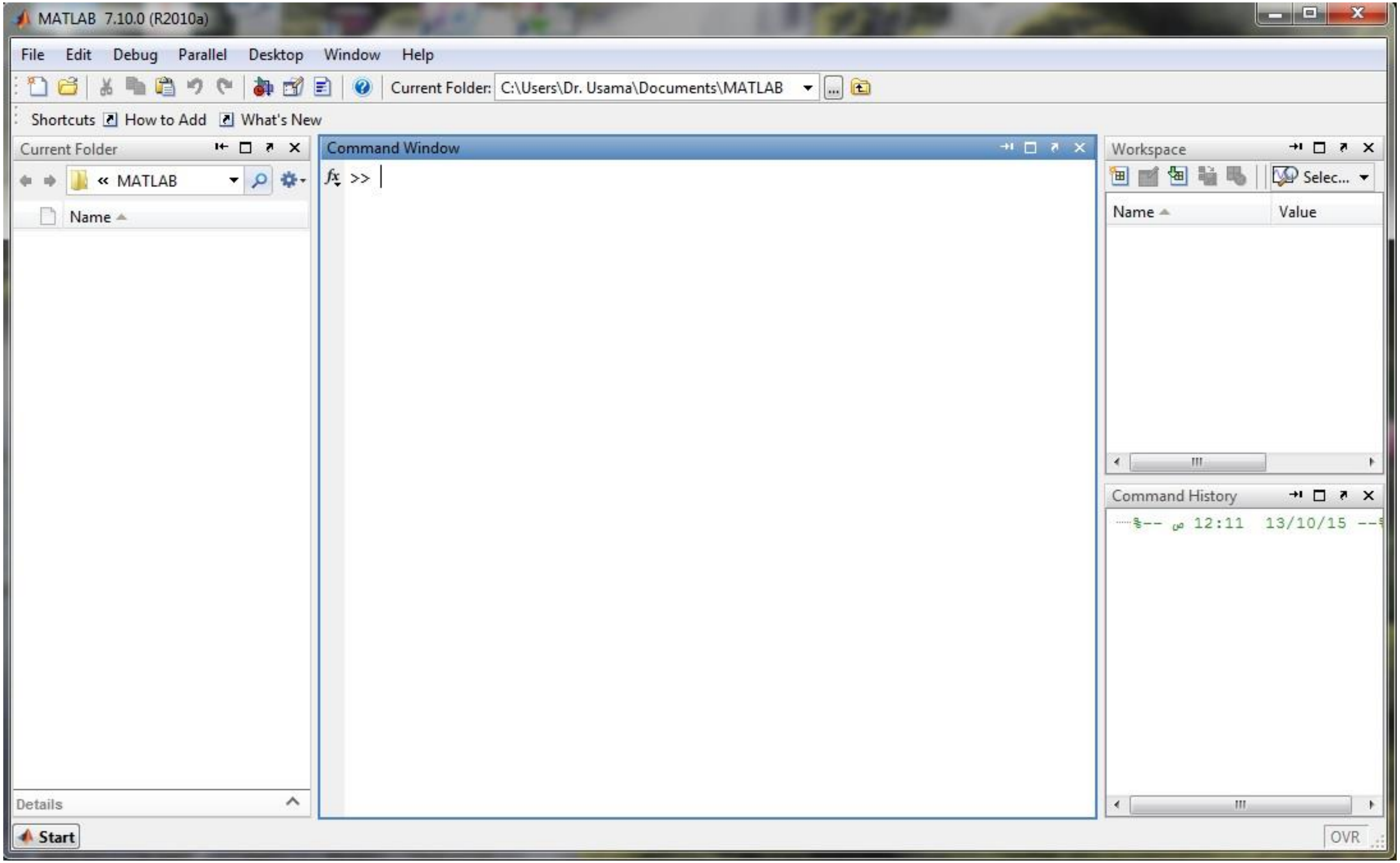
The goal of this minimum session (also called starting and exiting sessions) is to learn the first steps:

- ❑ How to log on
- ❑ Invoke MATLAB
- ❑ Do a few simple calculations
- ❑ How to quit MATLAB

Starting MATLAB

After logging into your account, you can enter MATLAB by double clicking on the MATLAB shortcut icon (MATLAB 2010) on your Windows desktop. When you start MATLAB, a special window called the MATLAB desktop appears. The desktop is a window that contains other windows. The major tools within or accessible from the desktop are:

❑ The Command Window
❑ The Command History
❑ The Workspace
❑ The Current Directory
❑ The Help Browser
❑ The Start button



Current Folder: C:\Users\Dr. Usama\Documents\MATLAB

```
fx >>
```

```
%-- ص 12:11   13/10/15 --%
```

Using MATLAB as a calculator

As an example of a simple interactive calculation, just type the expression you want to evaluate. Let's start at the very beginning. For example, let's suppose you want to calculate the expression, 1+2*3. You type it at the prompt command (>>) as follows,

```
>>1+2*3
  ans   =
          7
```

You will have noticed that if you do not specify an output variable, MATLAB uses a default variable **ans,** short for answer, to store the results of the current calculation. Note that the variable **ans** is created (or over written, if it is already existed). To avoid this, you may assign a value to a variable or output argument name. For example,

```
>>x=1+2*3
  X  =
     7
```

Will result in x being given the value 1+2*3=7. This variable name can always be used to refer to the results of the previous computations. Therefore, computing 4x will result in

```
>>4*x
   ans  =
      28.0000
```

Before we conclude this minimum session, Table1 gives the partial list of arithmetic operators.

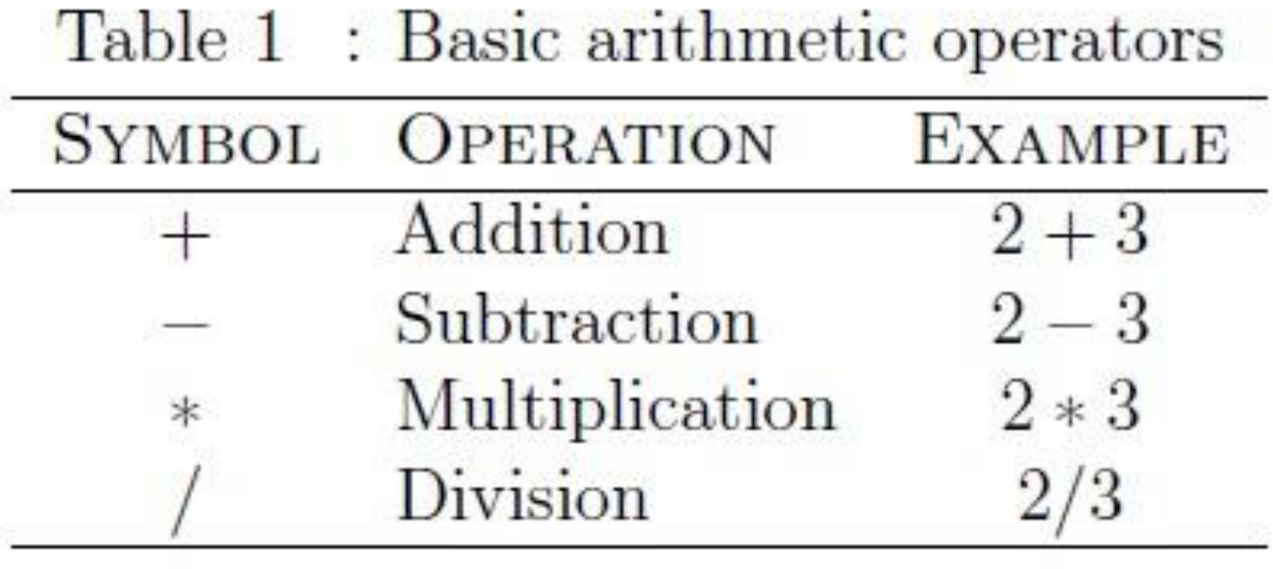


Table 1 : Basic arithmetic operators

Symbol	Operation	Example
$+$	Addition	$2+3$
$-$	Subtraction	$2-3$
$*$	Multiplication	$2*3$
$/$	Division	$2/3$

Quitting MATLAB

To end your MATLAB session, type quit in the Command Window, or select File → Exit MATLAB in the desktop main menu.

Creating MATLAB variables

variable name = a value (or an expression)

For example,

>> x = expression

where expression is a combination of numerical values, mathematical operators, variables, and function calls. On other words, expression can involve:

- ❖ Manual entry
- ❖ built-in functions
- ❖ user-defined functions

Overwriting variable

Once a variable has been created, it can be reassigned. In addition, if you do not wish to see the intermediate results, you can suppress the numerical output by putting a semicolon (;) at the end of the line.

Then the sequence of commands looks like this:

```
>> t = 5 ;
>> t = t +1
    t  =
          6
```

Error messages

If we enter an expression incorrectly,

```
>> x = 10 ;
>> 5 x
??? 5 x
     |
Error: Unexpected MATLAB expression.
```

Making corrections

To make corrections, we can, of course retype the expressions. A previously typed command can be recalled with the up-arrow key ↑ .

Controlling the hierarchy of operations or precedence

```
>> ( 1 + 2 ) * 3
   ans =
          9
```

and, from previous example

```
>> 1 + 2 * 3
   ans =
          7
```

Table 2: Hierarchy of arithmetic operations

Precedence	Mathematical operations
First	The contents of all parentheses are evaluated first, starting from the innermost parentheses and working outward.
Second	All exponentials are evaluated, working from left to right
Third	All multiplications and divisions are evaluated, working from left to right
Fourth	All additions and subtractions are evaluated, starting from left to right

Now, consider another example:

$$\frac{1}{2+3^2}+\frac{4}{5}\times\frac{6}{7}$$

In MATLAB, it becomes

```
>> 1/(2+3^2)+4/5*6/7
ans  =
     0.7766
```

or, if parentheses are missing,

```
>> 1/2+3^2+4/5*6/7
ans  =
     10.1857
```

Entering multiple statements per line

Use commas (,) or semicolons (;) to enter more than one statement at once. Commas (,) allow multiple statements per line without suppressing output.

```
>> a = 7 ; b = cos ( a ) , c = cosh ( a )
   b  =
       0.6570
   c  =
      548.3170
```

Controlling the appearance of floating point number

MATLAB does numerical calculations in double precision, which is 15 digits.

```
>> format short
>> c
     c =
      548.3170
```

If we want to see all 15 digits, we use the command format long

```
>> format long
>> c
     c  =
        5.483170351552120e+002
```

Miscellaneous commands

Here are few additional useful commands:

- ❑ To clear the Command Window, type clc
- ❑ To abort a MATLAB computation, type ctrl-c
- ❑ To continue a line, type ...

Getting help

To view help, select Help from MATLAB toolbar or by typing

```
>>  help
```

Another way to get help is to use the lookfor command. The help command searches for an exact function name match.

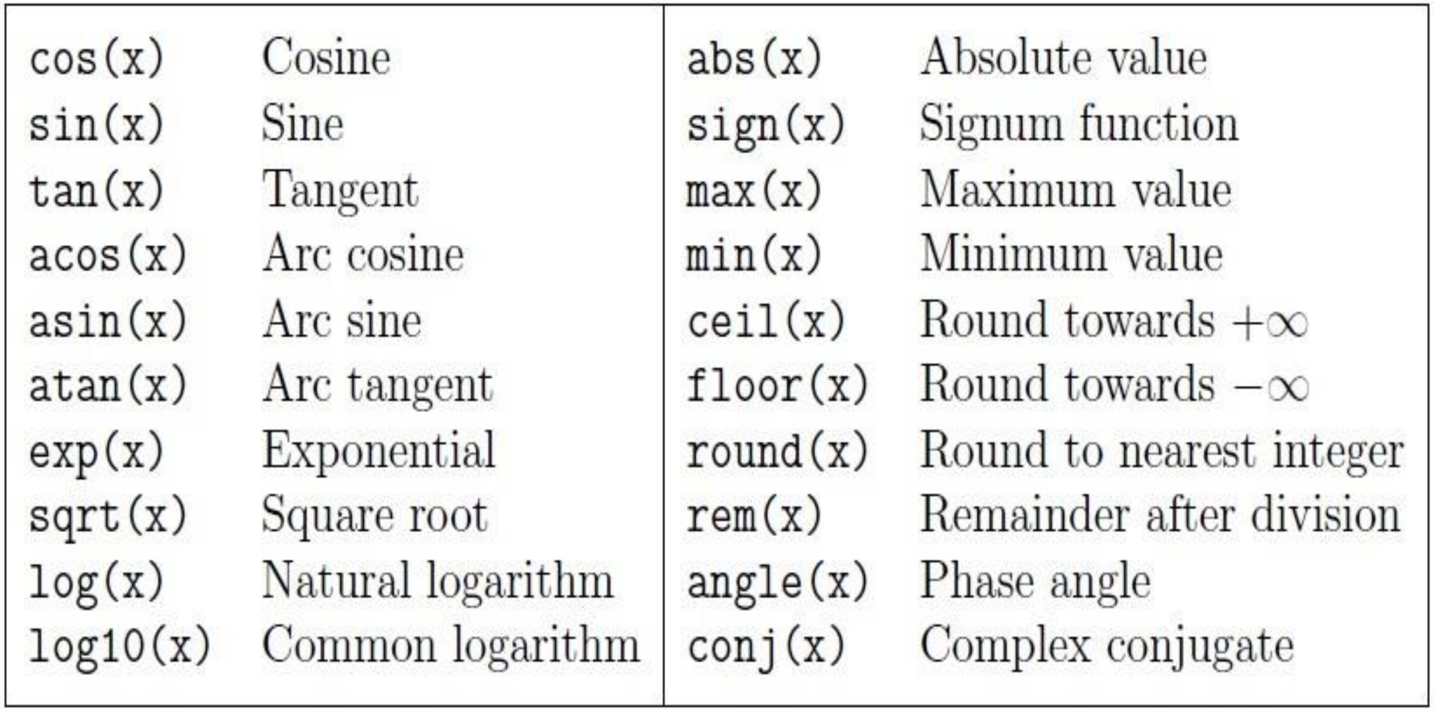
```
>>  lookfor inverse
```

1- Mathematical functions

MATLAB offers many predefined mathematical functions for technical computing which contains a large set of mathematical functions.

Typing **help elfun** and **help specfun** calls up full lists of elementary and special functions respectively

Table 3 : Elementary Functions



`cos(x)`	Cosine	`abs(x)`	Absolute value
`sin(x)`	Sine	`sign(x)`	Signum function
`tan(x)`	Tangent	`max(x)`	Maximum value
`acos(x)`	Arc cosine	`min(x)`	Minimum value
`asin(x)`	Arc sine	`ceil(x)`	Round towards $+\infty$
`atan(x)`	Arc tangent	`floor(x)`	Round towards $-\infty$
`exp(x)`	Exponential	`round(x)`	Round to nearest integer
`sqrt(x)`	Square root	`rem(x)`	Remainder after division
`log(x)`	Natural logarithm	`angle(x)`	Phase angle
`log10(x)`	Common logarithm	`conj(x)`	Complex conjugate

- Moreover, a list of the most common values is given in Table

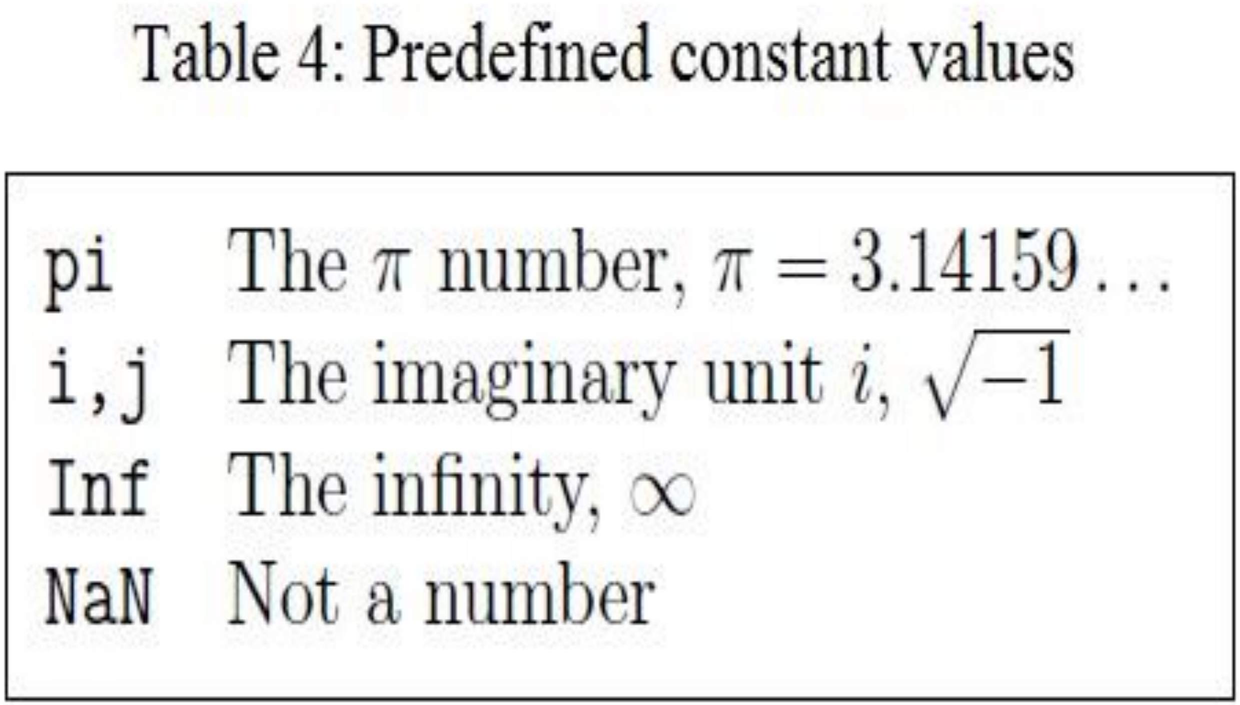


Table 4: Predefined constant values

`pi`	The π number, $\pi = 3.14159\ldots$
`i,j`	The imaginary unit i, $\sqrt{-1}$
`Inf`	The infinity, ∞
`NaN`	Not a number

Examples

the expression $y = e^{-a} \sin(x) + 10\sqrt{z}$, for $a = 5$, $x = 2$, and $z = 8$

```
 >> a = 5; x = 2; z = 8;
>> y = exp (-a) * sin(x) + 10* sqrt(z)
    y =
        28.2904
```

The subsequent examples are

```
>> log(142)
   ans =
         4.9558


>> log10(142)
    ans=
         2.1523
```

Note

the difference between the natural logarithm **log(x)** and the decimal logarithm (base 10) **log10(x)**.

To calculate **sin**$({}^{\pi}/_{4})$ and **e**^{10},

```
>> sin(pi/4)
    ans =
            0.7071
>> exp(10)
    ans =
             2.2026e+004
```

Notes:

- Only use built-in functions on the right hand side of an expression. Reassigning the value to a built-in function can create problems

- There are some exceptions. For example, **i** and **j** are pre-assigned to $\sqrt{-1}$. However, one or both of **i** or **j** are often used as loop indices.

- To avoid any possible confusion, it is suggested to use instead **ii** or **jj** as loop indices.