SLIIT UNI
THE KNOWLEDGE UNIVERSITY

Faculty of Computing

Year 1 Semester 1 (2024)

IT1130 - Mathematics for Computing — Lab Sheet 01

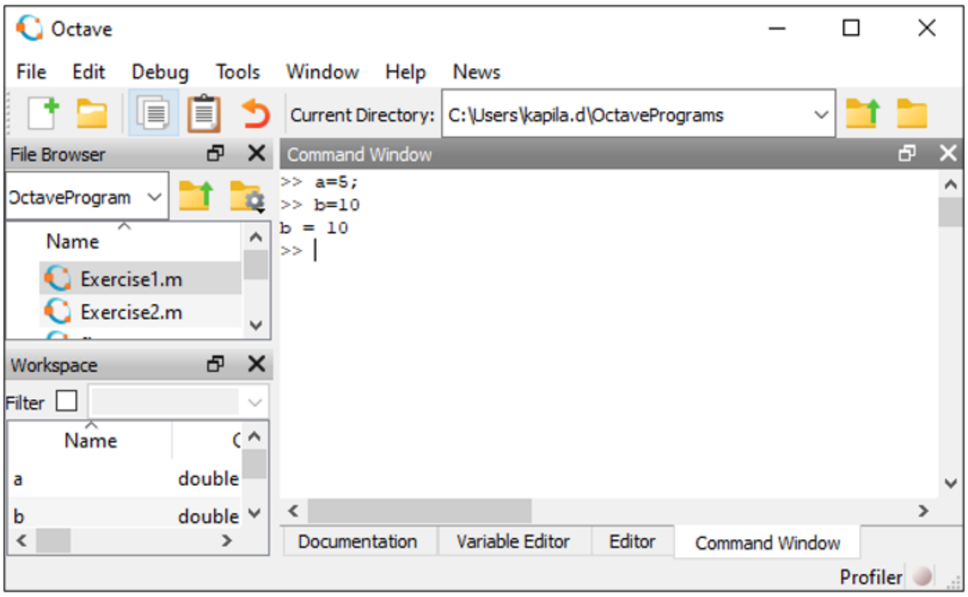
Objective:

Introduce GNU Octave for scientific computing.

Introduction:

GNU Octave is a high-level language primarily intended for numerical computations. It is typically used for such problems as solving linear and nonlinear equations, numerical linear algebra, statistical analysis, and for performing other numerical experiments (https://octave.org/). Also, the Octave syntax is largely compatible with MATLAB.

The current version of Octave executes in a graphical user interface (GUI). The GUI hosts an Integrated Development Environment (IDE) which includes a code editor with syntax highlighting, built-in debugger, documentation browser, as well as the interpreter for the language itself. A command-line interface (see the figure below) for Octave is also available to execute commands and script files in it temporally.



Current Directory: C:\Users\kapila.d\OctavePrograms

Name	C
a	double
b	double

```
>> a=5;
>> b=10
b = 10
>> |
```

Figure 1: GNU Octave IDE.

Script files: Script files are used to keep the code for later use in a simple text file and these have ".m" extension (i.e., Test.m). In order to create a script file, select New from the File menu and select New Script in the Octave IDE. This procedure brings up a text editor window where you can enter Octave commands. The script file can be run either via the command window by typing Test or by selecting Save File and Run/Continue option in the top of the Editor Window.

External functions: In the case of defining a script file for a function, the script file should be named exactly with the name of the function. The name of the file that includes the following function should be "sumproduct.m".

```
function [sum, product] = sumproduct(a,b)
    sum = a + b;
    product = a * b;
endfunction
```

Ending a command: A command can be ended with or without a semicolon (;). If the semi colon is not added when defining a variable (a value holder), Octave will display the value of the variable via the command window. To avoid displaying the value of the variable, put a semicolon at the end of the command.

Defining a variable: Variable are case sensitive and must be initialized when declaring. The type of the variable is automatically selected by Octave.

Commenting: A single-line comment can start with either `%` symbol or `#` symbol (e.g., `% this is a comment.`). Multiline comments can also be made using the following format.

```
%{

This is a
Multiline Comment.

%}
```

Some useful commands and functions:

help plot - to get a description on "plot" in the command window itself.

lookfor str – to search for the string "str" in the documentation.

doc - to open the Octave documentation.

clc - to clear screen.

clear - to clear all the variables from the memory.

close - to close all the figures in the memory.

disp(x) - to display the value of variable x.

plot(x,y) - plot the graph of y vs. x.

printf ('Hello') - to print Hello and enter a new line.

sqrt(x) - to determine the square root of x

nthroot (x,n) - to determine the nth root of real number x.

roots(a,b,c,. . .) - to determine the roots of a polynomial with coefficients given by a,b,c,. . . .

complex(a,b) - to define a complex number with real and imaginary components a, b respectively.

Some mathematical constants:

i, j (or I, J): returns a scalar whose element is pure imaginary unit $\sqrt{-1}$.

e (or exp (1)): returns 2.7183.

pi: returns 3.1416.

inf: returns a scalar whose element is all equal to the IEEE representation for positive infinity.

Arrays and matrices:

```
A=[1,2,3];              % generates 1-D array with A(1)=1, A(2)=2, A(3)=3.
A';                     % generates the transpose of A.
B= [1,2,3;4,5,6];       % generates 2(rows)×3(columns) matrix.
det(C);                 % computes the determinant of the square matrix C.
eig(C);                 % computes the Eigen values of the square matrix C.
inv(C);                 % computes the inverse of the square matrix C.
```

Exercise 1

1. Run Octave on your computer.
2. Use `help`, `lookfor`, and `doc` commands to get help.
3. Carry out few calculator functions on Octave command window. (e.g., `4*24+6*79-7`)
4. Try out the following:
   - Determine $2^{10}, \sqrt{10}, e^{10}, \log_e(10)$.
   - Find the values for: upper round 2.25, truncate 2.25, round 2.25, lower round 2.75.
   - Determine $\sin(1 \text{ rad}), \cos 90°, \sin^{-1}(0.5), \tan^{-1}(-1)$.
   - Do integer division 9 by 4 and find the quotient and remainder.
   - Create any 2×2 real matrix and find the determinant, inverse, and eigen values of it.
5. Recall previous commands with up and down arrows select the appropriate command.
6. Create `Exercise1.m` script file using a few tried out commands on Octave prompt.
7. Run the script file on Octave prompt.
8. Try the following commands:
   - `what`, `who`, `dir`, `ls`, `cd`, `chdir`, `pwd`
   - `which Exercise1`, `type Exercise1`, `delete Exercise1`

Exercise 2

Create `Exercise2.m` script file and write codes to do the following:

1. Assign value 10 to variable `a` and value 5 to variable `b` and calculate:
   - $c1 = a + b$
   - $c2 = a - b$
   - $c3 = 10 \times a + 5 \times b$
   - $c4 = a/b$
   - $c5 = a^b$
2. Print the above calculated values using `printf` function:

```
eg. printf('c1=%.2f\nc2=%.2f\nc3=%.2f\nc4=%.2f\n',c1,c2,c3,c4);
```

   Note that "`%.2f`" keeps the number to two decimal places (fixed-point format).