

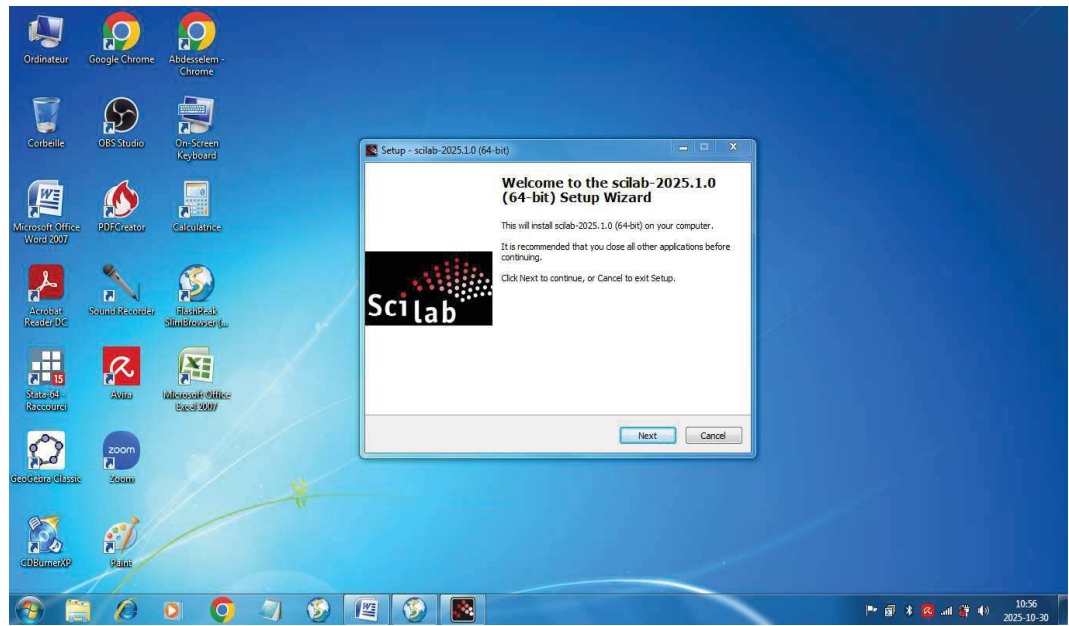
Chapter 1. Getting Started



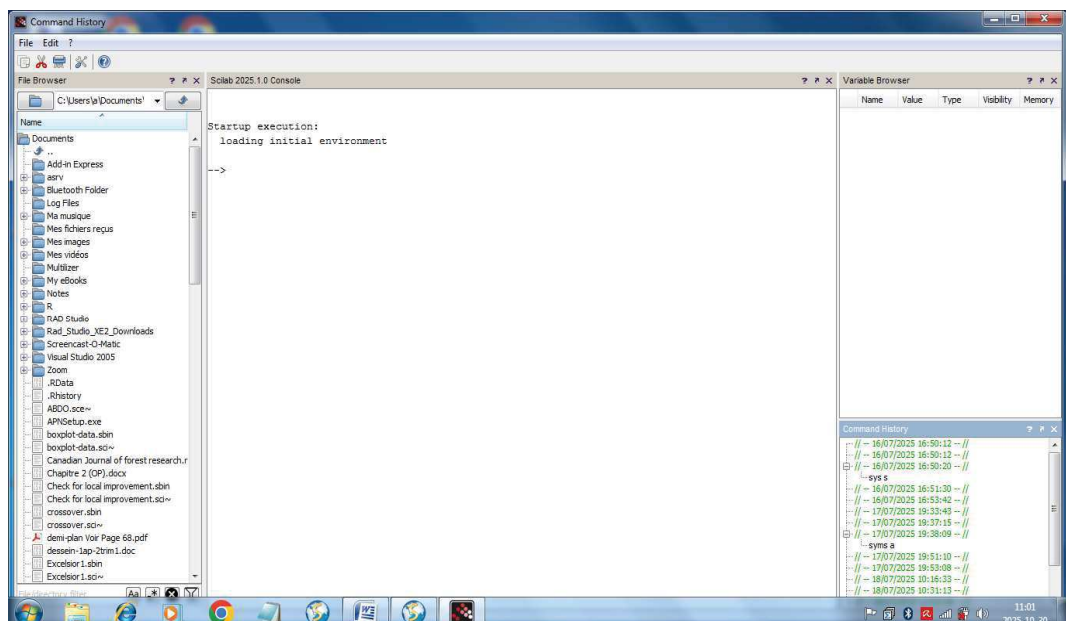
Scilab (a contraction of **Scientific Laboratory**) is open-source software available for download from the official webpage <https://scilab.org>. Scilab is supported on UNIX, Macintosh, and Windows environments. It includes libraries of a large number of built-in mathematical functions and a wide range of packages of pre-written programs, called *toolboxes*, which are managed through **ATOMS** (Automated **T**oolbox **M**anagement **S**ystem). Scilab also works as a programming language like C, C++ etc, and the built-in functions can be used into the users' programs. In addition, it has a variety of graphical capabilities. This feature of Scilab makes it user friendly interactive software. In Scilab, all variable types; including integer, real, complex, Boolean, string, polynomial, are treated as matrices. This chapter introduces the basic concepts needed to begin working with Scilab. It explains how to start the program, manage variables, set the working directory, save the work environment, and use essential functions and commands.

1. Startup and variable help:

The installation of Scilab is easy:



Once installed on your computer system, Scilab is ready to use. You can open Scilab by double-clicking the Scilab icon on your desktop or by clicking the icon that appears after typing "Scilab" in the Windows search bar. The Scilab window should then appear on your screen, looking like this:

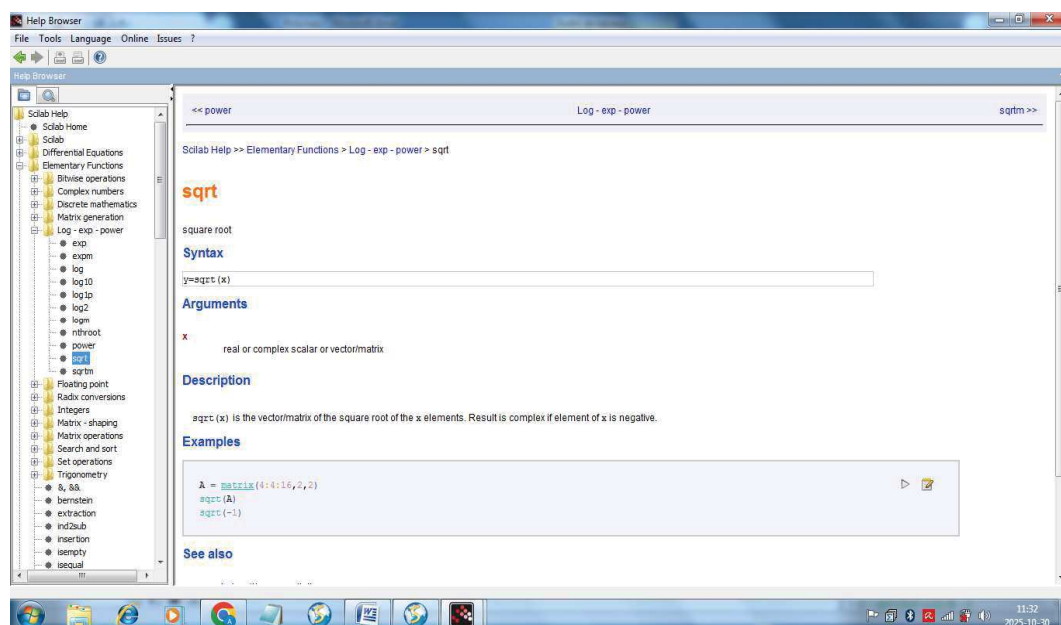


The window on the left, the *File Browser*, allows you to locate your files. In the center, the *Console*, is the area where you type commands and where the results are displayed. This is the part you will use most often. In this area, the prompt, appearing as `-->`, indicates that Scilab is waiting for a command. At the top right, the *Variable Browser* there is a summary of all the variables used, along with their properties. Below that is the *Command History*. If you need to find a command you've already entered, there's no need to retype it; it has been stored in memory and is available in this frame.

Scilab has an integrated *help* system that allows you to obtain information about commands and functions. For example, to get help on the **sqrt** (square root) command, simply type:

```
--> doc sqrt
```

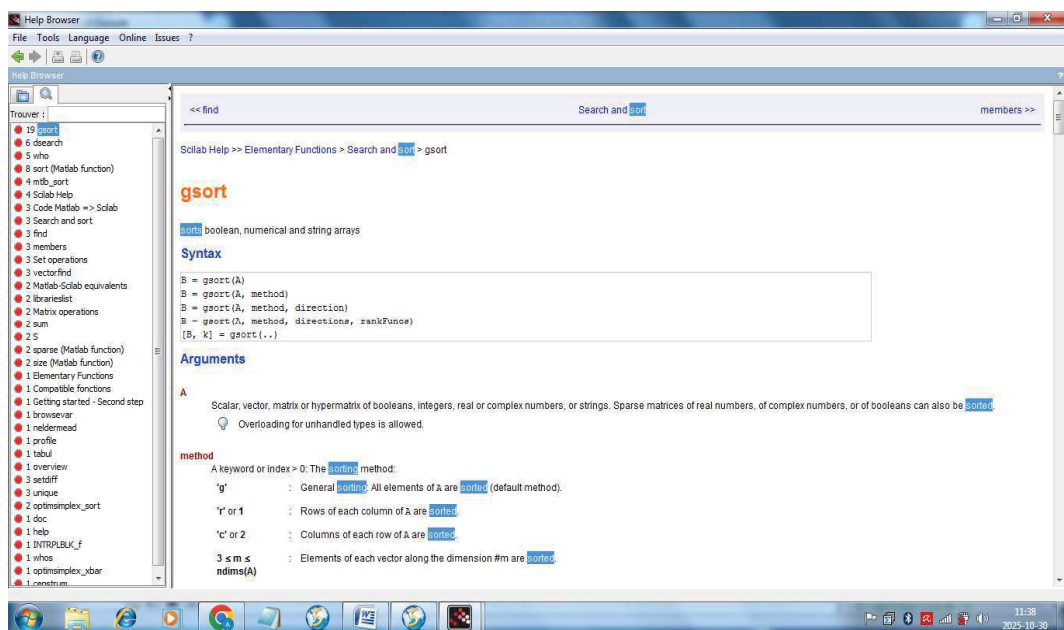
The help window should then appear on your screen, looking like this:



The Scilab help pages provide a detailed description of the **sort** command, its syntax, usage examples, and links to similar or related commands. When searching for a command to perform a particular task without knowing its name, you can use the **apropos** command, which searches the help pages for text containing a specific string of characters. For example, if you want to sort a vector or matrix, you would type:

```
--> apropos sort
```

The help browser will then suggest the **gsort** command, which is an excellent suggestion:



In general, you can also simply type **doc** and navigate through the table of contents.

3. Variables:

Scilab allows you to create variables to store data. Use the equal sign = to assign a value to a variable. In this example:

```
-->a = 5
```

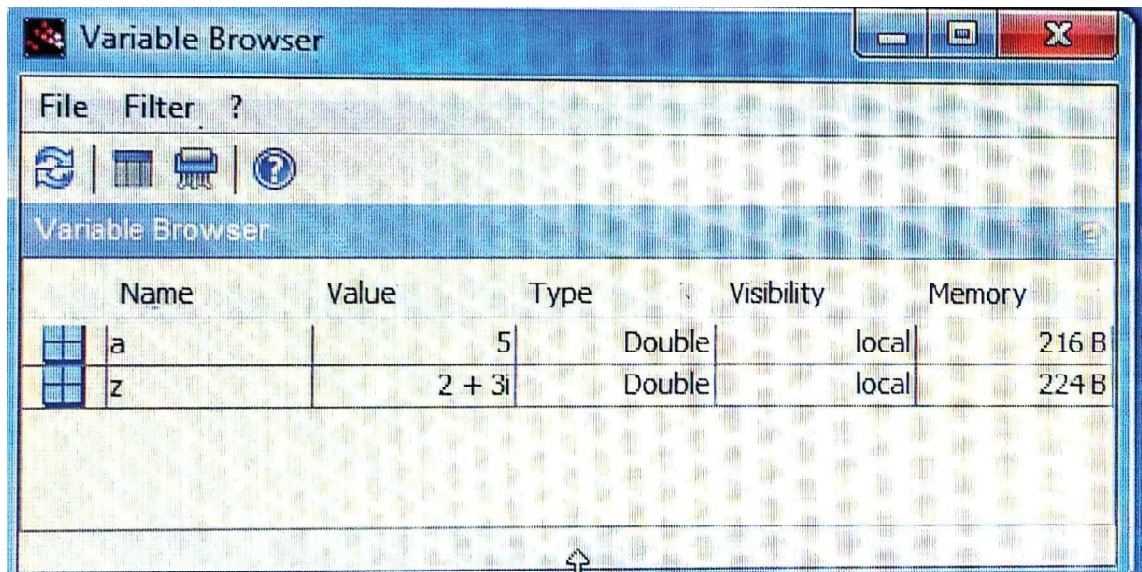
We have created a variable **a** to which we have assigned the value **5**.

We can also create complex variables by using the **%i** notation for the imaginary part. In the example:

```
-->z = 2 + 3*%i
```

The variable **z** now contains the complex number **2 + 3i**.

The properties of the variables **a** and **z** are summarized in the *Variable Browser* window below.



The screenshot shows the 'Variable Browser' window in Scilab. It contains a table with the following data:

Name	Value	Type	Visibility	Memory
a	5	Double	local	216 B
z	2 + 3i	Double	local	224 B

For example, in Scilab, the complex variable **z** has the **type “double”**, meaning it stores numerical values in double-precision format. Its **visibility** is **local**, which means it is only accessible within the function or workspace where it was created, and its **memory usage** is **224 B**.

4. Working directory:

The working directory is the folder on your computer where Scilab looks for files by default. You can find out the current working directory using the **pwd** (**P**rint **W**orking **D**irectory) command. This will display the path of the current working directory.

You can use the **cd** (**C**hange **D**irectory) command to change the working directory. For example, to change the working directory to "Documents", type:

```
--> cd("C:\...\Documents")
--> pwd()
```

This will change the working directory to the "Documents" folder and display the new path.

5. Saving the workspace:

To exit Scilab, simply type **quit** or **exit**. It is possible to save the workspace before exiting by using the **save** command. To load a previously saved workspace, use the **load** command. For example:

```
--> a = 1, b = -9
--> save('vals.sod', 'a', 'b')
--> quit
--> load('vals.sod', 'a', 'b')
--> a, b
```

6. Functions and commands:

Functions are essential in Scilab for performing mathematical operations and returns a result. Scilab offers many built-in functions. You usually call a function with parentheses () and sometimes with input arguments. Examples:

```
--> date()
--> x = -10
--> abs_x = abs(x)
--> disp(abs_x)
```

date() : shows the current date and time.

abs(x) : gives the absolute value of a number.

disp(abs_x): displays a value or message on the screen. It shows the result stored in **abs_x**.

A *command* is an instruction that tells Scilab to do something, but it may not return a value. Examples:

```
--> clear  
--> clc  
--> quit
```

clear: this command removes all variables from the workspace (memory).

clc: this command clears the console window, it removes all text shown on the screen but keeps the variables in memory.

quit: this command closes Scilab completely.