

# Write and Run MATLAB M-Files

## 1 Create m-files

1. Open Matlab. Notice the sections of the window: editor, command line, history, data window, etc.
2. Create a new folder and change the MATLAB current directory to the folder you created.
3. There are three ways to open a new m-file:



(1) Find and click this icon

(2) Go to File – New – Blank M-File

(3) Type `edit filename.m` in the Command Window; note that for function m-files, *filename* must match the name of the function you are going to write in this m-file.

## 2 Write function m-files



1. In the blank M-file you just opened in the editor:
  - Type `function f = myFunc(x, y)` as the first line. If your file name is, e.g. "lab4func.m", type `function f = lab4func(x, y)` instead. **The function name has to match the file name.** By including this first line in the M-file, you are telling MATLAB that this M-file is a user-specified function. A function file is not executable by itself; it can only be called in other commands.
  - Then choose a function, e.g.  $f(x, y) = x^2 + y$ , type `f = x.*x + y` as the content of the M-file.
2. Save your M-file; if you chose method (1) or (2) to open your blank M-file, now you'll need to give your M-file a name which matches your function name.

Note that you should use `.*` instead of `*` because we want the function input `x` and `y` to be vectors, and the dot-operator `.` is meant to repeat operations on the members of the vector. See Basic Operations: Dot-Operators for more details.

### 3 Write script m-files

- Now create another blank M-file to read and plot the function you just wrote:
  - In this file, you need to
    - generate a vector  $x$ , e.g. `x = linspace(-1, 1, 200);`
    - generate a number  $y$ , e.g. `y = 5;`
    - generate a vector *out* to be a function of  $x$  based on the function M-file you just wrote, e.g. `out = myFunc(x, y);`
    - make a plot of *out* vs.  $x$ , e.g. `plot(x, out)`
  - You can also add documentation for your plot using `xlabel`, `ylabel`, `title`, or `legend`.
- Save this file. This time, name your M-file something like “lab4plot.m” or “myPlot.m”, whatever you want, but NOT “plot.m” since “plot” is already a MATLAB internal function.

### 4 Run script m-files

- Test your script M-files by typing `lab4plot` or `myPlot` (whatever your filename is) at the command line, or find this icon  on the Toolbar in the Editor window. Do you see a figure window popping up?
- If you can't get through Steps 3–5, don't worry. Go to the Lab04 website and download two files: <functionExample.m> and <plotExample.m>. Save them into the folder you created on the desktop. Open each of these files into the editor in MATLAB and read through them. Pay close attention to the comment made at the end of each line telling what that command instructs MATLAB to do. Then run <plotExample.m> by typing `plotExample` in the Command window or click  in the Editor window. What do you see?
- Now let's turn the heat up. Open the Editor window with your function M-file and choose a new function to define in this function file. Examples: `sin()`, `cos()`, `exp()`,  $1/x$ , etc. Save and run your script M-file. (*Note:* you may need the dot-operator for some arithmetic operations. On the other hand, some MATLAB internal functions (like `sin()`, `cos()`, `log()`) are vectorized, which means they operate automatically over each member of an array without the need for an explicit loop.)