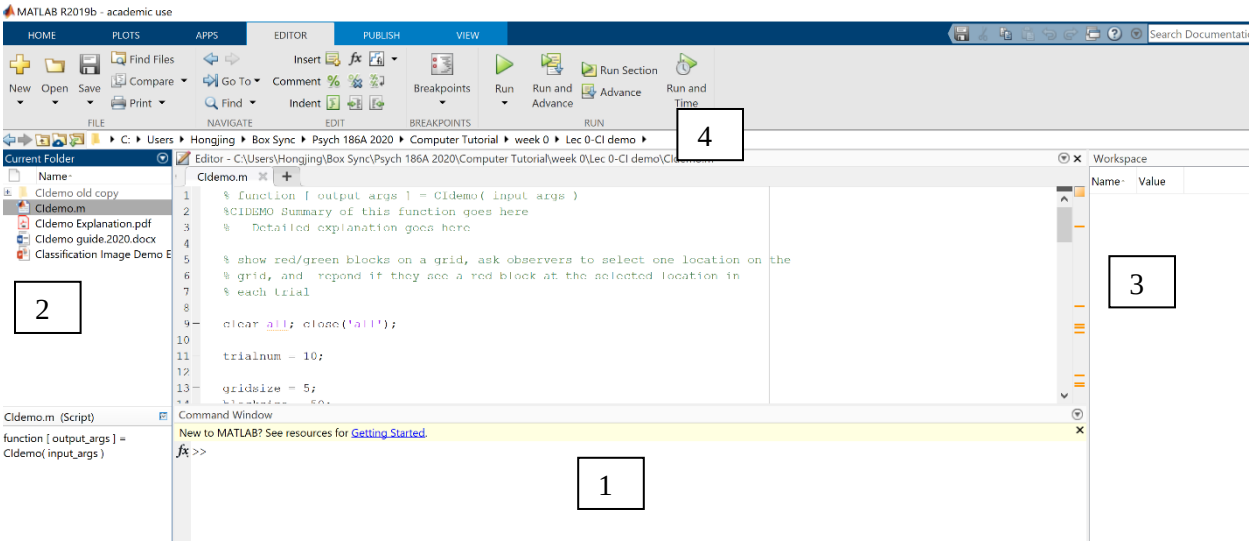
Psych 186A
Week 1

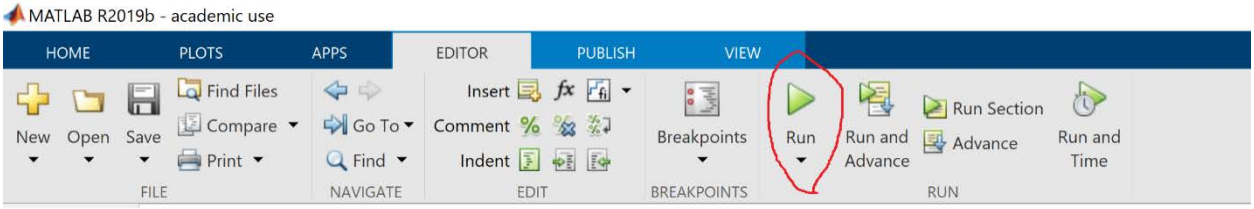
MATLAB Basics I: Running MATLAB statements and Basic MATLAB programming

I. Running MATLAB statements



1. **Command Window**: where you type your commands; it also displays answers of your operations
2. **Current Directory**: where MATLAB's current "attention" is
3. **Workspace**: what's in MATLAB's "working memory"
4. **Current Path**: the location of current directory

- MATLAB is a **script language**: You don't need to compile your program. It runs line by line.
- MATLAB codes can be written in an M-file (e.g., CIdemo.m). You can create a new .m file by going to **File → New → M-File** (or by clicking the icon "New M-File" on top left)
- After typing all your codes, you can click the **SAVE AND RUN** button in the **Editor** window (see the screencap on the next page), or hit the function key **F5** to run it.
- **Writing lines in a .m file is same as writing lines in the Command Window** (at least for almost all commands we are going to use). The only difference is that you're more organized in a .m file. Therefore, you can try out some simple commands/built-in functions in the Command Window.



- Command Window displays the answer of your computation. However, if you don't want your answers to be shown (especially when your answer is, e.g., a gigantic matrix), insert a **semicolon (;)** to the end of your statement. Try the difference between **11+4** and **11+4;**
- MATLAB stands for **MATrix LABoratory**. It is a very powerful **matrix calculator**. In fact, most variables in MATLAB are represented as matrices. E.g., when you define a variable **x = 3**, MATLAB actually takes **x** as a 1-by-1 matrix (technically, a scalar in linear algebra terms).

II. Basic MATLAB programming

MATLAB is quite similar to C++, except that MATLAB is more "high-level" (e.g., you don't need to define the type of a variable). In terms of syntax, the translation from C++ to MATLAB should be easy.

Variables: Variable names are case-sensitive. Underscore _ is the only accepted symbol in variable names. You CANNOT begin variable names with numbers or _ (e.g., Good: x, y30, Sum_2; Bad: 20x, _y2)
Matrix: first dimension = row, second dimension = column (you may go beyond two dimensions, but let's stick to normal, 2D matrices for now). In MATLAB, the first element in a vector is v(1) (unlike C++, which is v(0)).
Commenting: Use the ampersand % character

Defining variables / assigning values to variables:

```
x = 17;                      % as simple as that!
v = [3 4 5 6];               % defining an array, or 1x4 vector
v = 3:1:6 or v = 3:6;        % same as above, start:step:end
m = [11 22 33; 44 55 66];    % semicolon ; separates rows in a matrix
                             % this gives you a 2x3 matrix
ourclass = 'Psych186A';      % use single quotes ' for strings/chars
```

Recalling variables

```
x                            % just type it, case-sensitive
v(3)                         % the third element in v
v([4 1])                     % 4th and 1st elements in v
v(end)                       % the last element in v
m(1,2)                       % row 1, column 2 in m
m(2,:)                       % the entire row 2 in m
ourclass(5)                  % 5th character in ourclass
ourclass(9:-1:1)             % guess what it is and try it!
```

Modifying variables

```
v(3) = 7                     % change 3rd element of v to 7
m(2,2) = 100                 % change row 2, column 2 of m to 100
m(:,3) = [30; 60]            % change column 3 to [30; 60]
```

Logical values and operations

`true` (or 1)	`false` (or 0)
&& AND	|| OR
== equal to	~= not equal to
> larger than	< smaller than
>= larger than or equal to	<= smaller than or equal to
`xor` Exclusive OR	~ logical not

Arithmetic Operators:

`z = x + y` (also try `-`, `*`, `/` and `^`)
`c = a + b` (also try `-`)

`m = a.*b` (also try `./`, element-by-element multiplication/division)
`d = a.^b` (element-by-element power)
`bt = b'` (matrix transpose)
`mm = a*bt` (matrix multiplication)
`X = A\b` (matrix "division", for solving systems of equations)

Basic I/O

```
yourage = input('Enter your age: ');
disp('Hello! I guess you were born in the following year:');
disp(2020-yourage);
```

IF statements (* elseif and else are optional)

```
if AB(1,1) > 90
    disp('YEAH! I got an A!');
elseif AB(1,1) > 80
    disp('Not bad. Got a B.');
else
    disp('Darn it...');
end
```

Loops (loops are usually slow: try to use matrices whenever possible!)

```
% for loop: add 0.1 to x and divide v by 2, for 7 times
for j = 1:7
    x = x + 0.1;
    v = v/2;
end

% while loop: minus 2 from z until it is smaller than 10
while z >= 10
    z = z - 2;
end
```

In-class Exercise

Q1. Open a new .m file called **q1.m**.

a) Construct vector **x** = [11 12 13 14 … 30].
b) Construct vector **y** = [1 3 5 7 … 39].
c) Compute a vector **z**, which contains the reciprocals of the elements in **x**.
(i.e., **z** = [1/11 1/12 1/13 1/14 … 1/30])
d) Multiply each element in **y** by its correspond element in **z**.
Name the resulting vector **s**.
e) Sum up all the numbers in **s**, and display its value in the Command Window.

Q2. Study the following sequence of matrices:

$$[1] \quad \begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix} \quad \begin{bmatrix} 1 & 2 & 3 \\ 2 & 4 & 6 \\ 3 & 6 & 9 \end{bmatrix} \quad \begin{bmatrix} 1 & 2 & 3 & 4 \\ 2 & 4 & 6 & 8 \\ 3 & 6 & 9 & 12 \\ 4 & 8 & 12 & 16 \end{bmatrix} \quad \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 4 & 6 & 8 & 10 \\ 3 & 6 & 9 & 12 & 15 \\ 4 & 8 & 12 & 16 & 20 \\ 5 & 10 & 15 & 20 & 25 \end{bmatrix}$$

In a new .m file **q2.m**:
Construct the matrix **m**, which is the 10^{th} matrix in this sequence, and use "for" loop to construct the matrix and display it in the Command Window.

Q3. Suppose we've gathered 30 people's body measurements.
After sorting them by heights, we found that their heights are equally spaced by half an inch, with the shortest person being 60 inches.
In a new .m file **q3.m**:

a) Compute the tallest person's height, and store it in the variable **Tallest**.
b) Construct an array **Hinch** to represent the heights in ascending order.