

ENGG1003 - Thursday Week 11

MATLAB vs. Python

Sarah Johnson and Steve Weller

University of Newcastle

20 May 2021

Last compiled: May 20, 2021 3:27pm +10:00

Lecture overview

1 Context

- ▶ ENGG1003
- ▶ what is MATLAB?
- ▶ do we even need MATLAB?

2 MATLAB vs. Python

- ▶ features and philosophy
- ▶ key language details in MATLAB

3 Next steps

- ▶ getting MATLAB, if you need it
- ▶ Octave: free & mostly compatible with MATLAB

1) Context

- ≤ 2020 , ENGG1003 used *MATLAB* and *C*
 - ▶ **from 2021, ENGG1003 uses Python only**
 - ▶ ...yet some students will use MATLAB &/or C in later courses
- today's lecture: overview of MATLAB
- Monday week 12: overview of C

What is MATLAB?

- MATLAB is a computing environment and programming language for *matrix manipulation*
 - ▶ a matrix is a 2D array
 - ▶ MATLAB is an abbreviation for “matrix laboratory”
- MATLAB offers many additional “Toolboxes”:
 - ▶ control design
 - ▶ image processing
 - ▶ machine learning
 - ▶ digital signal processing
 - ▶ computational fluid dynamics
 - ▶ etc.

Do we even need MATLAB?

- **MATLAB is not assessable in ENGG1003**
- BUT...MATLAB is currently used in a number of later courses in Engineering programs
- non-exhaustive list:
 - CHEE4945, CHEE4975, ENGG2440, AERO3600, MCHA3400, MCHA3500, MCHA4100, ELEC2132, ELEC2430, ELEC3400, ELEC3410, ELEC3540, ELEC4100, many FYPs, ...

2) MATLAB vs. Python

Features and philosophy:

- **MATLAB** is proprietary, closed-source software
 - ▶ developed by MathWorks <https://www.mathworks.com/>
 - ▶ MATLAB license is free for students . . .
but very expensive otherwise
- **Python** is free and open-source software
 - ▶ you can keep programming in Python once you graduate!

Advantages of Python over MATLAB

- Python libraries offer similar functionality as MATLAB toolboxes
 - ▶ Python libraries growing much faster, are free, & supported by active online community
- Python is a very popular, in-demand language
 - ▶ many students are finding that employers are requesting Python coding skills for both positions and internships
 - ▶ MATLAB: 4 million users
 - ▶ Python: > 8 million users (2 million new users in 2018)
 - ▶ Python ranked #1 most popular language in 2020 (IEEE)
- *MATLAB vs Python: Why and How to Make the Switch*
<https://realpython.com/matlab-vs-python/>

MATLAB: key language details

- syntax
- arithmetic and relational operators
- flow control
- arrays
- plotting
- functions

**If you're familiar with Python at level of
ENGG1003, estimate transition to
MATLAB in 1–2 weeks**

MATLAB syntax

- MATLAB comments start with `%`. Python comments start with `#`
- white-space and indenting are very important in Python. MATLAB does not require the same (but it is highly recommended anyway for readability)
- MATLAB function `disp()` replaces `print()`
- help for a function via `help fname` rather than `help(fname)`

Arithmetic and relational operators

- addition, subtraction, multiplication and division are the same as Python. A difference is that
 - ▶ MATLAB uses `^` not `**` for exponential
 - ▶ ie: `x**2` becomes `x^ 2`
- relational operators `==`, `>`, `<`, `>=`, `<=` are the same, except
 - ▶ MATLAB uses `~=` instead of `!=` for 'not equal to'
- value of a MATLAB variable is *automatically* printed to the command window (console), unless `';` used to suppress
 - ▶ ie: `a = 3` prints value of `a`, whereas `a = 3;` does not

Flow control

- `if-else` statements and `for` / `while` loops work exactly the same way in both languages, with some small differences in syntax:
 - ▶ MATLAB does not need the `:` used in Python at the end of the loop definition / if condition
 - ▶ MATLAB designates the end of an if statement or loop by `'end'` instead of by indenting
 - ▶ Python shortens `elseif` to `elif`. MATLAB does not

Flow control

Nested If-Else

```
1 num = 10                                # Python
2 if num == 10:
3     print("num is equal to 10")
4 elif num == 20:
5     print("num is equal to 20")
6 else:
7     print("num is neither 10 nor 20")
```

```
1 num = 10;                               % MATLAB
2 if num == 10
3     disp("num is equal to 10")
4 elseif num == 20
5     disp("num is equal to 20")
6 else
7     disp("num is neither 10 nor 20")
8 end
```

Flow control

Example: use a `for` loop to add integers from 1 to 10

```
1 sum = 0                                # Python
2 for i in range(1,11):
3     sum = sum+i
```

```
1 sum = 0;                               % MATLAB
2 for i = 1:10
3     sum = sum+i;
4 end
```

Flow control

Example: use a `while` loop to find the first (smallest) integer which, when squared, is greater than 100

```
1 n = 1                                # Python
2 n_squared = 1
3 while n_squared < 100:
4     n = n+1
5     n_squared = n**2
```

```
1 n = 1                                % MATLAB
2 n_squared = 1
3 while n_squared < 100
4     n = n+1
5     n_squared = n^2
6 end
```

Arrays

- MATLAB arrays work a lot like `numpy` arrays. If you can work with Numpy arrays you will find MATLAB arrays are easy. Both do vectorisation in the same way
- There are a couple of syntax differences to note:
 - ▶ In MATLAB, when you want to index an array, you use round brackets `' () '`. Square brackets `' [] '` are used to create arrays

```
1 arr = np.array([10, 20, 30])    # Python
2 s = arr[i]
```

```
1 arr = [10, 20, 30]             % MATLAB
2 s = arr(i)
```

Arrays

- 2D arrays use ';' to designate the next row

```
1 a = np.array([[2,3],[4,5]])      # Python
2 s = arr[i,j]
```

```
1 a = [2 3; 4 5]                  % MATLAB
2 s = a(i,j)
```

- All zero / all one matrices are very similar

```
1 m = np.zeros([5,10,3])          # Python 5x10x3 array
```

```
1 m = zeros(5,10,3)              % MATLAB 5x10x3 array
```

- In MATLAB a new copy of an array is created by default

```
1 b = a.copy()                   # Python
```

```
1 b = a                          % MATLAB
```


Arrays

- in Python, the index of the first element in an array is '0', in MATLAB it is '1'
- in Python, the index of the last element in an array is '-1', in MATLAB it is 'end'
- length of an array is `length` rather than `len`
- array slicing works similarly in MATLAB to Python
- no need to call `linspace()` to create an array in MATLAB, `start:step:end` will do

```
1 a = 5:1:100           % MATLAB
```

Plotting

- plotting in MATLAB is very similar to `Matplotlib.pyplot`
- including the functions `plot()` `imshow()` `subplot()` `scatter()` `title()` `axis()` `xlabel()` `tick()` `figure()` and many more
- two tips:
 - ▶ in Python plotting a new curve will add it to the figure. In MATLAB it will replace the first curve unless you use a `hold on` command first
 - ▶ MATLAB does not need a `show()` command, the figure is shown automatically and does not need to be closed for the remainder of the program to be run

Plotting

```
1 import numpy as np                                # Python
2 import matplotlib.pyplot as plt
3 v0 = 5
4 g = 9.81
5 t = np.linspace(0,1,1001)
6 y = v0*t - 0.5*g*t**2
7 plt.plot(t, y)
8 plt.xlabel('t (s)')
9 plt.title('Velocity over time')
10 plt.show()
```

```
1 v0 = 5;                                           % MATLAB
2 g = 9.81;
3 t = 0:.001:1;
4 y = v0*t - 0.5*g*t.^2;
5 plot(t, y)
6 xlabel('t (s)')
7 title('Velocity over time')
```

Functions

- MATLAB functions work similarly to Python
 - ▶ declared by defining the function name, inputs, outputs and operation in the function declaration, then call the function by name as needed
 - ▶ functions start with the keyword `function` not `def`
 - ▶ function output is defined at the *start* of the function, not the end
 - ▶ end of function is indicated with the keyword `end`

```
1 def addition(num_1, num_2):                                # Python
2     total = num_1 + num_2
3     return total
```

```
1 function [total] = addition(num_1, num_2) % MATLAB
2     total = num_1 + num_2;
3 end
```

Next steps

- getting MATLAB, if you need it *for later courses*
 - ▶ **MATLAB is not assessable in ENGG1003**
 - ▶ <https://www.newcastle.edu.au/current-students/support/it/software-and-tools>
- Octave: free & mostly compatible with MATLAB
 - ▶ <https://www.gnu.org/software/octave/index>