

Session 1: Introduction to Python from the Matlab perspective

October 8th, 2018 | Sandra Diaz

Working with examples in this course

- Git repository
 - Work: Exercises we will be interactively working on
 - Slides

Starting with Python

- IDEs: Pyzo, PyCharm, Spyder, Wing
- http://scipy.github.io/old-wiki/pages/NumPy_for_Matlab_Users.html
- <https://www.python.org/>
- <https://docs.python.org/3/>
- <https://www.scipy.org/>
- <http://www.numpy.org/>
- <https://matplotlib.org>

Executing code in python

- The python interpreter
 - `$ python my-program.py`
- Ipython
 - Interactive shell for the python interpreter
- **Jupyter notebooks**
 - HTML-based notebook environment for Python, similar to the Matlab frontend

Running Python

- Versions 2 and 3 are commonly found
- In this course we focus on Python 3
 - \$ python --version

Introduction to Python

- Modules
 - `import math`
 - `from math import *`
 - `from math import cos`
 - `print(dir(math))`
 - `help(math.cos)` or `help(math)`

Introduction to Python

- Indentation is very important!
 - Do not use tabs
- Variable names
- Variable types
 - Fundamental types: int, float, bool, complex
 - Type casting
- Operators
 - All works the same as Matlab but:
 - / is always a floating point division in Python 3 (Python 2: from `__future__` import division)
 - // integer division
 - ** power
 - Booleans: and, not, or
 - = identical (objects or fundamental types)

Introduction to Python

- List of equivalent operations:
 - <http://mathesaurus.sourceforge.net/math-synonyms.pdf>

Introduction to Python

- Indexing
 - In Python indexing starts at 0
 - In Python, indexing is done with [] not ().
 - Use one [] for each dimension to index.
 - [start:stop] extract a portion of an indexable object from start until stop-1
 - [start:end:step] also works
 - Omitting one gives defaults
 - Index -1 is the last element, -2 the second last and so on so forth

Introduction to Python

- Indexable objects and basic data structures
 - Strings are arrays of characters
 - Lists are arrays of any type and do not have to be all of the same type: `x = [5, 'h', 2.0]`
 - Numerical arrays:
 - `range(start, stop, step)`
 - `drange`
 - Dictionaries: `{'key1': 'value', 'key2': 3}`

Introduction to Python

- Sending output to the screen: unlike Matlab, in Python we use the print command to send variable values and text to the screen
 - `print (variable)`
 - `print ("Hello Python!")`
 - `print ('{} {} {}'.format(2, 2*2, 2*2*2))`
 - `print ('Port your code from {} to {}'.format('Matlab', 'Python'))`

Introduction to Python

- Open a file
 - `file = fopen('myfile.dat', 'w') → file = open('myfile.dat', 'w')`
- Output to a file
 - `fprintf(file, 'From Matlab') → file.write("To Python")`
 - Variables must be converted to a string:
`file.write('{} {} {}'.format(2, 2*2, 2*2*2))`
- Read from a file
 - `content = fread(file) → content = file.read()`
 - `line = file.readline()`
- Close a file
 - `fclose(file) → file.close()`

Introduction to Python

- Flow control
 - Important that the scope of the flow control structure is defined by the indentation. There is no 'end' statement like in Matlab
 - Loops
 - for x in range(10):
 - while x<10:
 - parfor can be implemented in many ways in python: MPI or OpenMP
 - <http://pymotw.com/2/multiprocessing/basics.html>
 - http://www.reddit.com/r/Python/comments/j3qjb/parformatlabpool_replacement/
 - Control
 - if, elif, else
 - no real switch statement but can use a dictionary

Introduction to Python

- Tip: Code in Python is not checked for syntax errors before execution because it is an interpreted language
- Code that is not executed is not verified
- Test your code for all cases (unit testing)

Introduction to Python

- Functions
 - Use the **def** keyword to identify a function
 - Be careful, the extent of the function is defined by indentation
 - Use the return statement to send back one or more values from your function
 - Lambda functions are special functions without a name

Virtual environments

- Enclosed spaces with an independent Python installation in a self-contained directory
- Allow the installation of packages without administrator permissions
- Flexibility and portability
- Several independent Python spaces in the same computer
- *source env_path/bin/activate*
- <https://docs.python.org/3/tutorial/venv.html>

References

(1) Based on the work by J.R. Johansson <http://jrjohansson.github.io>