

Python

A quickstart into the very basics
Get to know your user environment

Thank You

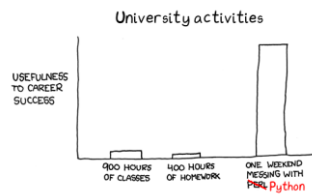
- <https://github.com/gibex/training-material/tree/master/Python>
- *Whirlwind Tour of Python* by Jake VanderPlas (O'Reilly).
Copyright 2016 O'Reilly Media, Inc., 978-1-491-96465-1.
<https://www.oreilly.com/programming/free/files/a-whirlwind-tour-of-python.pdf>
- University of Virginia, Advanced Research Computing Services, Python Quickstart
<https://arcs.virginia.edu/python-quickstart>
- <http://www.cs.cornell.edu/courses/cs1110/2018sp/>
- https://fabienmaussion.info/scientific_programming/html/00-introduction.html

See also

- <https://www.southampton.ac.uk/~fangohr/teaching/python/book.html>
- <https://www.math.ubc.ca/~pwalls/math-python/>
- <http://troll.cs.ua.edu/ACP-PY/index.html>
- <https://data-flair.training/blogs/python-lambda-expressions/>
- <http://pages.physics.cornell.edu/~myers/teaching/ComputationalMethods/GettingStarted.html>
- <https://anh.cs.luc.edu/python/hands-on/3.1/handsonHtml/index.html>
- <https://www2.cs.duke.edu/courses/spring18/compsci101/index.php>
- <https://github.com/parrt/msan501>
- <https://docs.python-guide.org/intro/learning/>

Tutorials

- <https://www.python.org/about/gettingstarted/>
- <https://realpython.com/>
- <https://www.learnpython.org/>
- Cheat sheets
- <https://www.datacamp.com/community/data-science-cheatsheets>



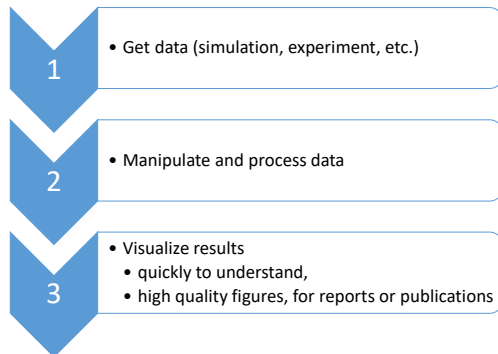
https://fabienmaussion.info/scientific_programming/img/00_messing_python.png

Python: setting the scene

get comfortable within the Python universe

What is the playfield?

- *The best way to learn a language is to "get your hands dirty"*



What is Python?

- From www.python.org: "Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance."
- Python is a general purpose programming language used for a huge variety of purposes. It's user community is growing rapidly! (<https://stackoverflow.blog/2017/09/06/incredible-growth-python/>)

What is Python?

- a **general purpose interpreted** programming language.
- a language that supports multiple approaches to software design, principally **structured** and **object-oriented** programming.
- provides **automatic memory management** and **garbage collection**
- **dynamically** typed.

Brian Gregor (BU): A Brief Introduction to Using Python for Computational Neuroscience

Why Python?

- Python is quick to program in
- Python is popular in research, and has lots of libraries for science
Widely used – extensive capabilities, documentation, and support
- Python interfaces well with faster languages
- Python is free
- Cross-platform (Windows, Mac, Linux)
- Access to advanced math, statistics, and database functions
- *Why write programs for research?*
 - Scripted research can be tested and reproduced
 - Programs are a rigorous way of describing data analysis for other researchers, as well as for computers. By sharing codes, which are much more easy for a non-author to understand than spreadsheets

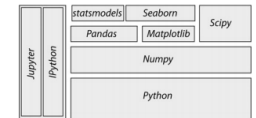
<http://github-pages.ucl.ac.uk/rsd-engineeringcourse/ch00python/00pythons.html>

Popular Python?

- Popular programming languages?
- <https://www.tiobe.com/tiobe-index/>
- What is Python used for?
- <https://www.jetbrains.com/research/devecosystem-2018/python/>

Python ecosystem

- Large and active ecosystem
- Core Python
 - Standard libraries
 - third-party packages:
 - *NumPy* for manipulation of homogeneous array-based data,
 - *Pandas* for manipulation of heterogeneous and labeled data,
 - *SciPy* for common scientific computing tasks,
 - *Matplotlib* for publication-quality visualizations,
 - *IPython* for interactive execution and sharing of code, etc. *Python versions*
- Versions
 - different versions of Python: 2.7 and 3.x.
 - Python 3.x introduced some backwards-incompatible changes to the language, so code written for 2.7 may not work under 3.x and vice versa.
 - Stick with 3.x



User Environment

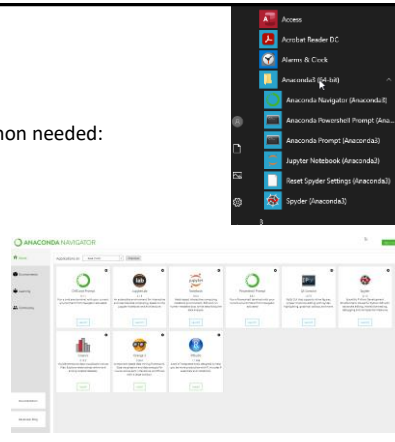
16

- <https://realpython.com/run-python-scripts/>
- <https://plot.ly/python/ipython-vs-python/>
- <https://yihui.name/en/2018/09/notebook-war/>
- <https://www.theatlantic.com/science/archive/2018/04/the-scientific-paper-is-obsolete/556676/>
- <https://fangohr.github.io/blog/installation-of-python-spyder-numpy-sympy-scipy-pytest-matplotlib-via-anaconda.html>

17

Where to start?

- 2 elements for programming in Python needed:
 - writing and editing Python code
 - running that code in an interpreter
- primary ways to run Python code: :
 1. Python interpreter
 2. IPython interpreter
 3. Running scripts
 4. IDE
 - Spyder
 5. Jupyter notebook.

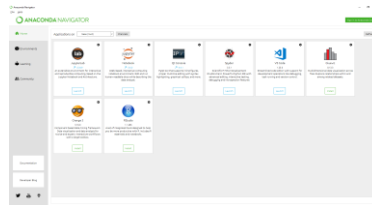


How do I get Python?

- core Python package (<https://www.python.org/downloads/>) is easy to install but *not* what you should choose.
- Using a distribution simplifies the process of setting up your python environment, includes core Python, necessary data packages, and integrates useful tools (IDE's, notebooks, etc)
Python Distributions (in order of preference):
 - Anaconda distribution (<https://www.anaconda.com/>)
 - Installation: <https://docs.anaconda.com/anaconda/install/>
 - Download: <https://www.anaconda.com/distribution/>
 - WinPython - <https://winpython.github.io/>
 - Windows specific data science distribution

Anaconda Navigator

- The Navigator is a main landing page for working with your python environment.
- Launch editors (spyder, jupyter notebook, etc.) to develop python code
- Manage (install packages, etc.) the python environment



21

Anaconda Applications

- Choose which environment (base(root)) to use to launch applications from.
- Click the **Launch** button on any of these applications will launch a separate window.



22

Anaconda Environments



- Clicking on the Environment tab will show what environments are available in Anaconda
 - In the simplest terms, an anaconda environment is a self-contained collection of python packages.
- See which packages are installed and which packages are available for installation.
- <https://www.geeksforgeeks.org/python-virtual-environment/>



How to start?

```
(base) C:\Users\w015831>python
Python 3.7.4 (default, Jul 26 2019, 08:04:48) [MSC v.1912 64 bit (AMD64)] :: Anaconda, Inc. on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> exit()

(base) C:\Users\w015831>ipython
Python 3.7.4 (default, Jul 26 2019, 08:04:48) [MSC v.1912 64 bit (AMD64)]
Type "copyright", "credits" or "license()" for more information
Python 3.7.4 -- An enhanced Interactive Python. Type '?' for help.

In [ ]:
>>>
```

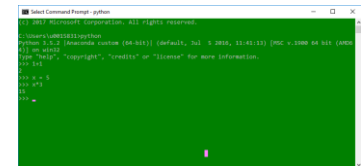
Hello World

- How to run Hello World code?
- `print('Hello World')` in python interpreter
- `python hello_world.py`
- Run in IDE
- `%run hello_world.py`
- Run in Jupyter notebook

25

Python interpreter

- The most basic way to execute Python code is line by line within the Python interpreter.
- The Python interpreter can be started by typing: `python`
 - Terminal on Mac OS X and Unix/Linux systems,
 - (anaconda)Command Prompt application in Windows
 - `>>>` by default

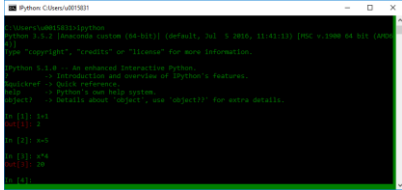


```
cmd - python
Python 3.7.2 [Anaconda Custom (64-bit)] (default, Jul 5 2019, 11:41:12) [AMD64]
>>>
x = 5
5
```

26

IPython interpreter

- Interactive shell
- Enhancements to the basic Python interpreter.
- <https://stackoverflow.com/questions/12370457/what-is-the-difference-between-python-and-ipython>




The screenshot shows a terminal window titled "Python 2.7.6 (IPython 1.1.0)". The prompt is "In [1]:". The output shows the IPython version and a list of features: "Interactive shell", "Enhancements and extensions of Python's features", "Magics", "Aliases", and "Aliases". The prompt is "In [2]:". The output shows the IPython version and a list of features: "Interactive shell", "Enhancements and extensions of Python's features", "Magics", "Aliases", and "Aliases".

- 27



IPython interpreter



- Python interactive interpreter (like running Python), but with some features that make it much easier and more pleasant to use. :
 - Tab autocompletion (on class names, functions, methods, variables)
 - More explicit and colour-highlighted error messages
 - Better history management
 - Basic UNIX shell integration (run simple shell commands such as cp, ls, rm, cp, etc. directly from the IPython command line)
 - Nice integration with many common GUI modules (PyQt, PyGTK, and tkinter)
 - <https://www.quora.com/What-is-the-difference-between-IPython-and-Python-Why-would-I-use-IPython-instead-of-just-writing-and-running-scripts>

- 28



Python scripts

- Programs: save code to file, and execute it all at once.
 - Script: A plain text file containing Python code that is intended to be directly executed by the user
 - By convention, Python scripts are saved in files with a `.py` extension.
- Two modes: normal and interactive.
 - Normal mode: entire `.py` files are provided to the interpreter.
 - Interactive mode: read-eval-print loop (REPL) executes statements piecewise.



29

Run Python script

Linux

- Write script in editor
- Run script using Python interpreter
`python hello_world.py`
- Make script executable
- `chmod u+x hello_world.py`
- Run script directly
`./hello_world.py`

Windows

- Write script in editor
- Run script using Python interpreter
`python hello_world.py`
- Run script directly
`hello_world.py`

30

Python scripts



- Linux

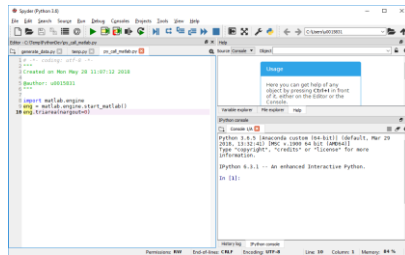
```
#!/usr/bin/env python
```

- determines the script's ability to be executed like a standalone executable without typing `python` in the terminal
- double clicking it in a file manager (when configured properly).

Spyder

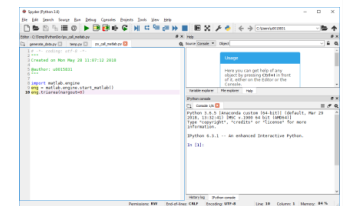
IDE: Spyder

- Integrate different aspects of programming and running code.
- SPyDER: "Scientific Python Development Environment"
<https://www.spyder-ide.org/>
- Several tools in one integrated environment (cfr MATLAB desktop)
 - a code editor
 - IPython interpreter / console
 - variable inspector
 - control icons



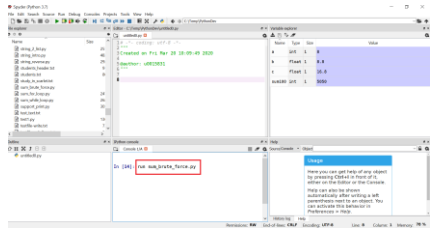
IDE: Spyder

- Spyder for code development.
 - Start from Anaconda Navigator
 - Command window: `spyder`
- Magic commands apply
 - Clear Console:
 - `%clear (ctrl-l)`
 - `%cls`
 - Clear all variables from Variable Explorer (reset the namespace):
`%reset`
 - With automagic on, `%` prefix not needed



Running scripts in Spyder

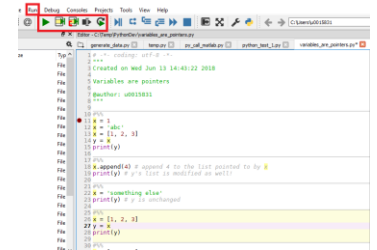
- Run a .py file from the console
 - run script.py
- Tab autocompletion works!



35

Running scripts in Spyder

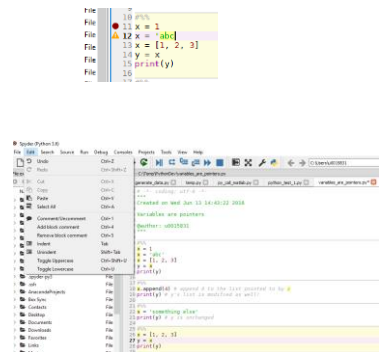
- Run scripts either with the green arrow icons or through the Run menu. *Run/green arrow* runs the entire script.
- *Run selection or current line* will run a highlighted portion of the script.
- Create **cells** by enclosing chunks of code with lines consisting of `###`
Run cell/green arrow with a box runs the cell.
- File: *first_prog_1.py*



36

Running scripts in Spyder

- A yellow triangle beside a line indicates a syntax error or potential problem.
- Tab completion for names familiar to it. It can show a list of members of a package for your selection, and when you have chosen a function it can show you a list of its arguments.
- Ability to indent and unindent blocks of code from the Edit menu. Click to highlight the text you wish to indent or unindent, then choose the appropriate option under Edit.
- "Add block comment." This will save you a lot of typing once you begin to write and debug your own programs. To remove the block comment, select it and choose "Remove block comment" from the Edit menu.



37

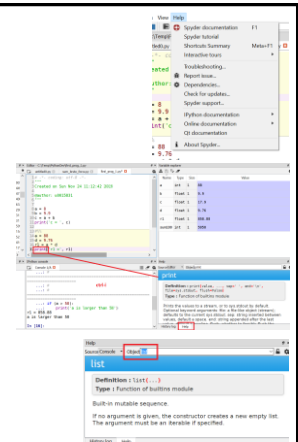
Spyder Help

- Help on Spyder from Help menu
- Help related to Python
 - Select a command and press `ctrl-I`
 - Information opens in help window
 - Enter object in help window
- `help(command)` in console

```
In [18]: help(print)
Help on built-in function print in module builtins:
print(...)
    print(value, ..., sep=' ', end='\n',
          file=sys.stdout, flush=False)

    Prints the values to a stream, or to sys.stdout by
    default.

    Optional keyword arguments:
```



38

Jupyter

getting_started_jupyter.ipynb

40

Jupyter notebook

- A nice idea popularized by Mathematica is a “notebook” interface, where you can run and re-run commands
- In the notebook, you can easily mix code with comments, and mix code with the results of that code; including graphics, ...
- Jupyter notebooks are the favorite environment for data science: data cleaning, data transformation, numerical simulation, machine learning, etc.
- <https://realpython.com/jupyter-notebook-introduction/>
- <https://docs.anaconda.com/ae-notebooks/4.2.2/user-guide/basic-tasks/apps/jupyter/>
- <https://towardsdatascience.com/5-reasons-why-jupyter-notebooks-suck-4dc201e27086>

41

Jupyter notebook

- Excellent for
 - Explorative programming
 - Data exploration
 - Communication, especially across domains
- Problems
 - What was (re-)executed, what not?
 - Version control?
- https://github.com/gibex/training-material/blob/master/Python/python_intro.pptx

42

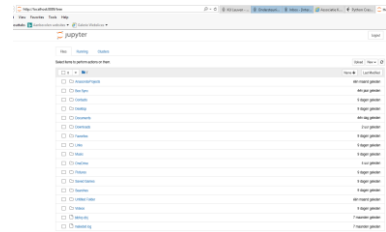
Jupyter: how to start

- Anaconda Navigator:
 - Start menu
 - Launch jupyter
- Anaconda prompt
 - open terminal and navigate to the **directory** where you would like to save your notebook
 - `jupyter notebook`

43

Jupyter

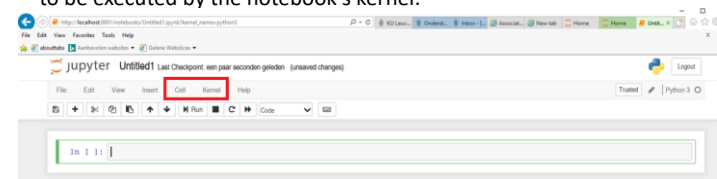
- Notebook Dashboard, specifically designed for managing your Jupyter Notebooks.
- Use it as the launchpad for exploring, editing and creating your notebooks.
- Start the dashboard on any system via the command prompt (or terminal on Unix systems):
`jupyter notebook`
The current working directory will be the start-up directory.



44

Jupyter notebook

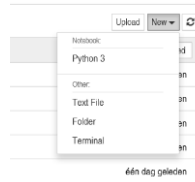
- Jupyter is essentially an advanced word processor.
- A kernel is a "computational engine" that executes the code contained in a notebook document.
- A cell is a container for text to be displayed in the notebook or code to be executed by the notebook's kernel.



45

Jupyter notebook

- Browse to the folder in which you would like to create your first notebook,
- Click the "New" drop-down button in the top-right and
- Select "Python 3" (or the version of your choice).



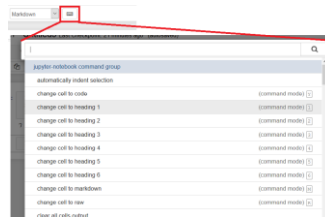
Jupyter: basics of editing

- Jupyter notebook: sequence of cells
 - Code
 - Label "In []" in front of the code
 - a * will appear when executing
 - replaced by a number that always increases by one with each cell execution. This allows for keeping track of the order in which the cells in the notebook have been executed.
 - Markdown
- Important shortcut: ctrl+Enter (execute cell)
- Color code
 - Blue bar on the left: active cell in command mode
 - Click in cell, changes in edit mode – Green bar
- Jupyter will periodically autosave the notebook

Jupyter: basics of editing



- try to know the basic shortcuts
- **Command mode** shortcuts:
 - Basic navigation: enter, **shift-enter**, up/k, down/j
 - Saving the notebook: s
 - Change Cell types: y, m, 1-6, t
 - m to change the current cell to Markdown,
 - y to change it back to code
 - Cell creation: a, b
 - a to insert a new cell above the current cell,
 - b to insert a new cell below
 - Cell editing: x, c, v, d, z
 - c copy selected cells
 - x cut selected cells
 - v paste copied cells
 - d + d (press the key twice) to delete the current cell
 - z undo cell deletion



Jupyter: some tips



- Jupyter notebook tips
 - <https://www.dataquest.io/blog/jupyter-notebook-tips-tricks-shortcuts/>
 - <https://www.dataquest.io/blog/jupyter-notebook-tutorial/>
 - <https://jupyter4edu.github.io/jupyter-edu-book/>
 - <https://reproducible-science-curriculum.github.io/workshop-RR-Jupyter/>
- Change the default startup directory
 - <https://stackoverflow.com/questions/35254852/how-to-change-the-jupyter-startup-folder>
- Change the default browser
 - <https://support.anaconda.com/customer/en/portal/articles/2925919-change-default-browser-in-jupyter-notebook>