Python for Scientific Research

Bram Kuijper

University of Exeter, Penryn Campus, UK

February 11, 2020

UNIVERSITY OF EXETER | DOCTORAL COLLEGE

Researcher Development

Acknowledgements

- This course is funded by UExeter's Institute for Data Science and Artificial Intelligence: IDSAI
- Big thanks to JJ Valletta as he has developed these lectures
- Big thanks to Deepak Kumar Panda for helping out this afternoon

UNIVERSITY OF EXETER | DOCTORAL COLLEGE

Researcher Development

Course Schedule

- Tuesday Feb 4: The basics of programming in Python
  - how to run Python code
  - data types
  - flow control
  - functions and modules

Course Schedule

- Tuesday Feb 4: The basics of programming in Python
  - how to run Python code
  - data types
  - flow control
  - functions and modules
- Tuesday Feb 11: Applying Python to simplify your life
  - text manipulation and regular expressions
  - working with files and streams
  - number crunching with `numpy` and `scipy`

Course Schedule

- Tuesday Feb 4: The basics of programming in Python
  - how to run Python code
  - data types
  - flow control
  - functions and modules
- Tuesday Feb 11: Applying Python to simplify your life
  - text manipulation and regular expressions
  - working with files and streams
  - number crunching with `numpy` and `scipy`
- Tuesday Feb 25: Advanced subjects
  - working with data using `pandas`
  - making graphs using `matplotlib`
  - data visualisation with `seaborn`

Schedule Tue Feb 4

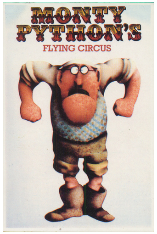
- Morning session, 0900 - 1100 DDM IT 3.037
  - 0900 - 0930: How to run Python
  - 0930 - 1000: Data types
  - 1000 - 1010: Break
  - 1010 - 1100: Data types practical
- Afternoon session, 1300 - 1600 DDM IT 3.037
  - 1300 - 1300: Flow control
  - 1330 - 1400: Flow control practical
  - 1400 - 1410: Break
  - 1410 - 1430: Flow control practical continued
  - 1430 - 1500: Functions
  - 1500 - 1510: Break
  - 1510 - 1600: Functions practical

Some important websites

- Course website: https://exeter-data-analytics.github.io
- Python documentation: https://docs.python.org

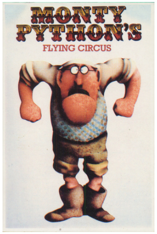
What is Python?

- A scripted, high-level programming language created by Guido Van Rossum and named after Monty Python's flying circus



What is Python?

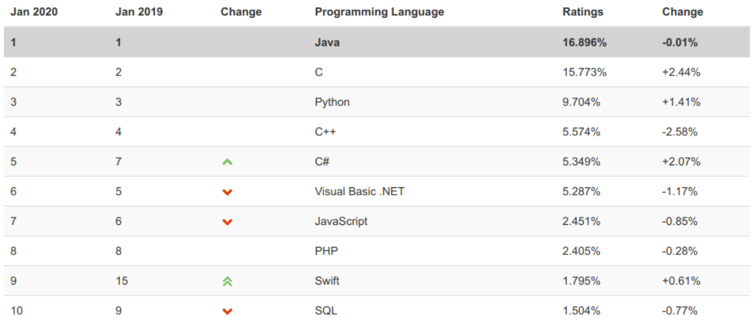
- A scripted, high-level programming language created by Guido Van Rossum and named after Monty Python's flying circus



- easy-to-use, highly standardized and with an emphasis on readability of code

Why use Python?

The TIOBE index is a measure of the popularity of programming languages:



Jan 2020	Jan 2019	Change	Programming Language	Ratings	Change
1	**1**		**Java**	**16.896%**	**-0.01%**
2	2		C	15.773%	+2.44%
3	3		Python	9.704%	+1.41%
4	4		C++	5.574%	-2.58%
5	7	▲	C#	5.349%	+2.07%
6	5	▼	Visual Basic .NET	5.287%	-1.17%
7	6	▼	JavaScript	2.451%	-0.85%
8	8		PHP	2.405%	-0.28%
9	15	▲▲	Swift	1.795%	+0.61%
10	9	▼	SQL	1.504%	-0.77%

Why Python?

- It is free! No licence costs

Why Python?

- It is free! No licence costs
- Runs on many platforms (Mac, Windows, Linux)

Why Python?

- It is free! No licence costs
- Runs on many platforms (Mac, Windows, Linux)
- Because of its ease of programming, Python minimises development effort

Why Python?

- It is free! No licence costs
- Runs on many platforms (Mac, Windows, Linux)
- Because of its ease of programming, Python minimises development effort
- A huge number of libraries, written by an active community

Why Python?

- It is free! No licence costs
- Runs on many platforms (Mac, Windows, Linux)
- Because of its ease of programming, Python minimises development effort
- A huge number of libraries, written by an active community
- Python can “glue” together functions written in C/C++ and Fortran to speed things up (we can also call R and MATLAB functions)

Why Python?

- It is free! No licence costs
- Runs on many platforms (Mac, Windows, Linux)
- Because of its ease of programming, Python minimises development effort
- A huge number of libraries, written by an active community
- Python can “glue” together functions written in C/C++ and Fortran to speed things up (we can also call R and MATLAB functions)
- Compared to other high-level scientific languages such as MATLAB and R, Python offers a much wider range of additional functionality (e.g web and GUI development)

Horses for courses

- Python is becoming the *de facto* standard for exploratory and interactive scientific research

  BUT

Horses for courses

- Python is becoming the *de facto* standard for exploratory and interactive scientific research

  BUT

- Python is no programming silver bullet

Horses for courses

- Python is becoming the *de facto* standard for exploratory and interactive scientific research

  BUT

- Python is no programming silver bullet
- Your application will dictate the tool (and a mixture of more than one language is ok). For example:

Horses for courses

- Python is becoming the *de facto* standard for exploratory and interactive scientific research

  BUT

- Python is no programming silver bullet
- Your application will dictate the tool (and a mixture of more than one language is ok). For example:
  - MATLAB excels at interfacing with hardware, e.g generating hardware description language (HDL) code to configure an integrated circuit board or connecting to a data acquisition card

Horses for courses

- Python is becoming the *de facto* standard for exploratory and interactive scientific research

  BUT

- Python is no programming silver bullet

- Your application will dictate the tool (and a mixture of more than one language is ok). For example:
  - MATLAB excels at interfacing with hardware, e.g generating hardware description language (HDL) code to configure an integrated circuit board or connecting to a data acquisition card

  - R is great for data wrangling and visualisation, and statistical modelling

Horses for courses

- Python is becoming the *de facto* standard for exploratory and interactive scientific research

  BUT

- Python is no programming silver bullet
- Your application will dictate the tool (and a mixture of more than one language is ok). For example:
  - MATLAB excels at interfacing with hardware, e.g generating hardware description language (HDL) code to configure an integrated circuit board or connecting to a data acquisition card
  - R is great for data wrangling and visualisation, and statistical modelling
  - C achieves the fastest runtimes, at the expense of a long development time

Why do *you* want to learn Python?

Some reasons:

Why do *you* want to learn Python?

Some reasons:

- Python has a simple syntax and is widely used; ideal language for beginners

Why do *you* want to learn Python?

Some reasons:

- Python has a simple syntax and is widely used; ideal language for beginners
- Development time is much quicker than for compiled languages like C or Java

Why do *you* want to learn Python?

Some reasons:

- Python has a simple syntax and is widely used; ideal language for beginners
- Development time is much quicker than for compiled languages like C or Java
- Broad uptake: Python (and R) have replaced Perl as the key programming language in Bioinformatics

Why do *you* want to learn Python?

Some reasons:

- Python has a simple syntax and is widely used; ideal language for beginners
- Development time is much quicker than for compiled languages like C or Java
- Broad uptake: Python (and R) have replaced Perl as the key programming language in Bioinformatics
- major GIS applications like ArcGIS use Python as their main scripting language
- Language of choice for machine learning (PyTorch, scikit, TensorFlow)

Why do *you* want to learn Python?

Some reasons:

- Python has a simple syntax and is widely used; ideal language for beginners
- Development time is much quicker than for compiled languages like C or Java
- Broad uptake: Python (and R) have replaced Perl as the key programming language in Bioinformatics
- major GIS applications like ArcGIS use Python as their main scripting language
- Language of choice for machine learning (PyTorch, scikit, TensorFlow)

Python version 2 vs 3

- Many systems (e.g., Mac OS X) still use Python 2 as the default
- Python 3 differs in various ways from Python 2
- Often, Python 3 code cannot be run using a Python 2 interpreter and vice versa
- Python 2 is a legacy version and will ultimately be replaced by Python 3
- **Current course will focus on Python 3**

Different Python distributions

Various distributions of Python available:

Different Python distributions

Various distributions of Python available:

- Official version from python.org
  - Caveat: libraries and other tools need to be installed separately via `pip` (Python's package manager)

Different Python distributions

Various distributions of Python available:

- Official version from python.org
  - Caveat: libraries and other tools need to be installed separately via `pip` (Python's package manager)
- Enthought Canopy: Python, libraries & tools in single installer

Different Python distributions

Various distributions of Python available:

- Official version from python.org
  - Caveat: libraries and other tools need to be installed separately via `pip` (Python's package manager)
- Enthought Canopy: Python, libraries & tools in single installer
- Anaconda: Python, libraries & tools in single installer: **used in this course**

Different Python distributions

Various distributions of Python available:

- Official version from python.org
  - Caveat: libraries and other tools need to be installed separately via `pip` (Python's package manager)
- Enthought Canopy: Python, libraries & tools in single installer
- Anaconda: Python, libraries & tools in single installer: **used in this course**

Executing Python code: Spyder IDE

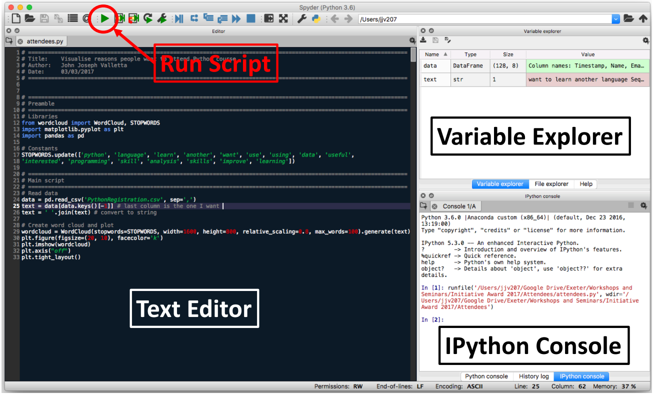
- Windows: Start Menu > Anaconda3 > Spyder

Executing Python code: Spyder IDE

- Windows: Start Menu > Anaconda3 > Spyder
- Mac: Applications > Spyder

Executing Python code: Spyder IDE

- Spyder is an integrated development environment (IDE) for scientific computing, akin to RStudio and MATLAB
- One place to write, execute and debug code, and explore variables



```python
# ==========================================================================
# Title:    Visualise reasons people want to attend Python course
# Author:   John Joseph Valletta
# Date:     03/03/2017
# ==========================================================================


# ==========================================================================
# Preamble
# ==========================================================================
# Libraries
from wordcloud import WordCloud, STOPWORDS
import matplotlib.pyplot as plt
import pandas as pd

# Constants
STOPWORDS.update(['python', 'language', 'learn', 'another', 'want', 'use', 'using', 'data', 'useful',
'interested', 'programming', 'skill', 'analysis', 'skills', 'improve', 'learning'])

# ==========================================================================
# Main script
# ==========================================================================
# Read data
data = pd.read_csv('PythonRegistration.csv', sep=',')
text = data[data.keys()[-1]] # last column is the one I want
text = ' '.join(text) # convert to string

# Create word cloud and plot
wordcloud = WordCloud(stopwords=STOPWORDS, width=1600, height=800, relative_scaling=0.8, max_words=100).generate(text)
plt.figure(figsize=(20, 10), facecolor='k')
plt.imshow(wordcloud)
plt.axis("off")
plt.tight_layout()
```

Name ▲	Type	Size	Value
data	DataFrame	(128, 8)	Column names: Timestamp, Name, Ema…
text	str	1	want to learn another language Seq…

```
Python 3.6.0 |Anaconda custom (x86_64)| (default, Dec 23 2016,
13:19:00)
Type "copyright", "credits" or "license" for more information.

IPython 5.3.0 -- An enhanced Interactive Python.
?         -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help      -> Python's own help system.
object?   -> Details about 'object', use 'object??' for extra
details.

In [1]: runfile('/Users/jjv207/Google Drive/Exeter/Workshops and
Seminars/Initiative Award 2017/Attendees/attendees.py', wdir='/
Users/jjv207/Google Drive/Exeter/Workshops and Seminars/Initiative
Award 2017/Attendees')

In [2]:
```

Permissions: RW   End-of-lines: LF   Encoding: ASCII   Line: 25   Column: 62   Memory: 37 %

Running standalone Python scripts without the IDE - I

Being able to run Python scripts without the Spyder IDE is particularly important when running scripts on computing clusters.

- Windows: don’t bother, use the Spyder IDE

Running standalone Python scripts without the IDE - I

Being able to run Python scripts without the Spyder IDE is particularly important when running scripts on computing clusters.

- Windows: don't bother, use the Spyder IDE
- Mac/Linux:
  - Write your code in a plain text file, say `my_script.py`

Running standalone Python scripts without the IDE - I

Being able to run Python scripts without the Spyder IDE is particularly important when running scripts on computing clusters.

- Windows: don't bother, use the Spyder IDE
- Mac/Linux:
  - Write your code in a plain text file, say `my_script.py`
  - In a terminal, run:

```
python3 my_script.py
```

Running standalone Python scripts without the IDE - II

By adding a so-called ‘shebang’ to the top of a Python script, one can run scripts as a standalone programme on Mac / Linux

Running standalone Python scripts without the IDE - II

By adding a so-called ‘shebang’ to the top of a Python script, one can run scripts as a standalone programme on Mac / Linux

- Add a shebang `#!/usr/bin/env python3`, to the first line of the python script file, here called `my_script.py`:

```
#!/usr/bin/env python3

print("This Python script prints something.")
...
```

Running standalone Python scripts without the IDE - II

By adding a so-called ‘shebang’ to the top of a Python script, one can run scripts as a standalone programme on Mac / Linux

- Add a shebang `#!/usr/bin/env python3`, to the first line of the python script file, here called `my_script.py`:

```
#!/usr/bin/env python3

print("This Python script prints something.")
...
```

- Close the file and make it executable by typing in a terminal

```
chmod +x my_script.py
```

Running standalone Python scripts without the IDE - II

By adding a so-called ‘shebang’ to the top of a Python script, one can run scripts as a standalone programme on Mac / Linux

- Add a shebang `#!/usr/bin/env python3`, to the first line of the python script file, here called `my_script.py`:

```
#!/usr/bin/env python3

print("This Python script prints something.")
...
```

- Close the file and make it executable by typing in a terminal

```
chmod +x my_script.py
```

- Run the script by typing in a terminal

```
./my_script.py
This Python script prints something.
```