Intro to Python

Markus Löning
PhD student
https://github.com/mloning
markus.loning.17@ucl.ac.uk

sktime – a Python package for machine learning with time series
https://github.com/alan-turing-institute/sktime

Technical Tuesdays

Introduction
15 Oct

R Scripting
29 Oct

JavaScript
19 Nov

Version Control
03 Dec

*nix Shell
22 Oct

Python
12 Nov

Databases
26 Nov

Mapping
10 Dec

Road map

1. Why Python?
2. How to type and run code?
3. Where to learn more?
4. Demo

Why Python?

- Readability
- Free and open-source, very popular
- Very rich scientific computing ecosystem
- Object oriented
- Powerful development environments
- Many libraries beyond scientific computing (general-purpose language)

In [1]:
```
import this
```

```
The Zen of Python, by Tim Peters

Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one-- and preferably only one --obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea -- let's do more of those!
```

Readability

- “Python is executable pseudocode”
- Write fast code fast
- Usually only one obvious way to do things

Popularity

	Python	R
TIOBE search engine results	3	14
Redmonk Github PRs	3	15
IEEE Spectrum multiple sources	1	5

Ecosystem

NumPy

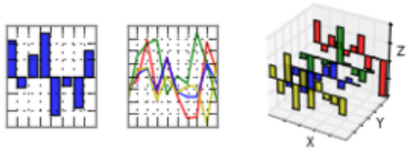
SciPy

matplotlib

[ ]: IPython
Interactive Computing

pandas

$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$



scikit learn

IDEs

Where to type and run code?

- PyCharm (professional license for students)
- Visual Studio Code
- Jupyter Notebook\Lab (interactive)
- Spyder (like RStudio)
- Atom
- Sublime Text

Resources

- **Scientific computing**: https://scipy-lectures.org
- **Geo-data science**: https://geo-python.github.io/site/
- **Research software engineering**: https://github.com/alan-turing-institute/rsd-engineeringcourse
- Numerous online books & tutorials (datacamp.com, LinkedIn Learning, etc.)

Community

- ~1.2m stackoverflow.com questions
- PyData London (annual conference, meet-ups, etc)
- Scipy community (conferences, tutorials, funding)

How to get started?

Anaconda distribution

- Easy to install
- Good online documentation and guides
- Core scientific computing libraries
- Manager for packages and virtual environment (conda)
- Anaconda Navigator (GUI)

Demo