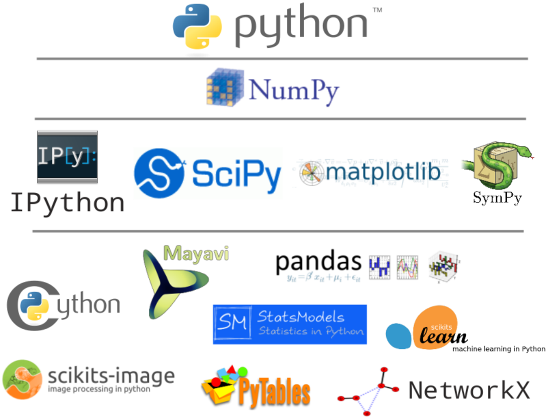
Scientific Programming: Analytics Tools and Visualisation

Scientific Programming with Python

Christian Elsasser

Based partially on a talk by Stéfan van der Walt

The Ecosystem of Homo Python Scientificus



[Ondřej Čertík/LANL]

Table of Contents

- Linear Algebra with Numpy
- Scipy
  - Basic Structure
  - Three Examples
- Interlude: Nice tools
  - datetime
  - requests & BeautifulSoup
- Visualisation
  - matplotlib
  - seaborn
  - bokeh
  - folium

A Few Technical Remarks

If you want to follow directly the code used in the lecture

- Download the code from the course homepage (Lecture 7)
- Start the virtual environment
  `$ .  venv/bin/activate` (from the home directory)
- Create a kernel for the notebook with the virtual environment
  ```
  $ python3 -m ipykernel install --user --name=ve3
  ```
- Unzip the file
  ```
  $ tar zxvf material_analytics_vis_lec.tar.gz
  ```
- Enter the created directory
  ```
  $ cd material_analytics_vis_lec
  ```
- . . . and start the notebook
  ```
  $ ipython3 notebook
  ```

Fundamental Tools – SciPy & NumPy

More than Arrays – NumPy and Matrices

NumPy offers a matrix framework for linear algebra calculations, allowing to defining one- and two-dimensional arrays as matrices

Matrices

```
»»» a = np.matrix([[1,2],[3,4]])
»»» b = np.matrix(np.random.rand(4))
»»» c = np.matrix(np.random.rand(3,3))
```

One-dimensional arrays → $1 \times n$ matrices, *i.e.* row vectors

Matrices have some additional functionality (*e.g.* inverse: `a.I`, hermitian: `a.H`)

Linear Algebra with SciPy – Bringing High-Performance Libraries to the Table

Light version of SciPy's linear algebra implementation at `np.linalg`

Examples of available functionality:

`np.linalg.cholesky`
`np.linalg.eigh`
`np.linalg.det`
`np.linalg.qr`
`np.linalg.eig`
`np.linalg.svd`

The functions are wrappers of the LAPACK linear algebra package

More functionality is embedded in the full SciPy implementation `scipy.linalg`, *e.g.*

Matrix Exponential

```
>>> a = np.matrix([[1,2],[3,4]])
>>> scipy.linalg.expm(a)
```

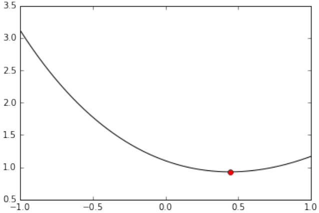
SciPy – or Where the Fun Really Starts

- Offering a large number of functionality for numerical computation
  - `scipy.linalg` → Linear Algebra
  - `scipy.optimize` → Numerical optimisation (incl. least square)
  - `scipy.integrate` → Numerical integration
  - `scipy.stats` → Statistics including a large set of distributions
  - more at http://docs.scipy.org/doc/scipy/reference/
- Eco-system of more advanced packages for data analysis, *e.g.*
  - scikits.learn: Machine-learning algorithms
  - scikits.image: Image processing
  - pytables: data structure (based on HDF5)
  - ...

Remark: `import scipy as sp` only imports the most basic tools ⇒ `from scipy import stats`

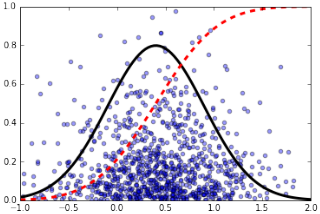
Three SciPy examples: Optimisation, Distributions and Fast-Fourier Transform

Find the minimum



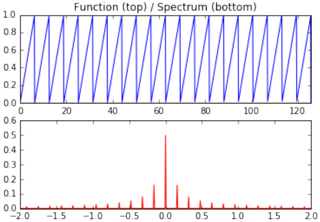
- Also for n-dim functions
- Basic functionality for least-square or maximum-likelihood estimation

Sample distributions



- Large variety of distributions
- Be careful with the order of parameters

Get the spectrum



- Fast frequency analysis
- Deals with the full spectrum (complex frequency values)

Time & Date

datetime – Easy Handling of Time

https://docs.python.org/3.4/library/datetime.html

- Collection of classes to manipulate date and time
- Most important class `datetime` to represent date (year, month, day) and time (hour, minute, second, millisecond)
- `strptime` and `strftime` to load and dump dates from and to a string, respectively → format defined via standard time fields (*i.e.* %Y for four-digit year, %b for three-letter month abbreviation, etc. using locale information)
- Timezone info encodable via abstract base class of `tzinfo`, *e.g.* `pytz` ⇒ No excuse for unannotated timestamps
- `timedelta` as difference between `datetime` objects allowing to make calculations

Web Tools

`requests` / `urllib` – The Web at Your Fingertip

http://docs.python-requests.org/en/master/
https://docs.python.org/3.4/library/urllib.html

requests

- User-friendly module for HTTP functionality
- POST and GET (and the others) functionality (→ extraction of web site content, download of files, low-level handling of APIs, etc.)
- Possiblity to specify sessions (`requests.Session`)
- Submission of additional parameters to specifiy proxy, authentification, etc.

urllib

- For some functionalities we need to fall back to `urllib`
  - Download files easily
  - Retrieve data from files iteratively

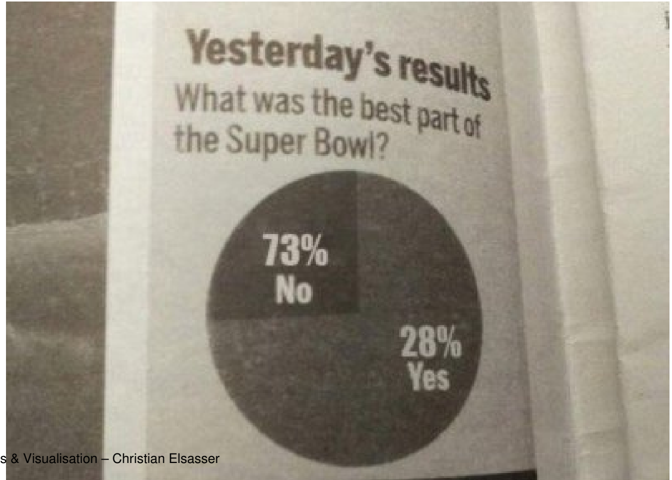
`BeautifulSoup` – Navigating through HTML and XML trees

https://www.crummy.com/software/BeautifulSoup/bs4/doc/

- Parsing of HTML or XML files into a tree structure
- Selection of sections based on tags including their attributes (class, id, name, etc.) possible
- Also extraction of attributes possible (*e.g.* href field for HTML links)
- `parent`, `children`, `siblings` methods allow to navigate in the structure of the document

Visualisation

Visualisation as well as Content Matters



Yesterday's results

What was the best part of the Super Bowl?

Visualisation Options in Python

Matplotlib

- Started as emulation for MATLAB
- Basic plotting also in more than one dimension

Seaborn

- Collection of more complex plots
- Based on Matplotlib

bokeh

- Web publishable graphics
- Large variety of usable interactions

Folium

- Python interface to leaflet (maps)
- Plotting of geo data

Advanced Python Modules

We omitted any modules with a large and specific purpose → otherwise you would sit here tomorrow

Left to the interested audience to explore them further

- NLTK (www.nltk.org) → Natural language processing
- scikit-learn (scikit-learn.org) → Machine learning
- scikit-image (scikit-image.org) → Image processing and analysis
- . . .

Rapidly growing and improving landscape of python modules, but with still some “whitish” spots (*e.g.* time series) ⇒ Reflection of available alternatives?

Conclusion

- Large variety of modules (growing every day), not just data analysis, but also for web interface, etc.
- Many packages targeting APIs
  - Twitter → `tweepy`
  - Yandex translator → `yandex.translate`
  - Quandl → `quandl`

  ⇒ Do not reinvent the wheel!
- `pip` is your friend and helper
- Learning by doing!
- . . . But knowing what functionalities are available and their potential is half the battle!