Universität Bremen

LAMOS

Introduction to Python

PIP Summer School on machine learning

Dr. Andreas Hilboll

24 September 2018

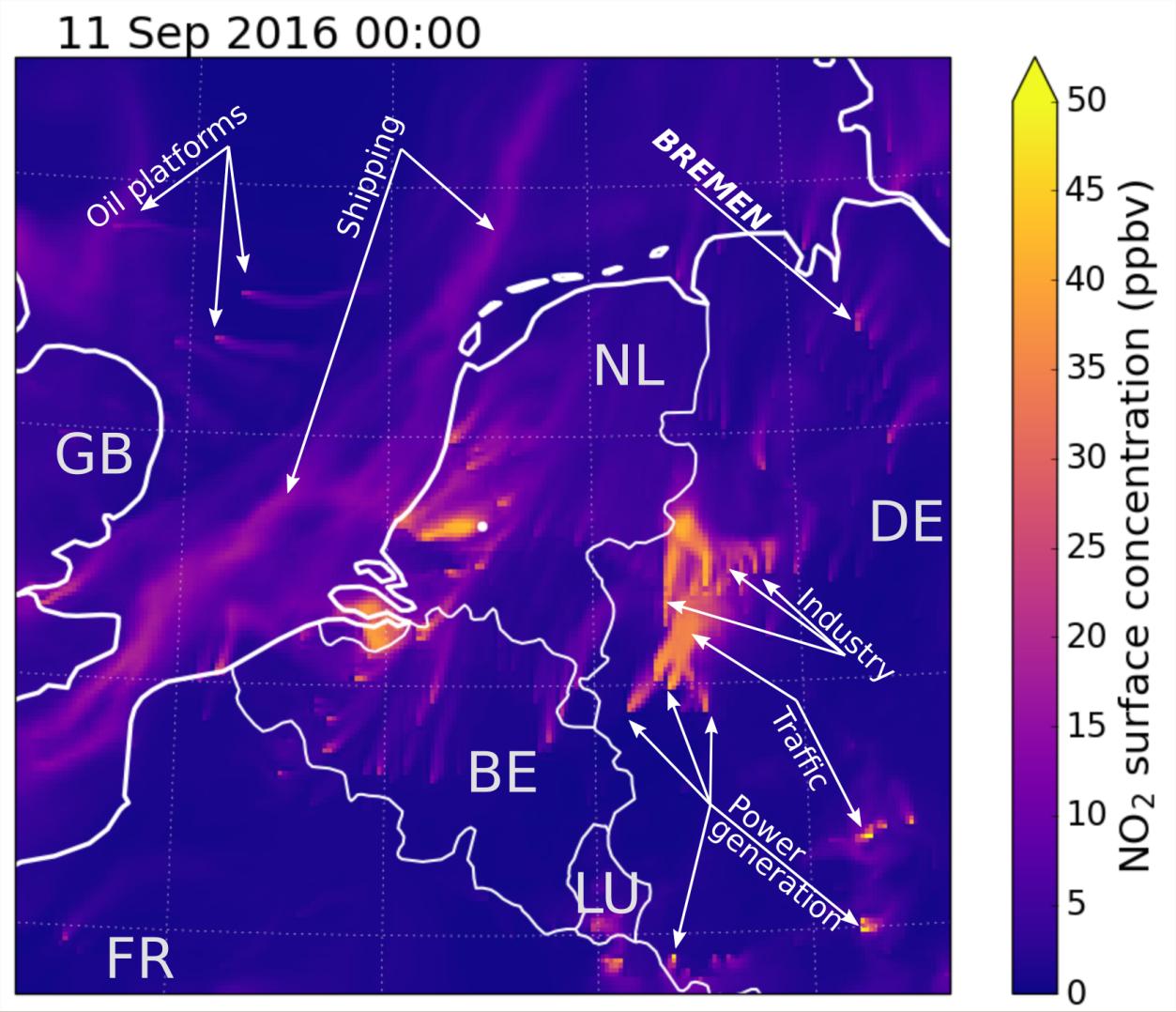
Who am I?

Who am I?

- Andreas Hilboll
- PostDoc at *Institute of Environmental Physics (IUP)*
  - atmospheric composition modeling (LAMOS group)
  - previously:
    - PhD in satellite remote sensing (DOAS)
    - M.Sc. in Mathematics
- hilboll@uni-bremen.de

What am I working on?

- Numerical simulation of atmospheric composition
  - i.e., air quality modeling



Who are you?

Show of hands ...

- Who knows what Python is?
- Who would recommend Python code by eye?
- Who has already written at least 1 line of Python code?
- Who is regularly working with Python?

Your Python setup

- Who has tried to follow the installation instructions?
- Who was successful?

An introduction to Python in 3 hours

Why Python?

- Free
- Good for beginners:
  - High-level language, no need to dive into internals
  - Interactive use, no compilation needed
  - Clear code (only few `( )`, `[ ]`, `{ }`, `;`, ...)
- Large ecosystem with modules for about everything
  - `https://pypi.io/`
- Strong adoption in science
  - `https://www.scipy.org/about.html`
- Strong adoption in data science / machine learning
  - `https://sklearn.org/`

General information about Python

Some free books about Python:

- The Hitchhiker's Guide to Python
- Think Python
- Dive into Python 3

This is not a general introduction to programming

Goals for today

1. Assumption: Python is working on all of your computers
    - I didn't get any questions regarding the installation...
2. Python basics
3. Work with the *Jupyter Notebook*
4. Work with data

Getting started

Setting up a Python environment on your computer

There are several different Python *distributions*. Here, I will describe how to use the Python 3 version of the Anaconda Python Distribution, which is very well documented. Additional help can be found here.

Also, I recommend to use conda environments within Anaconda. That way, the packages we use during the summer school won't impact on the rest of your system.

Usually, the installation of Anaconda Python is pretty straightforward, but can sometimes be tricky depending on your computer environment.

The file `environment.yml` contains the package specification for the conda environment. By using this file to setup your working environment, it is ensured that all participants are working with the same set of package versions.

environment.yml

These are the contents of the file `environment.yml` which describes the packages to be installed (I'm using three columns to save space!):

```
name: pip2018ml

channels:
- defaults

dependencies:
- python>=3.6
- pip
- mkl
- numpy
- scipy
- matplotlib
- cython
- pyqt>=5.9
- vtk>=8
- pandas
- scikit-learn
- h5py
- pillow
- statsmodels
- jupyter
- pytest
- pytest-cov
- joblib
- psutil
- numpydoc
- flake8
- spyder
- numexpr
- traits>=4.6.0
- pyface>=6
- traitsui>=6
- jupyter
- notebook
- nb_conda
- pip:
  - mne
  - mayavi
  - PySurfer
  - dipy
  - nitime
  - nibabel
  - nilearn
  - neo
  - pytest-sugar
  - pytest-faulthandler
  - pydocstyle
  - codespell
  - python-picard
```

Using the command line (1)

You can create a new environment from the *Anaconda Navigator*, or on the command line (i.e., *terminal*). Windows users can access the terminal by launching the *Anaconda Prompt* app from their start menu.

The command to create the environment is

```
conda env create --file environment.yml
```

This will create a new environment called `pip2018ml`, with the most recent *Python3* release. Also, a number of important packages for scientific computing will be installed into that environment.

Using the command line (2)

You might get some intermittent error messages, which you can probably ignore. At the end of a successful installation process, you should see this output:

```
#
# To activate this environment, use
#
#     $ conda activate pip2018ml
#
# To deactivate an active environment, use
#
#     $ conda deactivate
```

Using the command line (3): Activating the environment

When working on the command line, you can activate your new environment:

```
conda activate pip2018ml
```

You should now see (`pip2018ml`) at the left of your command prompt. Whenever this (`pip2018ml`) is displayed on the left of your prompt, the commands you enter will be executed in this environment.

Using the Anaconda Navigator

The Anaconda Navigator is a desktop graphical user interface to your Anaconda installation.

After installation of the Anaconda distrbibution, there should be an entry *Anaconda Navigator* entry in the start menu of your computer. Click it to launch the application. You can follow the documentation at the Anaconda Navigator website to perform the following tasks:

1. On the left-hand side, click on *Environments*
2. On the bottom of the second column, click *Import*
3. In the *Specification File* selector, choose the `environment.yml` file
4. Click *Import*

Python basics

What's special about Python?

- There are two different Python versions: Python 2 and Python3:
  - use Python 3!
- Whitespace is significant!
- comprehensive standard library, batteries included
  - "read" `https://docs.python.org/3/`
- No variable declaration: Just "use" a new variable:

```
newvar = 123.456
```

- No overloading: There is *only one instance* of a function definition

In many cases, you don't need indices

- loops iterate over all items in a list, one after the other:

```
for item in alist:
    do_stuff(item)
```

- use `zip(list1, list2)` to iterate over several lists simultaneously:

```
for item1, item2 in zip(list1, list2):
    do_stuff(item1, item2)
```

- when you *do* need an index, use `enumerate(alist)`:

```
for ix, item in enumerate(alist):
    do_stuff(ix, item)
```

List comprehension: one-line loops

Consider this code:

```
newlist = []
for item in oldlist:
    newlist.append(do_stuff(item))
```

This can be done within one single line:

```
newlist = [do_stuff(item) for item in oldlist]
```

The same works for dictionaries:

```
newdict = {key: value for key, value
   in zip(keylist, valuelist)}
```

Working with Python

There are different ways to *"work with Python"*

1. Write code with an *editor*, execute code on the *console*
    - Example: Vim, Emacs, Notepad, ...
2. Work interactively on the (I)Python *shell*
    - Example: IPython
3. Use an integrated IDE-like environment
    - Example: Spyder
4. Use a web-based, interactive environment
    - Example: Jupyter Notebook, Jupyter Lab

Here we will use the Jupyter Notebook

Starting the Jupyter Notebook

In order to start the Jupyter Notebook, you have two options:

1. On the command line, *activate* the environment (see above) and then execute the command `jupyter notebook`
2. From the *Anaconda Navigator*,
   - click on *Home*
   - make sure that the correct environment is selected (towards the top of the main screen, *Applications on*)
   - click the *Launch* button in the *Jupyter Notebook* tile

In either case, the Jupyter Notebook interface will launch in your default web browser (it might be in a different window or on a different screen, you'll have to find it).

Using the Jupyter Notebook

Numerical data: the NumPy `array`

Learning material

This lecture is based on a course from the Software Carpentry project. It contains more material on Python basics for novices:

1. NumPy: arrays and data
2. Loops: `for ...`
3. Lists: `[a, b, c]`
4. Conditionals: `if ...  else ...`
5. Functions: `def myfunc(arg1, arg2) ...`
6. Errors

Live coding

Now let’s hop over to the Notebook …