Python for data analysis in geosciences

General introduction

ATTENTION !

Signez la fiche de présence

Est-ce que tout le monde a les TPs et les données?

Today’s objectives

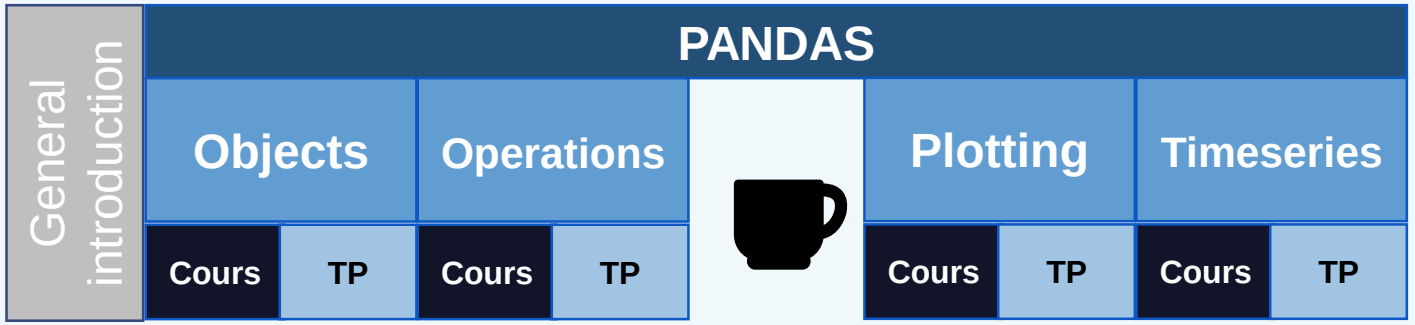
Main uses of pandas and xarray

Analyse geoscientific data in an efficient, fast and easily sharable way using python

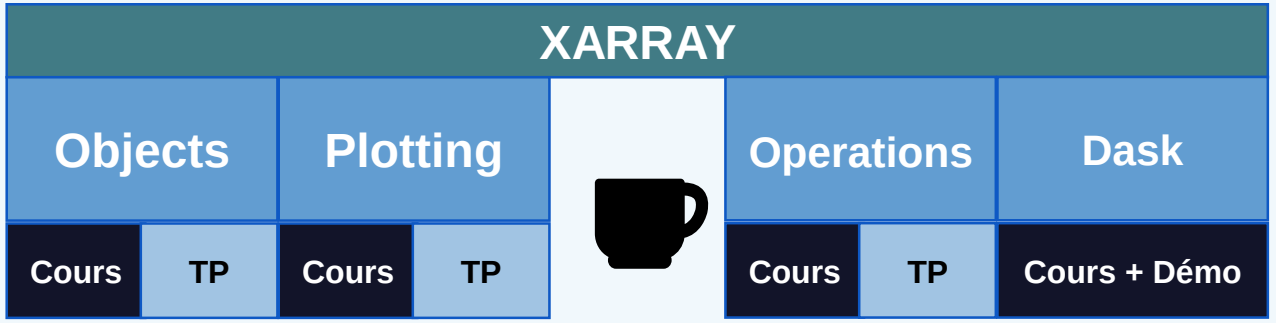
Scaling potential for huge datasets

Programme

Matin: pandas



Arpès-midi: xarray



Back to basic: numpy arrays

- Multi-dimensionnal arrays

```
array([27, 90, 55, ..., 97, 76, 87])
```

```
array([[ 5, 89, 40, ..., 78, 99, 70],
       [ 2, 43, 13, ..., 91, 73, 34],
       [27,  9, 91, ..., 68, 84, 62],
       ...,
       [44, 55, 63, ..., 65, 59, 37],
       [20,  8, 14, ..., 94, 49, 16],
       [45, 37, 89, ..., 53, 65, 84]])
```

- Vectorized operations
  - `array1 + array2`

- Statistical operations
  - `np.mean(array)`

Back to basic: numpy arrays

```
array([[[[29.569597, 29.58791 , 29.60622 , ..., 29.334484, 29.339611,
          29.349865],
         [29.554949, 29.57326 , 29.591572, ..., 29.32496 , 29.327158,
          29.33302 ],
         [29.536636, 29.554216, 29.573992, ..., 29.30958 , 29.311045,
          29.31544 ],
         ...,
```

- Access to data not really intuitive !

```
data.shape
```

```
(312, 5, 50, 50)
```

```
results = data[52, : , 10:12, 3]
results
```

```
array([[29.927763, 29.93289 ],
       [29.928495, 29.934355],
       [29.92996 , 29.937284],
       [29.931425, 29.940947],
       [29.932158, 29.943876]], dtype=float32)
```

Share functionnality 1: indexing

- Select data using **indices**
- Access variable/dimensions by their explicit **names**

Share functionnality 2: many formats for reading/writing

Tabular format

- csv
- excel (xls, xlsx)
- txt
- json
- …

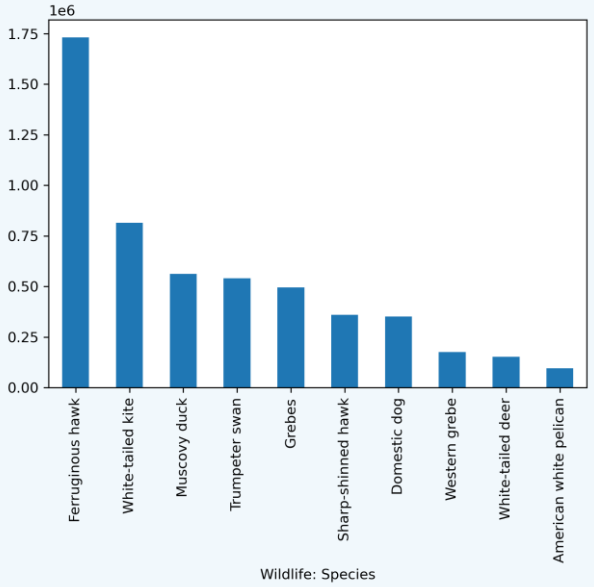
Grid format

- Netcdf
- GRIB
- geoTIFF
- zarr
- …

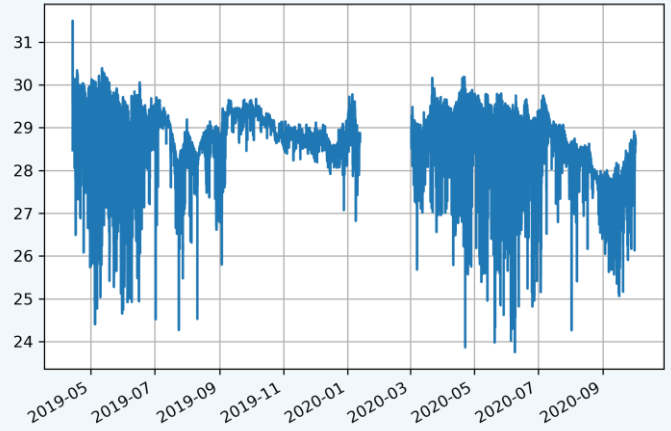
Share functionnality 3: easy plotting

- Integrated with **matplotlib** (and many other backends like hvplot, plotly, …)

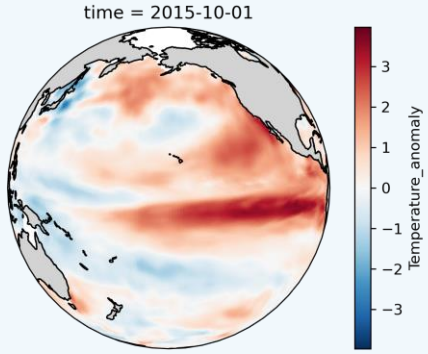
Categorical plots

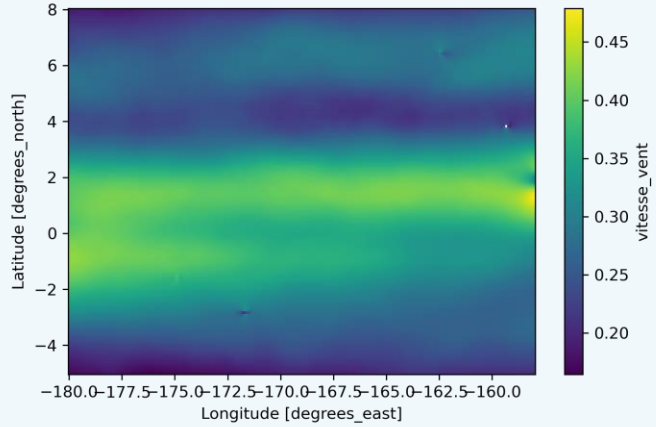


Time series



Maps





Some differences

pandas

- Better for **one dimension** data
- Different data types possible (string, datetime, int, float, …)
- **More methods** available

- Great for categorical data/timeseries

- Generalising pandas with **several dimensions**
- Better for **geographical data** with 3-4 dimensions

- Gridded data climate/ocean/atmosphere/satellite

Useful links

- https://pandas.pydata.org/
- https://docs.xarray.dev/en/stable/
- https://gallery.pangeo.io/
- https://www.dask.org/

Any question?

Quick introduction to jupyter notebooks

- Cells
- Code
- Markdown
- Outputs