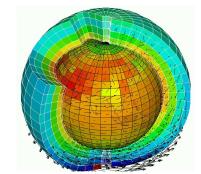


MC-Toolkit meetings

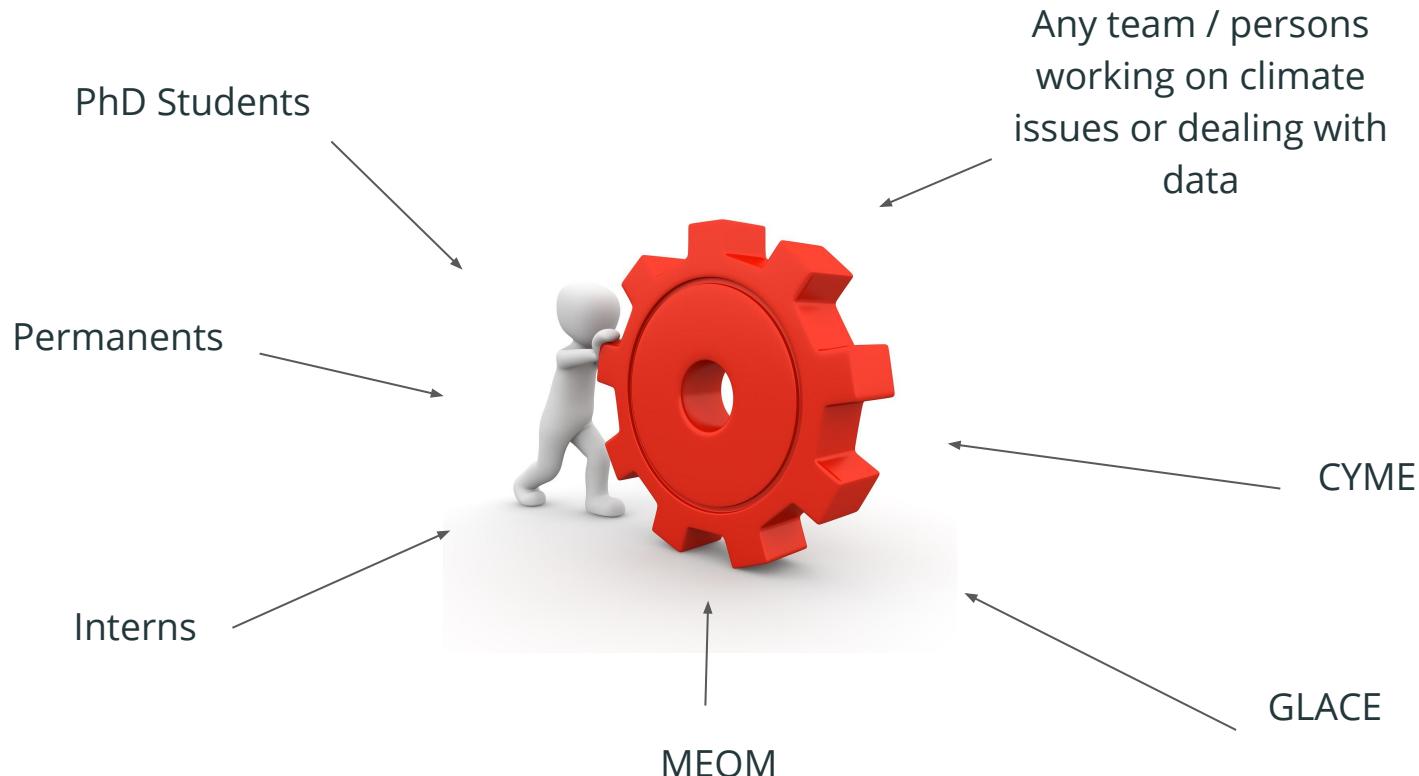
Mickaël Lalande



Thèse 2019-2022
Directeurs : Gerhard Krinner et Martin Ménégoz
Institut des Géosciences de l'Environnement (IGE)



Why this meeting?



My personal experience in MEOM's team -> SATIM meetings

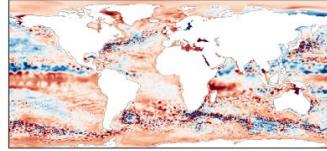


[brodeau/sosie: SOSIE is Only a Surface Interpolation Environment](#)

Screenshot of a GitHub repository page for `brodeau/sosie`. The repository title is "SOSIE is Only a Surface Interpolation Environment". The repository has 15 commits, 1 branch, 0 packages, 0 releases, and 1 contributor. The latest commit was made on Nov 2019. The repository contains files like `dist`, `powerspec.egg-info`, `powerspec`, `README.md`, and `setup.py`. The `README.md` file describes the `powerspec` package, which is a Python package for estimating wavenumber spectral density, kinetic energy spectral flux and spectral coherence of two-dimensional oceanic dataset such as SSH, vorticity. Installation instructions mention `pip install powerspec`.

[adeajayi-kunle/powerspec: Compute wavenumber spectrum for 2D field.](#)

M2 RESEARCH INTERNSHIP REPORT
Identification and filtering of oceanic chaos by Machine Learning



Mickaël LALANDE
Master 2 in Earth, Planetary and Environmental Sciences
Atmosphere-Climate-Continental Landmass Programme
UFR PhTTEM - Grenoble Alpes University

Supervisors: Thierry PENDUFF and Redouane LGUENSAT
Institute of Environmental Geosciences (IGE), Grenoble, France
MEOM Research Group
04/02/2019 - 28/06/2019 (5 months)

Ideas of topics

In the context of Mardi Café:

- xarray / cdo / climaf (netcdf, climatologies)
- cartopy / ferret / ncl / basemap / proplot (plot)
- cdo, basemap, scipy/stats, ESMx, climaf, sosie (regrid)
- Jupiter Notebook / Jupyter lab (pont ssh, Jupiter lab, plots interactifs)
- Github / Gitlab / svn / bitbucket
- dask (parallelization or MemoryError issues)
- Machine Learning ? (gpu, slack, gricad) -> ml-at-ige slack
- régression linéaire / tendances / analyses statistiques
- EOF
- analyses spectrales
- cmip6 data / reanalyses / obs
- MAR / LMDZ (how to install and launch a simulation?)

More general:

- How to write a paper (latex, overleaf, texmaker, etc.)
- How to deal with the bibliography (mendeley, zotero, etc.)
- How to publish an article (what journals, etc.)

Example of CLIMAF

[https://github.com/mickaellalande/MC-Toolkit/blob/master/Presentation/CLIMAF example.ipynb](https://github.com/mickaellalande/MC-Toolkit/blob/master/Presentation/CLIMAF%20example.ipynb)

The screenshot shows a Jupyter Notebook interface with the title "jupyter CLIMAF_example (auto-sauvegardé)". The menu bar includes Fichier, Édition, Affichage, Insérer, Cellule, Noyau, Widgets, Aide, Non flable, and Python 3. The toolbar includes icons for file operations like Open, Save, and Run, along with Execute and Cell.

The notebook content is titled "CLIMAF example (works only on CICLAD)" and contains the following code:

```
'LC debug :', False)
CliMAF install => /ciclad-home/jserver/Evaluation/CliMAF/climaf_installs/climaf_1.2.12
python => /prodigfs/ipslfs/dods/jserver/miniconda/envs/analyse_env_2.7/bin/python
...
Required softwares to run CliMAF => you are using the following versions/installations:
CliMAF version = 1.2.12
ncl 6.6.2 => /prodigfs/ipslfs/dods/jserver/miniconda/envs/analyse_env_2.7/bin/ncl
cdt 1.9.6 => /opt/nco/1.9/bin/cdt
nco (ncols) 4.5.2 => /opt/nco-4.5.2/bin/ncols
ncdump fichier => /prodigfs/ipslfs/dods/jserver/miniconda/envs/analyse_env_2.7/bin/ncdump
...
Cache directory set to : /data/mlalande/climafcache (use $CLIMAF_CACHE if set)
Cache directory for remote data set to : /data/mlalande/climafcache/remote_data (use $CLIMAF_REMOTE_CACHE if set)
warning : Binary cdftools not found. Some operators won't work
Available macros read from ~/.climaf.macros are : []

Entrée [2]: # The years considered to plot the biases between IPSL and observational references
first_year=1984
last_year=2014

"Load" model data

Entrée [3]: req_snow = ds(
    project='CMIP6',
    model='IPSL-CM6A-LR',
    variable='sfc',
    table='Limon',
    frequency='monthly',
    realization='r1i1p1f1',
    period=str(first_year)+ '-' +str(last_year),
```

Example for xarray

https://github.com/mickaellalande/MC-Toolkit/blob/master/Presentation/xarray_example.ipynb

The screenshot shows a Jupyter Notebook interface with the title "jupyter xarray_example (auto-sauvegardé)". The menu bar includes Fichier, Édition, Affichage, Insérer, Cellule, Noyau, Widgets, Aide, and a Python 3 kernel indicator. The toolbar includes icons for file operations like new, open, save, and execute.

The main content area displays a notebook titled "Xarray example". The first cell, "Import main modules", contains code to import xarray, matplotlib, numpy, and xesmf:

```
Entrée [1]: # This first line allows to have interactive plots inside the notebook
%matplotlib notebook

import xarray as xr # xarray is for dealing with NetCDF files
import matplotlib.pyplot as plt # for plot
import numpy as np # for maths
import xesmf as xe # for regridding
```

The second cell, "Load model data", contains code to open a dataset from a file:

```
Entrée [2]: model_dataset = xr.open_dataset(
    "data/snc_LIMon_IPSL-CM6A-LR_historical_r1i1p1f1_gr_185001-201412.nc"
)
```

The output of this cell, "Out[2]", shows the dataset structure:

```
Out[2]: <xarray.Dataset>
Dimensions:    (axis_nbounds: 2, lat: 143, lon: 144, time: 1980)
Coordinates:
* lat          (lat) float32 -90.0 -88.73239 -87.46479 ... 88.73239 90.0
* lon          (lon) float32 0.0 2.5 5.0 7.5 10.0 ... 350.0 352.5 355.0 357.5
* time         (time) datetime64[ns] 1850-01-16T12:00:00 ... 2014-12-16T12:00:00
Dimensions without coordinates: axis_nbounds
Data variables:
    time_bounds (time, axis_nbounds) datetime64[ns] ...
    snc         (time, lat, lon) float32 ...
Attributes:
    Conventions:      CF-1.7 CMIP-6.2
    creation date:   2018-07-11T07:36:35Z
```

Example “ma cuisine”

https://github.com/mickaellalande/MC-Toolkit/blob/master/Presentation/ma_cuisine.py

jupyter ma_cuisine.py il y a quelques secondes

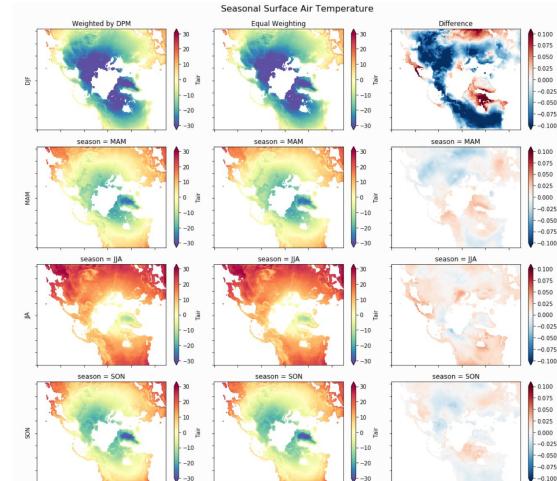
Fichier Édition Affichage Langage Se déconnecter Python

```
49 # =====
50 # Compute monthly weighted data
51 #
52 # http://xarray.pydata.org/en/stable/examples/monthly-means.html
53 dpm = {'no_leap': [0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31],
54     '365_day': [0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31],
55     'standard': [0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31],
56     'gregorian': [0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31],
57     'proleptic_gregorian': [0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31],
58     'all_leap': [0, 31, 29, 31, 30, 31, 31, 30, 31, 30, 31, 30, 31],
59     '366_day': [0, 31, 29, 31, 30, 31, 31, 30, 31, 30, 31, 30, 31],
60     '360_day': [0, 30, 30, 30, 30, 30, 30, 30, 30, 30, 30, 30, 30]}
61
62 def leap_year(year, calendar='standard'):
63     """Determine if year is a leap year"""
64     leap = False
65     if ((calendar in ['standard', 'gregorian',
66         'proleptic_gregorian'], 'julian']) and
67         (year % 4 == 0)):
68         leap = True
69     if ((calendar == 'proleptic_gregorian') and
70         (year % 100 == 0) and
71         (year % 400 != 0)):
72         leap = False
73     elif ((calendar in ['standard', 'gregorian']) and
74         (year % 100 == 0) and (year % 400 != 0) and
75         (year < 1583)):
76         leap = False
77     return leap
78
79 def get_dpm(time, calendar='standard'):
80     """
81     return a array of days per month corresponding to the months provided in `months`
82     """
83     month_length = np.zeros(len(time), dtype=np.int)
84
85     cal_days = dpm[calendar]
86
87     for i, (month, year) in enumerate(zip(time.month, time.year)):
88         month_length[i] = cal_days[month]
89         if leap_year(year, calendar=calendar) and month == 2:
90             month_length[i] += 1
91
92     return month_length
93
94 # Seasonal climatology (on monthly data set)
95 def season_clim(ds, calendar='standard'):
96     # Make a DataArray with the number of days in each month, size = len(time)
97     month_length = xr.DataArray(get_dpm(ds.time.to_index()), calendar=calendar),
```

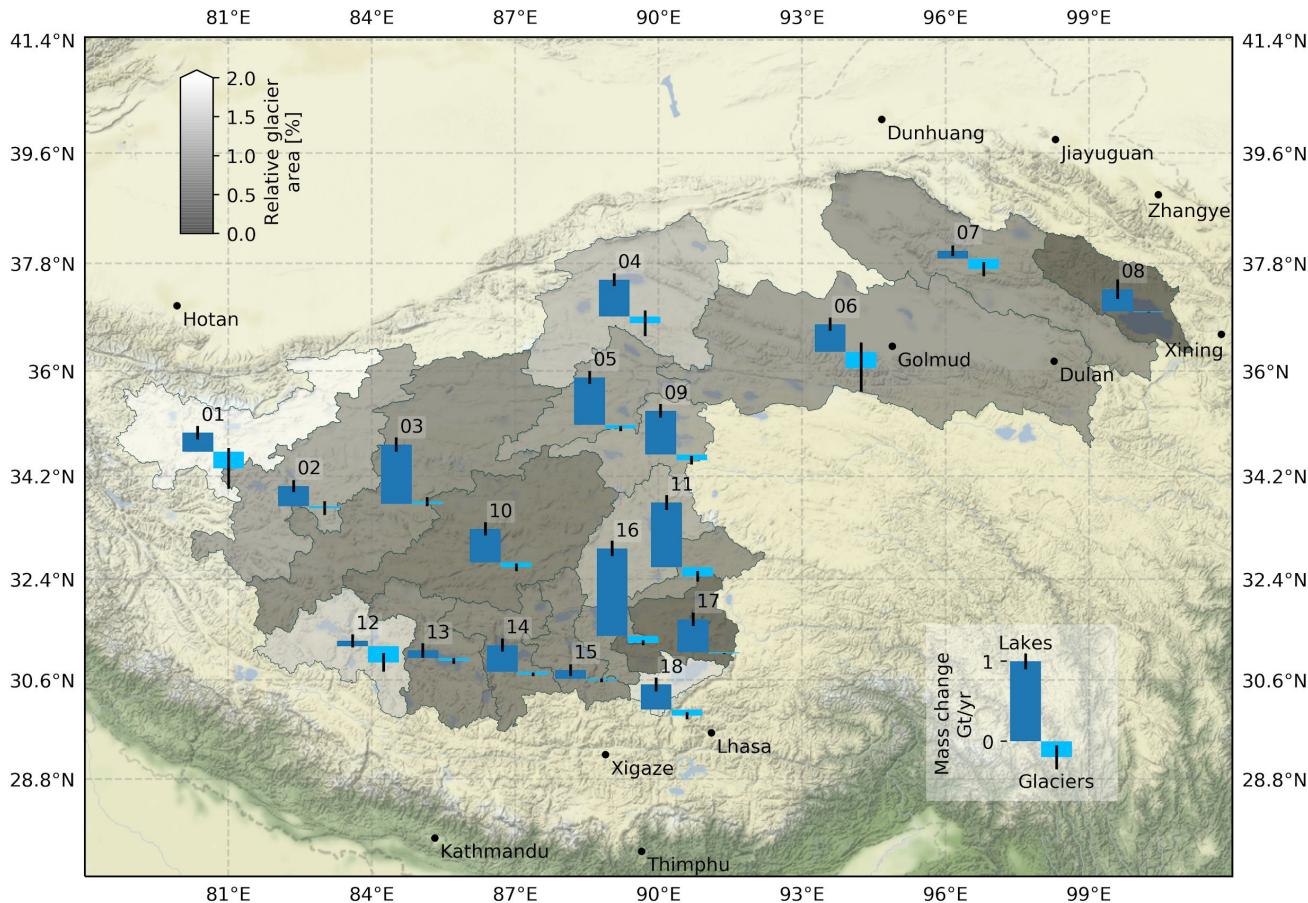
How to take into account the number of days in a month for monthly data analyse?

Code inspired from:

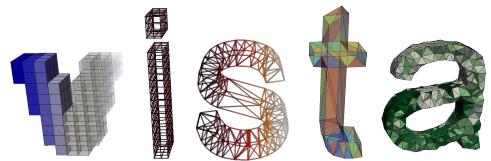
[Calculating Seasonal Averages from Timeseries of Monthly Means — xarray 0.14.1 documentation](http://xarray.pydata.org/en/stable/examples/monthly-means.html)



Example plots Fanny (with Cartopy)

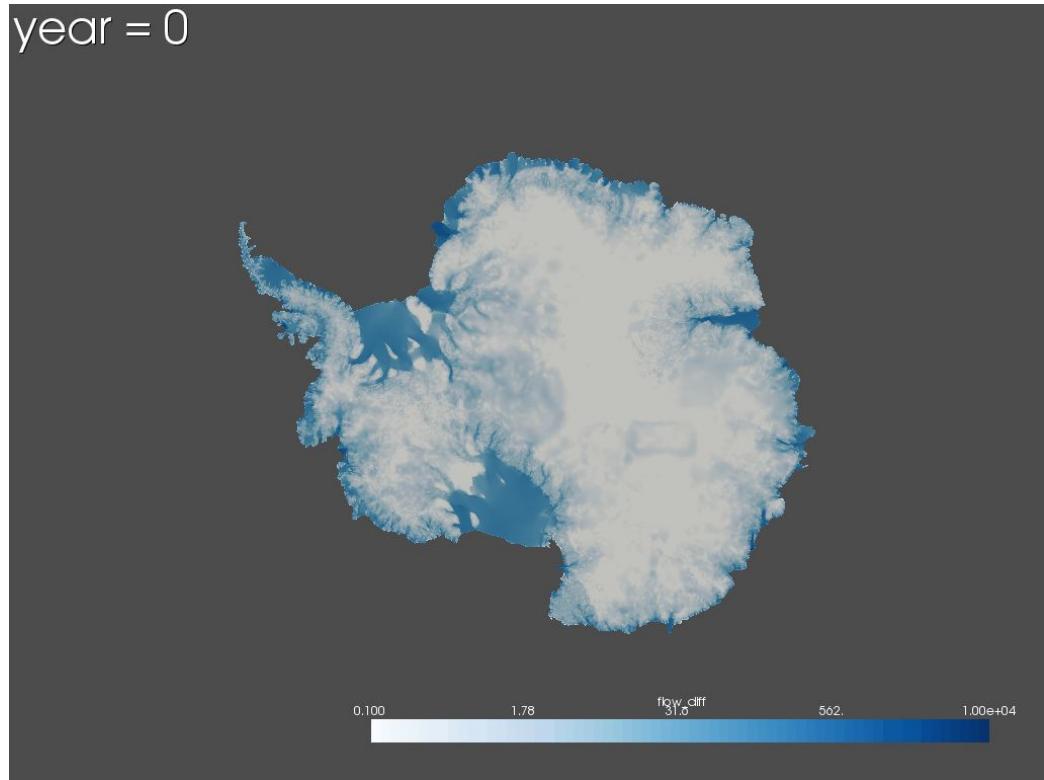


Example Benoit: Velocity differences over Antarctica (with PyVista)



3D plotting and mesh analysis through a streamlined interface for the Visualization Toolkit (VTK)

<https://docs.pyvista.org/>



MEOM's tuto

<https://github.com/meom-group/tutos/blob/master/software.md>

The screenshot shows a GitHub repository page for 'meom-group / tutos'. The repository has 3 stars, 5 forks, and 5 issues. The 'Code' tab is selected, showing the 'software.md' file. The file was last updated by 'auraoupa' on 27 Feb 2024. It contains 181 lines (128 sloc) and is 21.9 KB in size. The file content is as follows:

```
## Data analysis software used in MEOM group and how to learn it.

authors : Julien Le Sommer, Aurélie Albert and Redouane Lguensat (MEOM group, IGE)

This page provides a curated list of software used for data analysis in the MEOM group, online resources on how to use it and general advice on how to proceed with ocean data analysis. Please, keep in mind that (i) this list is not exhaustive and that (ii) it may evolve with time.

## General advice

A nice mental picture for understanding most of our data analysis tasks is the notion of data analysis pipeline. Our data analyses generally combine several steps, all corresponding to individual pieces of software. Our data flows through the pipeline and gets transformed at each step by a particular piece of software. Ideally these pipelines should be as automated as possible so that our work is easily reproducible.

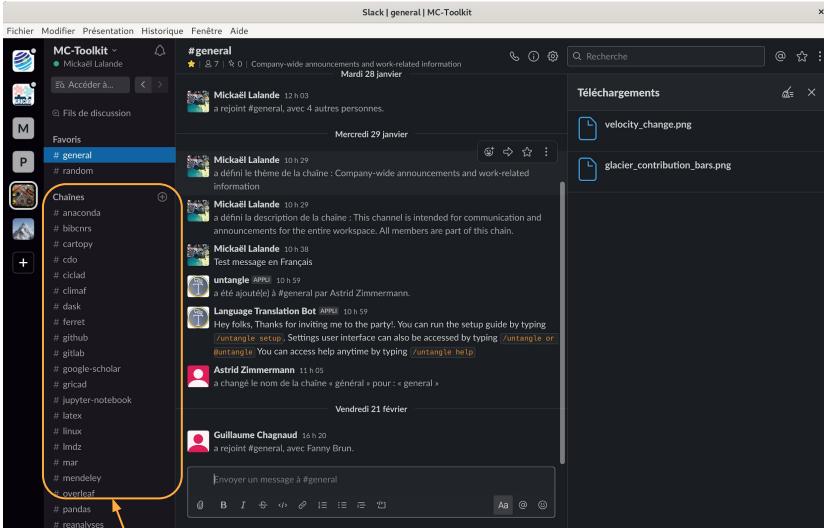
A key principle for building data analysis pipelines is to try to rely as much as possible on pre-existing software. In practice, most of the steps in an analysis pipeline are very generic (as eg. reading/writing/plotting data) so that we can just use preexisting code. So a large fraction of our work just involves glueing together existing pieces of code. This is why modern software is now made as modular as possible.

If you need to write new code, it should focus in priority on what is specific to your analysis. Some building-blocks of your data analysis pipeline are indeed more specific to your needs than others. For those key specific steps, you might have to write a
```

Slack + github

Lien d'invitation **Slack**:

https://join.slack.com/t/mc-toolkit-ige/shared_invite/zt-dc3fgldi-wFNV9AlbXzWb9dd0gdlHg



Add the channels that you are interested
So that we can ask question and help each other
with the regarding tool

Github

(to find back the code examples shown during these meetings):

<https://github.com/mickaellalande/MC-Toolkit>

A screenshot of the GitHub repository page for "mickaellalande / MC-Toolkit". The repository has 4 commits, 1 branch, 0 packages, and 0 releases. It has 1 contributor. The repository description is "Support for MC-Toolkit meetings at IGE". The README.md file contains a section titled "MC-Toolkit" with a goal to provide support materials for meetings at IGE about tools: MC-Toolkit (MC stand for Mardi-Clé, Modélisation&Climate... who knows?), in addition of a Slack channel. It lists several ideas for topics/tools to talk about, such as xarray (+dask) / cdo / climaf (climatology), How to compute climatologies (DJF, days in months, etc. what do you do?), cartopy / ferret / basemap / proplot (plot), regrid (cdo, basemap, scipy/stats, xESMF), and Jupyter Notebook / Anaconda.

Next meeting?

Same time?

In the context of Mardi Café or other?

MC for Mardi Café / Modélisation Climat / ??

I can start with xarray, climato (ma cuisine), proplot, xesmf
(+ anaconda/jupyter-notebooks/github ? May be before ?)

Possible next meetings:

0. Anaconda/Miniconda + Jupyter-Notebook?
 1. Martin: CLIMAF + R tool
 2. Mickaël: xarray + "ma cuisine"
 3. Mickaël / Fanny: plots (cartopy/proplot)
 4. Benoit: GeoPandas / PyVista
 5. Aurélie: Dask+xarray / Pangeo
 6. visit / FlowVR (Basile HECTOR)
 7. ...