

A Climate Analytics Hub for multi-model analysis

IS-ENES3/ESGF Virtual Workshop
on Compute and Analytics

2nd December, 2019

D. Elia^{1,2}, P. Nassisi¹, C. Palazzo¹,
F. Antonio¹, S. Fiore¹, G. Aloisio^{1,2}

¹ *Fondazione Centro Euro-Mediterraneo sui Cambiamenti Climatici
(CMCC), Lecce, Italy*

² *University of Salento, Lecce, Italy*



IS-ENES3 receives funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 824084

Multi-model climate analysis challenges & issues

Multi-model data analysis requires

- ✓ access to data produced by large-scale simulations for multiple climate models
- ✓ running workflows with hundreds of data analytics operators

Several **key challenges** and practical **issues** related to large-scale climate analysis

- ✓ Input data from multiple models needed
- ✓ **Data download** is a **big barrier** for climate scientists
- ✓ **Data analysis** mainly performed using **client-side & sequential** approaches
- ✓ **Installation** and update of data analysis **tools and libraries** needed
- ✓ Strong **requirements** in terms of **computational** and **storage** resources



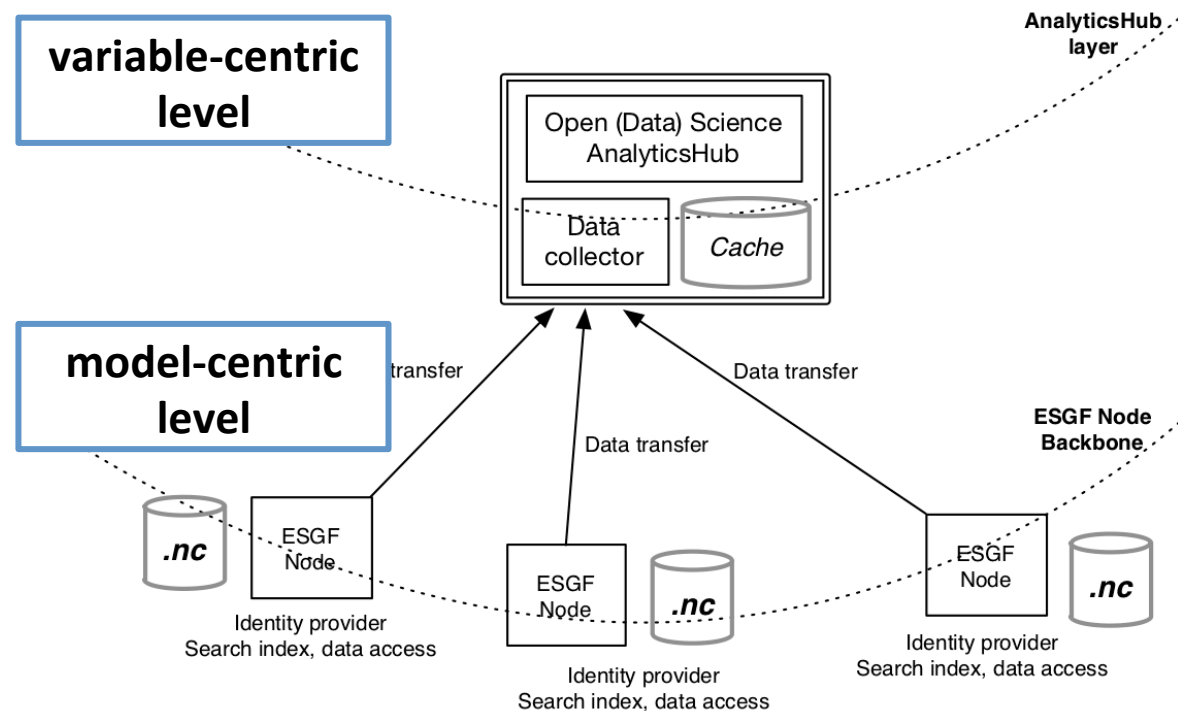
Climate Analytics-Hub

The **Climate Analytics-Hub** builds on top of the ESGF data nodes to allow the execution of multi-model climate analyses on a single location.

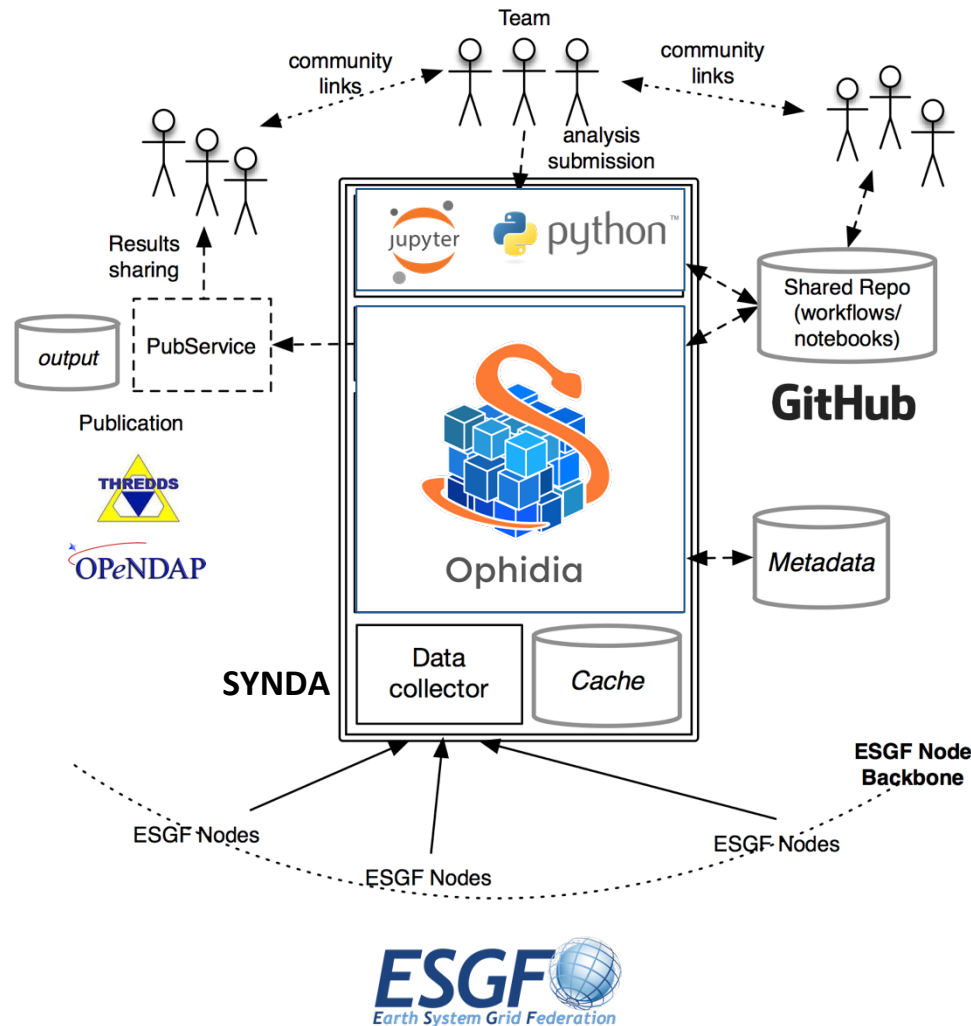
The Analytics-Hub provides Open Data Science oriented computing and analytics capabilities.

The **data collector** layer

- ✓ pre-stages and caches data relevant to the analyses from the different ESGF data nodes
- ✓ synchronizes the local copy of the data with the ESGF remote repositories



The CMCC Analytics Hub



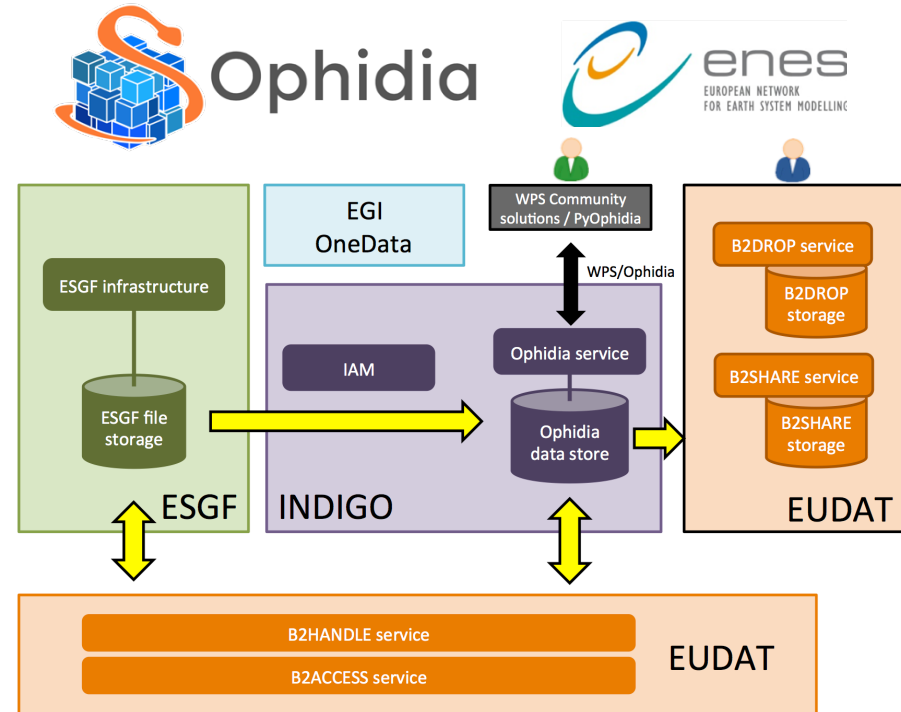
The **Analytics-Hub** consists of several components:

- ✓ A graphical **virtual environment** for (open) data science (e.g. JupyterHub)
- ✓ A wide set of high-level (Python) **scientific libraries** for analysis and plotting
- ✓ **Data Analytics and Machine Learning (HPDA) frameworks** for data science
- ✓ A user-oriented **monitoring system** to track application execution
- ✓ The **data collector** (Synda) and the local storage to gather relevant datasets from ESGF



ENES Climate Analytics Service (ECAS)

- ✓ The Analytics-Hub is a paradigm joining data and computing able to provide a **multi-model environment** for CMIP-based analytics experiments in ESGF
- ✓ The **ENES Climate Analytics Service (ECAS)**, proposed by CMCC & DKRZ in EOSC-hub supports climate data analysis
- ✓ It is one of the **EOSC-Hub Thematic Services**
- ✓ ECAS builds on top of the **Ophidia big data analytics framework** with components from INDIGO-DataCloud, EUDAT and EGI



The European Commission launched the European Open ScienceCloud Initiative to capitalise on the data revolution. EOSC will provide European science, industry and public authorities with world-class digital infrastructure that bring state of the art computing and data storage capacity to the fingertips of any scientists and engineer in the EU.



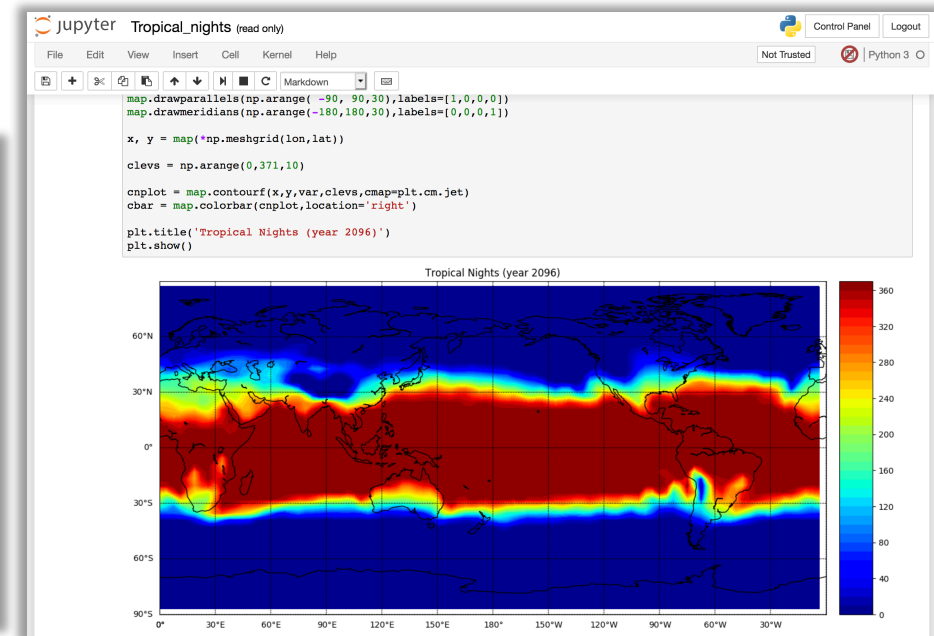
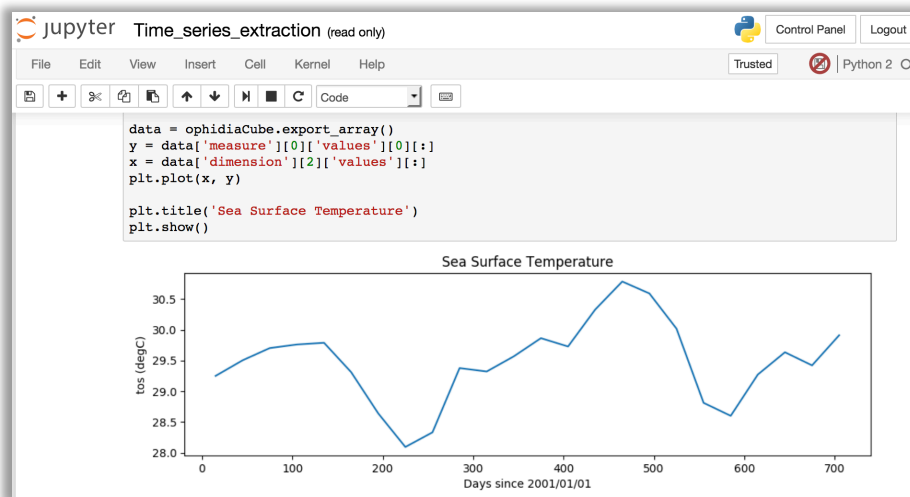
EOSC-hub receives funding from the EU's Horizon 2020 research and innovation programme under grant agreement No. 777536.



ECASLab: Python environment for Data Science

ECASLab provides a ready-to-use environment based on JupyterHub, Ophidia and other services from EUDAT and EGI, bundled with a wide set of well-known Python scientific and data management modules, some examples:

- NumPy, SciPy, Pandas
- NetCDF, PyOphidia, Scikit-learn
- Matplotlib, basemap, Cartopy



Programmatic access to ECAS through the PyOphidia class

- ✓ **PyOphidia** provides a Python interface to submit commands to ECAS and to retrieve/deserialize the results (e.g. in Jupyter Notebooks)
- ✓ Two modules implemented:
 - ✓ **Client**: supports the submissions of Ophidia commands and workflows, as well as the management of session from Python code (similar to the Ophidia Terminal)
 - ✓ **Cube class**: provides the datacube type abstraction and the methods to manipulate, process and get information on cubes objects

```
from PyOphidia import cube, client
cube.Cube.setclient(read_env=True)

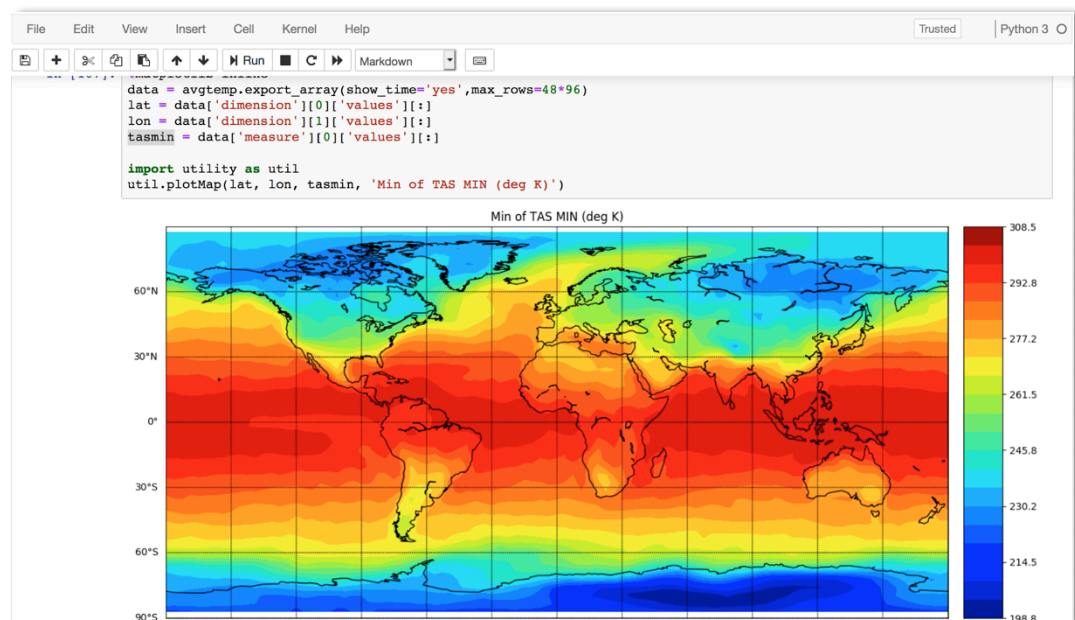
mycube =
cube.Cube.importnc(src_path='/public/data/ecas_training
/file.nc', measure='tos', imp_dim='time',
import_metadata='yes', ncores=5)
mycube2 = mycube.reduce(operation='max', ncores=5)
mycube3 = mycube2.rollup(ncores=5)
data = mycube3.export_array()

mycube3.exportnc2(output_path='/home/test',
export_metadata='yes')
```

<https://github.com/OphidiaBigData/PyOphidia>

<https://pypi.org/project/PyOphidia/>

<https://anaconda.org/conda-forge/pyophidia>

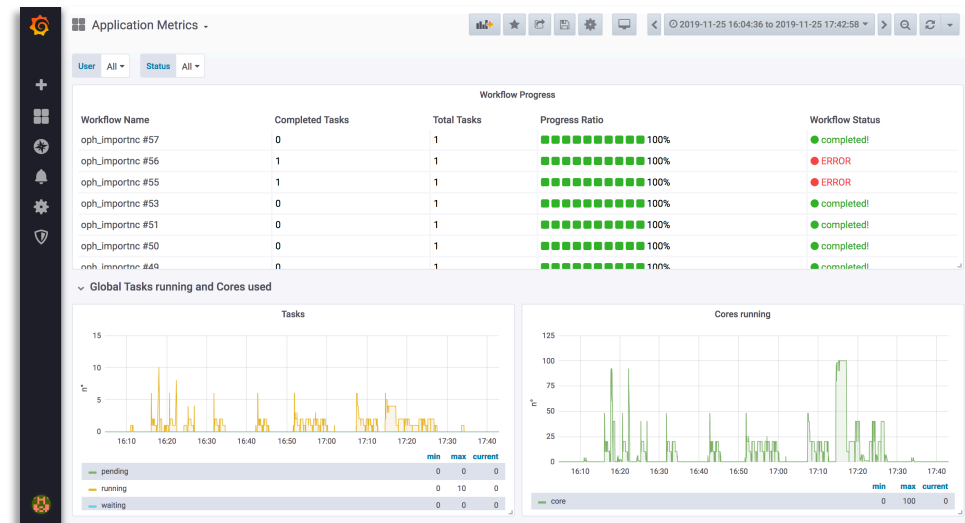
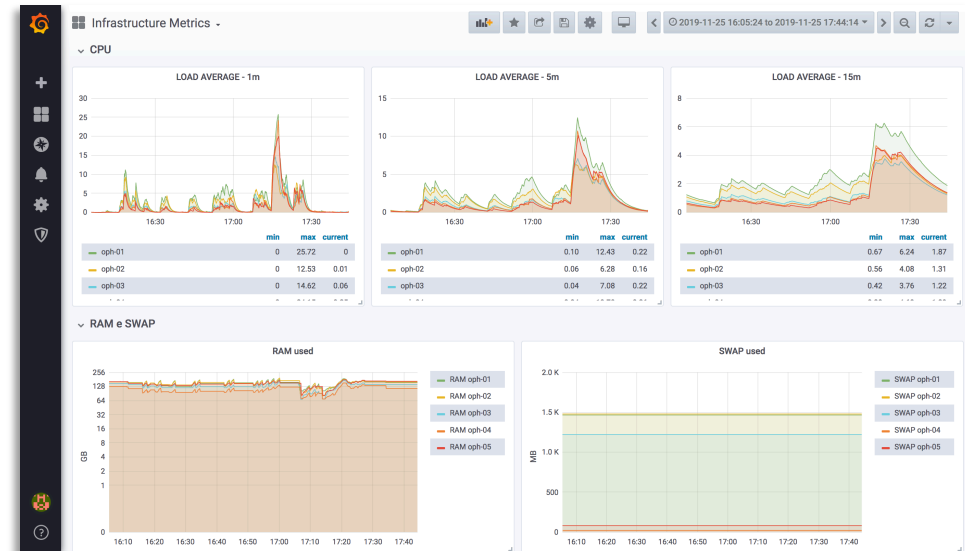


Real-time monitoring of applications

A **Grafana**-based system is used for real-time monitoring of the analytics-hub environment and the applications being executed.

Custom dashboards have been designed to track:

- ✓ **Infrastructure** metrics:
 - resource usage on the environment nodes (CPU, memory, disk)
- ✓ **Application** metrics:
 - status of each operators and number of workflows/operator running



Multi-model experiment demo

jupyter Precipitation Trend Analysis Demo Last Checkpoint: 8 minuti fa (autosaved) Logout Control Panel

File Edit View Insert Cell Kernel Widgets Help Trusted Python 2.0

```

In [ ]: ### Workflow for the analysis of precipitation trends related to different scenarios. ###
from PyOphidia import cube, client
cube.Cube.setclient(read_env=True)

In [ ]: # workflow parameters
ncores = 5 # number of cores
list_of_models = ['CanESM2', 'CCSM4', 'CMCC-CM', 'CMCC-CMS', 'CNRM-CM5', 'HadGEM2-ES', 'INM-CM4', 'IPESM2', 'MIROC5', 'MRI-CGCM3', 'NCAR-CCSM', 'OASU-CCCMA', 'OASU-CCCMA-CR', 'OASU-CCCMA-CR2', 'OASU-CCCMA-CR3', 'OASU-CCCMA-CR4', 'OASU-CCCMA-CR5', 'OASU-CCCMA-CR6', 'OASU-CCCMA-CR7', 'OASU-CCCMA-CR8', 'OASU-CCCMA-CR9', 'OASU-CCCMA-CR10', 'OASU-CCCMA-CR11', 'OASU-CCCMA-CR12', 'OASU-CCCMA-CR13', 'OASU-CCCMA-CR14', 'OASU-CCCMA-CR15', 'OASU-CCCMA-CR16', 'OASU-CCCMA-CR17', 'OASU-CCCMA-CR18', 'OASU-CCCMA-CR19', 'OASU-CCCMA-CR20', 'OASU-CCCMA-CR21', 'OASU-CCCMA-CR22', 'OASU-CCCMA-CR23', 'OASU-CCCMA-CR24', 'OASU-CCCMA-CR25', 'OASU-CCCMA-CR26', 'OASU-CCCMA-CR27', 'OASU-CCCMA-CR28', 'OASU-CCCMA-CR29', 'OASU-CCCMA-CR30', 'OASU-CCCMA-CR31', 'OASU-CCCMA-CR32', 'OASU-CCCMA-CR33', 'OASU-CCCMA-CR34', 'OASU-CCCMA-CR35', 'OASU-CCCMA-CR36', 'OASU-CCCMA-CR37', 'OASU-CCCMA-CR38', 'OASU-CCCMA-CR39', 'OASU-CCCMA-CR40', 'OASU-CCCMA-CR41', 'OASU-CCCMA-CR42', 'OASU-CCCMA-CR43', 'OASU-CCCMA-CR44', 'OASU-CCCMA-CR45', 'OASU-CCCMA-CR46', 'OASU-CCCMA-CR47', 'OASU-CCCMA-CR48', 'OASU-CCCMA-CR49', 'OASU-CCCMA-CR50', 'OASU-CCCMA-CR51', 'OASU-CCCMA-CR52', 'OASU-CCCMA-CR53', 'OASU-CCCMA-CR54', 'OASU-CCCMA-CR55', 'OASU-CCCMA-CR56', 'OASU-CCCMA-CR57', 'OASU-CCCMA-CR58', 'OASU-CCCMA-CR59', 'OASU-CCCMA-CR60', 'OASU-CCCMA-CR61', 'OASU-CCCMA-CR62', 'OASU-CCCMA-CR63', 'OASU-CCCMA-CR64', 'OASU-CCCMA-CR65', 'OASU-CCCMA-CR66', 'OASU-CCCMA-CR67', 'OASU-CCCMA-CR68', 'OASU-CCCMA-CR69', 'OASU-CCCMA-CR70', 'OASU-CCCMA-CR71', 'OASU-CCCMA-CR72', 'OASU-CCCMA-CR73', 'OASU-CCCMA-CR74', 'OASU-CCCMA-CR75', 'OASU-CCCMA-CR76', 'OASU-CCCMA-CR77', 'OASU-CCCMA-CR78', 'OASU-CCCMA-CR79', 'OASU-CCCMA-CR80', 'OASU-CCCMA-CR81', 'OASU-CCCMA-CR82', 'OASU-CCCMA-CR83', 'OASU-CCCMA-CR84', 'OASU-CCCMA-CR85', 'OASU-CCCMA-CR86', 'OASU-CCCMA-CR87', 'OASU-CCCMA-CR88', 'OASU-CCCMA-CR89', 'OASU-CCCMA-CR90', 'OASU-CCCMA-CR91', 'OASU-CCCMA-CR92', 'OASU-CCCMA-CR93', 'OASU-CCCMA-CR94', 'OASU-CCCMA-CR95', 'OASU-CCCMA-CR96', 'OASU-CCCMA-CR97', 'OASU-CCCMA-CR98', 'OASU-CCCMA-CR99', 'OASU-CCCMA-CR100']
scenario = 'rcp85' # scenario (e.g. rcp45 or rcp85)
frequency = 'day' # time frequency (e.g. day or mon)
percentile = 0.9 # percentile (e.g. 0.9)
past_time_subset = '1976_2006' # past time subset
future_time_subset = '2071_2101' # future time subset
spatial_subset = '-90:90|0:360' # geographic subset
output_grid = 'r360x180' # output grid using the format r<lon>x<lat>, i.e. a global
import_type = 1 # import type (optional), set to '1' in case only subsetting
io_server_type = 'ophidiaio_memory' # I/O server type (optional), default 'mysql_table'
base_path = '/PTA/precip_trend_data/'

In [ ]: # general import and other variables
import datetime
import multiprocessing
import multiprocessing.pool
from multiprocessing import Pool
from IPython.display import Image, display
import numpy as np

hist_subset = ','.join([str(y)+"-06_"+str(y)+"-09" for y in range(1976,2006)])
future_subset = ','.join([str(y)+"-06_"+str(y)+"-09" for y in range(2071,2101)])
workflow_id = int(np.random.random_sample() * 1000)

```

Infrastructure Metrics - Last 2 minutes 5s

LOAD AVERAGE - 1m

	min	max	current
oph-01	0.0200	0.1500	0.0200
oph-02	0.0200	0.1400	0.0200
oph-03	0.0300	0.2000	0.0300
oph-04	0.0400	0.1200	0.0500
oph-05	0.0500	0.1500	0.0500

jupyter Logout Control Panel

Every 2.0s: squeue Mon Dec 2 11:50:08 2019

JOBID	PARTITION	NAME	USER	ST	TIME	NODES	NODELIST(REASON)
[Job information is obscured in the screenshot]							

Catalog http://ophidialab.cmcc.it:8180/thredds/catalog/PTA/precip_trend_output/111238695229505952271558621818154495/catalog.html

Dataset	Size	Last Modified
111238695229505952271558621818154495		--

[OphidiaLab at CMCC see Info](#)
THREDDS Data Server [Version 4.3.23 - 20140826.1617] Documentation



Conclusions & next steps

Recap:

- ✓ The Analytics-Hub
 - defines a **variable-centric** environment for data analysis
 - **reduces time-to-solution** by removing the download barrier and providing server-side and high performance data analytics
 - provides a **user-friendly** environment for development, testing and execution of data analytics experiments

Next steps include:

- ✓ Implementation of multi-model experiments with CMIP6 data
- ✓ Deployment of the Analytics-Hub on CMCC new supercomputer
- ✓ Integration of additional data analytics tools to target a wider set of climate applications



Thanks

Useful links:

- ✓ CMCC ECASLab instance: <https://ecaslab.cmcc.it/>
- ✓ Ophidia Website: <http://ophidia.cmcc.it>
- ✓ Ophidia Doc: <http://ophidia.cmcc.it/documentation>
- ✓ PyOphidia repository: <https://github.com/OphidiaBigData/PyOphidia>
- ✓ Contact us at: ophidia-info@cmcc.it

