ESiWACE2 HPDA & Vis Training 2021

High-Perfomance Data Analytics in eScience with the Ophidia framework

Donatello Elia, Fabrizio Antonio
Fondazione Centro Euro-Mediterraneo sui Cambiamenti Climatici (CMCC), Lecce, Italy

esiwace
CENTRE OF EXCELLENCE IN SIMULATION OF WEATHER AND CLIMATE IN EUROPE

Session 3
15 September 2021

cmcc
Centro Euro-Mediterraneo sui Cambiamenti Climatici

Session outline

Introduction to HPDA and data challenges in eScience

Overview of the Ophidia HPDA framework

Ophidia core concepts: architecture, storage model, operators and primitives, terminal and deployment

Ophidia Python bindings: PyOphidia

DEMO: Introduction to PyOphidia

HANDS-ON: Data analytics examples with PyOphidia

Climate analysis challenges & issues

Effective scientific analysis requires *novel solutions* able to cope with ***big data volumes***

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

- Setup of a data analysis experiment requires the ***download of (multiple) input data***
  - *Data download is a big barrier for climate scientists*
  - *Reducing data movement is essential*
- The complexity of the analysis leads to the need for ***end-to-end workflow support***
  - *Data analysis* requires *highly-scalable solutions* able to *parallelise* the processing
  - Analysing large datasets involves *running tens/hundreds of analytics operators*
- Large data volumes pose ***strong requirements in terms of computational and storage resources***

High Performance Data Analytics for eScience

- *Computational science modeling and data analytics are both crucial in scientific research*
  - *Their coexistence in the same (current) software infrastructure is not trivial*
- *The convergence of the solutions and technology from the Big Data and HPC software ecosystems is a key factor for accelerating scientific discovery*

High-Performance Data Analytics (HPDA)

- *New computing paradigms, data management approaches and job management solutions are being designed by the scientific software community*
- *Higher-level programming approaches* for data analytics are required to effectively exploit the resources and improve scientists' productivity

Session outline

Introduction to HPDA and data challenges in eScience

Overview of the Ophidia HPDA framework

Ophidia core concepts: architecture, storage model, operators and primitives, terminal and deployment

Ophidia Python bindings: PyOphidia

DEMO: Introduction to PyOphidia

HANDS-ON: Data analytics examples with PyOphidia

Ophidia HPDA framework

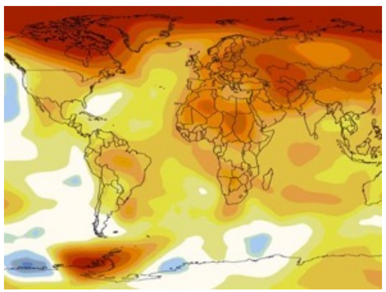
Ophidia (http://ophidia.cmcc.it) is a CMCC Foundation research project addressing data challenges for eScience

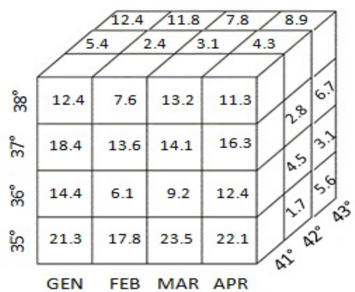
- A **HPDA framework** for multi-dimensional scientific data joining HPC paradigms with scientific data analytics approaches
- **In-memory** and **server-side** data analysis exploiting parallel computing techniques
- Multi-dimensional, array-based, storage model and partitioning schema for scientific data leveraging the **datacube** abstraction
- End-to-end mechanisms to support **interactive analysis, complex experiments** and **large workflows** on scientific data

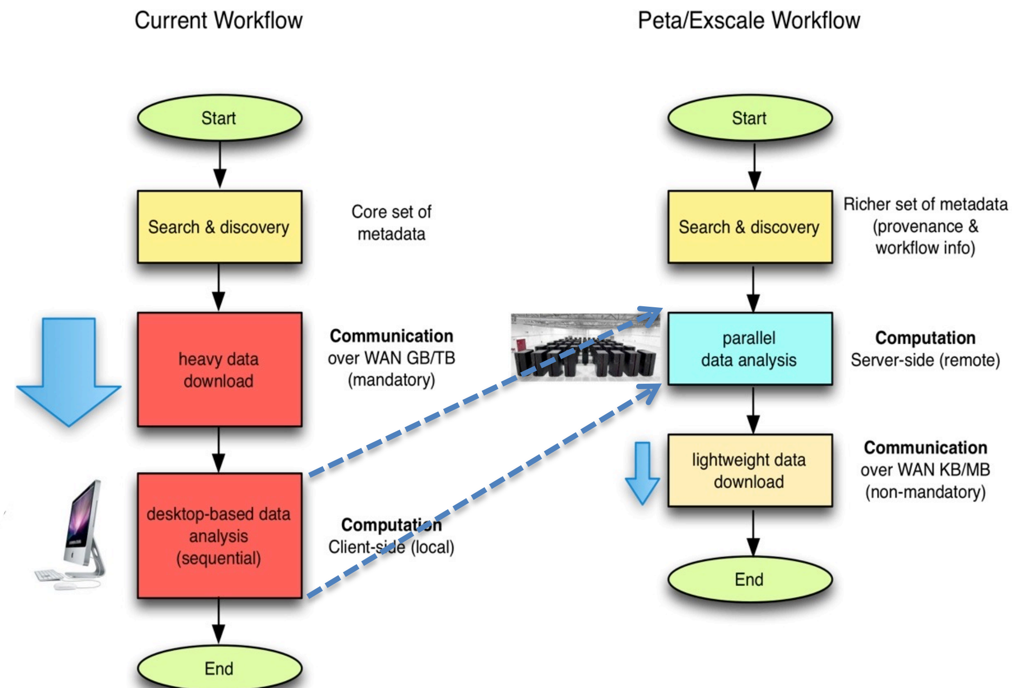
S. Fiore, D. Elia, C. Palazzo, F. Antonio, A. D'Anca, I. Foster, G. Aloisio, "Towards High Performance Data Analytics for Climate Change", ISC High Performance 2019, LNCS Springer, 2019

A paradigm shift

Volume, variety, velocity are key challenges for big data in general and for climate sciences in particular. Client-side, sequential and disk-based workflows are three limiting factors for the current scientific data analysis tools.





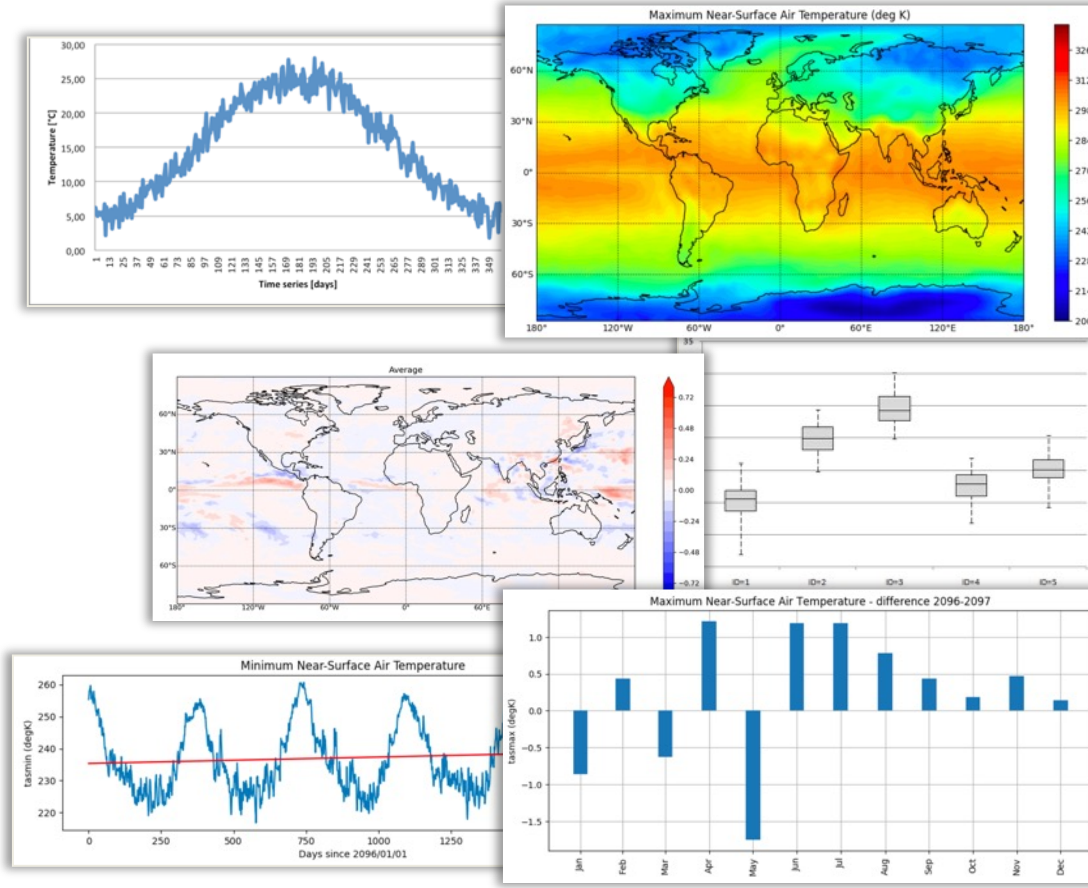


S. Fiore, A. D'Anca, C. Palazzo, I. Foster, D. N. Williams, G. Aloisio, "Ophidia: toward bigdata analytics for eScience", ICCS2013 Conference, Procedia Elsevier, 2013

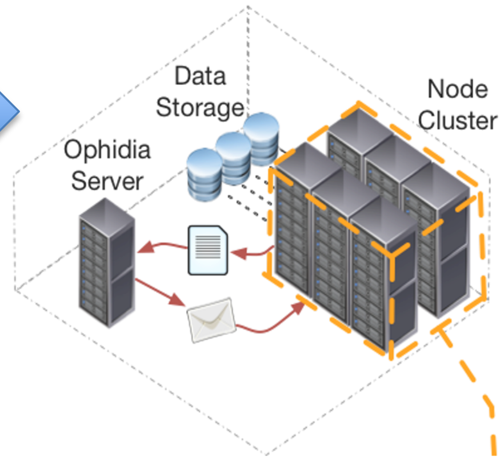
Data analytics requirements and use cases

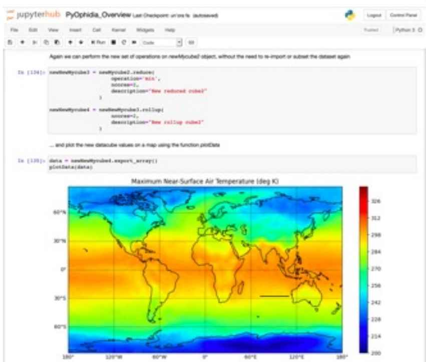
Requirements and needs focus on:

- *Time series analysis*
- *Data subsetting*
- *Model intercomparison*
- *Multi-model means*
- *Massive data reduction*
- *Data transformation*
- *Parameter sweep experiments*
- *Maps generation*
- *Ensemble analysis*
- *Data analytics workflow support*



Server-side paradigm and execution modes





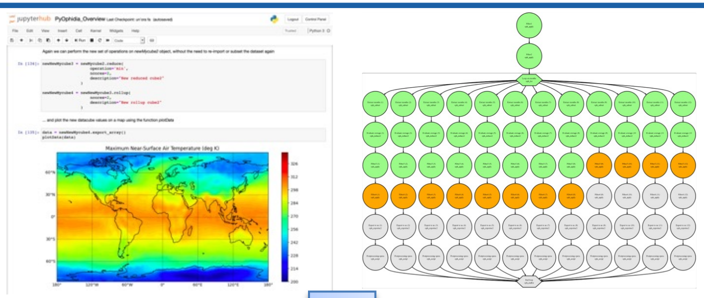
Oph_Term: a terminal-like commands interpreter serving as a client for the Ophidia framework

PyOphidia: a Python interface for datacube management & analytics with Ophidia

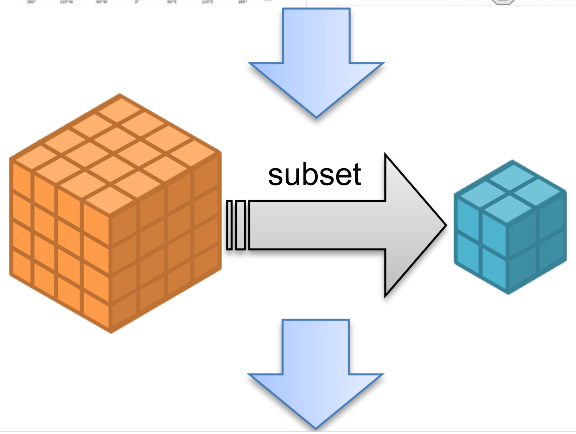
Multiple execution modes:

- *Interactive data analysis*
- *Batch processing*
- *Python notebooks and applications*
- *Workflows of operators*

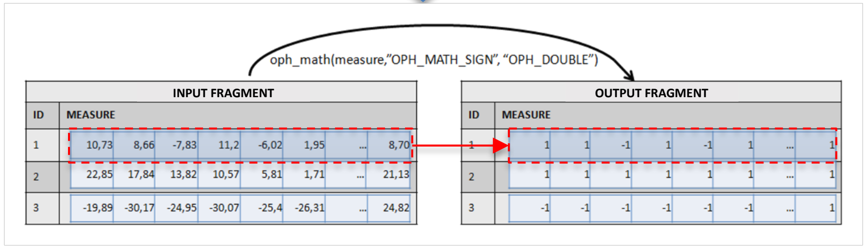
Granularity of operations in Ophidia



Workflows/applications: combine multiple Ophidia Operators to compute from complex experiments (e.g., multi-model analysis) to simple indicators (e.g., Summer Days)



Ophidia Operators: datacube-level operations on multi-dimensional data. Both data and metadate. Some examples: subsetting, aggregation, comparison



INPUT FRAGMENT

ID	MEASURE							
1	10,73	8,66	-7,83	11,2	-6,02	1,95	...	8,70
2	22,85	17,84	13,82	10,57	5,81	1,71	...	21,13
3	-19,89	-30,17	-24,95	-30,07	-25,4	-26,31	...	24,82

OUTPUT FRAGMENT

ID	MEASURE							
1	1	1	-1	1	-1	1	...	1
2	1	1	1	1	1	1	...	1
3	-1	-1	-1	-1	-1	-1	...	1

Ophidia Primitives: low-level functions applied on the single binary arrays of the datacube fragments. Some examples: time series analysis, array transformations

Session outline

Introduction to HPDA and data challenges in eScience

Overview of the Ophidia HPDA framework

Ophidia core concepts: architecture, storage model, operators and primitives, terminal and deployment

Ophidia Python bindings: PyOphidia

DEMO: Introduction to PyOphidia

HANDS-ON: Data analytics examples with PyOphidia

Ophidia architecture: overview

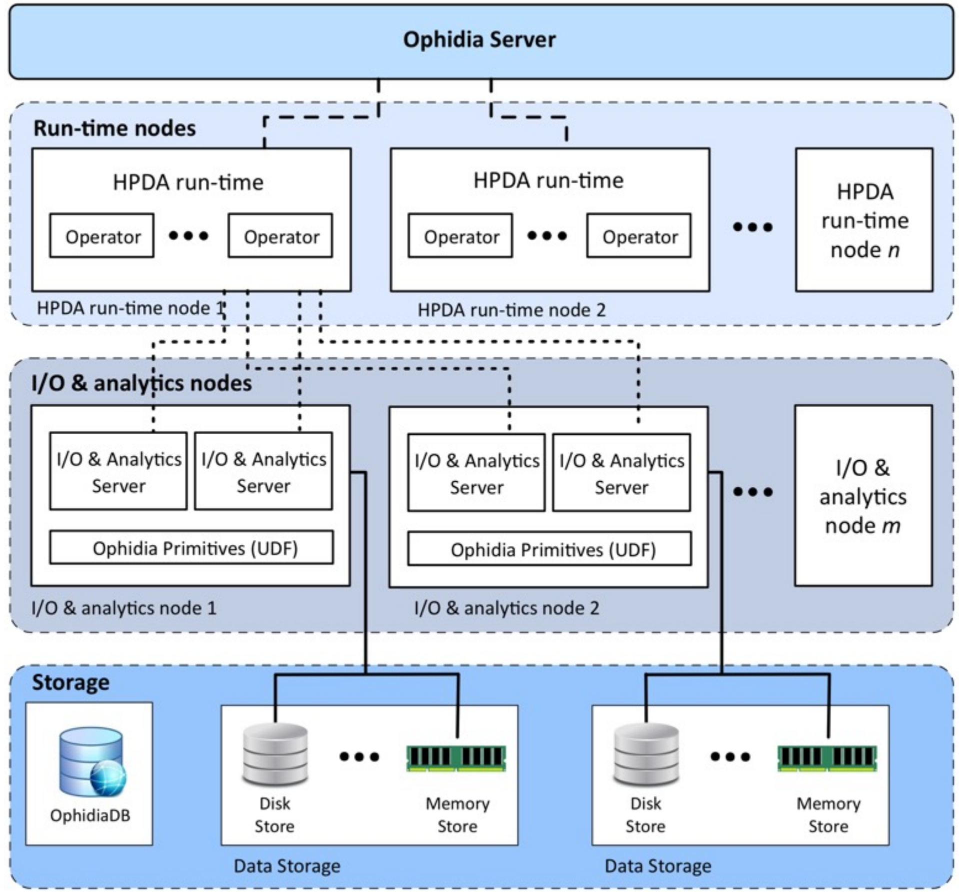
Front end

Runtime system

I/O & Analytics servers

Storage layer

System catalog



Ophidia architecture: storage layer & model

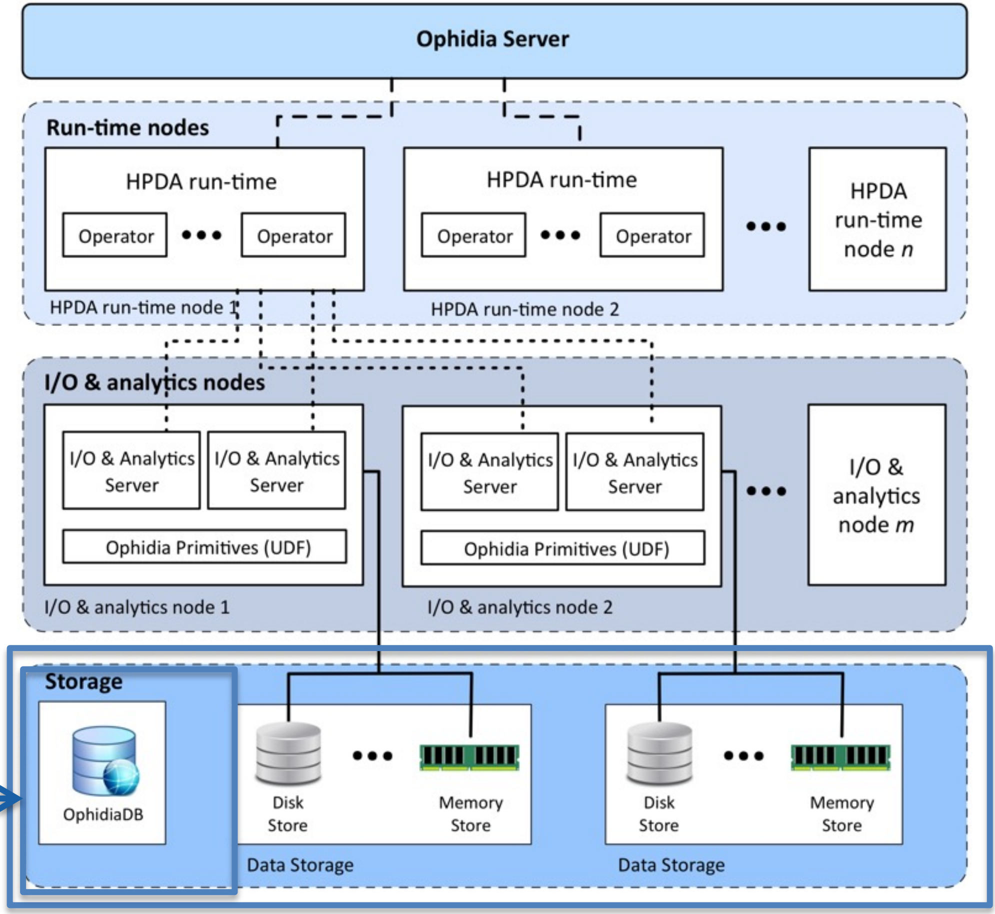
Distributed *hardware resources to manage storage*

Ophidia implements the ***datacube abstraction*** *from OLAP*

The storage model relies on ***implicit*** *(array-based) and* ***explicit*** *(tuple-based)* ***dimensions*** *for specific representations of data*

Data partitioned *in a hierarchical fashion over the storage according to the storage model & partitioning schema*

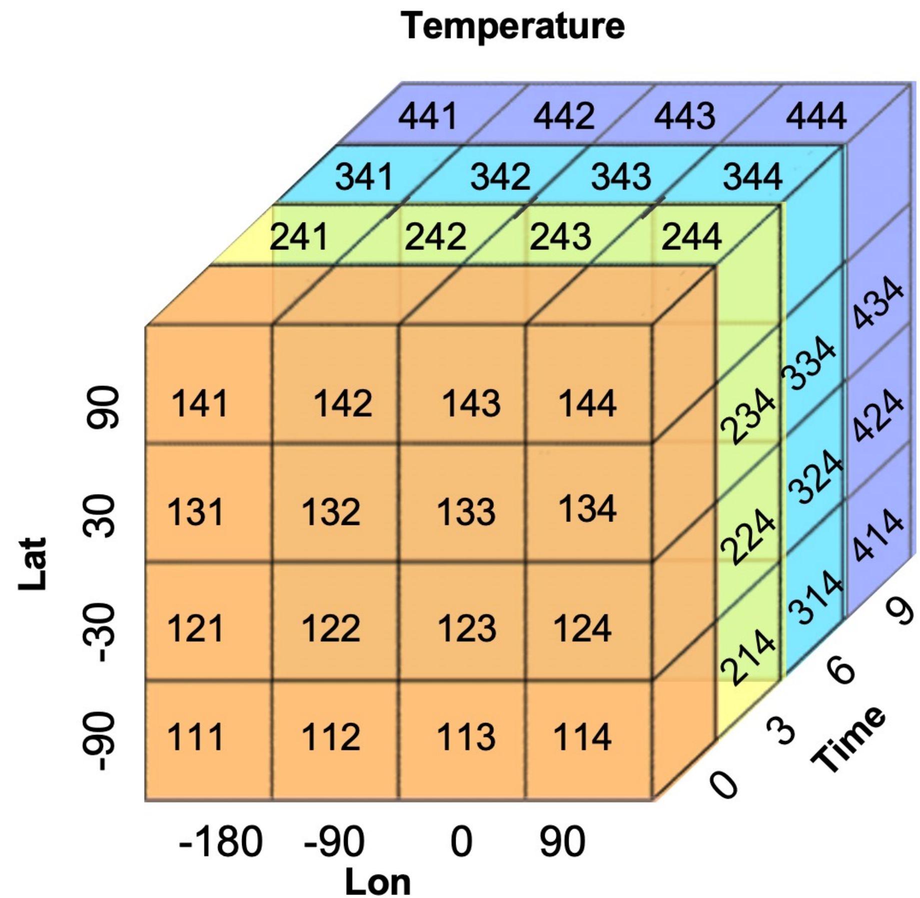
OphidiaDB is the system catalog: maps data fragmentation and tracks metadata



S. Fiore, D. Elia, C. Palazzo, F. Antonio, A. D'Anca, I. Foster, G. Aloisio, "Towards High Performance Data Analytics for Climate Change", ISC High Performance 2019, LNCS Springer, 2019

From NetCDF to datacube

```
netcdf test {
dimensions:
    lat = 4 ;
    lon = 4 ;
    time = UNLIMITED // (4 currently) ;
variables:
    double lon(lon) ;
    double lat(lat) ;
    double time(time) ;
    float Temperature(time, lat, lon) ;
data:
    lon = -180, -90, 0, 90 ;
    lat = -90, -30, 30, 90 ;
    time = 0, 3, 6, 9 ;
    temperature =
        111, 112, 113, 114,
          121, 122, 123, 124,
          131, 132, 133, 134,
          141, 142, 143, 144,
        211, 212, 213, 214,
          221, 222, 223, 224,
          231, 232, “33, 234,
          241, 242, 243, 244,
        ...
```



The datacube abstraction naturally fits for scientific multi-dimensional data, like climate data

From NetCDF to Ophidia

```
netcdf test {
dimensions:
    lat = 4 ;
    lon = 4 ;
    time = UNLIMITED // (4 currently) ;
variables:
    double lon(lon) ;
    double lat(lat) ;
    double time(time) ;
    float Temperature(time, lat, lon) ;
data:
    lon = -180, -90, 0, 90 ;
    lat = -90, -30, 30, 90 ;
    time = 0, 3, 6, 9 ;
    temperature =
        111, 112, 113, 114,
          121, 122, 123, 124,
          131, 132, 133, 134,
          141, 142, 143, 144,
        211, 212, 213, 214,
          221, 222, 223, 224,
          231, 232, 233, 234,
          241, 242, 243, 244,
        311, 312, 313, 314,
          ...
```

NetCDF

Defined as:
implicit dimension

		Temperature			
lat	lon	time[0]	time[1]	time[2]	time[3]
-90	-180	111	211	311	411
-90	-90	112	212	312	412
-90	0	113	213	313	413
-90	90	114	214	314	414
-30	-180	121	221	321	421
-30	-90	122	222	322	422
-30	0	123	223	323	423
-30	90	124	224	324	424
30	-180	131	231	331	431
30	-90	132	232	332	432
30	0	133	233	333	433
30	90	134	234	334	434
90	-180	141	241	341	441
90	-90	142	242	342	442
90	0	143	243	343	443
90	90	144	244	344	444

Ophidia

From NetCDF to Ophidia

```
netcdf test {
dimensions:
    lat = 4 ;
    lon = 4 ;
    time = UNLIMITED // (4 currently) ;
variables:
    double lon(lon) ;
    double lat(lat) ;
    double time(time) ;
    float Temperature(time, lat, lon) ;
data:
    lon = -180, -90, 0, 90 ;
    lat = -90, -30, 30, 90 ;
    time = 0, 3, 6, 9 ;
    temperature =
        111, 112, 113, 114,
          121, 122, 123, 124,
          131, 132, 133, 134,
          141, 142, 143, 144,
        211, 212, 213, 214,
          221, 222, 223, 224,
          231, 232, 233, 234,
          241, 242, 243, 244,
        311, 312, 313, 314,
          ...
```

NetCDF

Defined as:
explicit dimensions

		Temperature			
lat	lon	time[0]	time[1]	time[2]	time[3]
-90	-180	111	211	311	411
-90	-90	112	212	312	412
-90	0	113	213	313	413
-90	90	114	214	314	414
-30	-180	121	221	321	421
-30	-90	122	222	322	422
-30	0	123	223	323	423
-30	90	124	224	324	424
30	-180	131	231	331	431
30	-90	132	232	332	432
30	0	133	233	333	433
30	90	134	234	334	434
90	-180	141	241	341	441
90	-90	142	242	342	442
90	0	143	243	343	443
90	90	144	244	344	444

Ophidia

From NetCDF to Ophidia

```
netcdf test {
dimensions:
    lat = 4 ;
    lon = 4 ;
    time = UNLIMITED // (4 currently) ;
variables:
    double lon(lon) ;
    double lat(lat) ;
    double time(time) ;
    float Temperature(time, lat, lon) ;
data:
    lon = -180, -90, 0, 90 ;
    lat = -90, -30, 30, 90 ;
    time = 0, 3, 6, 9 ;
    temperature =
        111, 112, 113, 114,
          121, 122, 123, 124,
          131, 132, 133, 134,
          141, 142, 143, 144,
        211, 212, 213, 214,
          221, 222, 223, 224,
          231, 232, 233, 234,
          241, 242, 243, 244,
        311, 312, 313, 314,
          ...
```

NetCDF

Mapped to a single unique key

ID	Array			
1	111	211	311	411
2	112	212	312	412
3	113	213	313	413
4	114	214	314	414
5	121	221	321	421
6	122	222	322	422
7	123	223	323	423
8	124	224	324	424
9	131	231	331	431
10	132	232	332	432
11	133	233	333	433
12	134	234	334	434
13	141	241	341	441
14	142	242	342	442
15	143	243	343	443
16	144	244	344	444

Ophidia

From NetCDF to Ophidia

```
netcdf test {
dimensions:
    lat = 4 ;
    lon = 4 ;
    time = UNLIMITED // (4 currently) ;
variables:
    double lon(lon) ;
    double lat(lat) ;
    double time(time) ;
    float Temperature(time, lat, lon) ;
data:
    lon = -180, -90, 0, 90 ;
    lat = -90, -30, 30, 90 ;
    time = 0, 3, 6, 9 ;
    temperature =
      111, 112, 113, 114,
        121, 122, 123, 124,
        131, 132, 133, 134,
        141, 142, 143, 144,
      211, 212, 213, 214,
        221, 222, 223, 224,
        231, 232, 233, 234,
        241, 242, 243, 244,
      311, 312, 313, 314,
        ...
```

NetCDF

Data reorganised based on implicit and explicit dimensions

		Temperature			
lat	lon	time[0]	time[1]	time[2]	time[3]
-90	-180	111	211	311	411
-90	-90	112	212	312	412
-90	0	113	213	313	413
-90	90	114	214	314	414
-30	-180	121	221	321	421
30	-90	122	222	322	422
30	0	123	223	323	423
30	90	124	224	324	424
30	-180	131	231	331	431
30	-90	132	232	332	432
30	0	133	233	333	433
30	90	134	234	334	434
90	-180	141	241	341	441
90	-90	142	242	342	442
90	0	143	243	343	443
90	90	144	244	344	444

Ophidia

From NetCDF to Ophidia

```
netcdf test {
dimensions:
    lat = 4 ;
    lon = 4 ;
    time = UNLIMITED // (4 currently) ;
variables:
    double lon(lon) ;
    double lat(lat) ;
    double time(time) ;
    float Temperature(time, lat, lon) ;
data:
    lon = -180, -90, 0, 90 ;
    lat = -90, -30, 30, 90 ;
    time = 0, 3, 6, 9 ;
    temperature =
        111, 112, 113, 114,
          121, 122, 123, 124,
          131, 132, 133, 134,
          141, 142, 143, 144,
        211, 212, 213, 214,
          221, 222, 223, 224,
          231, 232, 233, 234,
          241, 242, 243, 244,
        311, 312, 313, 314,
          ...
```

NetCDF

Table horizontally partitioned in multiple fragments

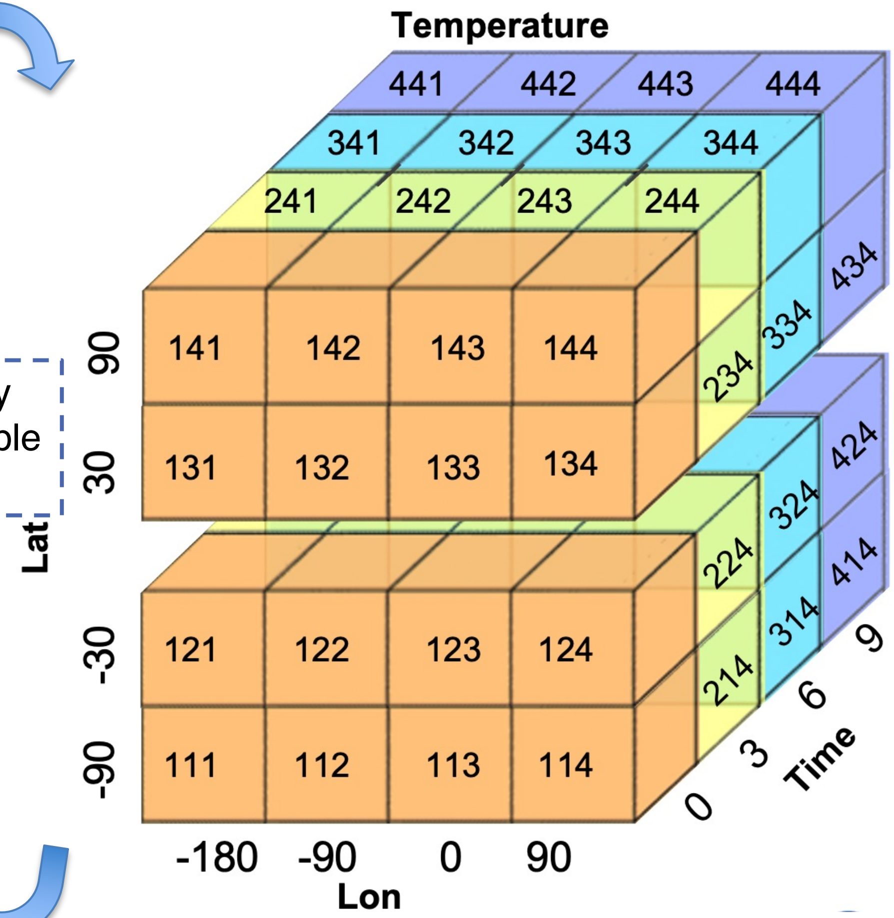
lat	lon	Temperature			
		time[0]	time[1]	time[2]	time[3]
-90	-180	111	211	311	411
-90	-90	112	212	312	412
-90	0	113	213	313	413
-90	90	114	214	314	414
-30	-180	121	221	321	421
-30	-90	122	222	322	422
-30	0	123	223	323	423
-30	90	124	224	324	424
30	-180	131	231	331	431
30	-90	132	232	332	432
30	0	133	233	333	433
30	90	134	234	334	434
90	-180	141	241	341	441
90	-90	142	242	342	442
90	0	143	243	343	443
90	90	144	244	344	444

From NetCDF to Ophidia

```
netcdf test {
dimensions:
    lat = 4 ;
    lon = 4 ;
    time = UNLIMITED // (4 currently) ;
variables:
    double lon(lon) ;
    double lat(lat) ;
    double time(time) ;
    float Temperature(time, lat, lon) ;
data:
    lon = -180, -90, 0, 90 ;
    lat = -90, -30, 30, 90 ;
    time = 0, 3, 6, 9 ;
    temperature =
        111, 112, 113, 114,
          121, 122, 123, 124,
          131, 132, 133, 134,
          141, 142, 143, 144,
        211, 212, 213, 214,
          221, 222, 223, 224,
          231, 232, 233, 234,
          241, 242, 243, 244,
        311, 312, 313, 314,
          ...
```

NetCDF

Table horisontally partitioned in multiple fragments

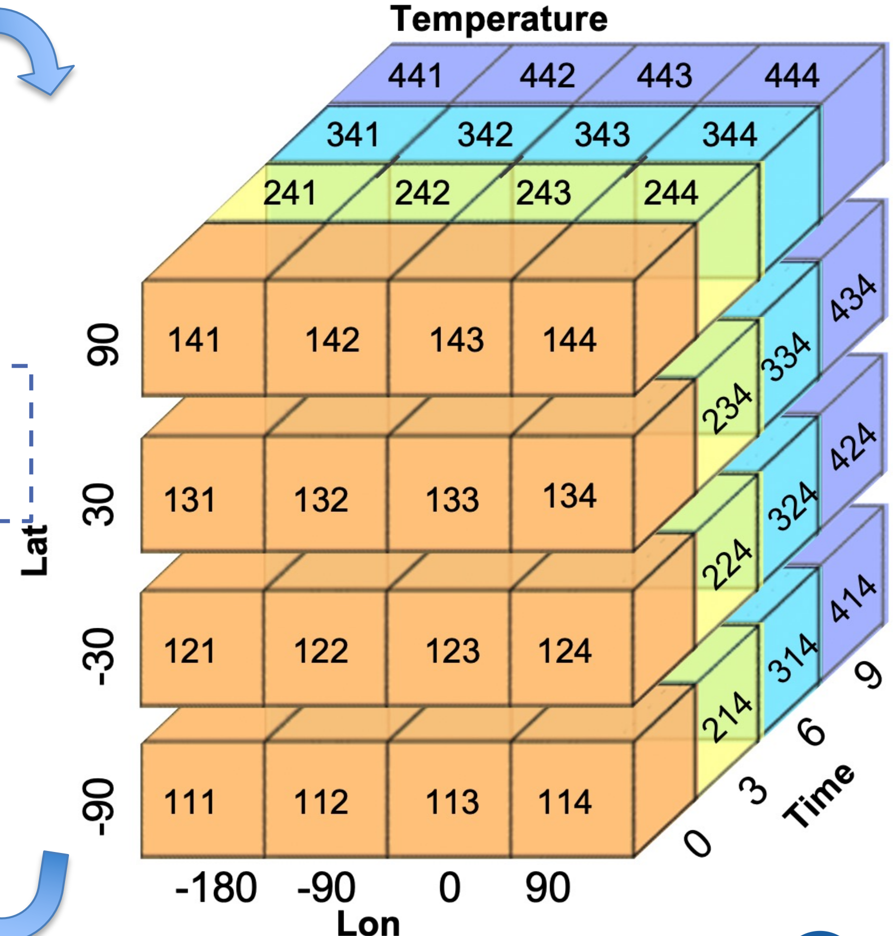


From NetCDF to Ophidia

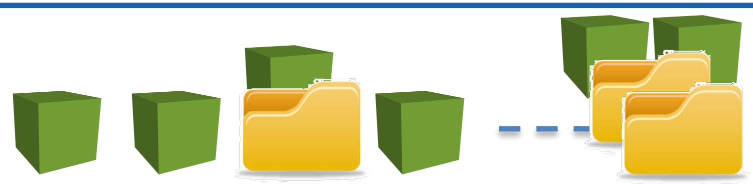
```
netcdf test {
dimensions:
    lat = 4 ;
    lon = 4 ;
    time = UNLIMITED // (4 currently) ;
variables:
    double lon(lon) ;
    double lat(lat) ;
    double time(time) ;
    float Temperature(time, lat, lon) ;
data:
    lon = -180, -90, 0, 90 ;
    lat = -90, -30, 30, 90 ;
    time = 0, 3, 6, 9 ;
    temperature =
        111, 112, 113, 114,
          121, 122, 123, 124,
          131, 132, 133, 134,
          141, 142, 143, 144,
        211, 212, 213, 214,
          221, 222, 223, 224,
          231, 232, 233, 234,
          241, 242, 243, 244,
        311, 312, 313, 314,
          ...
```

NetCDF

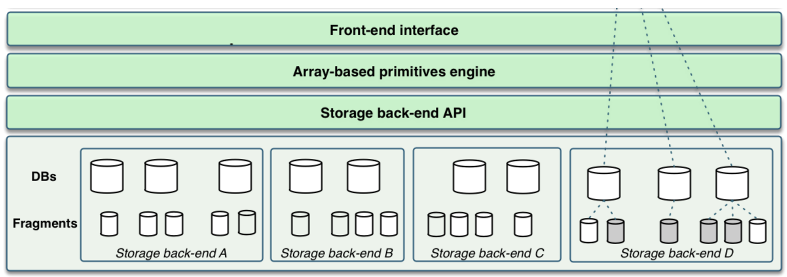
Fragmentation can be increased for parallel analysis



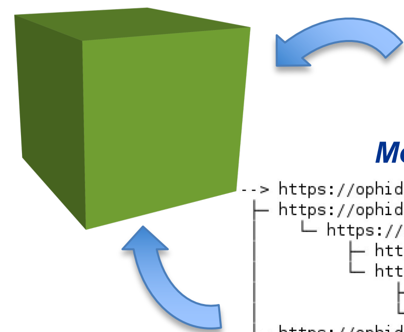
Data abstraction: cube space perspective



System perspective (internal storage representation)



User perspective (datacube abstraction)



System metadata of the datacube (size, distribution, etc.)

Metadata provenance

```
--> https://ophidia.cmcc.it:8443/162/169 (ROOT)
 ├ https://ophidia.cmcc.it:8443/162/170 (oph_reduce)
 │    └ https://ophidia.cmcc.it:8443/162/171 (oph_merge)
 │         ├ https://ophidia.cmcc.it:8443/162/172 (oph_aggregate2)
 │         └ https://ophidia.cmcc.it:8443/162/173 (oph_rollup)
 │              ├ https://ophidia.cmcc.it:8443/162/174 (oph_reduce)
 │              └ https://ophidia.cmcc.it:8443/162/175 (oph_reduce)
 ├ https://ophidia.cmcc.it:8443/162/176 (oph_aggregate)
 └ https://ophidia.cmcc.it:8443/162/177 (oph_aggregate)
```

CMD	BEHAVIOUR
cd	change directory
mkdir	create a new folder
rm	remove an empty folder or hide (logically delete) a container
ls	list subfolders and containers in a folder
mv	move/rename a folder or a container

Metadata associated to the datacubes

Searching results

Id	Variable	Key	Type	Value
73693068	tas	standard_name	text	air_temperature
73693069	tas	long_name	text	Air Temperature
73693070	tas	comment	text	This is sampled synoptically.
73693071	tas	units	text	K
73693072	tas	original_name	text	temp2

S. Fiore, D. Elia, C. Palazzo, F. Antonio, A. D'Anca, I. Foster, G. Aloisio, "Towards High Performance Data Analytics for Climate Change", ISC High Performance 2019, LNCS Springer, 2019

Ophidia architecture: I/O & Analytics layer

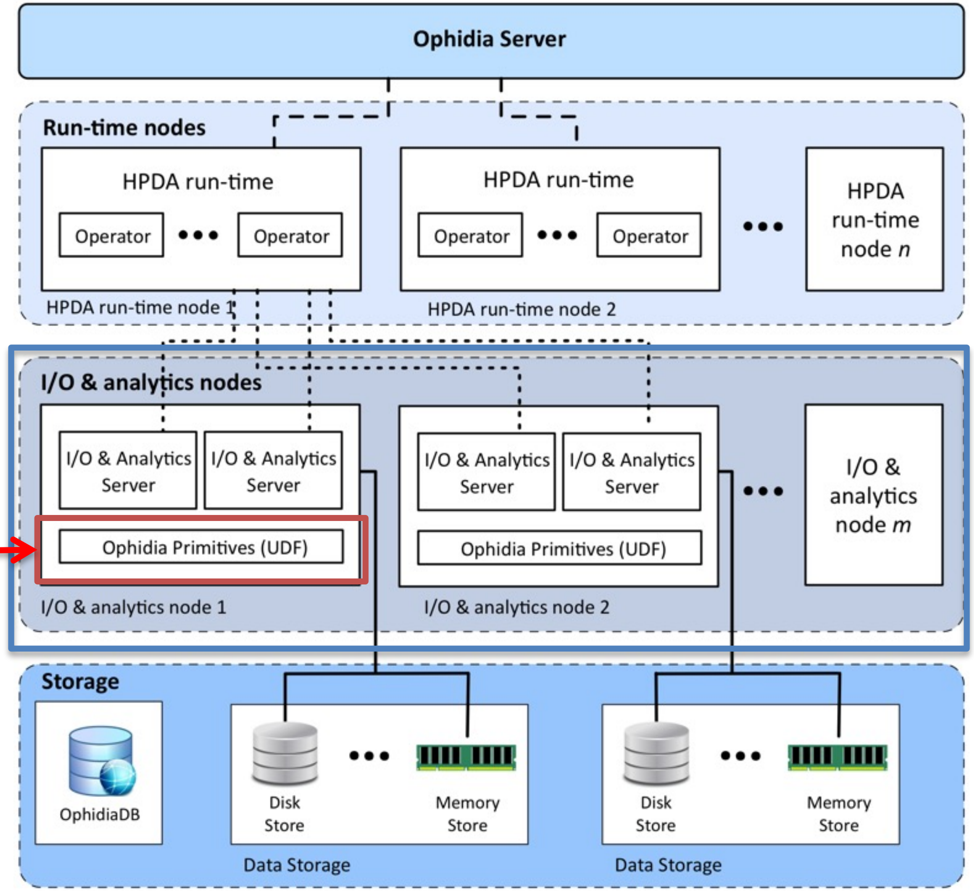
*Multiple **I/O & analytics nodes** execute one or more servers*

*Native **in-memory** analytics & I/O **engine** for **n-dimensional arrays***

Handles also I/O with NetCDF files, access and management of datacubes

*Servers run the (binary) array-based **Ophidia primitives** (UDF)*

Servers can transparently interface to different storage back-ends



D. Elia, S. Fiore, A. D'Anca, C. Palazzo, I. Foster, D. N. Williams, G. Aloisio (2016). "An in-memory based framework for scientific data analytics". In Proc. of the ACM Int. Conference on Computing Frontiers (CF '16), pp. 424-429.

Ophidia array-based primitives

Ophidia provides a **wide set of array-based primitives** (around 100) to perform:

- data summarisation, sub-setting, predicates evaluation, statistical analysis, array concatenation, algebraic expression, regression, etc.

Primitives come as plugins (UDF) and are applied on a single datacube chunk (fragment)

Primitives can be nested to get more complex functionalities

New primitives can be easily integrated as additional plugins

oph_apply operator to run any primitive on a datacube

*oph_apply(oph_predicate(measure,'**x-298.15**','**>0**','**1**','**0**'))*

Ophidia Primitives documentation: http://ophidia.cmcc.it/documentation/users/primitives/index.html

Array-based primitives: nesting support

oph_boxplot(oph_subarray(oph_uncompress(measure), 1,18))

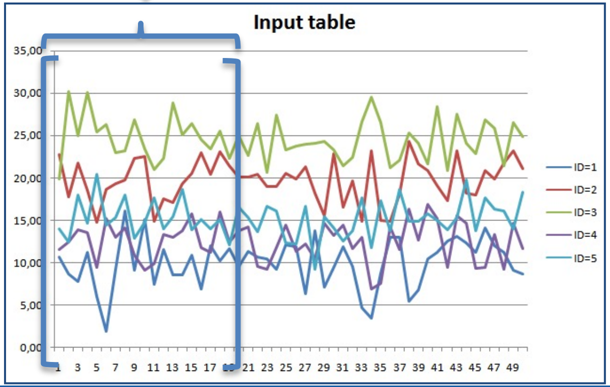
Single chunk or fragment (input)

INPUT TABLE 5 tuples x 50 elements										
ID	**MEASURE**									
1	10,73	8,66	7,83	11,20	6,02	1,95	...	16,11	...	8,70
2	22,85	17,84	21,82	18,57	14,81	18,71	...	19,83	...	21,13
3	19,89	30,17	24,95	30,07	25,40	26,31	...	23,18	...	24,82
4	11,60	12,49	13,91	13,53	9,48	15,27	...	14,17	...	11,66
5	13,94	12,43	17,95	14,70	20,41	14,46	...	18,00	...	18,30

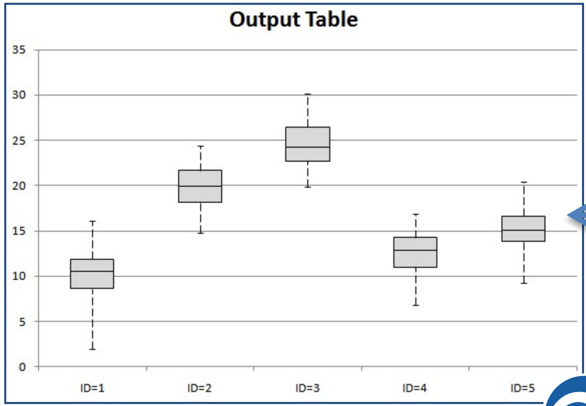
subarray(measure, 1,18)

Single chunk or fragment (output)

OUTPUT TABLE 5 tuples x 5 elements (summary)					
ID	**MEASURE**				
1	1,95	8,64	10,47	11,87	16,11
2	14,81	18,14	19,93	21,66	24,35
3	19,89	22,74	24,24	26,45	30,17
4	6,87	10,99	12,85	14,28	16,93
5	9,23	13,87	15,05	16,61	20,41



Scientific representation



Ophidia architecture: HPDA runtime layer

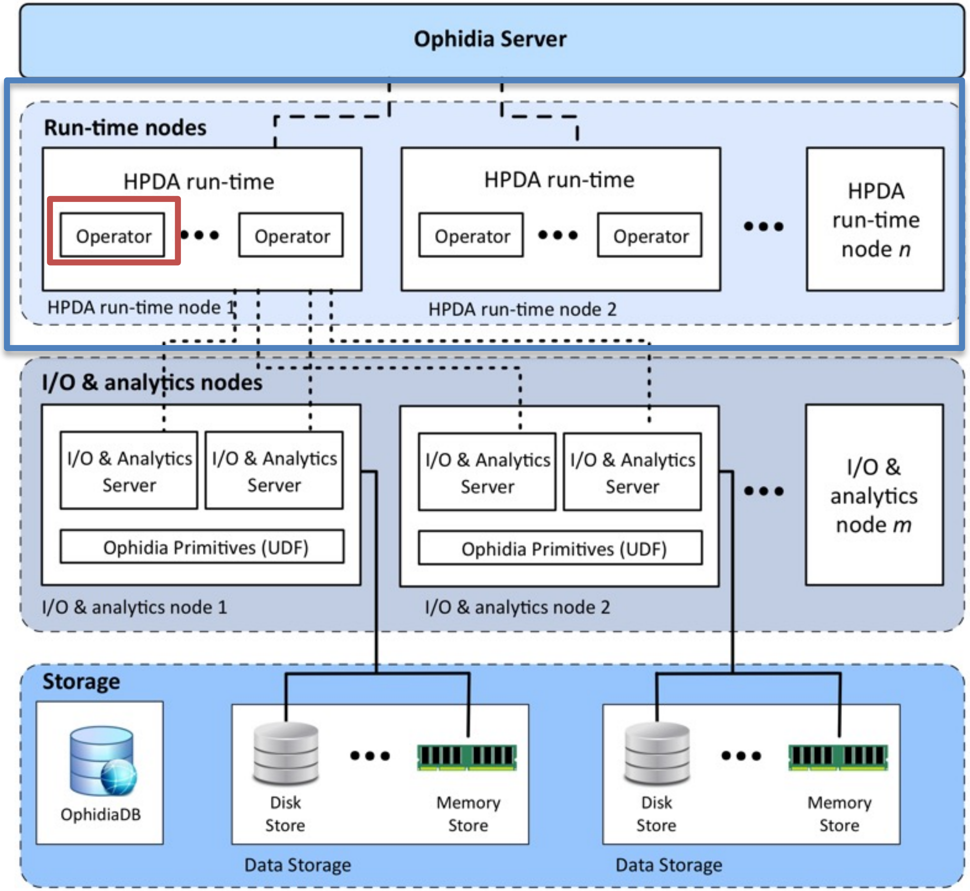
*The Ophidia HPDA runtime system can be executed with **multiple processes/threads** and **distributed over multiple nodes***

*Runtime defines a **multi-level parallel execution model:***

- *Datacube-level (HTC-based)*
- *Fragment-level (HPC-based: MPI+X)*

*Provides the environment for the execution of **parallel** MPI/Pthread-based **operators***

*Operators interact with the I/O & analytics servers to manipulate the entire set of fragments associated to a **whole datacube***



D. Elia, S. Fiore and G. Aloisio, "Towards HPC and Big Data Analytics Convergence: Design and Experimental Evaluation of a HPDA Framework for eScience at Scale," in IEEE Access, vol. 9, pp. 73307-73326, 2021

Ophidia operators

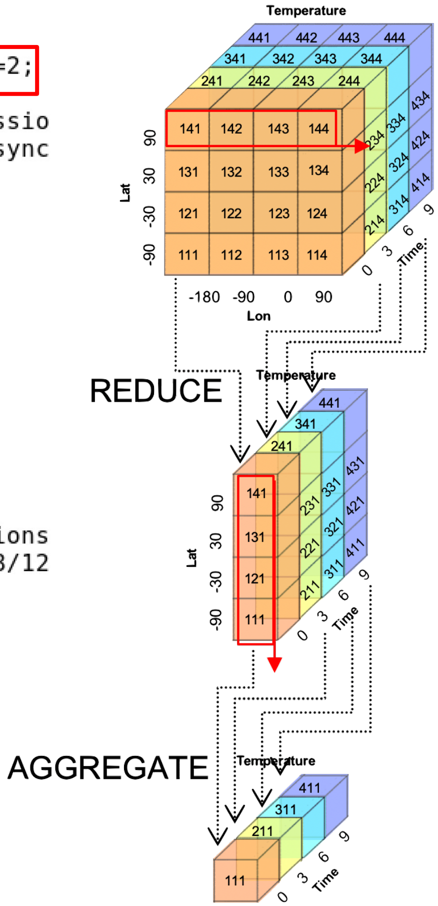
CLASS	PROCESSING TYPE	OPERATOR(S)
I/O	Parallel	OPH_IMPORTNC, OPH_EXPORTNC, OPH_CONCATNC, OPH_RANDUCUBE
Time series processing	Parallel	OPH_APPLY
Datacube reduction	Parallel	OPH_REDUCE, OPH_REDUCE2, OPH_AGGREGATE
Datacube subsetting	Parallel	OPH_SUBSET
Datacube combination	Parallel	OPH_INTERCUBE, OPH_MERGECUBES
Datacube structure manipulation	Parallel	OPH_SPLIT, OPH_MERGE, OPH_ROLLUP, OPH_DRILLDOWN, OPH_PERMUTE
Datacube/file system management	Sequential	OPH_DELETE, OPH_FOLDER, OPH_FS
Metadata management	Sequential	OPH_METADATA, OPH_CUBEIO, OPH_CUBESCHEMA
Datacube exploration	Sequential	OPH_EXPLORECUBE, OPH_EXPLORENC

About 50 operators for data and metadata processing

Ophidia operators documentation: http://ophidia.cmcc.it/documentation/users/operators/index.html

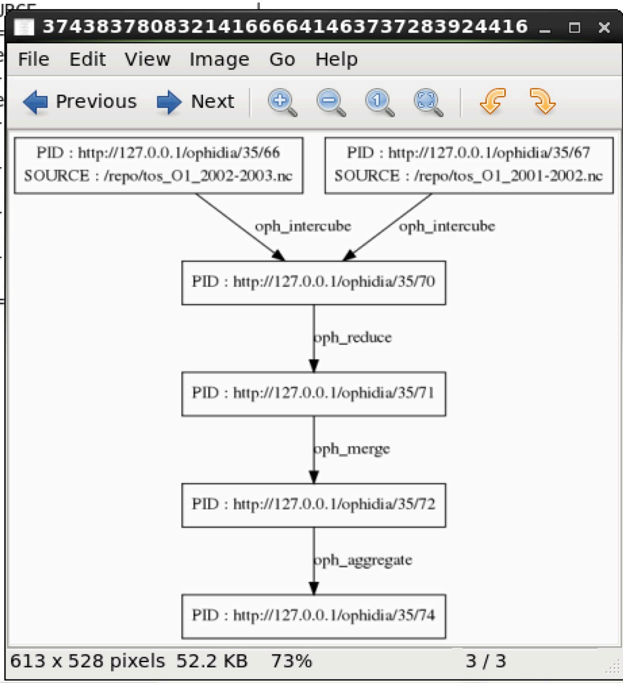
“data” operators

```
[12..3289] >> oph_reduce cube=http://127.0.0.1/ophidia/418/12717;operation=avg;ncores=2;nthreads=2;
[Request]:
operator=oph_reduce;cube=http://127.0.0.1/ophidia/418/12717;operation=avg;ncores=2;nthreads=2;sessio
nid=http://127.0.0.1/ophidia/sessions/127028404128222463341617004437753289/experiment;exec_mode=sync
;cwd=/;cdd=/;host_partition=auto;

[JobID]:
http://127.0.0.1/ophidia/sessions/127028404128222463341617004437753289/experiment?239#582

[Response]:
Output Cube
-----------
http://127.0.0.1/ophidia/418/12722

Execution time: 0.35 seconds
[12..3289] >> oph_aggregate operation=avg;ncores=2;nthreads=2;
[Request]:
operator=oph_aggregate;operation=avg;ncores=2;nthreads=2;sessionid=http://127.0.0.1/ophidia/sessions
/127028404128222463341617004437753289/experiment;exec_mode=sync;cube=http://127.0.0.1/ophidia/418/12
722;cwd=/;cdd=/;host_partition=auto;

[JobID]:
http://127.0.0.1/ophidia/sessions/127028404128222463341617004437753289/experiment?240#584

[Response]:
Output Cube
-----------
http://127.0.0.1/ophidia/418/12723

Execution time: 0.17 seconds
```



“metadata” operators

```
[37..4416] >> oph_cubeio
[Request]:
operator=oph_cubeio;sessionid=http://127.0.0.1/ophidia/sessions/374383780832141666641463737283924416/experiment;exec_mode=sync;ncores=1;cube=http://127.0.0.1/ophidia/35/74;cwd=/
;

[JobID]:
http://127.0.0.1/ophidia/sessions/374383780832141666641463737283924416/experiment?82#176

[Response]:
Cube Provenance
---------------

+=================================================================+===============+================================+=========================+
| INPUT CUBE                                                      | OPERATION     | OUTPUT CUBE                    | SOURCE
+=================================================================+===============+================================+=====
|                                                                 | ROOT          | http://127.0.0.1/ophidia/35/66 | /re
|-----------------------------------------------------------------|---------------|--------------------------------|----
|                                                                 | ROOT          | http://127.0.0.1/ophidia/35/67 | /re
|-----------------------------------------------------------------|---------------|--------------------------------|----
| http://127.0.0.1/ophidia/35/66 - http://127.0.0.1/ophidia/35/67 | oph_intercube | http://127.0.0.1/ophidia/35/70 |
|-----------------------------------------------------------------|---------------|--------------------------------|----
| http://127.0.0.1/ophidia/35/70                                  | oph_reduce    | http://127.0.0.1/ophidia/35/71 |
|-----------------------------------------------------------------|---------------|--------------------------------|----
| http://127.0.0.1/ophidia/35/71                                  | oph_merge     | http://127.0.0.1/ophidia/35/72 |
|-----------------------------------------------------------------|---------------|--------------------------------|----
| http://127.0.0.1/ophidia/35/72                                  | oph_aggregate | http://127.0.0.1/ophidia/35/74 |
+=================================================================+===============+================================+====

Cube Provenance Graph
---------------------

Directed Graph DOT string :
digraph DG {
        node    [shape=box]

        0       [label="PID : http://127.0.0.1/ophidia/35/74\n"]
        1       [label="PID : http://127.0.0.1/ophidia/35/72\n"]
        2       [label="PID : http://127.0.0.1/ophidia/35/71\n"]
        3       [label="PID : http://127.0.0.1/ophidia/35/70\n"]
        4       [label="PID : http://127.0.0.1/ophidia/35/66\nSOURCE : /repo/tos_O1_2002-2003.nc\n"]
        5       [label="PID : http://127.0.0.1/ophidia/35/67\nSOURCE : /repo/tos_O1_2001-2002.nc\n"]

        1->0    [label="oph_aggregate"]
        2->1    [label="oph_merge"]
```



Ophidia architecture: front-end layer

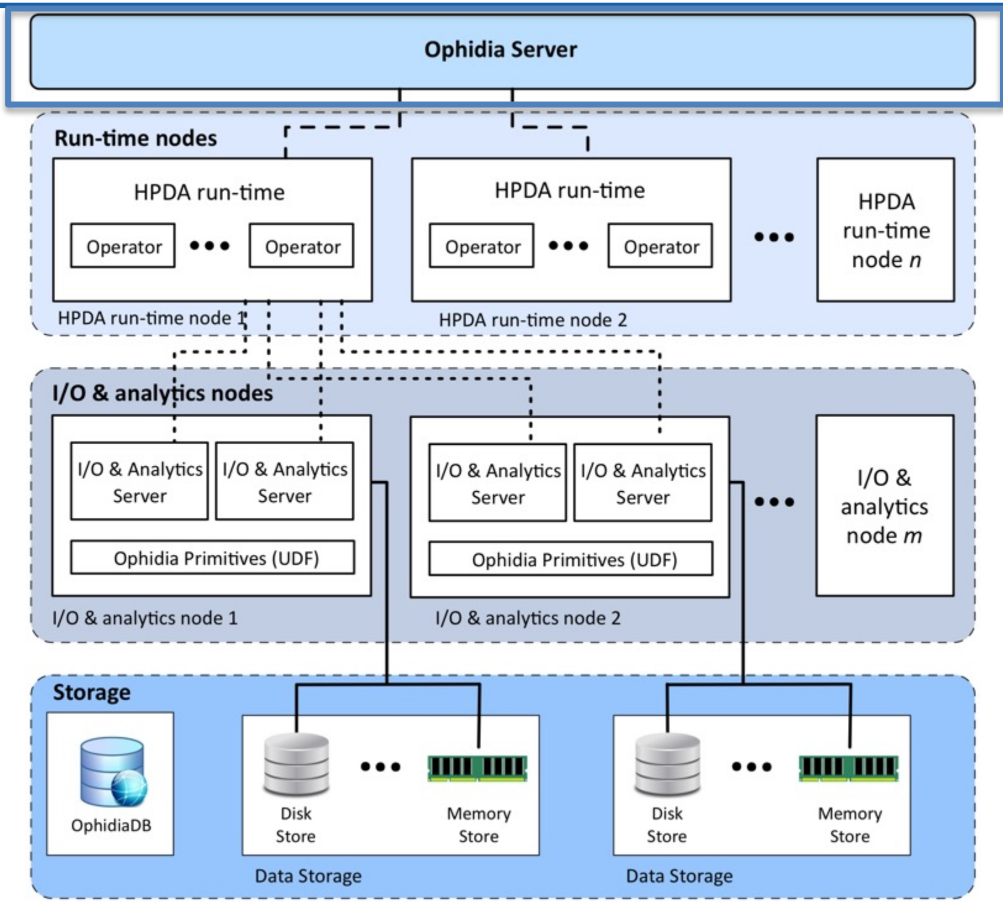
*The **Ophidia Server** is the **multi-interface** server front-end*

*Manages user **authN/authZ, sessions** and enables server-side computation*

*Handles **single task** and **workflows** execution and monitors their execution*

Remote interactions with:

- *the Ophidia terminal CLI*
- *PyOphidia Python API*
- *WPS clients*



C. Palazzo, A. Mariello, S. Fiore, A. D'Anca, D. Elia, D. N. Williams, G. Aloisio, "A Workflow-Enabled Big Data Analytics Software Stack for eScience", HPCS 2015, pp. 545-552

On-demand deployment on HPC infrastructures

Target environment: *HPC cluster*

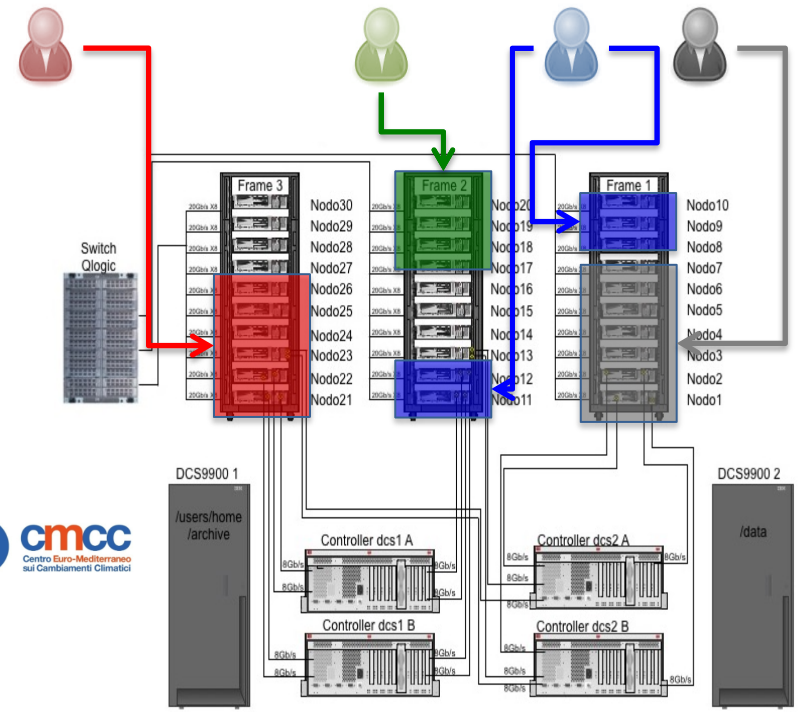
On-demand deployment of I/O & analytics servers

- `oph_cluster action=deploy;nhost=64;cluster_name=new;`
- `oph_cluster action=undeploy;cluster_name=new;`

Transparent interaction with scheduling systems

Zeus SuperComputer at CMCC: 1.2 PetaFlops, 348 nodes

Multiple isolated instances can be deployed simultaneously by different teams/users

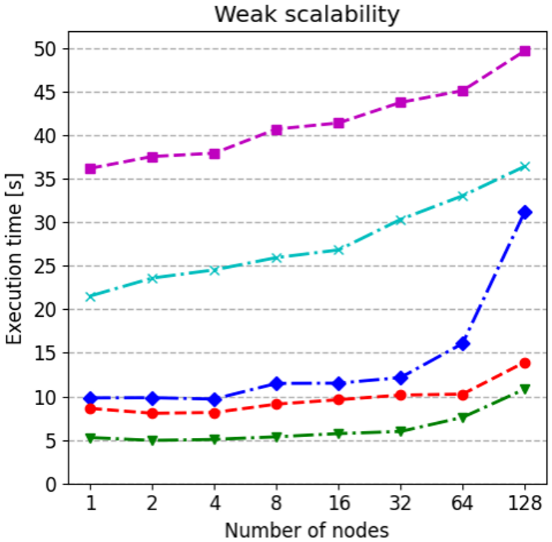


Ophidia HPDA framework benchmark

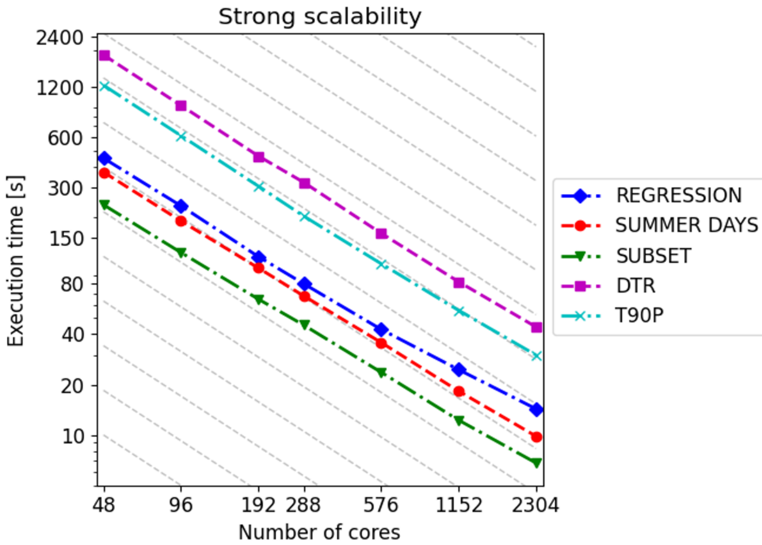
Goal: *benchmarking, tuning and optimisation over a large-scale HPC machine of the Ophidia HPDA framework*

Evaluate the scalability of Ophidia analytics kernels on a few thousands of cores:

- various **strong** and **weak scalability** tests run
- **good scalability** in most the cases until **3k cores**



Data size per node: 67GiB



Data size fixed: 3.2TiB

PRACE

We acknowledge PRACE for awarding access to MareNostrum 4 at Barcelona Supercomputing Center (BSC), Spain and the support provided by BSC (PRACE resources for CoE, in the context of ESiWACE).

BSC Barcelona Supercomputing Center
Centro Nacional de Supercomputación

D. Elia, S. Fiore and G. Aloisio, "Towards HPC and Big Data Analytics Convergence: Design and Experimental Evaluation of a HPDA Framework for eScience at Scale," in IEEE Access, vol. 9, pp. 73307-73326, 2021

Session outline

Introduction to HPDA and data challenges in eScience

Overview of the Ophidia HPDA framework

Ophidia core concepts: architecture, storage model, operators and primitives, terminal and deployment

Ophidia Python bindings: PyOphidia

DEMO: Introduction to PyOphidia

HANDS-ON: Data analytics examples with PyOphidia

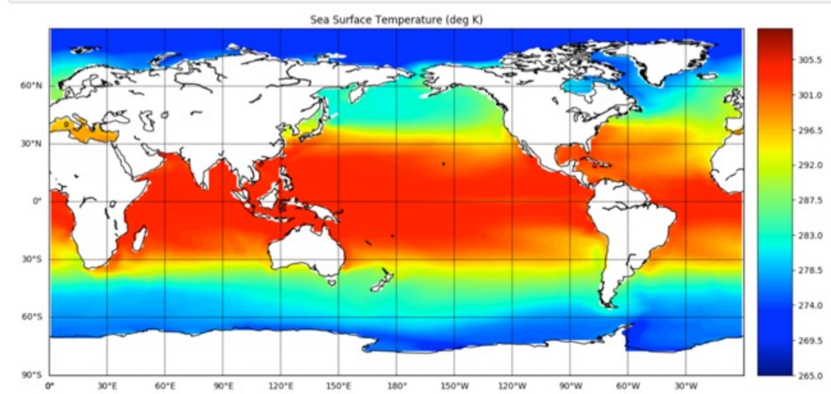
Programmatic support for data science applications

PyOphidia is a Python module to interact with the Ophidia framework.

It provides a programmatic access to Ophidia features, allowing:

- *Submission of commands to the Ophidia Server and retrieval of the results*
- *Management of (remote) data objects in the form of datacubes*
- *Easy exploitation from Jupyter Notebooks and integration with other Python modules*

```python
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')
```

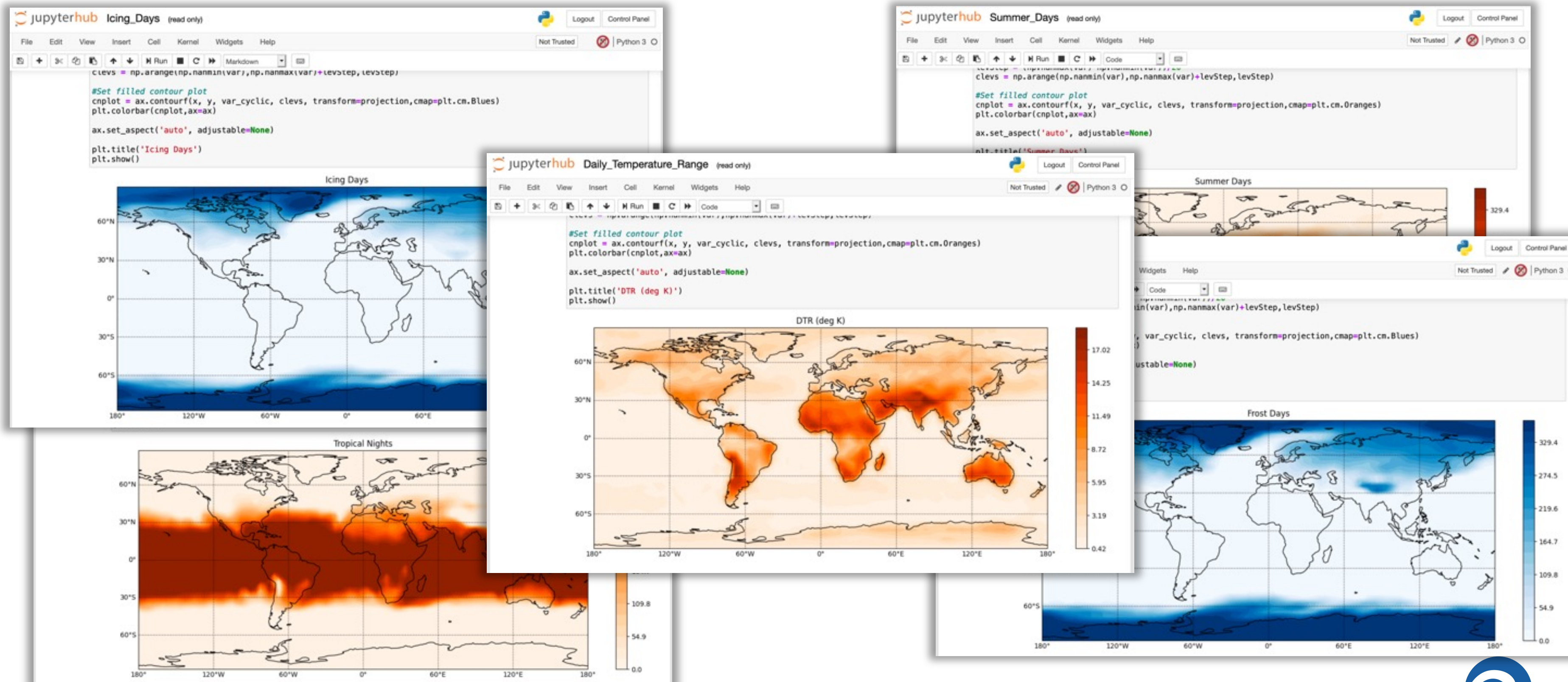


Export result to NetCDF file

```python
]: mycube3.exportnc2(output_path='/home/' + cube.Cube.client.username,export_metadata='yes')
```

Interactive climate data analytics

PyOphidia can be combined with other Python libraries (e.g., cartopy, matplotlib) and Notebooks for interactive prototyping, computation and visualisation of climate indices



What have we learned so far?

Joining HPC and data analytics is an enabling factor for scientific applications

Challenges for efficient climate (scientific) data management and analytics should be addressed: novel and efficient software solution are required

Overview of the Ophidia HPDA framework main aspects and how it addresses data analytics challenges for scientific analysis

- *Datacube abstraction for multi-dimensional scientific (climate) data*
- *Scalable architecture, data distribution, parallel operators*

PyOphidia Python module provides a high-level interface for parallel data management and analysis abstracting from the infrastructure complexity

Next: Demo and hands-on with PyOphidia

References and further readings

- *D. A. Reed and J. Dongarra. (2015). Exascale computing and big data. Commun. ACM 58, 7 (July 2015), 56–68.*
- *Asch, M., et al. (2018). Big data and extreme-scale computing: Pathways to convergence-toward a shaping strategy for a future software and data ecosystem for scientific inquiry. Int. J. High Perform. Comput. Appl., 32(4), 435-479.*
- *S. Fiore, et al. (2013). Ophidia: Toward Big Data Analytics for eScience. ICCS 2013, volume 18 of Procedia Computer Science, pp. 2376-2385.*
- *S. Fiore, et al. (2014). "Ophidia: A Full Software Stack for Scientific Data Analytics", proc. of the 2014 Int. Conference on High Performance Computing & Simulation (HPCS 2014), pp. 343-350.*
- *S. Fiore, D. Elia, C. Palazzo, F. Antonio, A. D'Anca, I. Foster and G. Aloisio (2019), "Towards High Performance Data Analytics for Climate Change", ISC High Performance 2019. Lecture Notes in Computer Science, vol. 11887, pp. 240-257.*
- *D. Elia, S. Fiore and G. Aloisio, "Towards HPC and Big Data Analytics Convergence: Design and Experimental Evaluation of a HPDA Framework for eScience at Scale," in IEEE Access, vol. 9, pp. 73307-73326, 2021*
- *D. Elia, et al. (2016). "An in-memory based framework for scientific data analytics". In Proc. of the ACM Int. Conference on Computing Frontiers (CF '16), pp. 424-429.*
- *C. Palazzo, et al. (2015), "A Workflow-Enabled Big Data Analytics Software Stack for eScience", HPCS 2015, pp. 545-552*

Questions?

__ESiWACE2__ has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 823988

More about Ophidia?

Ophidia website: *http://ophidia.cmcc.it*

GitHub repo: *https://github.com/OphidiaBigData*

Contact: *ophidia-info [at] cmcc.it*

Twitter channel: *https://twitter.com/OphidiaBigData*