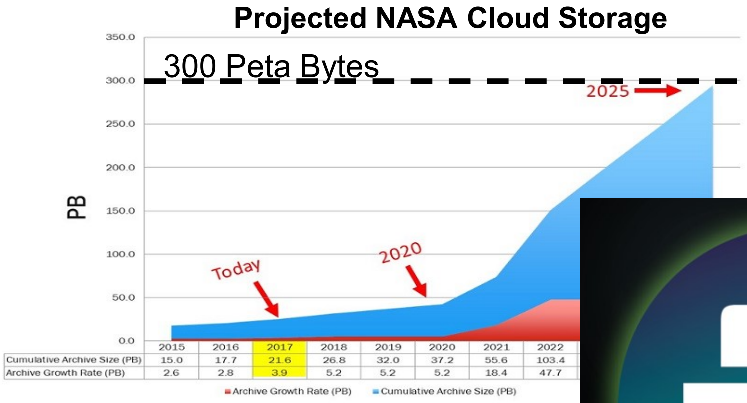
Pangeo community platform for scientific data processing

OMP 2019/04/02, Guillaume Eynard-Bontemps, CNES/Pangeo

Pangeo community goals and motivation



	2015	2016	2017	2018	2019	2020	2021	2022
Cumulative Archive Size (PB)	15.0	17.7	21.6	26.8	32.0	37.2	55.6	103.4
Archive Growth Rate (PB)	2.6	2.8	3.9	5.2	5.2	5.2	18.4	47.7

Problems

- Data volume crisis in (geo)sciences
- Software multiplication, non reproducibility
- Many copies of the same datasets
- Local vs HPC vs Cloud
- Technology gap: industry vs academia

Mission

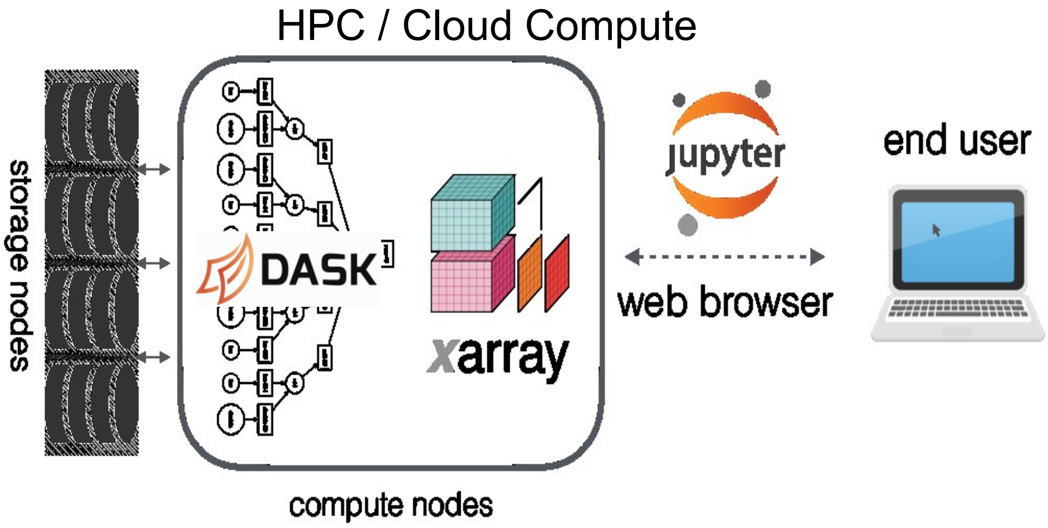
To cultivate an ecosystem in wh
the next generation of open-sou
analysis tools for the geosciences can be developed, distributed, and sustained.

als/vision

- oster collaboration around the open source Scientific Python ecosystem:
  - open and collaborative development
  - Welcoming and inclusive culture
- Support the development with domain-specific (geo)science and transverse packages
- Improve scalability of these tools to handle gigabytes to petabytescale datasets

Pangeo ecosystem

- Set of tools that will facilitate science at all scales
- Platform agnostic
- The core of the Pangeo ecosystem includes:
  - **Xarray** (data-model and toolkit for working with N-dimensional labeled arrays)
  - **Dask** (parallel computing)
  - **Jupyter** (interactive computing)
- Extensible: Series of 3rd party packages that build on top of core libraries
- Flexible: Individual components may be swapped in/out



Examples of 3rd party packages in the Pangeo Ecosystem:

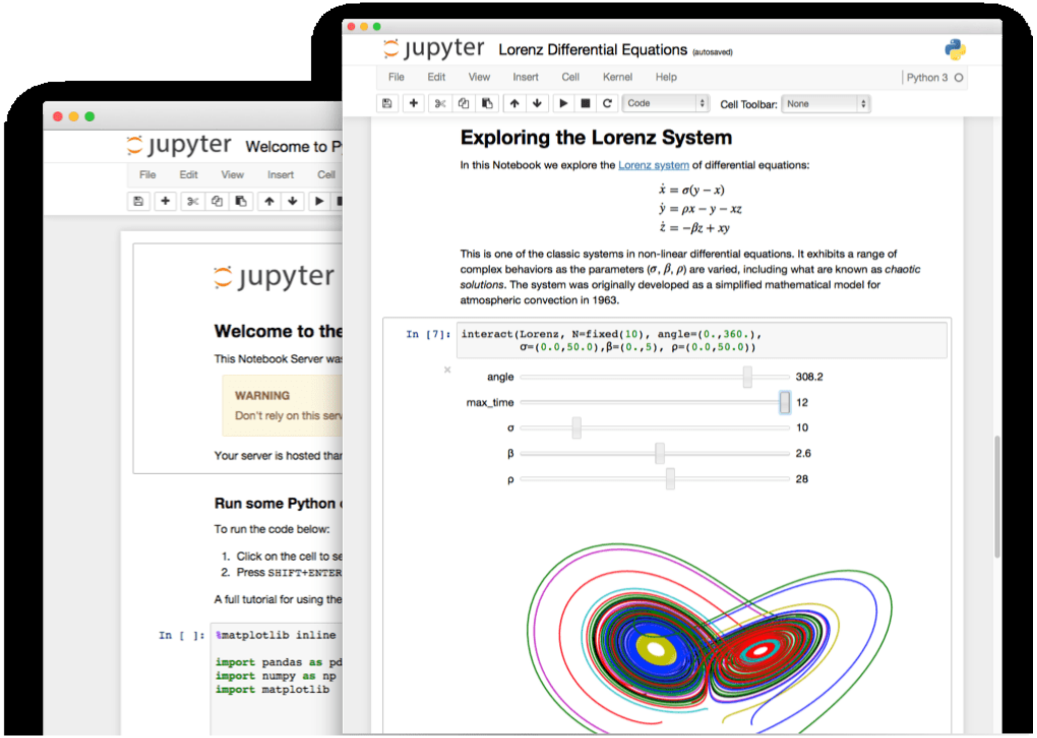
- Data discovery
- Regridding and GIS
- Vector calculus
- Signal processing
- Thermodynamics

Jupyter

- ***"The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and explanatory text. Uses include: data cleaning and transformation, numerical simulation, statistical modeling, machine learning and much more."***

- **Originally "Python-centric" but has been expanded to include over 40 popular programming languages (e.g. Julia and R)**

- **Check it out at: http://jupyter.org**



jupyter Welcome to P

jupyter

Welcome to the

This Notebook Server wa

WARNING

Don't rely on this serv

Your server is hosted thar

Run some Python

To run the code below:

1. Click on the cell to se
2. Press SHIFT+ENTER

A full tutorial for using the

```
In [ ]: %matplotlib inline

        import pandas as pd
        import numpy as np
        import matplotlib
```

jupyter Lorenz Differential Equations (autosaved)

Exploring the Lorenz System

In this Notebook we explore the Lorenz system of differential equations:

$$\dot{x} = \sigma(y - x)$$
$$\dot{y} = \rho x - y - xz$$
$$\dot{z} = -\beta z + xy$$

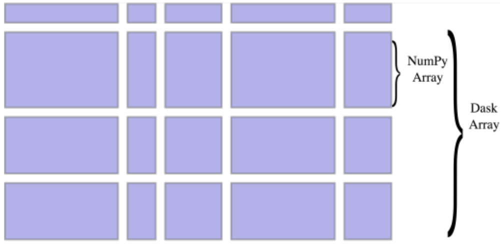
This is one of the classic systems in non-linear differential equations. It exhibits a range of complex behaviors as the parameters (σ, β, ρ) are varied, including what are known as *chaotic solutions*. The system was originally developed as a simplified mathematical model for atmospheric convection in 1963.

```
In [7]: interact(Lorenz, N=fixed(10), angle=(0.,360.),
                 σ=(0.0,50.0),β=(0.,5), ρ=(0.0,50.0))
```

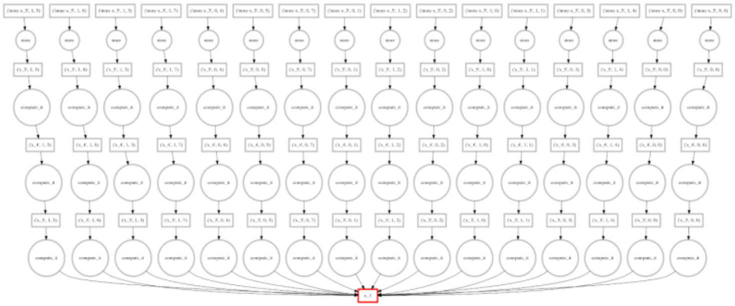
angle 308.2
max_time 12
σ 10
β 2.6
ρ 28

Dask

- **Dask is a flexible parallel computing library for analytic computing**
- **Parallel arrays allow us to seamlessly scale serial programs and workflows**
- **Dynamic task scheduling is optimized for computation**
- **Can be utilized on a single machine or a cluster of machines**



Dask arrays coordinate many NumPy arrays arranged into a grid. These NumPy arrays may live on disk or on other machines.

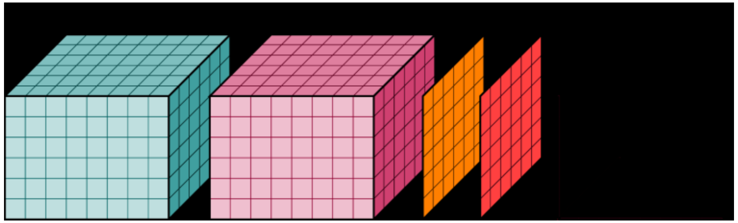


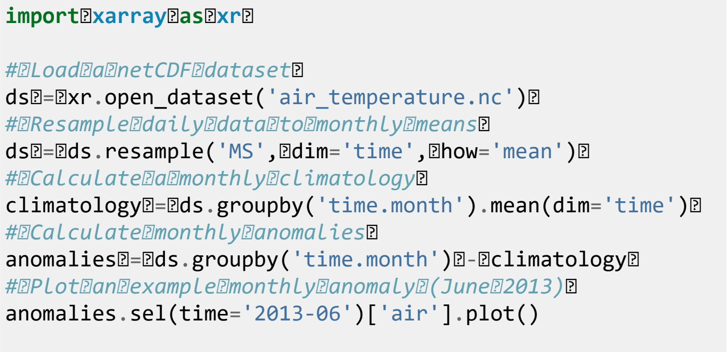
Example of a Dask task graph for a simple, embarrassingly parallel reduction operation.

Source: Dask documentation

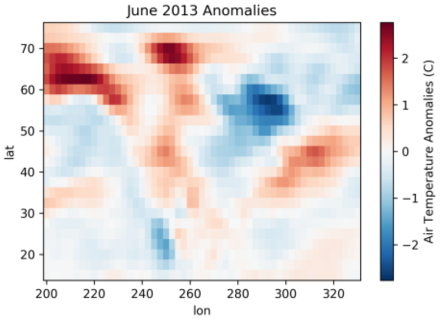
Xarray

- **N-D labeled arrays and datasets in Python**
- **Data model emulates the Common Data Model (e.g. NetCDF)**
- **Key features:**
  - Label-based indexing
  - Interoperability with core scientific Python packages
  - Parallel computation using Dask
  - Wide range of input/output options
  - Robust data analysis and manipulation toolkit





```python
import xarray as xr

# Load a netCDF dataset
ds = xr.open_dataset('air_temperature.nc')
# Resample daily data to monthly means
ds = ds.resample('MS', dim='time', how='mean')
# Calculate a monthly climatology
climatology = ds.groupby('time.month').mean(dim='time')
# Calculate monthly anomalies
anomalies = ds.groupby('time.month') - climatology
# Plot an example monthly anomaly (June 2013)
anomalies.sel(time='2013-06')['air'].plot()
```



Hoyer, S. & Hamman, J., (2017). Xarray: N-D labeled Arrays and Datasets in Python. Journal of Open Research Software. 5(1), p.10. DOI: http://doi.org/10.5334/jors.148

Pangeo diversity

BUILD YOUR OWN PANGEO

Storage Formats	HDF	OPeNDAP	Cloud Optimized COG/Zarr/Parquet/etc.
ND-Arrays	NumPy	DASK	More coming…
Data Models	xarray	Iris	pandas $y_it = \beta' x_{it} + \mu_i + \epsilon_{it}$
Processing Mode	jupyter Interactive	Batch	Serverless λ
Compute Platform	HPC	Cloud Google Cloud Platform	Local

Pangeo HPC deployment

NCAR's Cheyenne Super Computer

- 145,152 processors
- 52.7 Pb of parallel disk storage
- InfiniBand high-speed interconnect

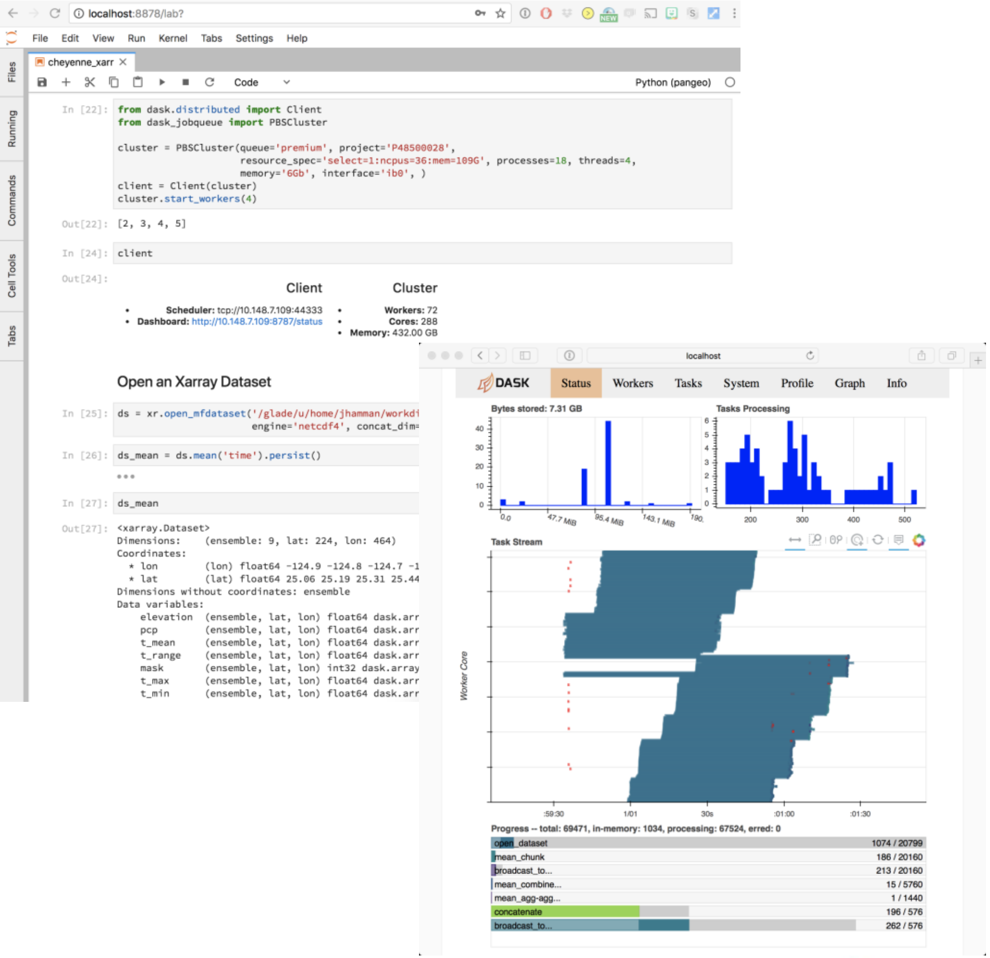
dask.distributed: parallel workers across many HPC nodes

Xarray for computational toolkit and I/O

Jupyter notebooks for interactive computing

New tools for deploying dask clusters on HPC

- e.g. **dask-jobqueue**[1]



```
In [22]: from dask.distributed import Client
         from dask_jobqueue import PBSCluster

         cluster = PBSCluster(queue='premium', project='P48500028',
                              resource_spec='select=1:ncpus=36:mem=109G', processes=18, threads=4,
                              memory='6Gb', interface='ib0', )
         client = Client(cluster)
         cluster.start_workers(4)

Out[22]: [2, 3, 4, 5]

In [24]: client

Out[24]:
```

Client	Cluster
Scheduler: tcp://10.148.7.109:44333	**Workers:** 72
Dashboard: http://10.148.7.109:8787/status	**Cores:** 288
	Memory: 432.00 GB

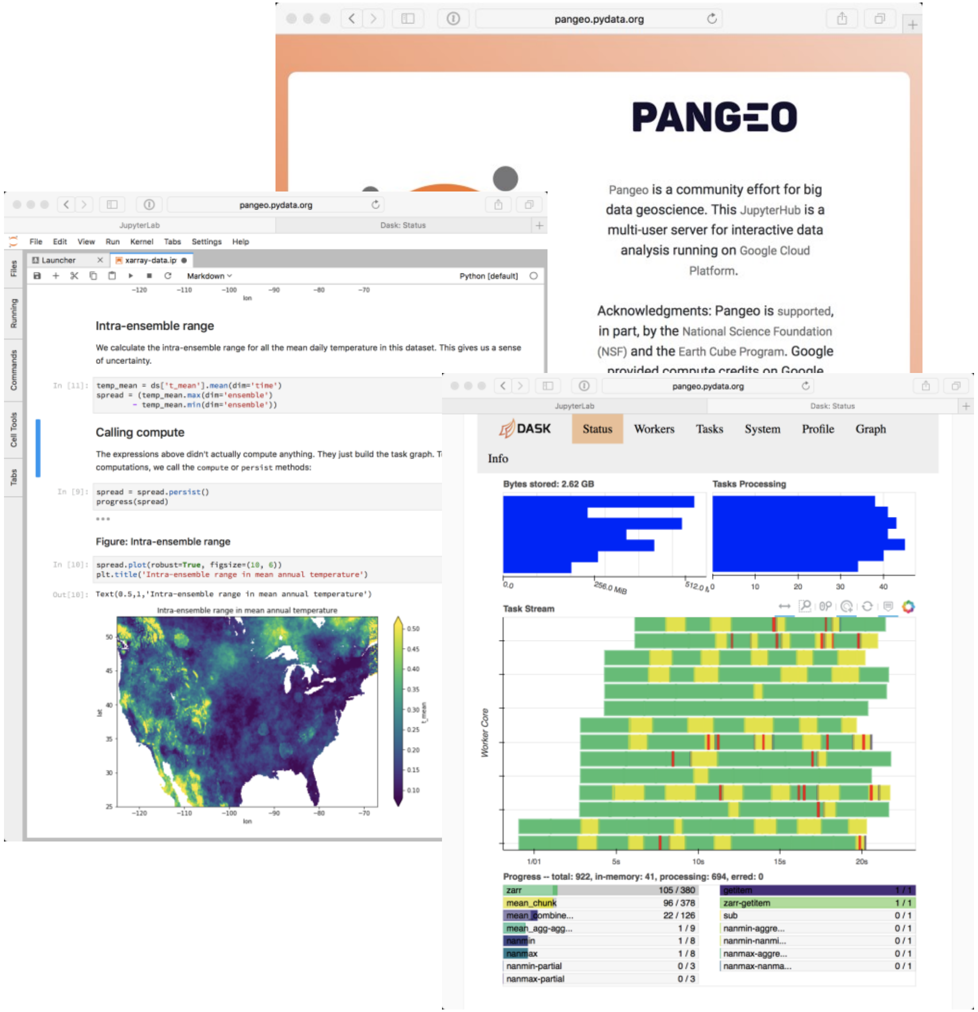
Open an Xarray Dataset

```
In [25]: ds = xr.open_mfdataset('/glade/u/home/jhamman/workdi
                                 engine='netcdf4', concat_dim=

In [26]: ds_mean = ds.mean('time').persist()

         •••

In [27]: ds_mean

Out[27]: <xarray.Dataset>
         Dimensions:    (ensemble: 9, lat: 224, lon: 464)
         Coordinates:
           * lon        (lon) float64 -124.9 -124.8 -124.7 -1
           * lat        (lat) float64 25.06 25.19 25.31 25.44
         Dimensions without coordinates: ensemble
         Data variables:
             elevation  (ensemble, lat, lon) float64 dask.arr
             pcp        (ensemble, lat, lon) float64 dask.arr
             t_mean     (ensemble, lat, lon) float64 dask.arr
             t_range    (ensemble, lat, lon) float64 dask.arr
             mask       (ensemble, lat, lon) int32 dask.array
             t_max      (ensemble, lat, lon) float64 dask.arr
             t_min      (ensemble, lat, lon) float64 dask.arr
```

```
Progress -- total: 69471, in-memory: 1034, processing: 67524, erred: 0
open_dataset          1074 / 20799
mean_chunk             186 / 20160
broadcast_to...        213 / 20160
mean_combine...         15 / 5760
mean_agg-agg...          1 / 1440
concatenate            196 / 576
broadcast_to...        262 / 576
```

[1] https://github.com/dask/dask-jobqueue

Pangeo public cloud deployment



Pangeo is a community effort for big data geoscience. This JupyterHub is a multi-user server for interactive data analysis running on Google Cloud Platform.

Acknowledgments: Pangeo is supported, in part, by the National Science Foundation (NSF) and the Earth Cube Program. Google provided compute credits on Google

Intra-ensemble range

We calculate the intra-ensemble range for all the mean daily temperature in this dataset. This gives us a sense of uncertainty.

```
In [11]: temp_mean = ds['t_mean'].mean(dim='time')
         spread = (temp_mean.max(dim='ensemble')
                 - temp_mean.min(dim='ensemble'))
```

Calling compute

The expressions above didn't actually compute anything. They just build the task graph. T computations, we call the compute or persist methods:

```
In [9]: spread = spread.persist()
        progress(spread)
```

Figure: Intra-ensemble range

```
In [10]: spread.plot(robust=True, figsize=(10, 6))
         plt.title('Intra-ensemble range in mean annual temperature')

Out[10]: Text(0.5,1,'Intra-ensemble range in mean annual temperature')
```

```
Progress -- total: 922, in-memory: 41, processing: 694, erred: 0
zarr              105 / 380     getitem             1 / 1
mean_chunk         96 / 378     zarr-getitem        1 / 1
mean_combine...    22 / 126     sub                 0 / 1
mean_agg-agg...     1 / 9       nanmin-aggre...     0 / 1
nanmin              1 / 8       nanmin-nanmi...     0 / 1
nanmax              1 / 8       nanmax-aggre...     0 / 1
nanmin-partial      0 / 3       nanmax-nanma...     0 / 1
nanmax-partial      0 / 3
```

hub.pangeo.io

pangeo.binder.io

JupyterHub/BinderHub running on the Google Cloud

- Kubernetes for both Jupyter and Dask-distributed
  - Dask-kubernetes
- Exploring/evaluating:
  - Cloud storage
  - User environment customization
  - Data discovery
- Kubernetes Helm-chart (github.com/pangeo-data/helm-chart)
- CI/CD with Hubploy and CircleCI
- Deployments exist on AWS and Azure.

Pangeo vs state of the art

Spark	vs	DASK
Mature		Less Mature
Robust		Pretty strong
JVM/Python		Python only
Query optimized		Science optimized
Collections & Dataframes		Collections, DF, Arrays, Futures…
Python overhead		Python only
For big tabular data		For science data
Hadoop/Cloud/HPC		Hadoop/Cloud/HPC

PANGEO VS y **databases** *(Rasdaman, SciDB…)*

I don't know much about Array DBs!!!

Laptop to cluster

Serverless

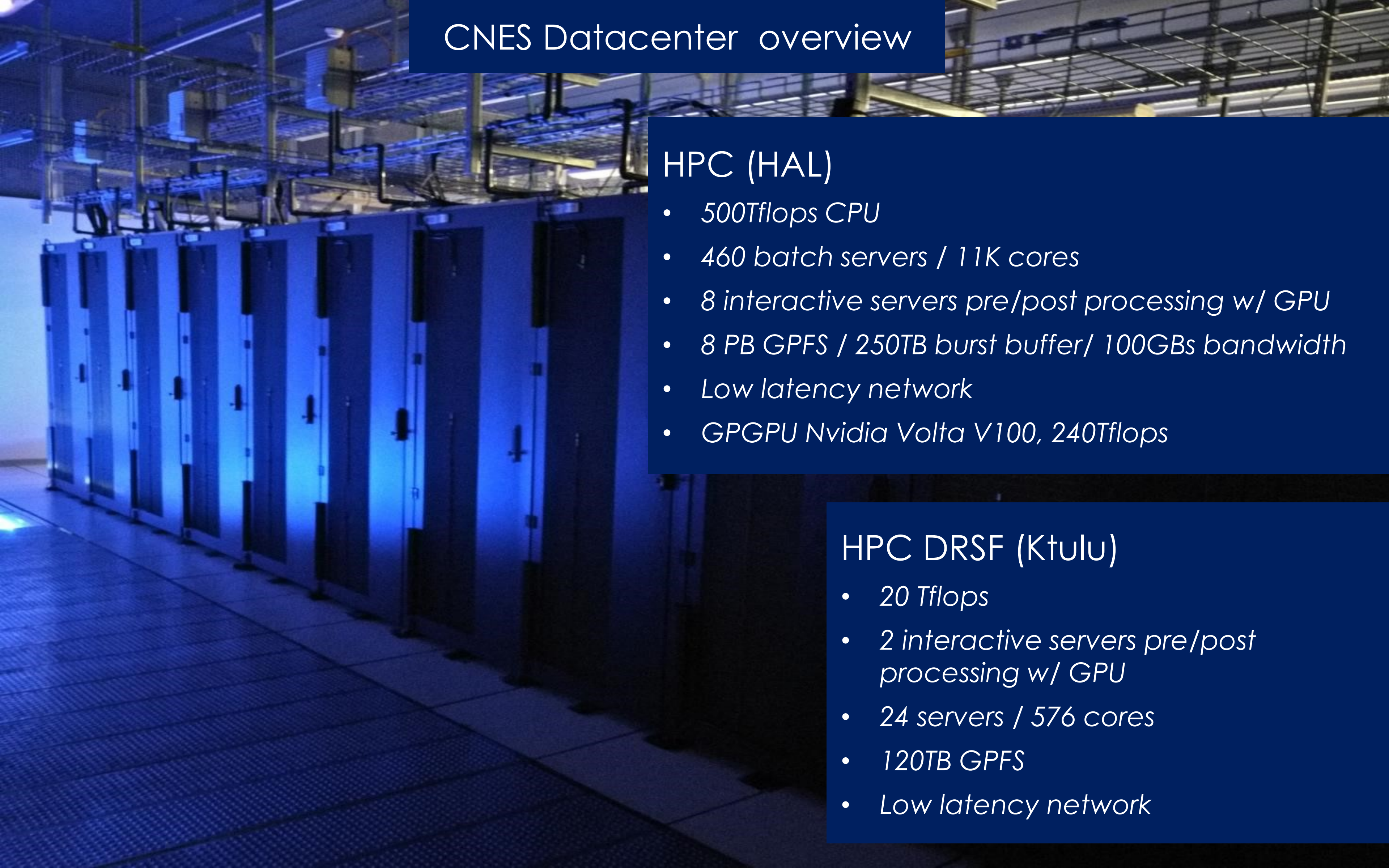
NetCDF/TIFF no ingestion

Scales with Dask

Python only

Can build array db with Pangeo (Open data Cube)

CNES Datacenter overview



HPC (HAL)

- *500Tflops CPU*
- *460 batch servers / 11K cores*
- *8 interactive servers pre/post processing w/ GPU*
- *8 PB GPFS / 250TB burst buffer/ 100GBs bandwidth*
- *Low latency network*
- *GPGPU Nvidia Volta V100, 240Tflops*

HPC DRSF (Ktulu)

- *20 Tflops*
- *2 interactive servers pre/post processing w/ GPU*
- *24 servers / 576 cores*
- *120TB GPFS*
- *Low latency network*

HPC usecases in CNES

Two main kinds of processing

Numerical simulation (HPC)

- Upstream phase, R&D
- Highly optimized technics
- Fine grain parallelism

Trends : multiscale, multiphysics

peps

Plateforme d'exploitation des produits Sentinel

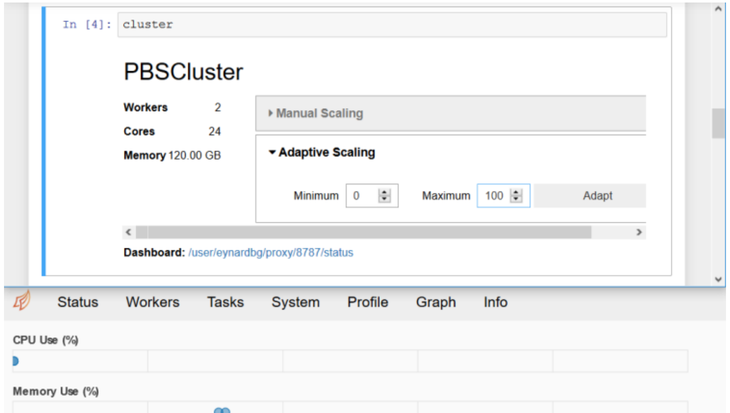
Data Processing (HTC)

- Downstream phase, operation
- Sensors data → scientific data
- Coarse grain parallelism

Trends : data volume explosion

Pangeo at CNES

- JupyterHub and notebooks for interactive computing
    - Hub on a VM with qsub access
    - Batchspawner, Wrapspawner
- dask.distributed: parallel workers across many HPC nodes
- Xarray for computational toolkit and I/O
- New tool for deploying dask clusters on HPC: **dask-jobqueue**
    - Start a cluster from a notebook
    - Interactive (or not) distributed computing
    - Auto scaling capabilities



```
In [4]: cluster
```

PBSCluster

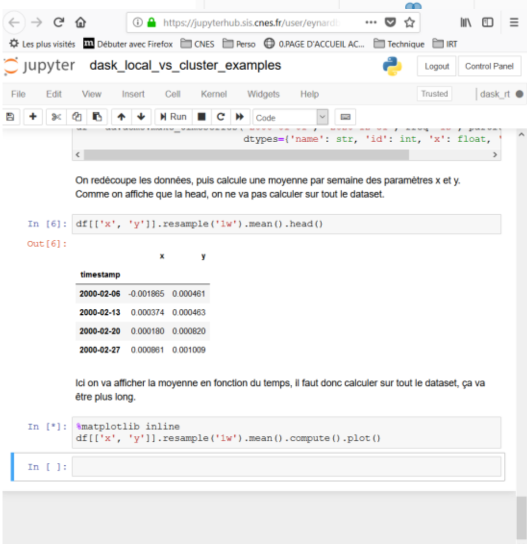
Workers	2
Cores	24
Memory	120.00 GB

▸ Manual Scaling

▾ Adaptive Scaling

Minimum 0 Maximum 100 Adapt

Dashboard: /user/eynardbg/proxy/8787/status



dask_local_vs_cluster_examples

```
                         dtypes={'name': str, 'id': int, 'x': float, '
```

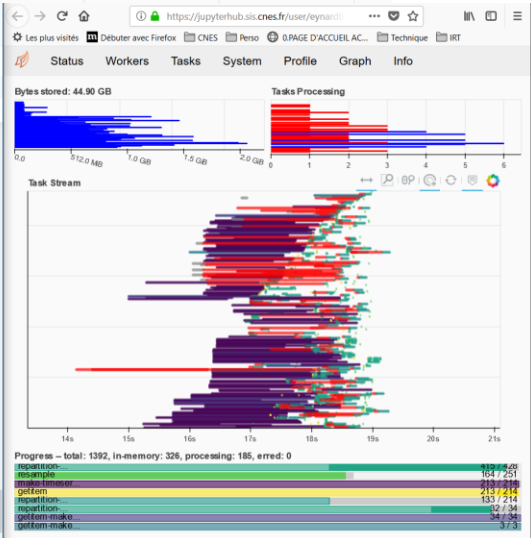
On redécoupe les données, puis calcule une moyenne par semaine des paramètres x et y. Comme on affiche que la head, on ne va pas calculer sur tout le dataset.

```
In [6]: df[['x', 'y']].resample('1w').mean().head()
Out[6]:
```

	x	y
timestamp		
2000-02-06	-0.001865	0.000461
2000-02-13	0.000374	0.000463
2000-02-20	0.000180	0.000820
2000-02-27	0.000861	0.001009

Ici on va afficher la moyenne en fonction du temps, il faut donc calculer sur tout le dataset, ça va être plus long.

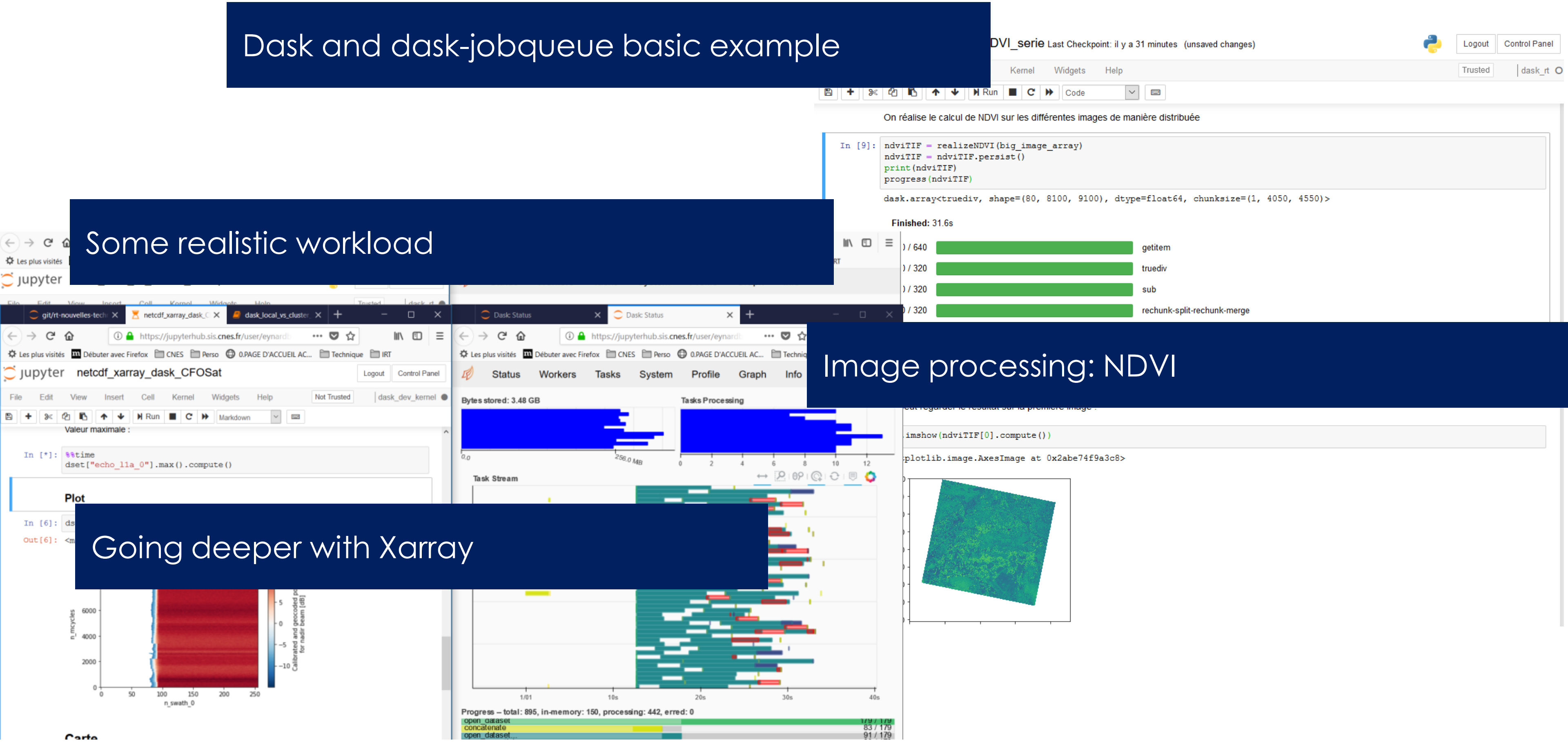
```
In [*]: %matplotlib inline
        df[['x', 'y']].resample('1w').mean().compute().plot()

In [ ]:
```



Progress – total: 1392, in-memory: 326, processing: 185, erred: 0

```
repartition-...     415 / 428
resample            164 / 251
make-timeser...     213 / 214
getitem             213 / 214
repartition-...     133 / 214
repartition-...      32 / 34
getitem-make...      34 / 34
getitem-make...       3 / 3
```

Demos @ CNES



Dask and dask-jobqueue basic example

Some realistic workload

Image processing: NDVI

Going deeper with Xarray

DVI_serie Last Checkpoint: il y a 31 minutes (unsaved changes)

On réalise le calcul de NDVI sur les différentes images de manière distribuée

```
In [9]: ndviTIF = realizeNDVI(big_image_array)
        ndviTIF = ndviTIF.persist()
        print(ndviTIF)
        progress(ndviTIF)

dask.array<truediv, shape=(80, 8100, 9100), dtype=float64, chunksize=(1, 4050, 4550)>
```

Finished: 31.6s

```
) / 640   getitem
) / 320   truediv
) / 320   sub
) / 320   rechunk-split-rechunk-merge
```

```
.imshow(ndviTIF[0].compute())

plotlib.image.AxesImage at 0x2abe74f9a3c8>
```

jupyter netcdf_xarray_dask_CFOSat

Valeur maximale :

```
In [*]: %%time
        dset["echo_l1a_0"].max().compute()
```

Plot

```
In [6]: ds
Out[6]: <m
```

Carte

Bytes stored: 3.48 GB

Tasks Processing

Task Stream

Progress – total: 895, in-memory: 150, processing: 442, erred: 0

```
open_dataset     179 / 179
concatenate       83 / 179
open_dataset...   91 / 179
```

Pangeo Cloud use

Geoscience use cases:
http://pangeo.io/use_cases/index.html

Astronomy with GAIA catalog:

https://github.com/pangeo-data/pangeo/issues/255#issuecomment-427186915

Image processing and visualization

https://medium.com/pangeo/cloud-native-geoprocessing-of-earth-observation-satellite-data-with-pangeo-997692d91ca2

Conclusion, Useful links

Conclusions

- Pangeo ecosystem greatlty facilitates distributed computing and data analysis at scale
- It changes ways of doing it too
- Non monolithic platform built on top of existing Scientific Python stack and new related packages
- Community is always here to help
- Dask more versatile and easy to use than Spark.

Next steps

- Broaden users and use cases at CNES
- Encourage people to get in touch with Pangeo community
- Work in cooperation with others (Ongoing with Ifremer and CLS on SWOT aval data processing)
- Get involved!!
- Pangeo french meeting ont May 23rd

Pangeo wesite and discussions:

https://pangeo.io

https://github.com/pangeo-data/pangeo/issues

https://medium.com/pangeo

Pangeo Example + Binder:

https://github.com/pangeo-data/pangeo-example-notebooks

http://binder.pangeo.io/v2/gh/pangeo-data/pangeo-example-notebooks/master

Dask jobqueue:

https://github.com/dask/dask-jobqueue

Dask simple examples:

https://github.com/dask/dask-examples

My email

Guillaume.Eynard-Bontemps@cnes.fr

Tutorial

Github and binder link :
https://github.com/guillaumeeb/pangeo-tutorial-agu-2018
https://binder.pangeo.io/v2/gh/guillaumeeb/pangeo-tutorial-agu-2018/some_fixes

Dask only tutorial:
https://github.com/mrocklin/pydata-nyc-2018-tutorial

Other binder resources :
https://github.com/pangeo-data/pangeo-example-notebooks
https://github.com/pangeo-data/pangeo_ocean_examples
https://github.com/dask/dask-examples