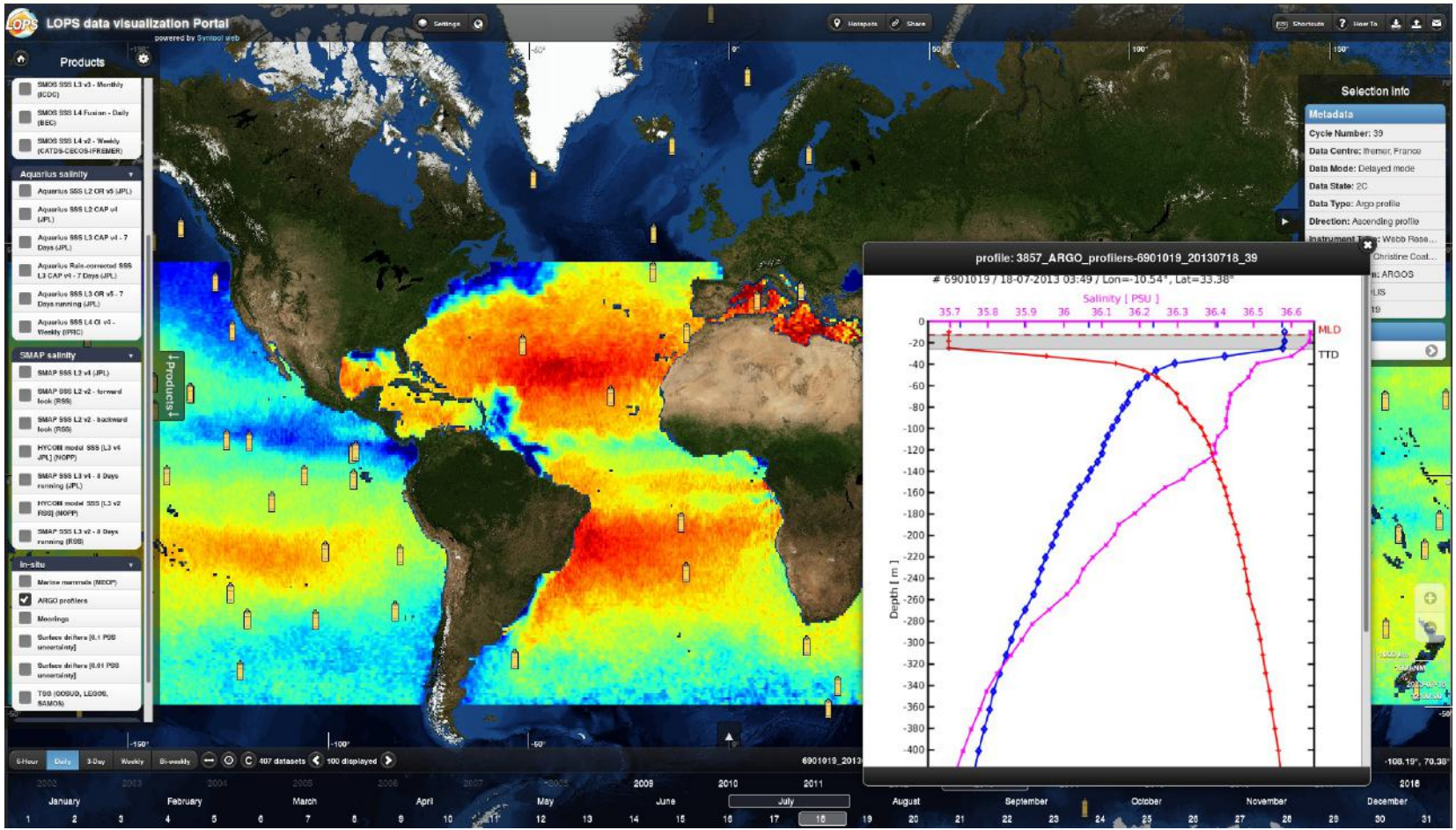
VISUALISATION - SYNTOOL

http://syntool.ifremer.fr



EUMETSAT

- Analyse visuelle, découverte
- Adresse données satellite sous la trace (L1/L2) + autres types
- Aspect temporel, colocalisation “basique”
- Open-source, développé/maintenu par ODL

- Statique (images prégénérées + json pour in situ) – pas de support WMS, intégration limitée avec autres services
- Lourd à administrer (processus d’ingestion), stockage important nécessaire
- Vite limitant en terme d’analyse scientifique : résultats statiques, pas de lien direct avec outils d’analyse

JUPYTER

jupyter

- http://jupyter.org/
- Python (but not only) in your web browser
- Embeds and mixcode, visualisation, explainations, equations in « notebooks »
- Growingly popular for interactive science
- Can run different languages (over 40)
- Can mix in some shell instructions
- Can be exported as html pages, pdf documents, .rst documents, LateX, python script
- Widgets for more interactivity, small task interfaces
- Complemented by **jupyterhub** which is single-user => allow multi-user access : a jupyter notebook server is spawned for each user

- Travail à distance
- Mise au point, analyse de données
- Scenarios d'analyse
- Combinaison de données
- Intégration de middleware pour analyse/traitement avancé
-
- Partage de résultats
- Training, enseignement

- Tableaux de bord, interfaces ad hoc pour certaines problématiues

Interactive integration of our different pieces of software

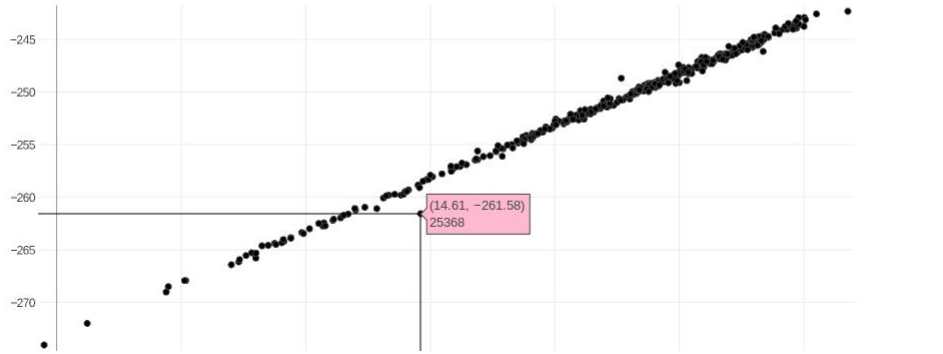
Interactive match-up outlier investigation with Jupyter

In [2]:
```
from s3analysis.slstr.mdb.analysis import get_basic_mask
from s3analysis.slstr.cloud import cloud_mask, DEFAULT_CLOUDMASK

# basic validity mask (sat angl < 55., wind speed > 6 m/s)
basic_mask = get_basic_mask(data_sat)

print "Number of match-ups : ", len(basic_mask)

print "Number of valid match-ups : ", (numpy.count_nonzero(basic_mask))
print "Number of invalid match-ups : ", (basic_mask).size - (numpy.count_nonzero(basic_mask))


# select only the match-ups where WST fields are defined
valid_sst = (
        basic_mask &
        ~numpy.ma.getmaskarray(data_sat['WST']["sst_theoretical_uncertainty"]) &
        (data_sat['WST']["quality_level"] > 2) &
        (numpy.ma.fabs(data_sat['WST']["dt_analysis"]) <= 5.) &
        ~numpy.ma.getmaskarray(data_insitu["water_temperature"])
        )


print "Final number of valid clear sky match-ups : ", numpy.count_nonzero(valid_sst)


slstr_sst = data_sat['WST']["sea_surface_temperature"] - 273.15
insitu_sst = data_insitu["water_temperature"]


cloudybox = cloud_mask(data_sat_box['WCT']["cloud_in"])
confidence = (data_sat_box['WCT']['confidence_in'][:] & 16384) > 0
```

In [3]:
```
# additional filter to keep only nighttime data
night = (data_insitu['solar_zenith_angle'] > 90.)
```

In [4]:
```
# achtung! plotly neeeds to be installed in your environment (pip install plotly
import plotly.graph_objs as go
import numpy as np
from plotly.offline import download_plotlyjs, init_notebook_mode, plot, iplot

# allow inline plot with plotly
init_notebook_mode(connected=True)

# Create a interactive scatterplot SST vs in situ with plotly
trace = go.Scattergl(
    x = insitu_sst[night & valid_sst],
    y = slstr_sst[night & valid_sst] - 273.15,
```

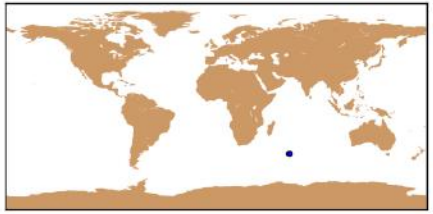
```
print len(insitu_sst[night & valid_sst]), ' match-ups'
```

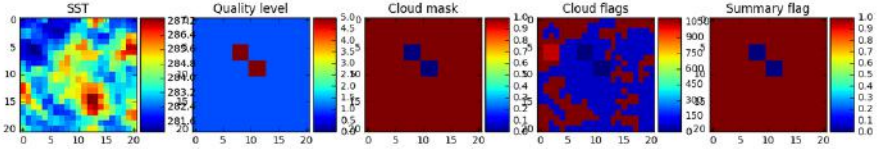


EUMETSAT

Interactive match-up outlier investigation with Jupyter

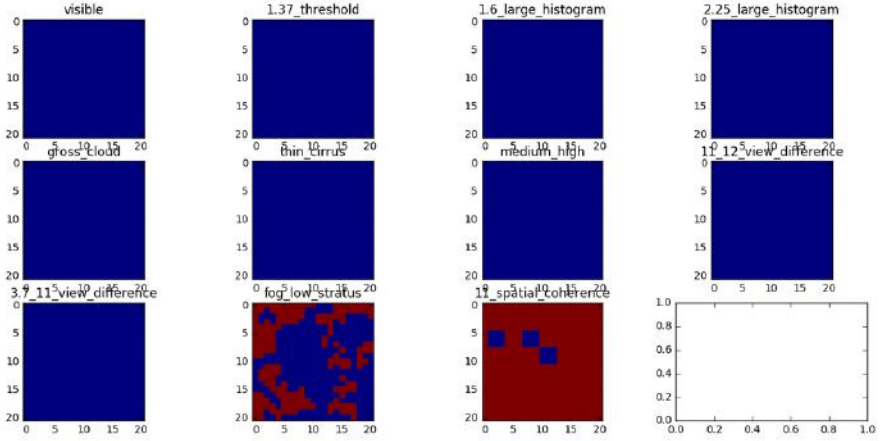
```
In situ value : 14.610000 K
SST - in situ difference : -3.040000 K
Traceability:
....WST file :  S3A SL 2 WST    20170628T185854 20170628T190154 20170628T202840 0179 019 198 5220 MAR F NR 002.SEN3
```






```
Used mask flags :  ['visible', '1.37_threshold', '1.6_large_histogram', '2.25_large_histogram', 'gross_cloud', 'thin_cirr
us', 'medium_high', '11_12_view_difference', '3.7_11_view_difference', 'fog_low_stratus', '11_spatial_coherence']
```


```
Used mask flags :  ['visible', '1.37_threshold', '1.6_large_histogram', '2.25_large_histogram', 'gross_cloud', 'thin_cirr
us', 'medium_high', '11_12_view_difference', '3.7_11_view_difference', 'fog_low_stratus', '11_spatial_coherence']
```


```
# display match-up info
print "SST value : %f K" % slstr_sst[choice]
print "In situ value : %f K" % insitu_sst[choice]
print "SST - in situ difference : %f K" % (slstr_sst[choice] - insitu_sst[choice])

print "Traceability:"
print "....WST file : ", data_sat['WST']['origin'][choice]

# locate match-up on map
from mpl_toolkits.basemap import Basemap
m = Basemap()
m.drawmapboundary()
m.fillcontinents(color='#cc9966')
x, y = m(data_insitu['lon'][choice], data_insitu['lat'][choice])
m.scatter(x, y)


# plot cloud and SST
plot_mask(choice)
```

Interactive match-up outlier investigation with Jupyter

trace back to original file

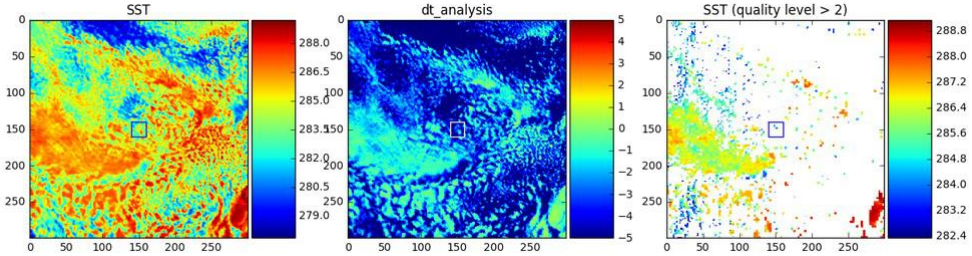
Here we access the content of the original file from which the match-up was extracted, and display a larger area around the match-up location.

This require to have access to the original SLSTR files!

In [6]:
```
print data_sat['WST']['dynamic_target_center_index'][choice]
```

```
[681 200]
```

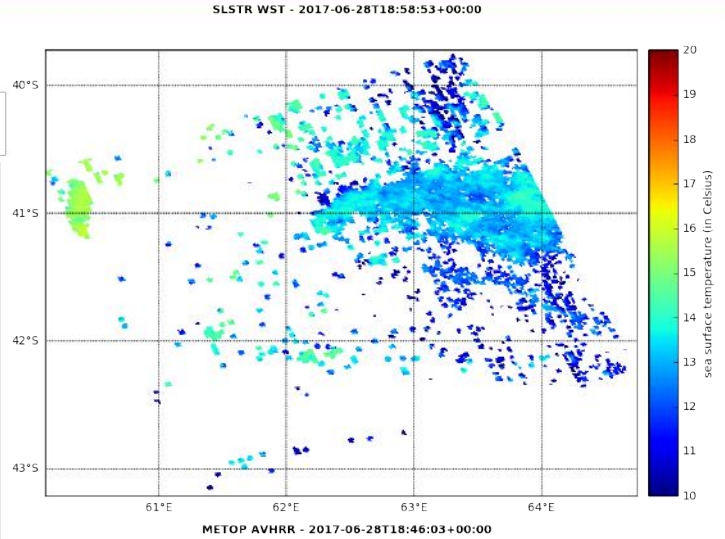
In [7]:
```
# get full path name
from naiad.utils.filelocator import FileLocator
locator = FileLocator()
fname = locator.get_full_path(data_sat['WST']['origin'][choice], 's3a_sl_2_wst__ref')

# define large subset
row, cell = data_sat['WST']['dynamic_target_center_index'][choice]
boxwidth = 300
boxheight = 300
larger_box = {'row': slice(max(0, row - boxheight / 2), row + boxheight / 2),
              'cell': slice(max(0, cell - boxwidth / 2), cell + boxheight / 2)
             }

# load data into a cerbere swa=th object
from cerbere.mapper.safeslfile import SAFESLWSTFile
from cerbere.datamodel.swath import Swath
wstfile = SAFESLWSTFile(fname)
swath = Swath()
swath.load(wstfile)
```

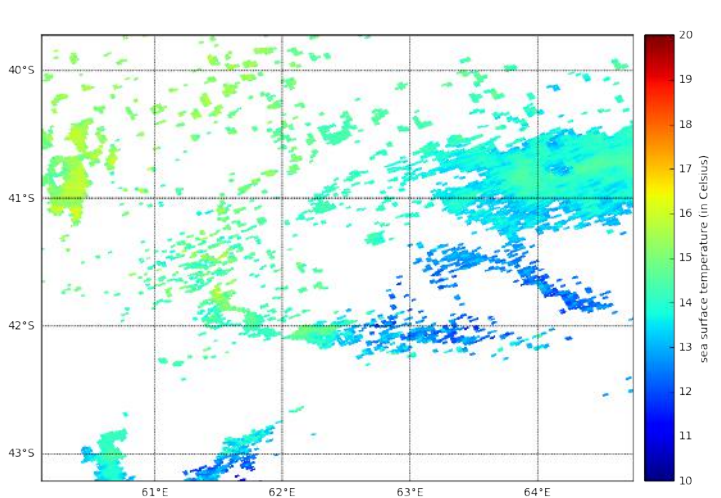


EUMETSAT

fetch Metop image

In [19]:
```
# import necessary packages
import shapely

from naiad.utils.filelocator import FileLocator
from naiad.queries.server import Server
from naiad.queries.search import SpatioTemporalSearch

from cerbere.mapper.ghrsstncfile import GHRSSTNCFile
from cerbere.datamodel.swath import Swath
from cerplot.mapping import CerMap

%matplotlib inline

# provides Naiad server URL
es = Server("http://eumetsat-gses-5:9200/")

# ========== DEFINE HERE YOUR SEARCH CRITERIA ==========

# define the geographical search box
lats = swath.get_lat(slices=larger_box)
lons = swath.get_lon(slices=larger_box)
area = shapely.geometry.box(lons.min(), lats.min(), lons.max(), lats.max())

# define start and end of search interval
start = swath.get_start_time() - datetime.timedelta(hours=1)
end = swath.get_end_time() + datetime.timedelta(hours=1)

# define the naiad indice (product name) to crawl
product = 'avhrr_sst_metop_b-osisaf-l2p-v1.0'

# compose the query
search = SpatioTemporalSearch([product], area, start, end)

# execute the query
res = search.run(es)
```





middleware

Cerbere : data abstraction layer in python

- Generic **python API** to access and describe file content (different data formats) and observation patterns
- Abstract layer to build generic tools and applications upon it
- Implemented at Ifremer, used by a few other people, also access layer for softwares like felyx, naiad, syntool, cal/val tools and routines
- Generic data file model (similar to netCDF) – **mapper** :
  - Standard geolocation dimensions : row/cell, x/y, lat/lon, time,…
  - Other dimensions
  - Standard geolocation fields
  - Instrumental / geophysical fields : multi-dimensional arrays (incl.)
    - Variables attributes : no explicit scale factor, transformation performed in memory
  - Metadata (global attributes)
- Generic observation patterns – **datamodel**
  - Swath, Grid, Trajectory (along-track) , Image, TimeSeries, GridTimeSeries,….
  - Generic functions
    - save : format to similar format (dimensions, global attributes, etc…) any data following the same observation pattern
    - extraction of subsets, etc…
- Complemented by some companion packages
  - mappers for other formats (Sentinel-3/SAFE, IASI/EPS)
  - **Also alleviates complexity of SLSTR products**
  - generic packages based on the cerbere datamodel concept : ancillary fields, display, resampling/interpolation, ocean parameter calculation,

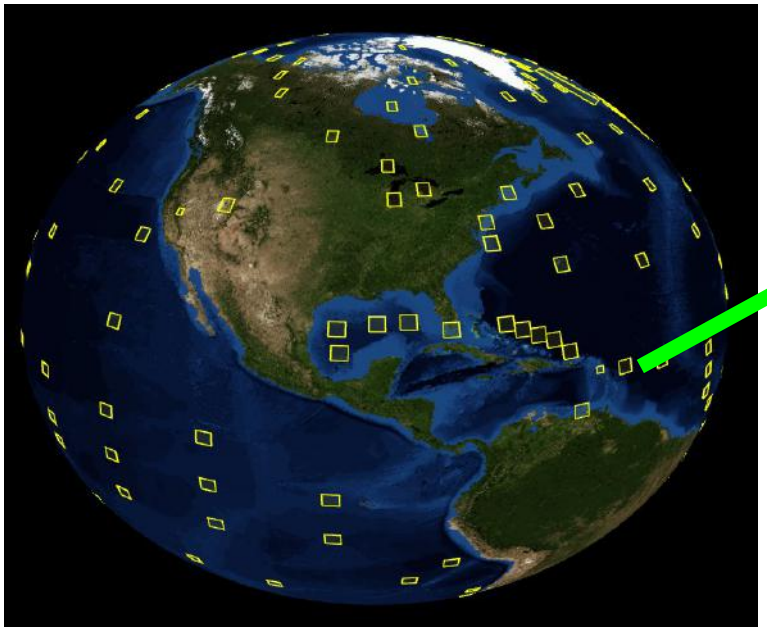
Doc/tutorial : http://cerbere.readthedocs.io/en/latest/

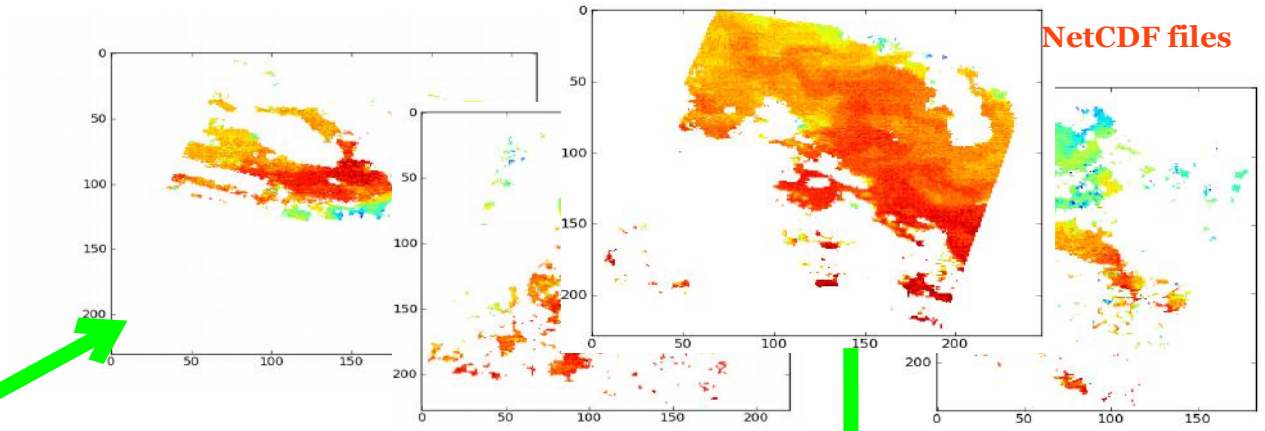
EUMETSAT

Felyx for MDB production

extract **miniprods** (subsets) over static and dynamic sites

process quantitative, qualitative, stat metrics over miniprods





NetCDF files

```
source: 20130101-IFR-L4_GHRSST-SSTfnd-ODYSSEA-GLOB_010_v2.0-fv1.0.nc
felyx_dataset_name: ifr-l4-sstfnd-odyssea-glob_010_v2.
percentage_coverage_of_site_by_miniprod: 100.0
date_modified: 2014-04-18T10:30:21
felyx_site_identifier: ukm005
date_created: 2014-04-18T10:30:21
time_coverage_start: 2013-01-01T00:00:00
time_coverage_stop: 2013-0101T00:00:00

sst_standard_deviation : 1.34
mean_sst : 286.289
ice_presence: 0
cloud_presence": 46.80
day_or_night: "night"
mean_wind_speed: 4.8388
```

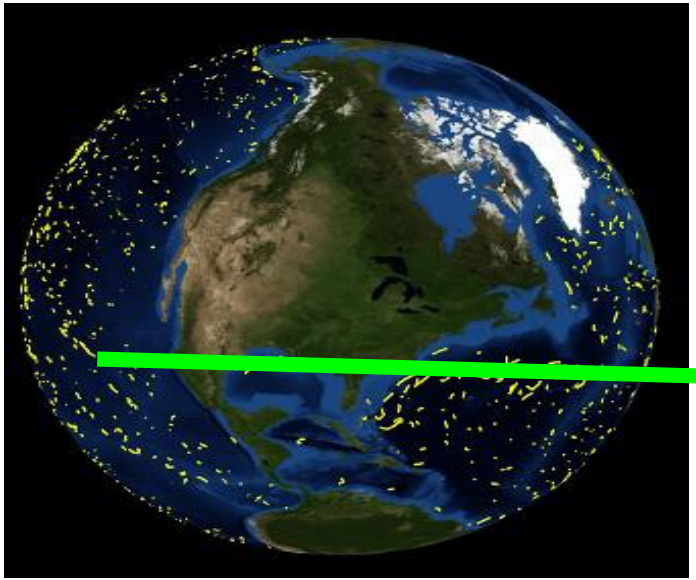
JSON files
indexed in a search engine
(ElasticSearch)

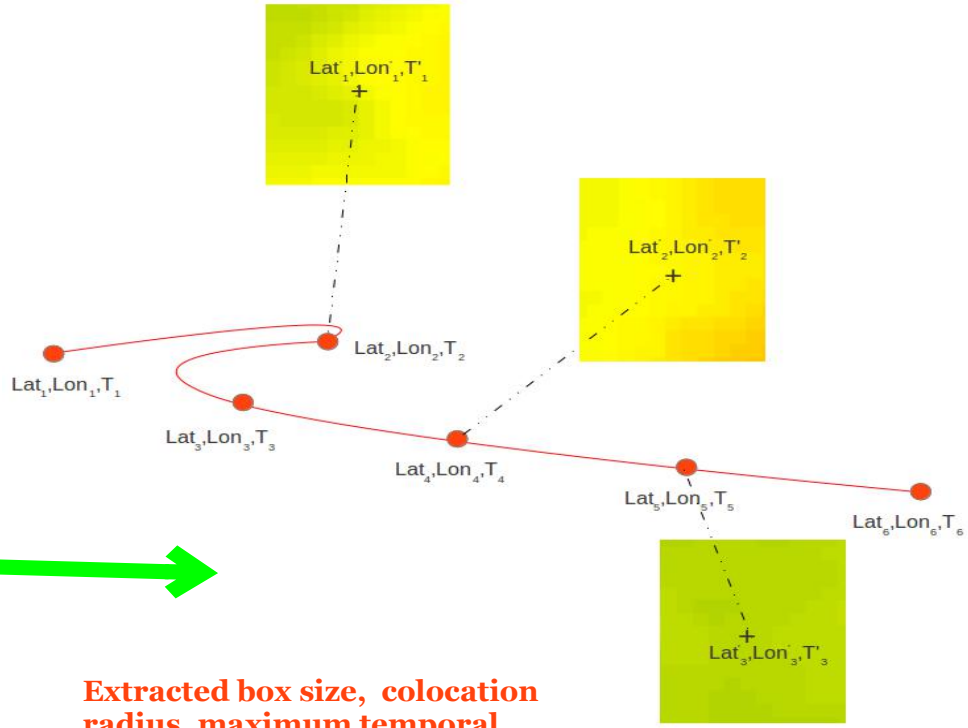
Felyx for MDB production

sites may be trajectories (buoys, cruise, hurricane)

MINIPROD's centred on trajectory locations closest in time locations closest in time

trajectory files ingested through import web service (CSV file)





Extracted box size, colocation radius, maximum temporal difference can be adjusted for each dataset

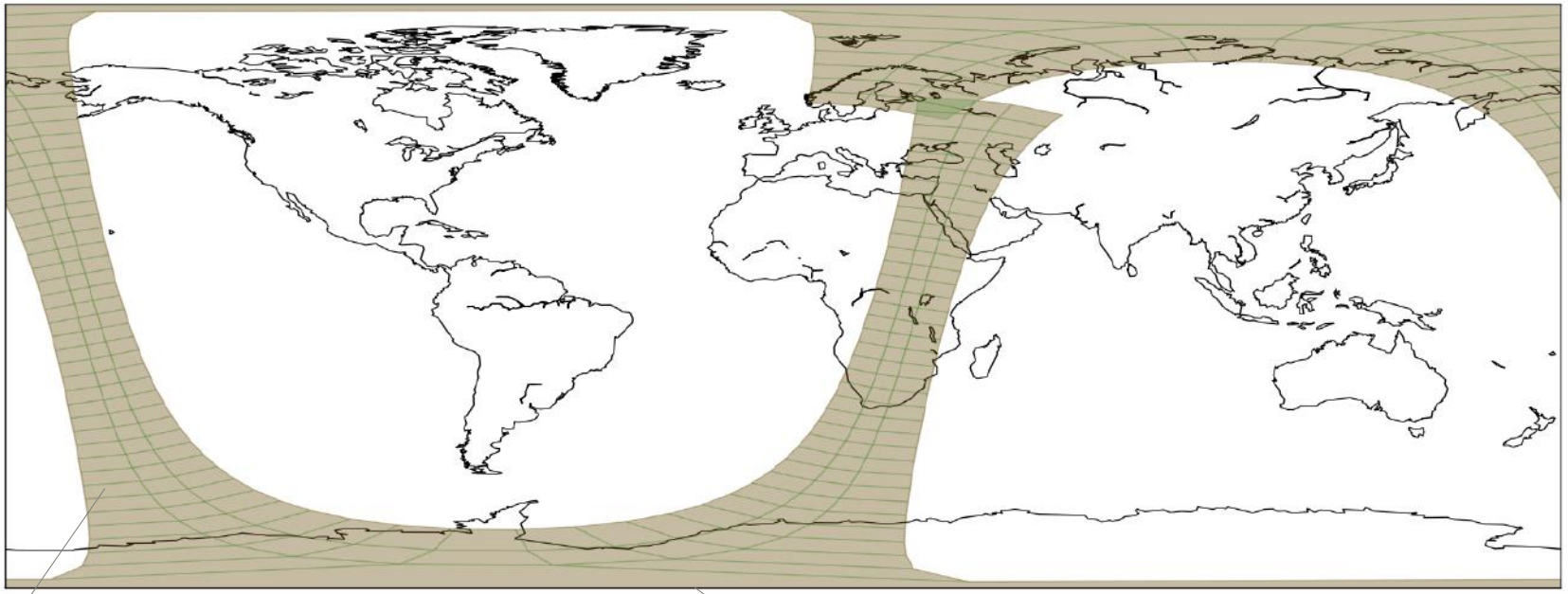
Naiad

- Intended for **satellite to satellite cross-over** detection
- Indexing of observation data as temporally bounded geographical shapes
- Command line or API based
- **Main functions**
  - Search file or file subset wrt multiple criteria : spatial, temporal, properties and metadata
  - Cross-search in different datasets, with time window constraint (cross-overs)
- **Main outputs**
  - List of file subsets (file name, indices)

http://naiad.readthedocs.io/en/develop/

Naiad ENHANCED SATELLITE ARCHIVE DATAMINER

Naiad – data tiling



Tile level

Product / file

Tile geographical shape (polygon)

temporal coverage

any numerical (quantitative) or text (tag, qualititative, metadata) properties

Granule level

Product / file

Sensor footprint geographical shape

temporal coverage

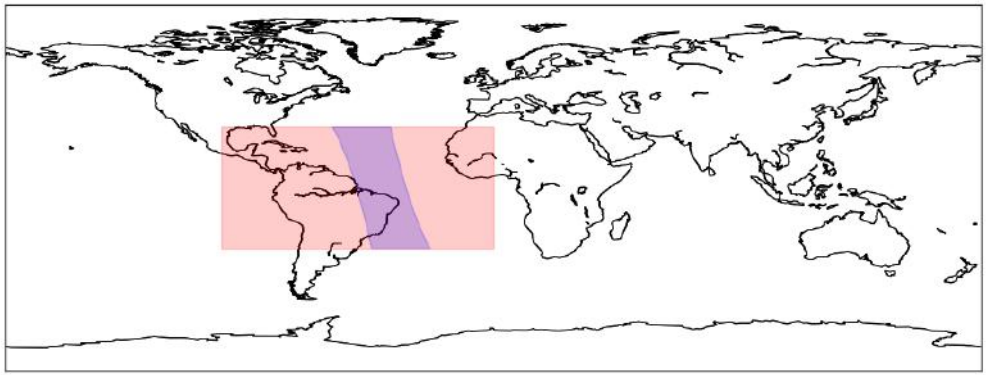
any numerical (quantitative) or text (tag, qualititative, metadata) properties

All tile metadata indexed in Elasticsearch search engine

Naiad - queries

Command line tools or python API

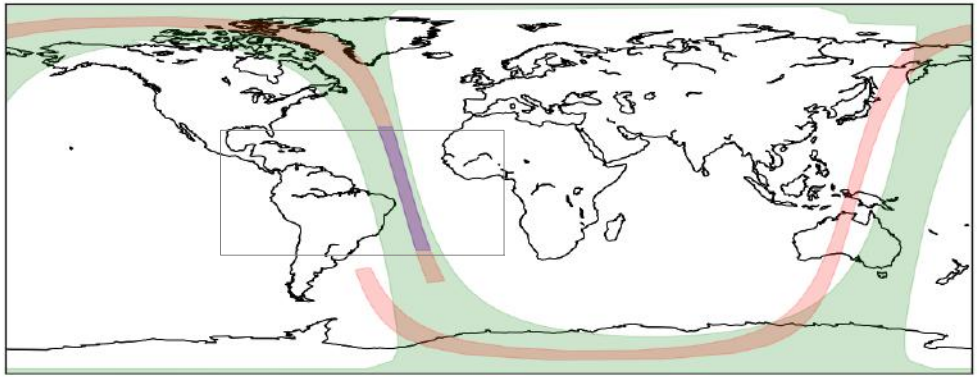
Simple search



Simple search

product(s)
time and space criteria
constraints on properties

Cross-over search

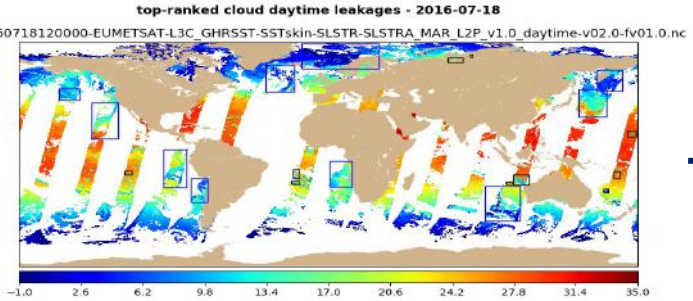


```
SHOW 0 8
W_XX-EUMETSAT-Darmstadt,HYPERSPECT+SOUNDING,MetOpA+IASI_C_EUMP_20100701004153_19184_eps_o_l1.nc

Reference
Name       : W_XX-EUMETSAT-Darmstadt,HYPERSPECT+SOUNDING,MetOpA+IASI_C_EUMP_20100701004153_19184_eps_o_l1.nc
Time range: 2010-07-01 01:45:32 to 2010-07-01 02:04:44
Slice      : {'cell': slice(40, 119, None), 'row': slice(477, 621, None)}
Geometry   :POLYGON ((-74.3743285021517 30, -68.90709065955365 30, -68.65699768066406 29.06100082397461, -60.12200164794922 -9.196999549865723, -56.58900070190
43 -23.4950008392334, -54.69640015258144 -30, -60.13921621269841 -30, -62.56700134277344 -19.75699996948242, -70.01499938964844 13.71700000762939, -72.4209976
1962891 23.25600051879883, -74.3743285021517 30))
Crossover :
Name       : 20100701-ATS_NR_2P-UPA-L2P-ATS_NR__2PNPDE20100701_020310_000046842090_00432_43572_3066-v01.nc
Time range: 2010-07-01 02:08:32 to 2010-07-01 02:27:22
Slice      : {'cell': slice(0, 511, None), 'row': slice(2152, 9684, None)}
Geometry   :POLYGON ((-74.3743285021517 30, -68.90709065955365 30, -68.65699768066406 29.06100082397461, -60.12200164794922 -9.196999549865723, -56.58900070190
43 -23.4950008392334, -54.69640015258144 -30, -60.13921621269841 -30, -62.56700134277344 -19.75699996948242, -70.01499938964844 13.71700000762939, -72.4209976
1962891 23.25600051879883, -74.3743285021517 30))
```

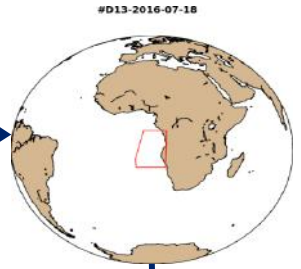
Results as images, text or json document

Use case : cloud leakage detection workflow

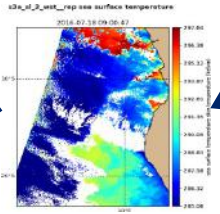
Binning of sea surface temperature (day / night)



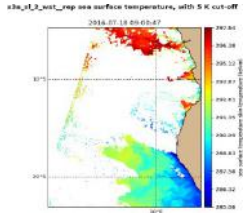
Extraction of anomalies vs reference and feature extraction

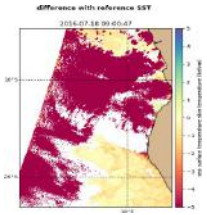


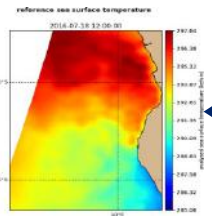
Search of granules and cross-overs



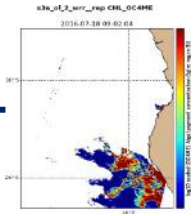
Cut-off vs OSTIA

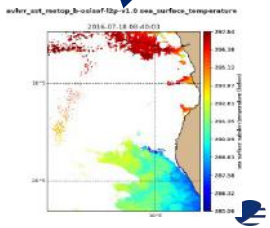




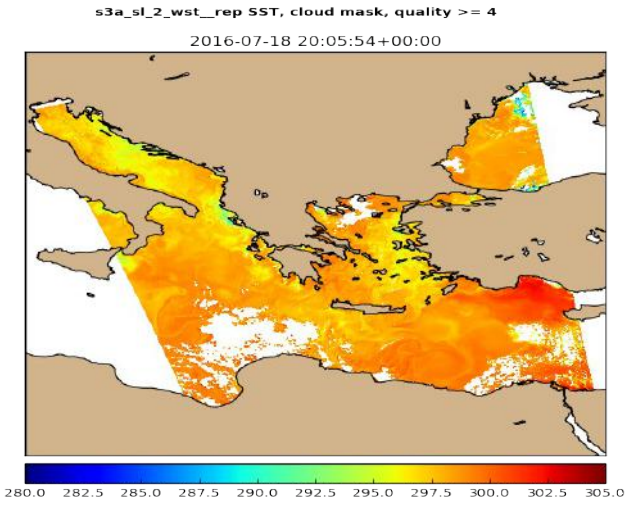


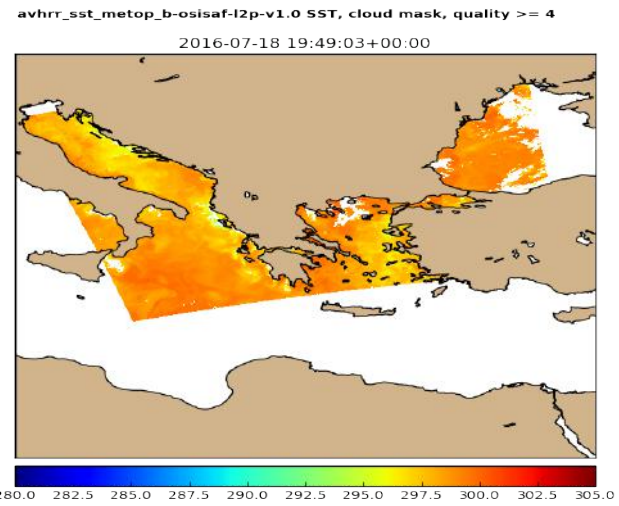
Intercomparison and assessment

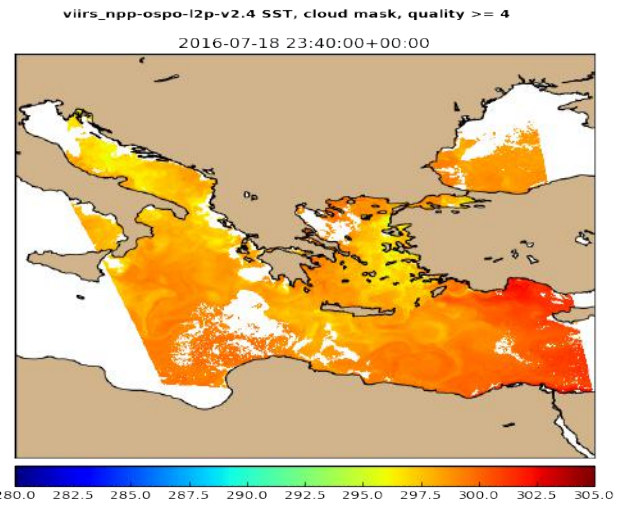


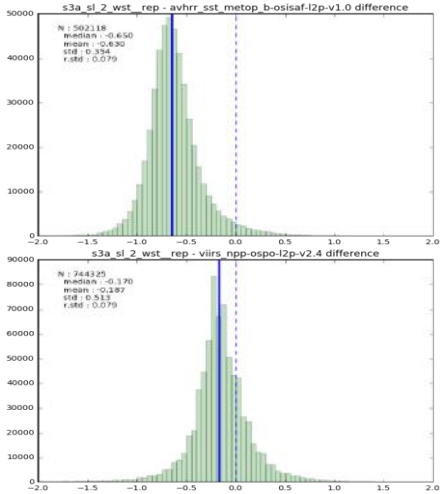


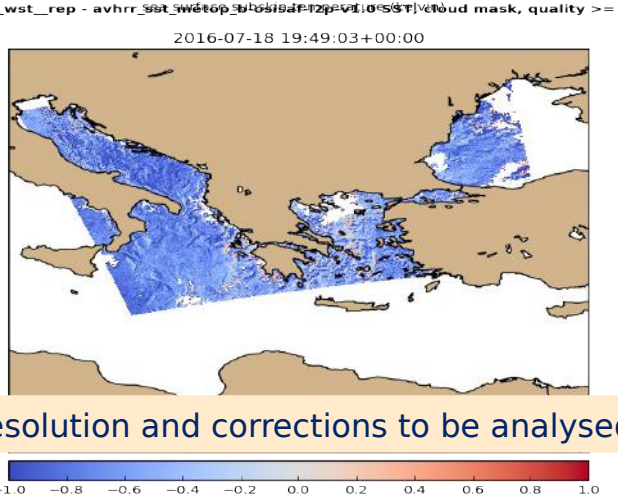
Use case : cross-over comparisons

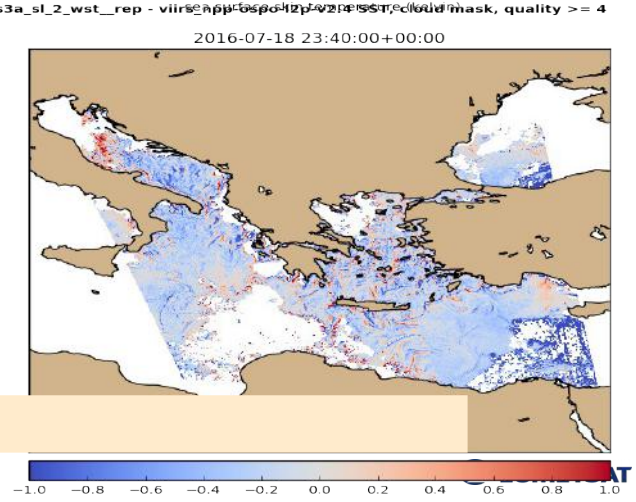








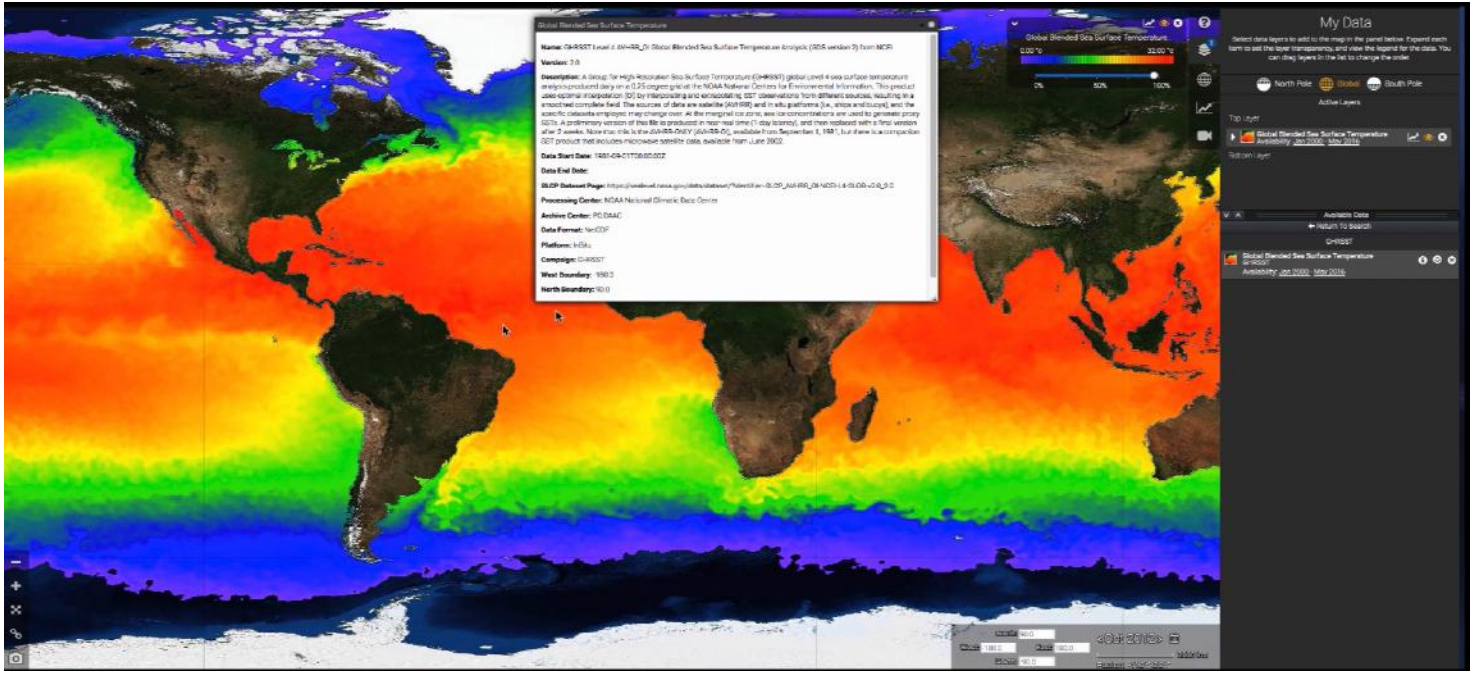


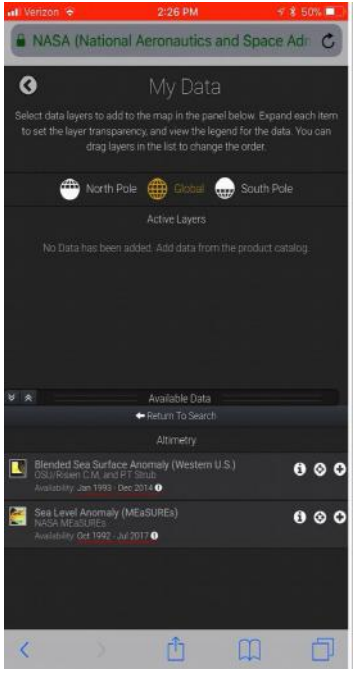


Feature resolution and corrections to be analysed

“big data” analytics (ex: SDAP, JPL solution)

Analyze Sea Level On-The-Fly



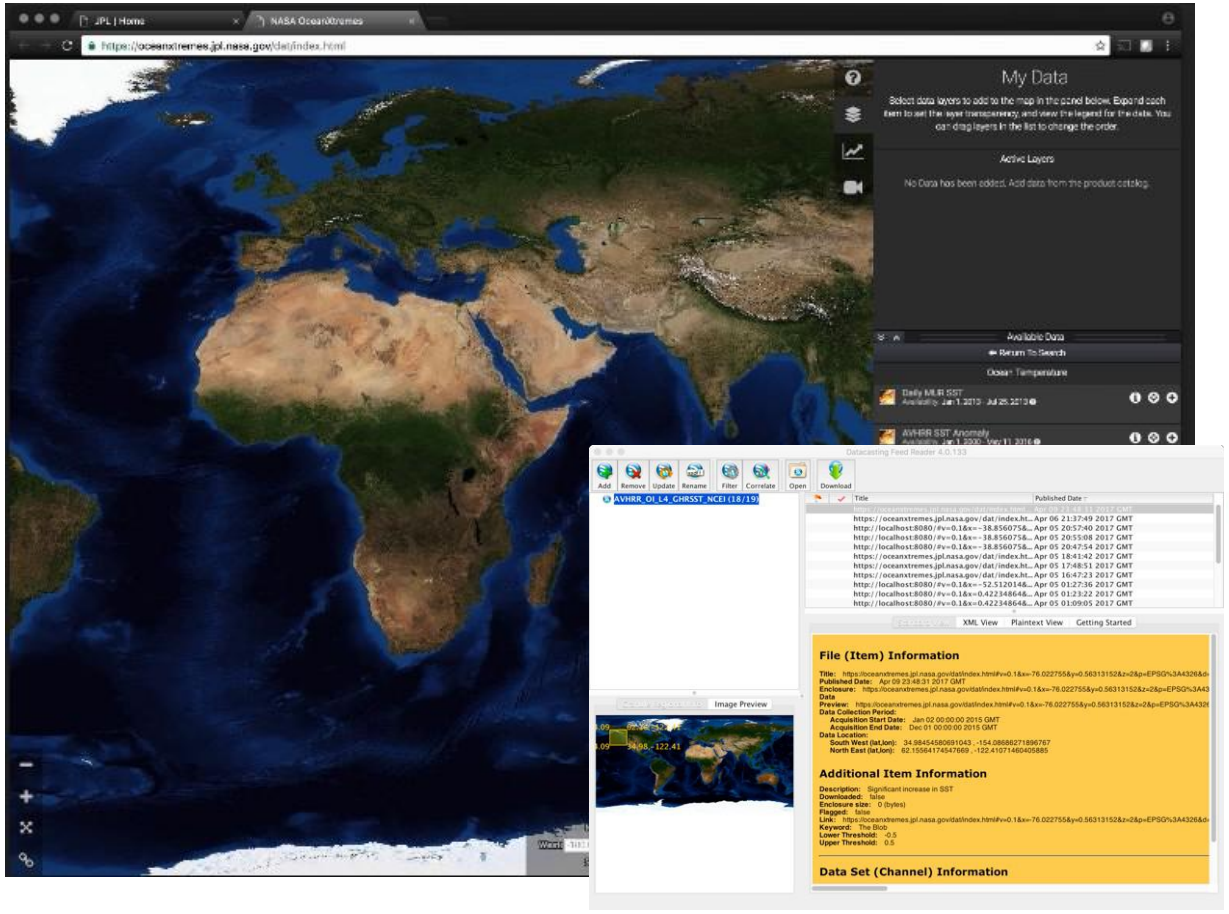


Sea Level Change - Data Analysis Tool

Visualizations | Hydrological Basins | Argo Profile | Time Series | Deseason | Data Comparison | Scatter Plot | Latitude/Time Hovmöller | Etc.

National Aeronautics and Space Administration
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

Analyze Ocean Anomaly – “The Blob”



- **Visualize** parameter
- **Compute** daily differences against climatology
- **Analyze** time series area averaged differences
- **Replay** the anomaly and visualize with other measurements
- **Document** the anomaly
- **Publish** the anomaly

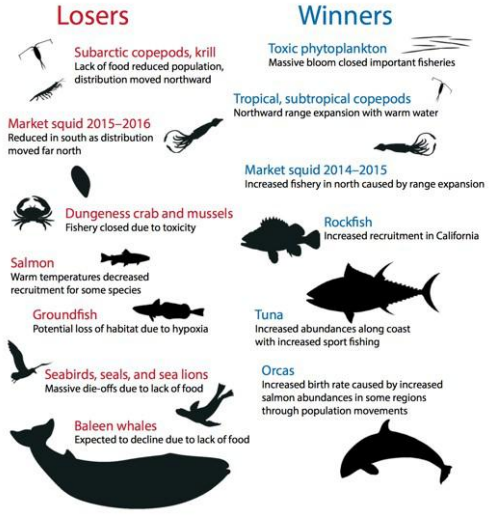
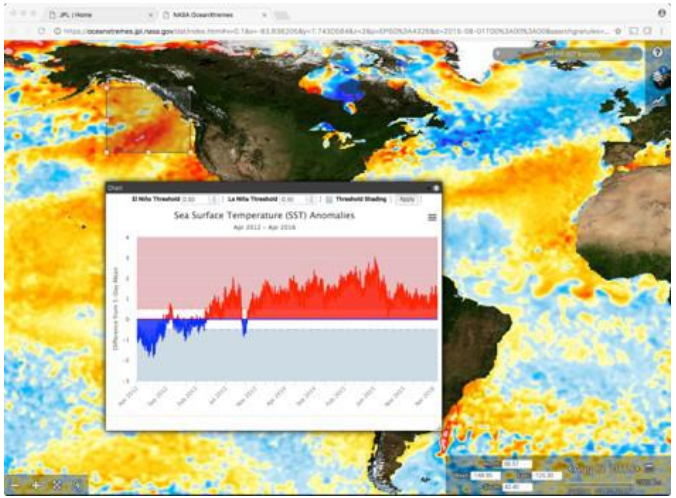


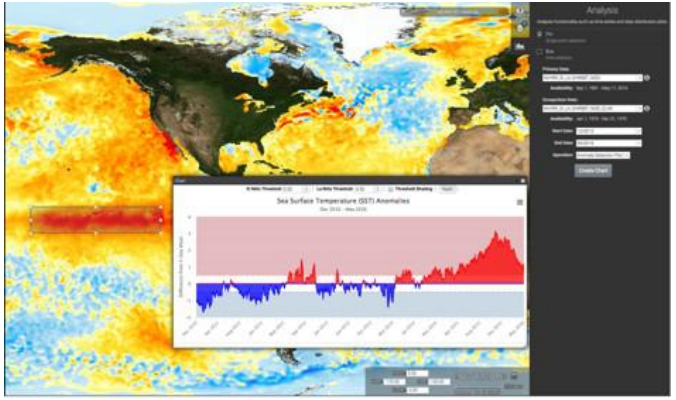
Figure from Cavole, L. M., et al. (2016). "Biological Impacts of the 2013–2015 Warm-Water Anomaly in the Northeast Pacific: Winners, Losers, and the Future." Oceanography 29.

National Aeronautics and Space Administration
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

More Anomalies



"The Blob"



El Niño 3.4 regional signal

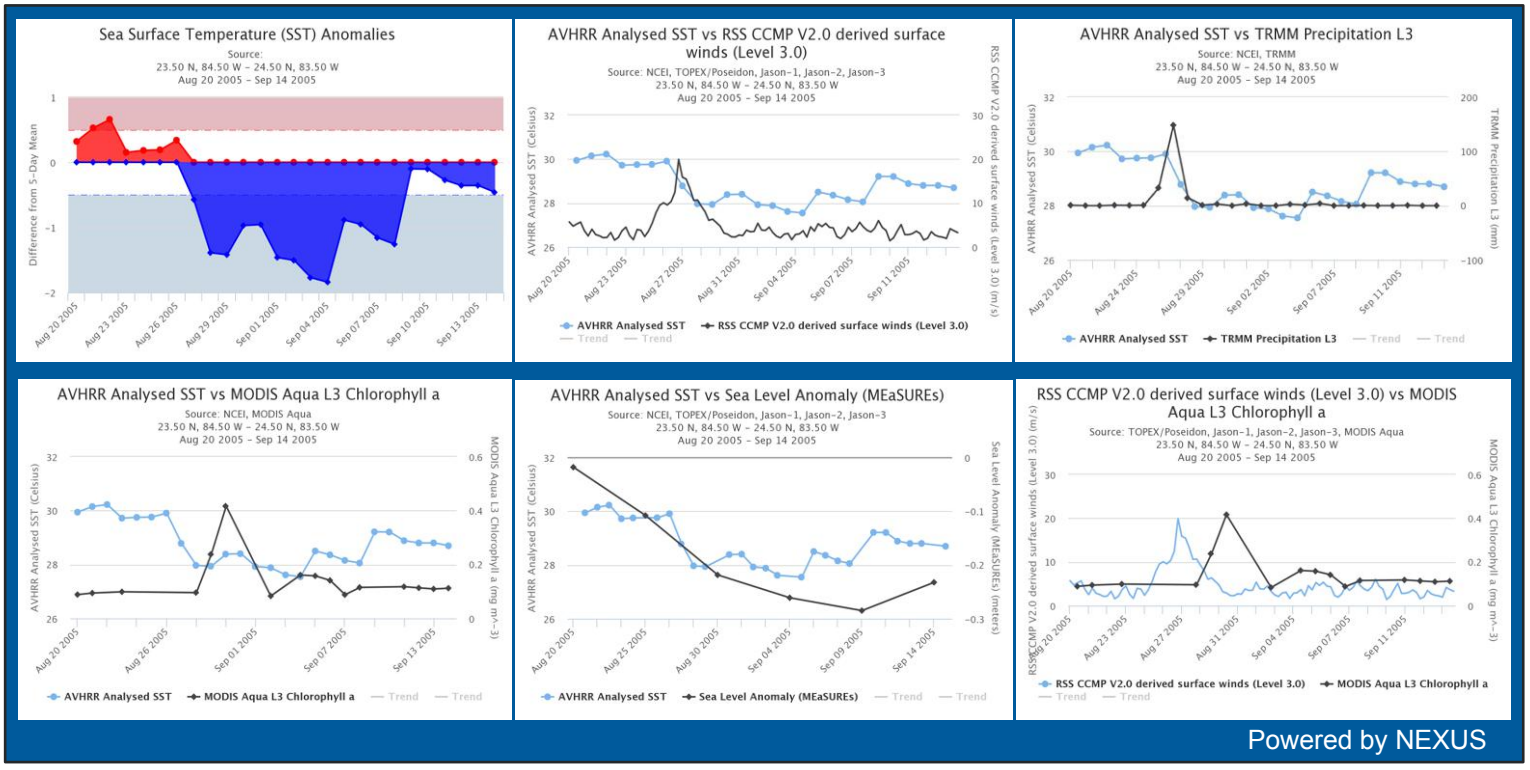
Recreated identification of "The Blob"

- **The Blob** is the name given to a large mass of relatively warm water in the Pacific ocean off the coast of North America. It was first detected in late 2013 and continued to spread throughout 2014 and 2015.
- SST anomaly = SST – SST Climatology at each location to compare with standard deviation - Chelle Gentemann, Senior Scientist at Earth & Space Research

Recreated the El Niño 3.4 regional signal

- **El Niño** is a phenomenon in the equatorial Pacific Ocean characterized by a five consecutive 3-month running mean of sea surface temperature (SST) anomalies in the Niño 3.4 region that is above (below) the threshold of +0.5°C (-0.5°C). This standard of measure is known as the Oceanic Niño Index (ONI).
- https://www.ncdc.noaa.gov/teleconnections/enso/indicators/sst.php

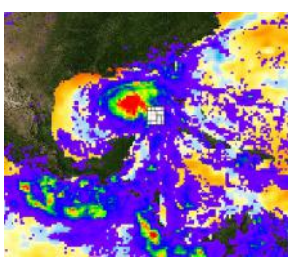
Hurricane Katrina Study



Hurricane Katrina passed to the southwest of Florida on Aug 27, 2005. The ocean response in a 1 x 1 deg region is captured by a number of satellites. The initial ocean response was an immediate cooling of the surface waters by 2 °C that lingers for several days. Following this was a short intense ocean chlorophyll bloom a few days later. The ocean may have been "preconditioned' by a cool core eddy and low sea surface height.

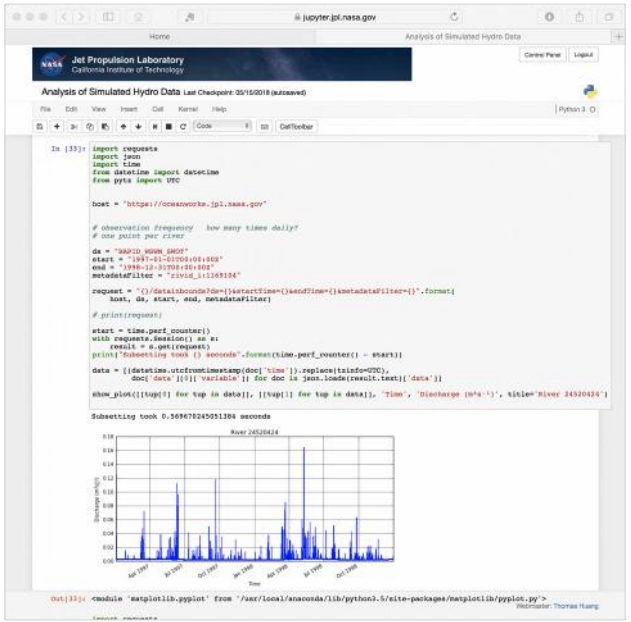
The SST drop is correlated to both wind and precipitation data. The Chl-A data is lagged by about 3 days to the other observations like SST, wind and precipitation.

A study of a Hurricane Katrina–induced phytoplankton bloom using satellite observations and model simulations
Xiaoming Liu, Menghua Wang, and Wei Shi
JOURNAL OF GEOPHYSICAL RESEARCH, VOL. 114, C03023, doi:10.1029/2008JC004934, 2009

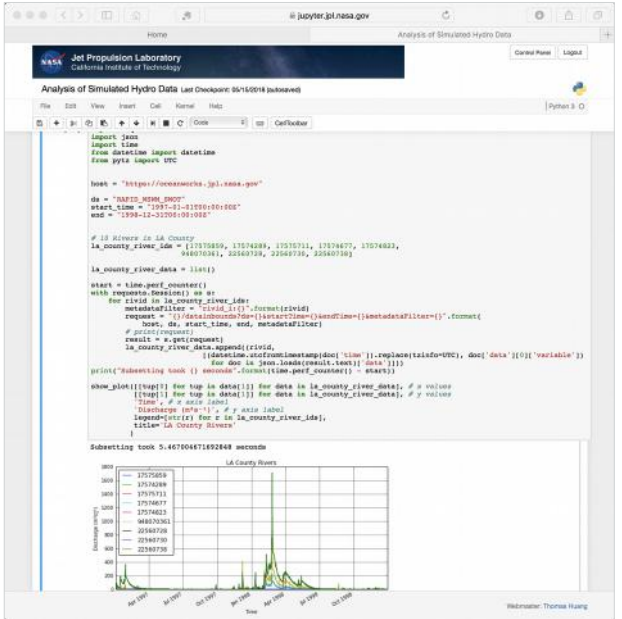


Hurricane Katrina TRMM overlay SST Anomaly

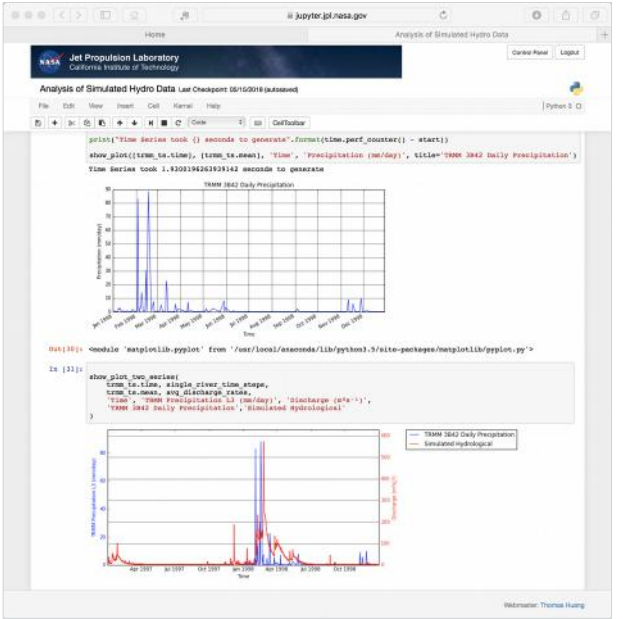
Performance example: support for hydrology



Retrieval of a single river time series



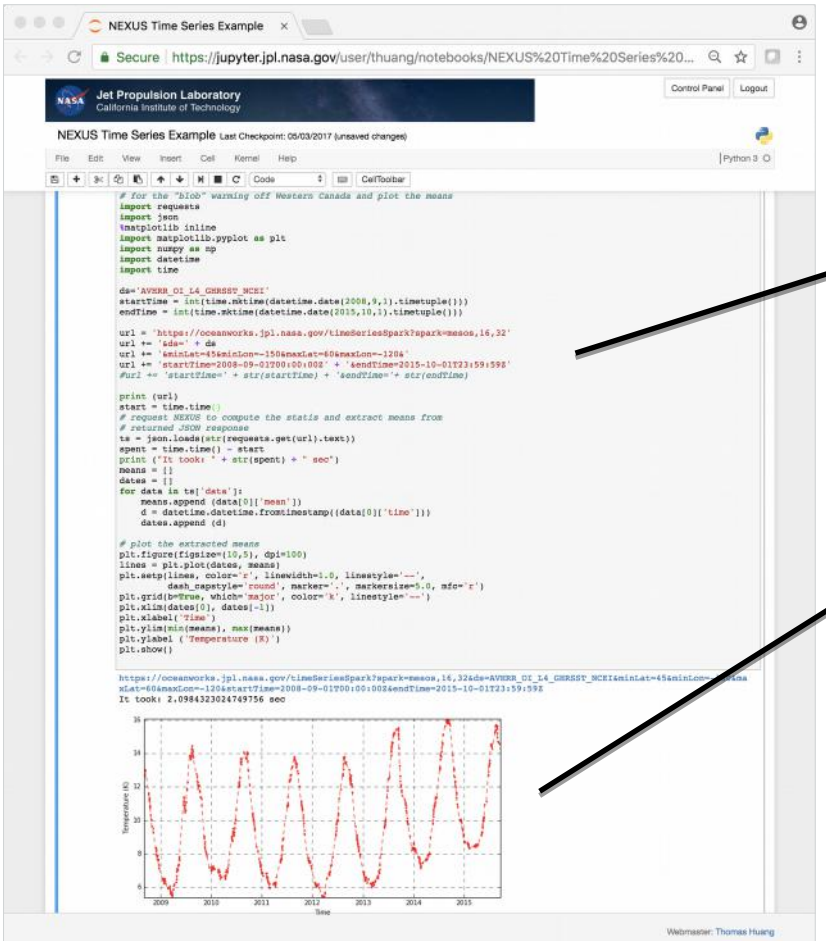
Retrieval of time series from 9 rivers



Time series coordination between TRMM and river

- Simulated hydrology data in preparation for SWOT hydrology
- **River data**: ~3.6 billion data points. 3-hour sample rate. Consists of measurements from ~600,000 rivers
- **TRMM data**: 17 years, .25deg, 1.5 billion data points
- Sub-second retrieval of river measurements
- On-the-fly computation of time series and generate coordination plot

Enable Science without File Download



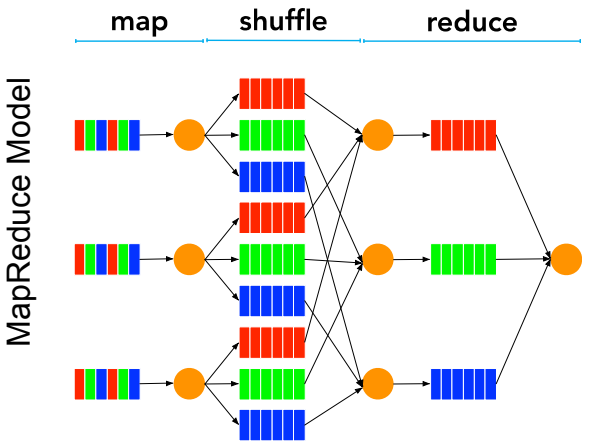

```
# Request NEXUS to compute SST Time Series 2008/9/1 - 2015/10/1
# for the "blob" warming off Western Canada and plot the means
…
ds='AVHRR_OI_L4_GHRSST_NCEI'

url = … # construct the webservice URL request


# make request to NEXUS using URL request
# save JSON response in local variable
ts = json.loads(str(requests.get(url).text))

# extract dates and means from the response
means = []
dates = []
for data in ts['data']:
    means.append (data[0]['mean'])
    d = datetime.datetime.fromtimestamp((data[0]['time']))
    dates.append (d)

# plot the result
…
```

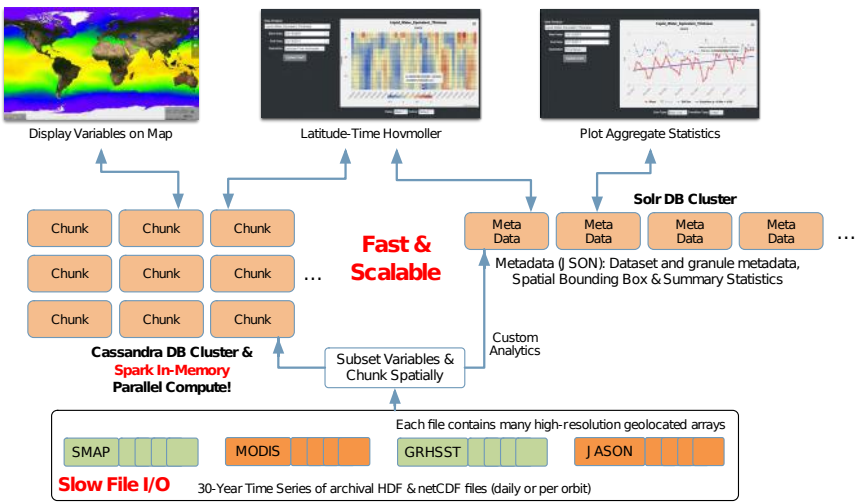
https://oceanworks.jpl.nasa.gov/timeSeriesSpark?**spark**=mesos,16,32&**ds**=AVHRR_OI_L4_GHRSST_NCEI&**minLat**=45&**minLon**=-150&**maxLat**=60&**maxLon**=-120&**startTime**=2008-09-01T00:00:00Z&**endTime**=2015-10-01T23:59:59Z

It took: 2.0984323024749756 sec

NEXUS: Scalable Data Analytic Solution

- **MapReduce**: A programming model for expressing distributed computations on massive amount of data and an execution framework for large-scale data processing on clusters of commodity servers. - J. Lin and C. Dyer, "*Data-Intensive Text Processing with MapReduce*"
  - **Map**: splits processing across cluster of machines in parallel, each is responsible for a record of data
  - **Reduce**: combines the results from Map processes
- **NEXUS** is a data-intensive analysis solution using a new approach for handling science data to enable large-scale data analysis
  - Streaming architecture for horizontal scale data ingestion
  - Scales horizontally to handle massive amount of data in parallel
  - Provides high-performance geospatial and indexed search solution
  - Provides tiled data storage architecture to eliminate file I/O overhead
  - A growing collection of science analysis webservices

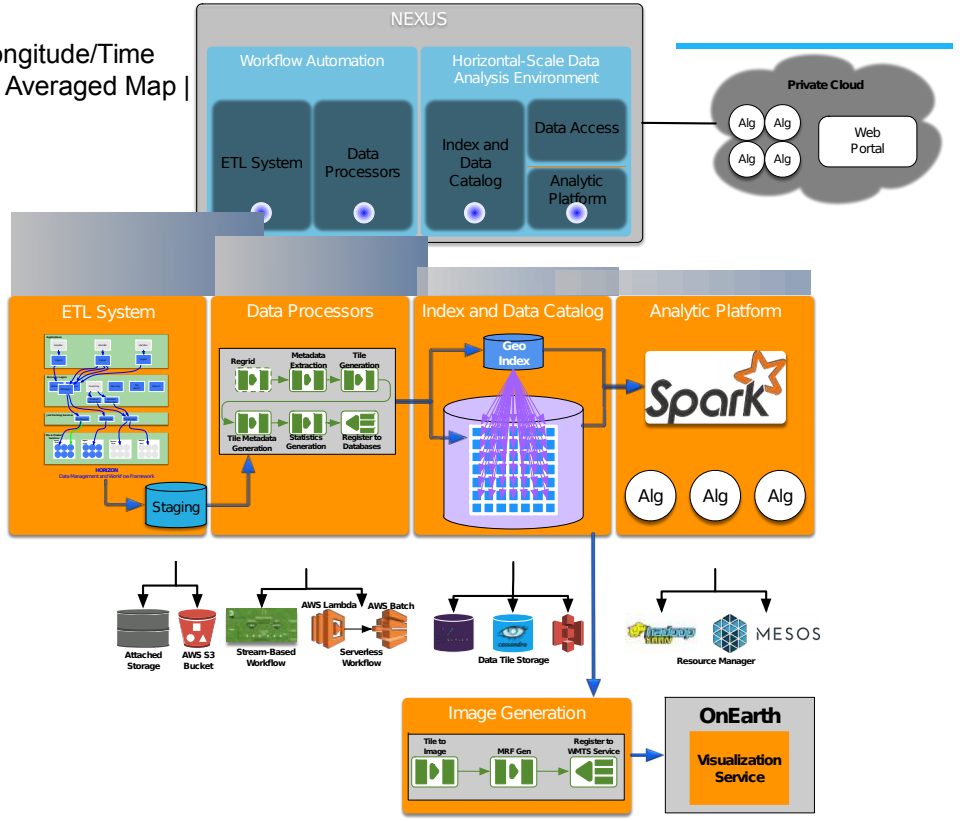




NEXUS' Two-Database Architecture

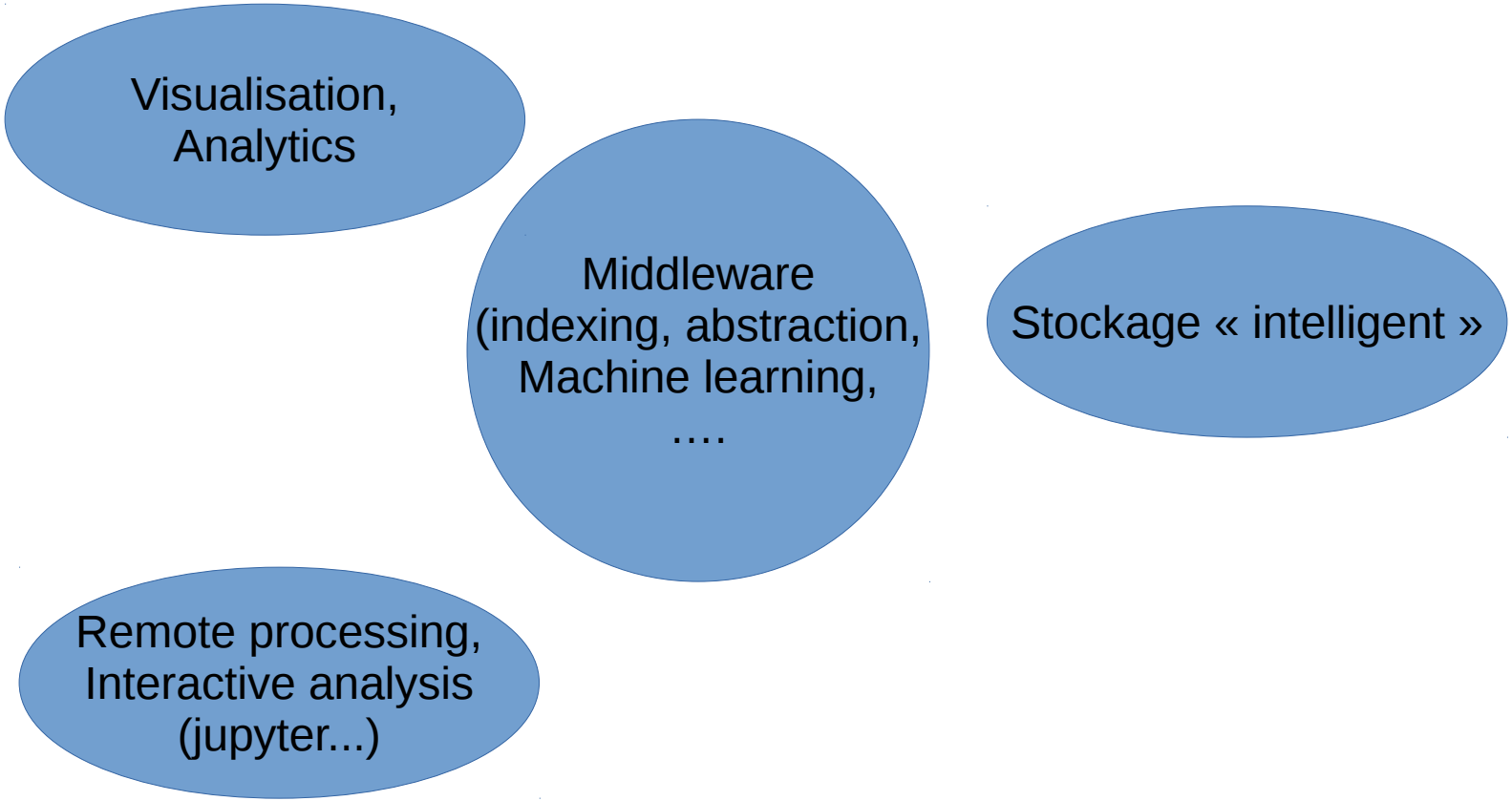
NEXUS' Pluggable Architecture for different Operation Needs

- **NEXUS supports public/private Cloud and local cluster deployments**
- **It has a growing set of algorithms –** Time Series | Latitude/Time Hovmöller| Longitude/Time Hovmöller| Latitude/Longitude Time Average | Area Averaged Time Series | Time Averaged Map | Climatological Map | Correlation Map | Daily Difference Average
- **It offers several container-based deployment options**
  - Local on-premise cluster
  - Private Cloud
  - Amazon Web Service
- **Automate Data Ingestion with Image Generation**
  - Cluster based
  - Serverless (Amazon Lambda and Batch)
- **Data Store Options**
  - Apache Cassandra
  - ScyllaDB
  - Amazon Simple Storage Service (S3)
- **Resource Management Options**
  - Apache YARN
  - Apache MESOS
- **Analytic Engine Options**
  - Custom Apache Spark Cluster
  - Amazon Elastic MapReduce (EMR)
  - Amazon Athena (work-in-progress)



Big data analytics

- Multi-dimensional analysis
- Fast, high integration level
- Storage key/value (or chunks)
-
- Resource consuming (distributed storage)
- Requires new storage concept, fragmented and indexing – file concept would remain virtual
- Duplication of data required if no revision of our storage design
- Multiple solutions existing, each one with its own storage model and technology => need careful rationalization (only one technology per product or group of products, API standardization)


Visualisation,
Analytics
Middleware
(indexing, abstraction,
Machine learning,
….
Stockage « intelligent »
Remote processing,
Interactive analysis
(jupyter...)