Ifremer

IN SITU DATA FROM THE OCEAN TO YOUR DESKTOP

ACCESS & USE IN SITU DATA

Agenda

- How to access and use Coriolis data
- How to access and use Copernicus Marine Service In Situ TAC data

How to access In Situ Data ? – Coriolis

https://dataselection.coriolis.eu.org/

Select a region,

a period

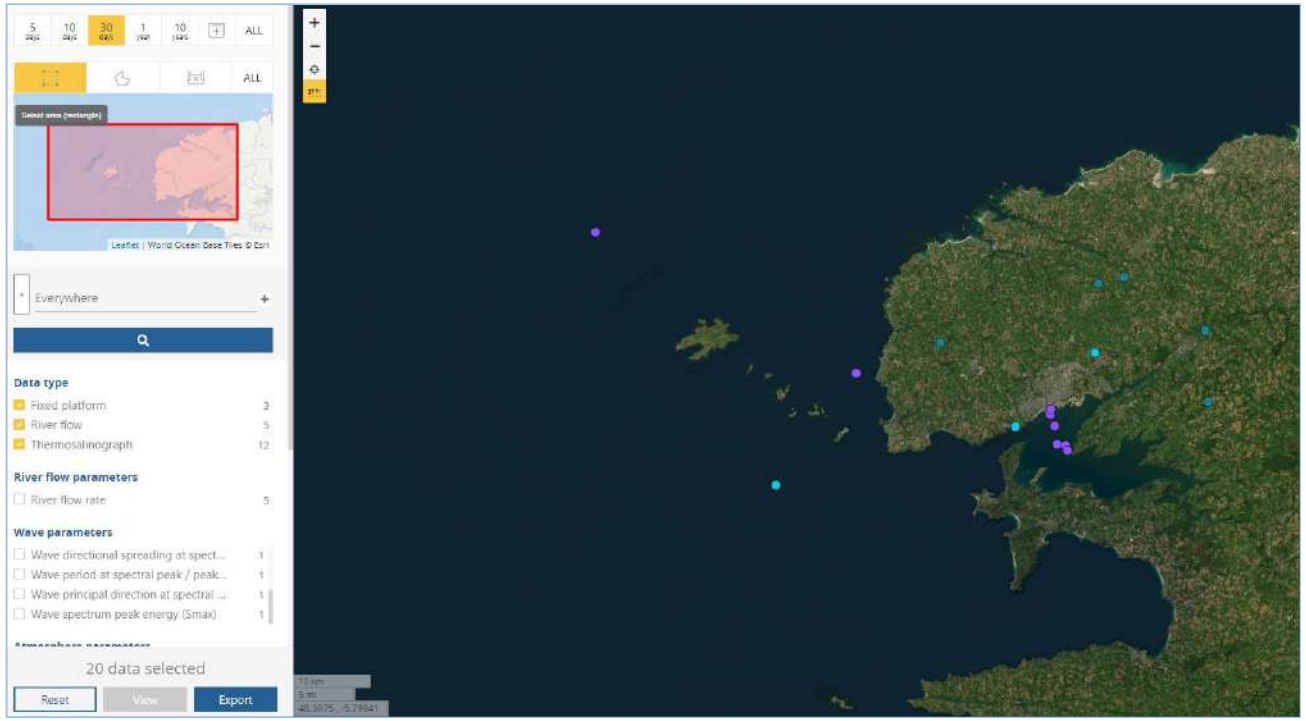
around Brittany – 30 last days

<u>3 storms:</u>

9th of march : LARISA storm

31st of march : MATHIS storm

12nd of april NOA storm

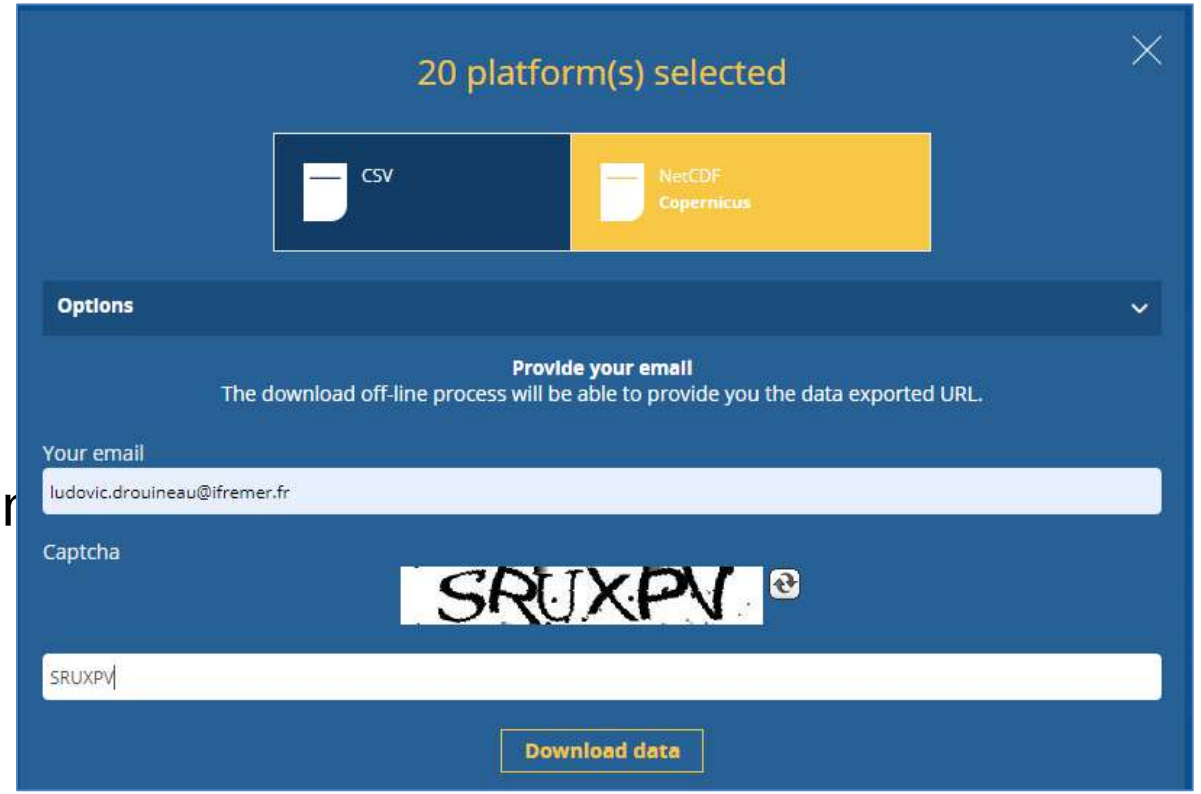


How to access In Situ Data ? – Coriolis

Choose your format (csv / NetCDF)

<u>NetCDF :</u>
format Copernicus Marine Service
In Situ TAC

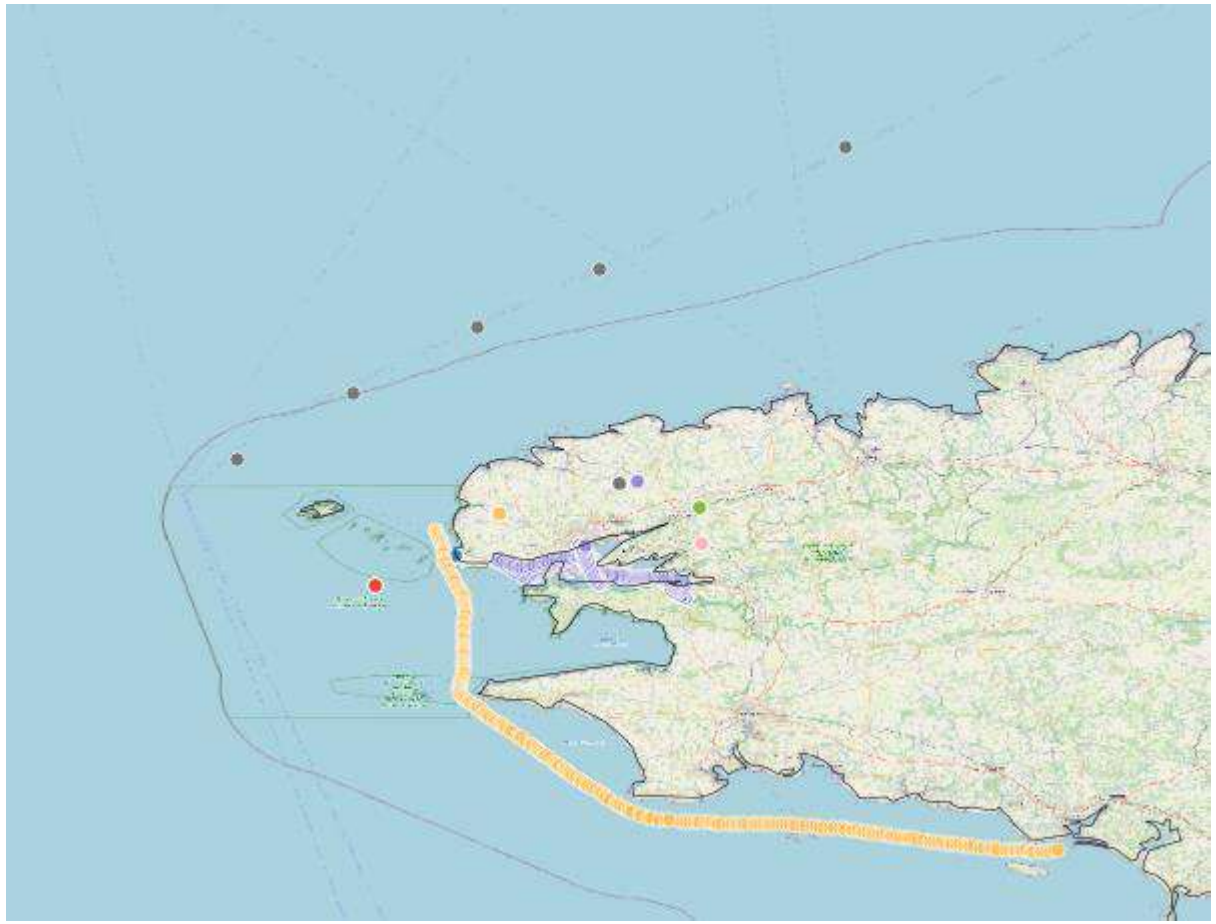
Several mails to show the progressior
Output : A tar.gz file



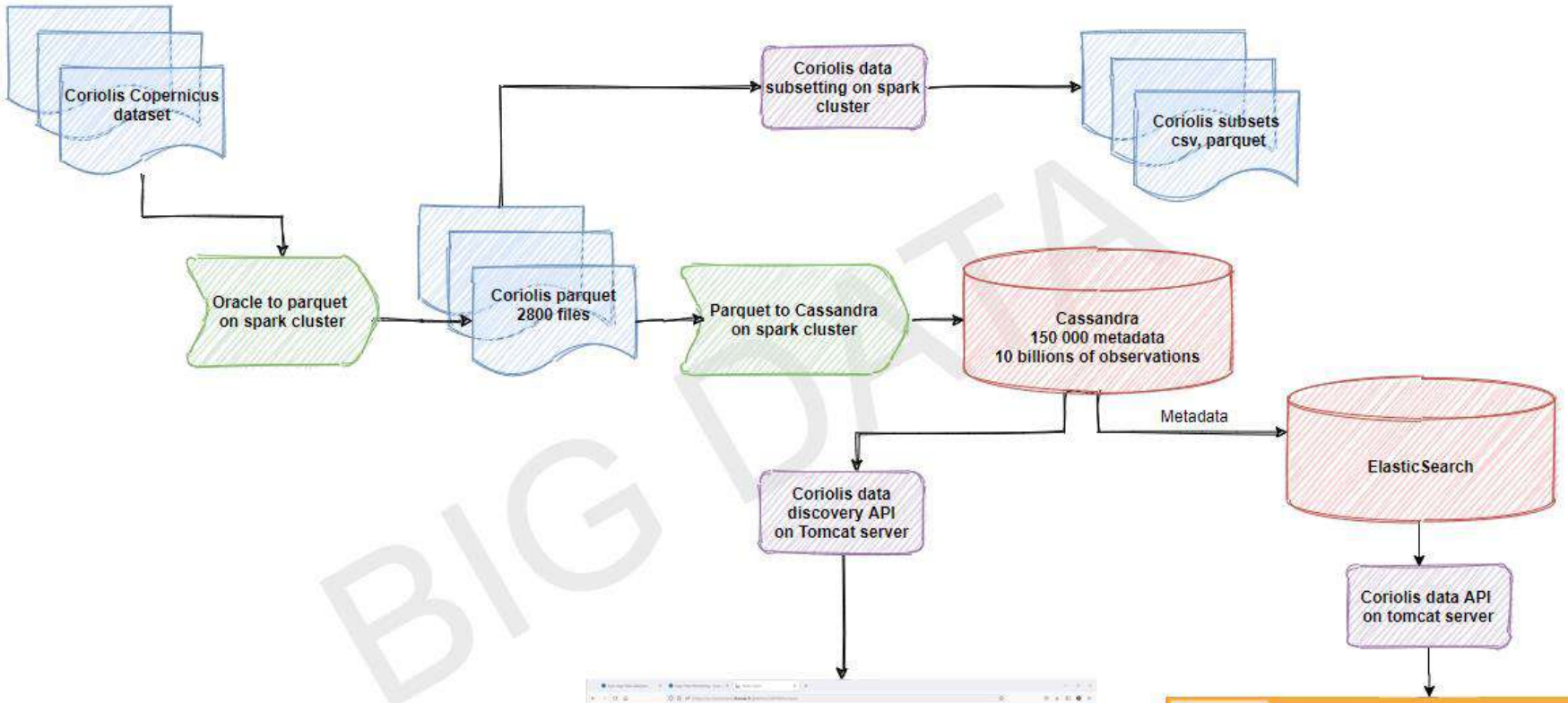
Manipulate your data with Xarray

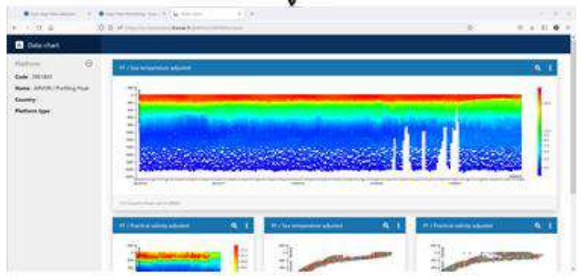
- GL_TS_MO_6200069.nc
- GL_TS_MO_6200069_changed_ONP.nc
- GL_TS_MO_Iroise.nc
- GL_TS_RF_EXSC0049.nc
- GL_TS_RF_EXSC0050.nc
- GL_TS_RF_EXSC0051.nc
- GL_TS_RF_EXSC0079.nc
- GL_TS_RF_EXSC0080.nc
- GL_TS_TG_BrestTG_01minute.nc
- GL_TS_TG_LeConquetTG_01minute.nc
- GL_TS_TS_FGG8669.nc
- GL_TS_TS_FMNB.nc
- GL_TS_TS_FQBE.nc

```
fig = plt.figure(figsize=(16,12))
ax = plt.axes(projection=ccrs.PlateCarree())
ax.set_extent([-6,-2,47,50])
ax.coastlines('10m')
imagery = OSM()
ax.add_image(imagery, 10)

for f in file:
    datasets = xr.open_dataset(path + '/' + f)
    time = datasets.TIME
    geom_ds = xr.Dataset({'geometry': time})
    geom_ds.plot.scatter(x='LONGITUDE', y='LATITUDE', s=50)

plt.title('')
plt.show()
```



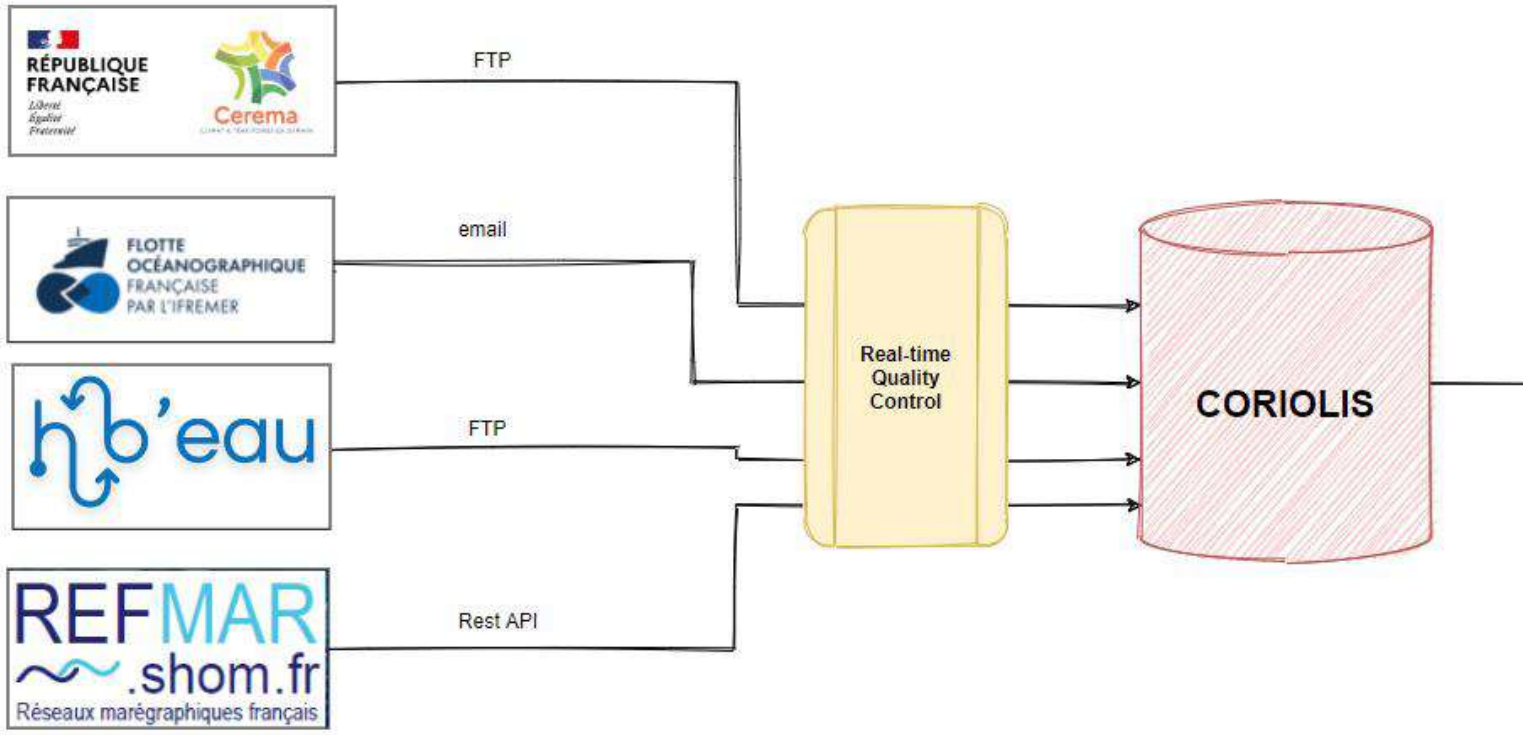
Let’s look under the hood





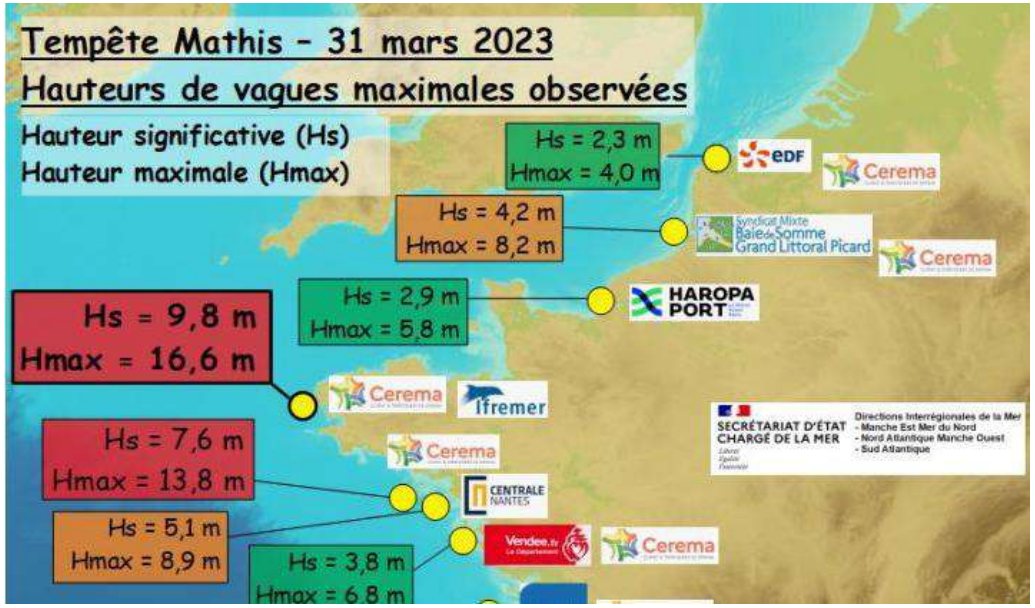
Important task: Improve the data refresh rate (goal : 15min)
Current refresh rate: 24h

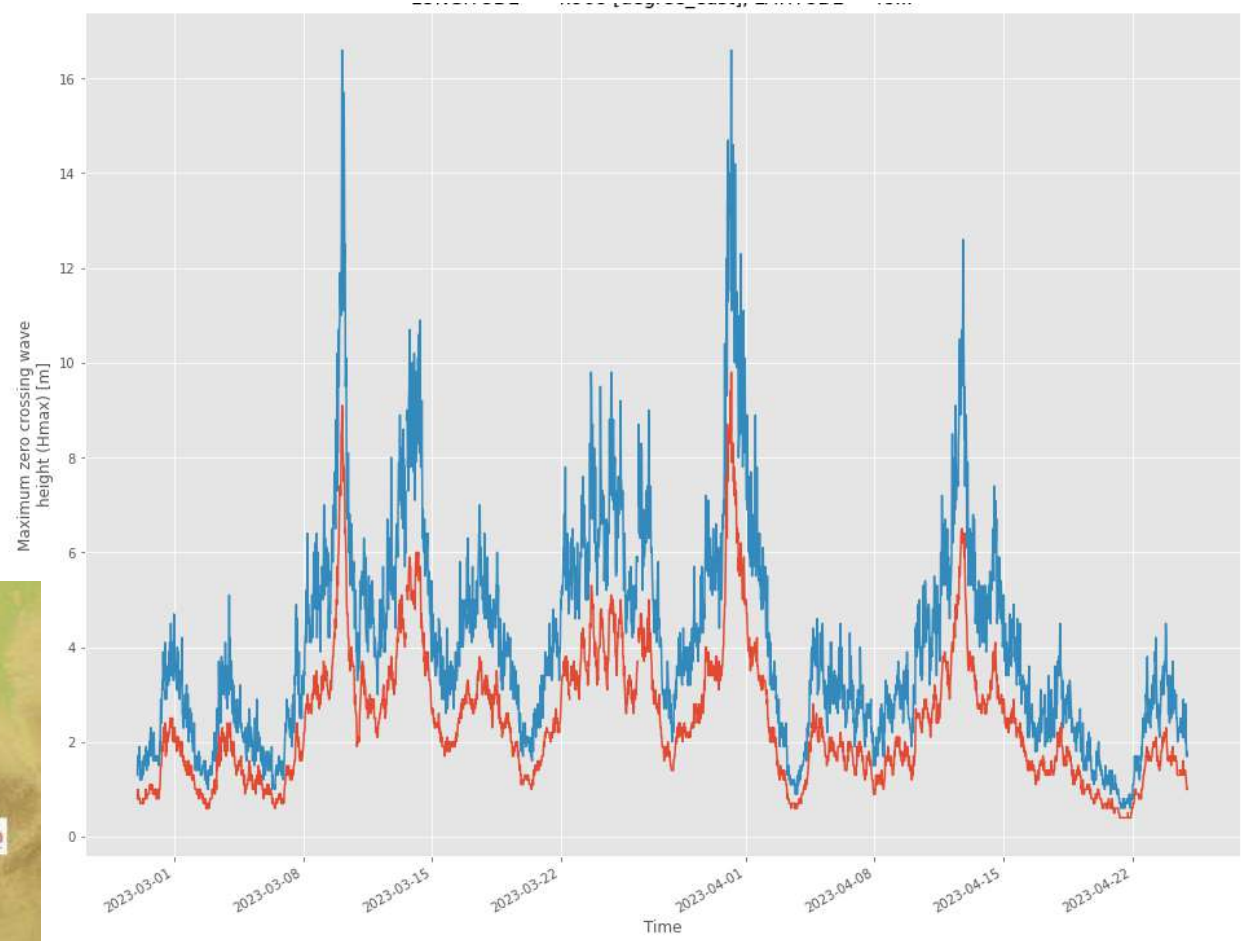
Where all these data comes from ?



Fixed platforms: **Moorings**, Tide-gauges, riverflows

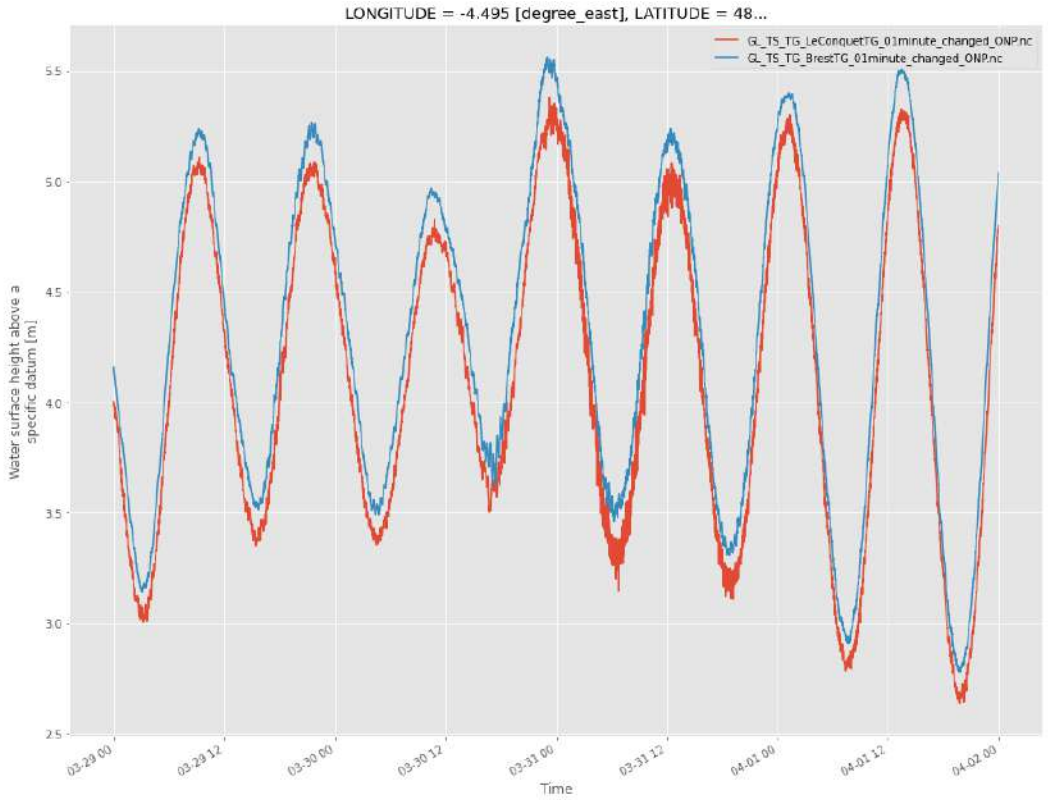
```
fig = plt.figure(figsize=(16,12))
ax = fig.add_subplot(111)

for f in file:
    datasets = xr.open_dataset(path + '/' + f)
    vavh = datasets.VAVH.isel(DEPTH=0)
    vzmx = datasets.VZMX.isel(DEPTH=0)
    vavh.plot()
    vzmx.plot()

plt.show()
```



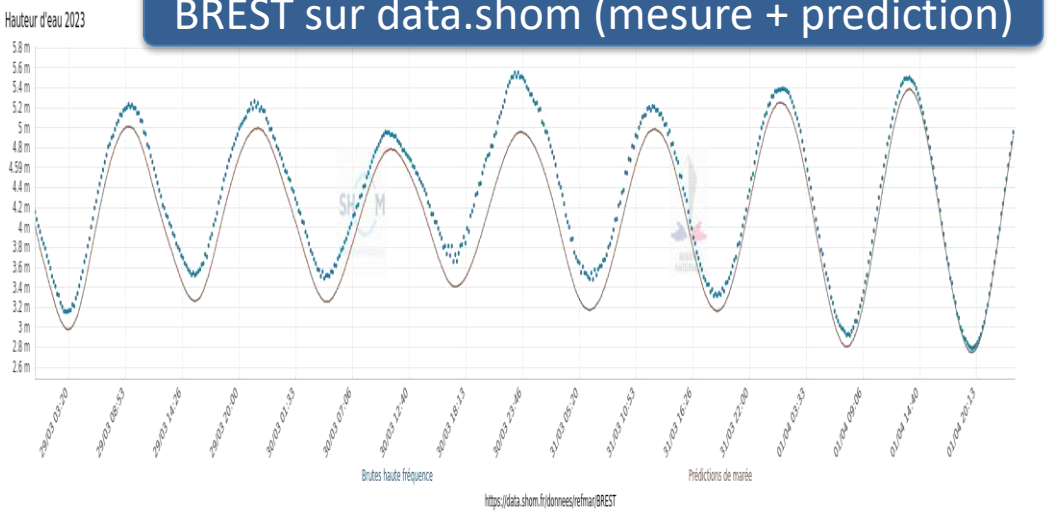


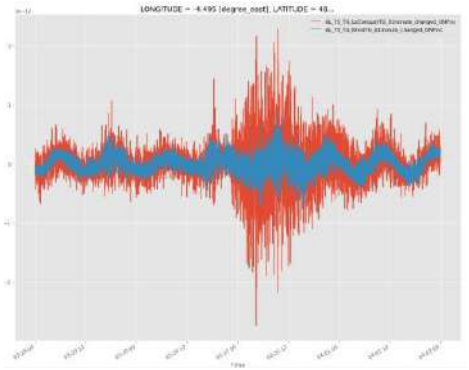
Fixed platforms: Moorings, **Tide-gauges**, riverflows

```
fig = plt.figure(figsize=(16,12))
ax = fig.add_subplot(111)

for f in file:
    datasets = xr.open_dataset(path + '/' + f)
    slev = datasets.SLEV.sel(TIME=slice('2023-03-29', '2023-04-01'))
    slev.plot()

plt.show()

fig = plt.figure(figsize=(16,12))
ax = fig.add_subplot(111)
for f in file:
    datasets = xr.open_dataset(path + '/' + f)
    slev = datasets.SLEV.sel(TIME=slice('2023-03-29', '2023-04-01')).differenti
    slev.plot()

plt.show()
```



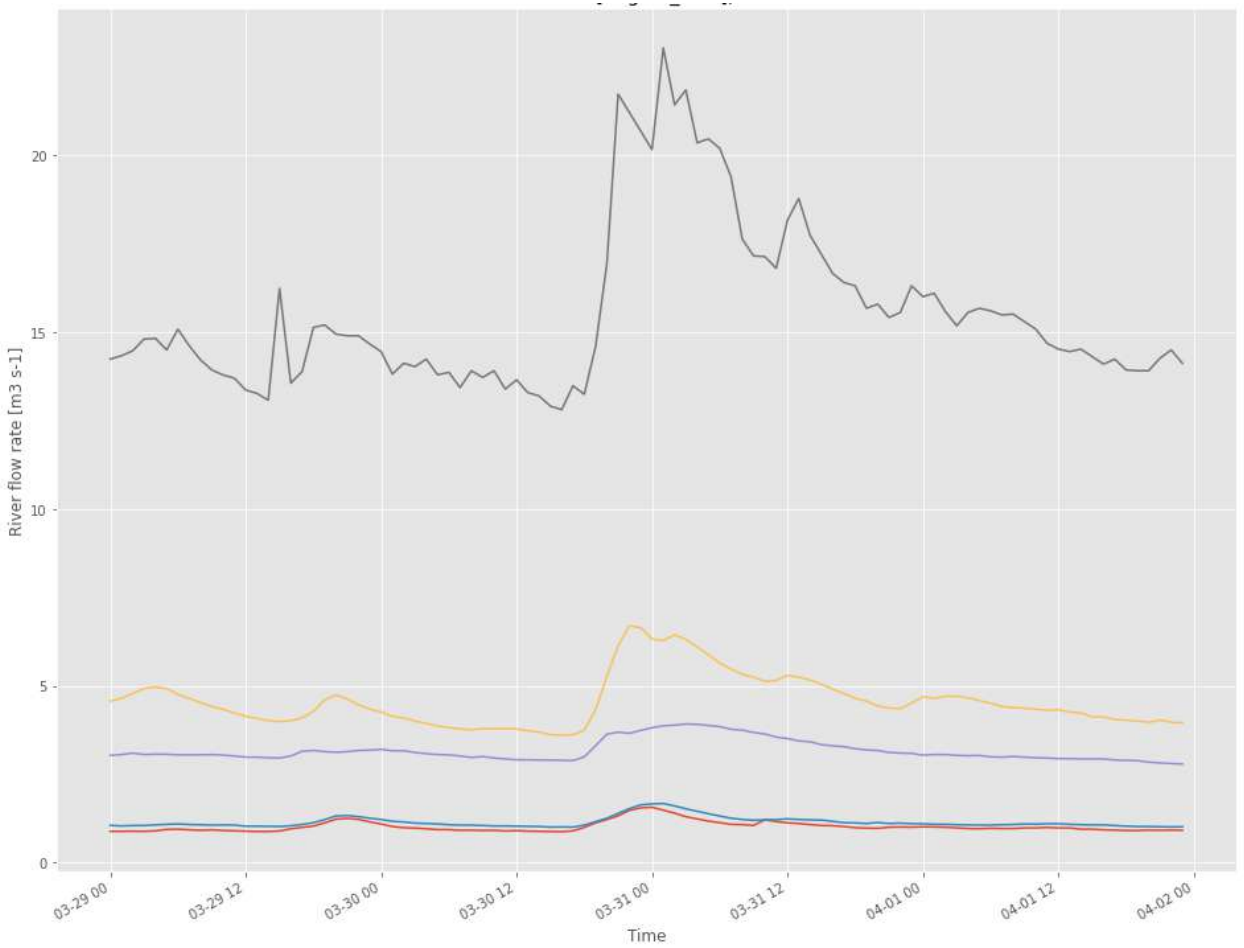
BREST sur data.shom (mesure + prediction)





Fixed platforms: Moorings, Tide-gauges, **riverflows**

```
fig = plt.figure(figsize=(16,12))
ax = fig.add_subplot(111)

for f in file:
    datasets = xr.open_dataset(path + '/' + f)
    rvfl = datasets.RVFL.sel(TIME=slice('2023-03-29', '2023-04-01'))
    #rvfl = datasets.RVFL
    rvfl.plot()

plt.show()
```



How to access In Situ Data ? – Coriolis

https://dataselection.coriolis.eu.org/

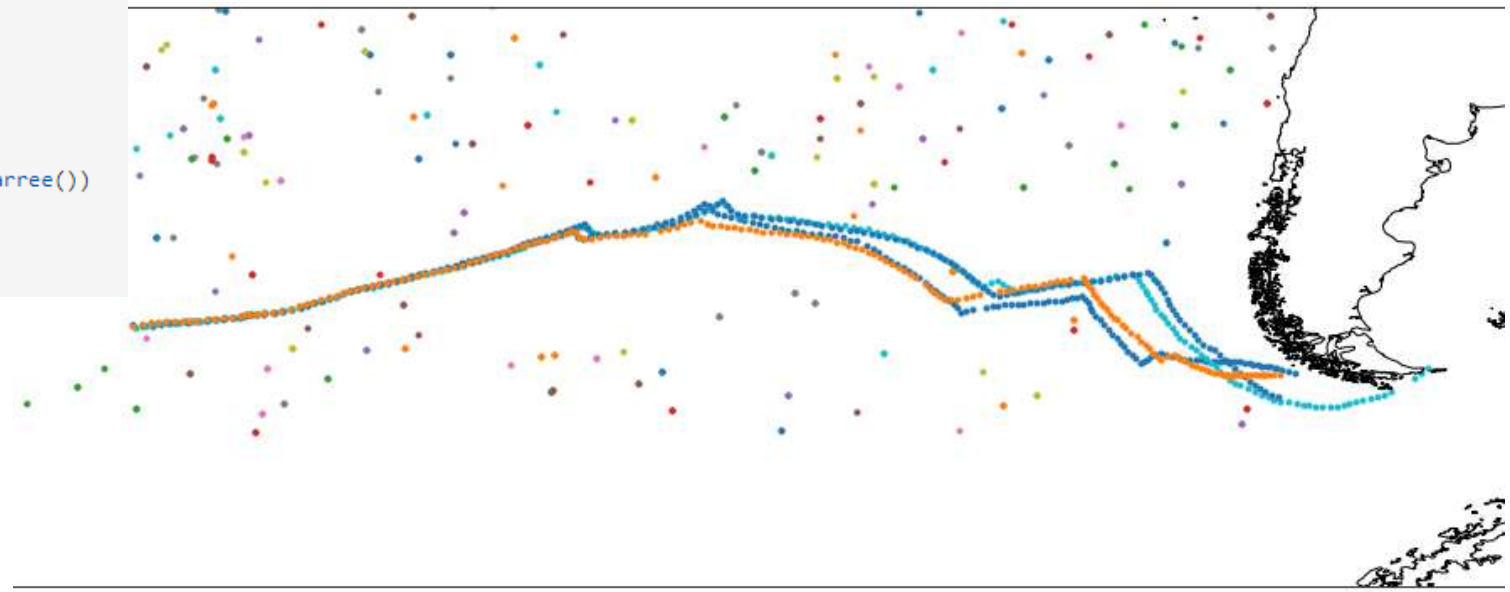
Select a region,

a period (20/03/2023 – 06/04/2023)

Compare Argo salinity with Salinity from sail boats Measured by a Ferrybox (OceanPack):

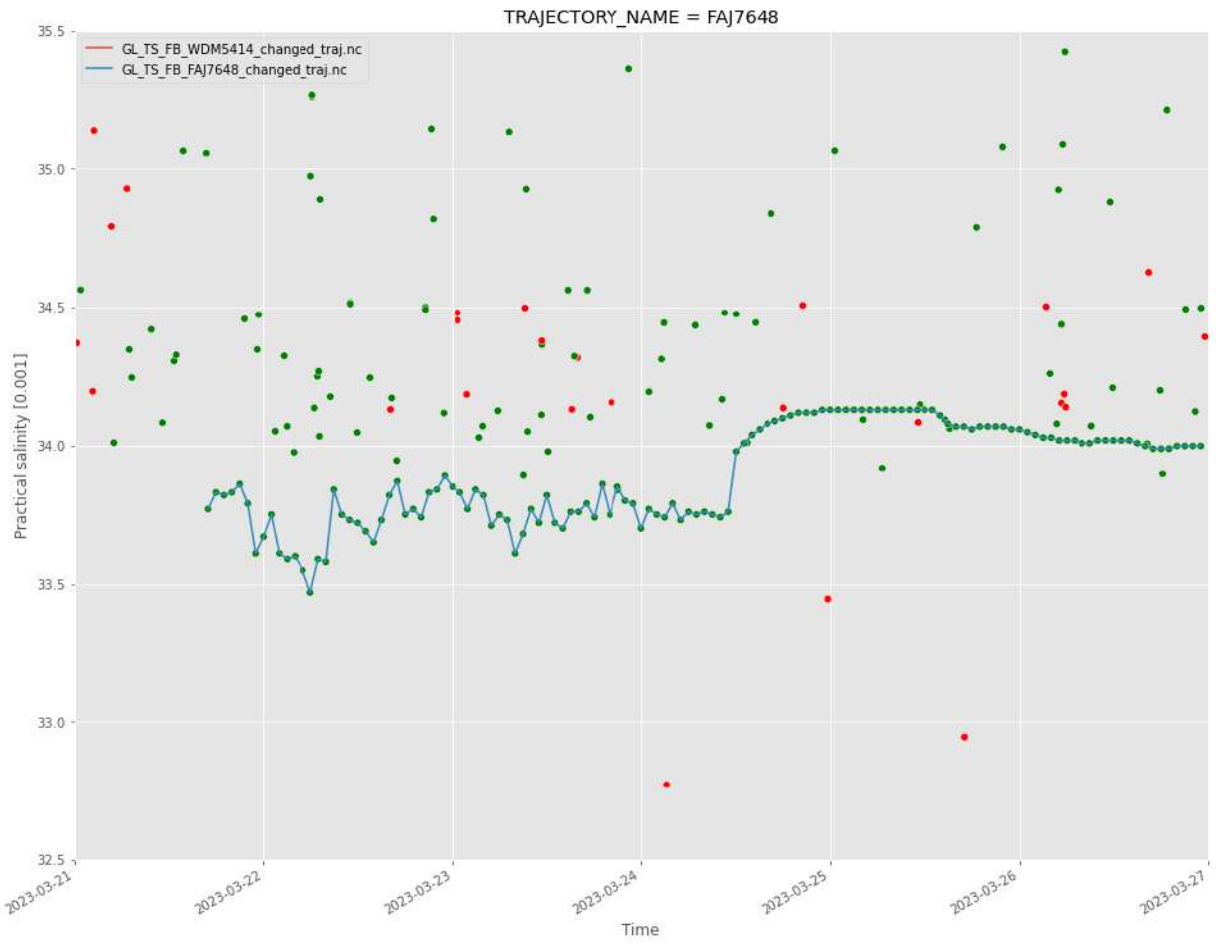
Manipulate your data with Xarray

```
fig = plt.figure(figsize=(16,12))
ax = plt.axes(projection=ccrs.PlateCarree())
ax.set_extent([-150,-60,-60,-33])
ax.coastlines('10m')
#imagery = OSM()
#ax.add_image(imagery, 10)
ax.set_xlabel('Longitude')
ax.set_ylabel('Latitude')

for f in files:
    datasets = xr.open_dataset(dir_path + '/' + f)
    if 'PSAL' in datasets:
        if 'PR_PF' in f:
            psal = datasets['PSAL'].isel(DEPTH=0)
        else : psal = datasets['PSAL'].isel(DEPTH=1)
    lat = datasets['LATITUDE']
    lon = datasets['LONGITUDE']
    ax.scatter(lon, lat, marker='.', transform=ccrs.PlateCarree())

plt.title('')
plt.show()
```



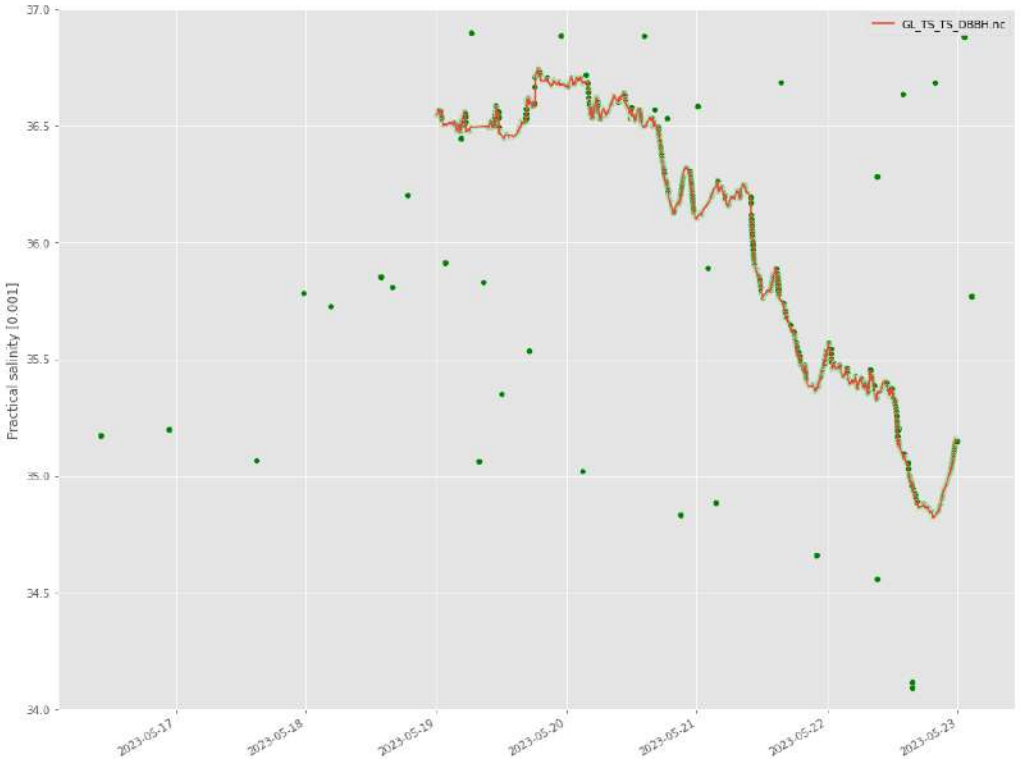
Compare salinity

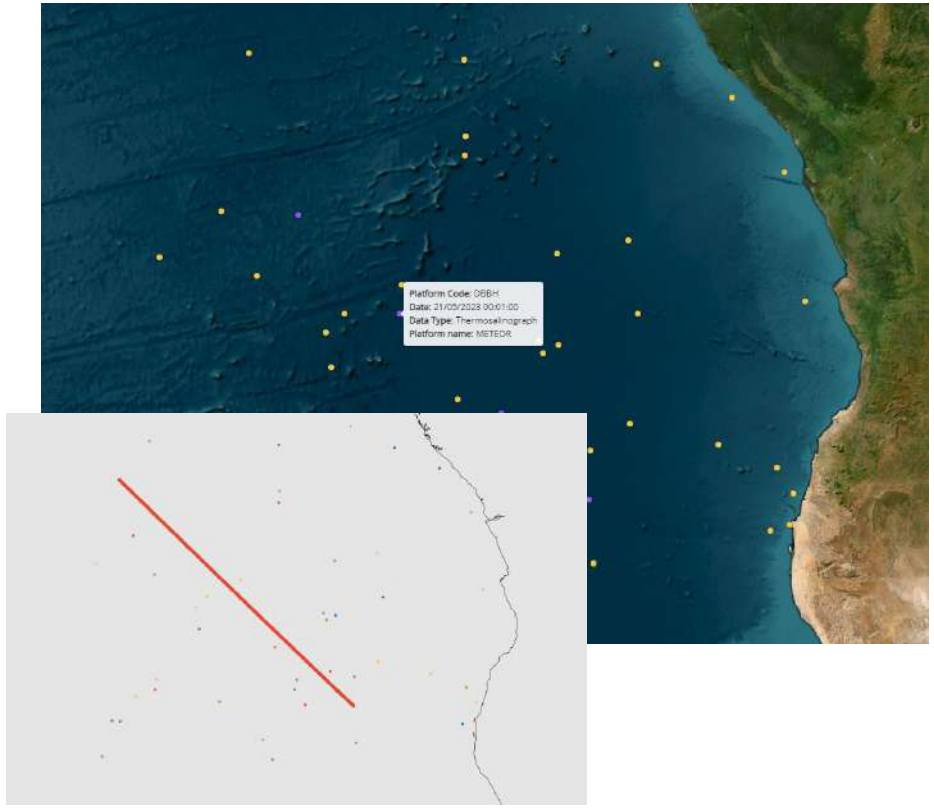
```
fig = plt.figure(figsize=(16,12))
ax = fig.add_subplot(111)
ax.set_ylim([32.5, 35.5])
ax.set_xlim([datetime.date(2023, 3, 21), datetime.date(2023, 3, 27)])
for f in files:
    datasets = xr.open_dataset(dir_path + '/' + f)
    if 'PSAL' in datasets:
        if 'PR_PF' in f:
            psal = datasets['PSAL'].isel(DEPTH=0)
            psal_qc = datasets['PSAL_QC'].isel(DEPTH=0)
            colors = np.where(psal_qc==1,'g','r')
            psal.plot.scatter(c=colors)
        else :
            psal = datasets['PSAL'].isel(DEPTH=1)
            psal_qc = datasets['PSAL_QC'].isel(DEPTH=0)
            colors = np.where(psal_qc==1,'g','r')
            if len(psal) >=2:
                psal.plot(label=f)
                psal.plot.scatter(c=colors)
plt.legend()
plt.show()
```



Using DBBH (Meteor – German R/V)

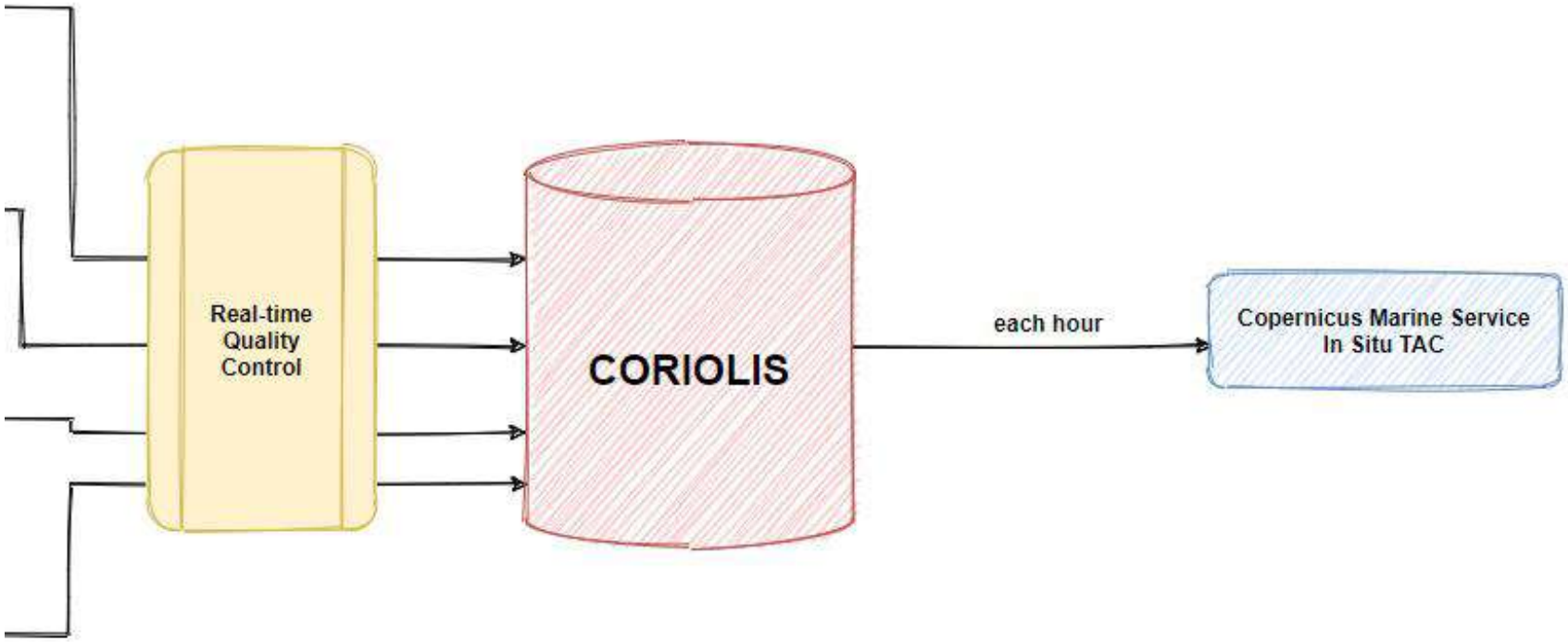
Same work but compared to Argo collocation algorithm from SNO SSS



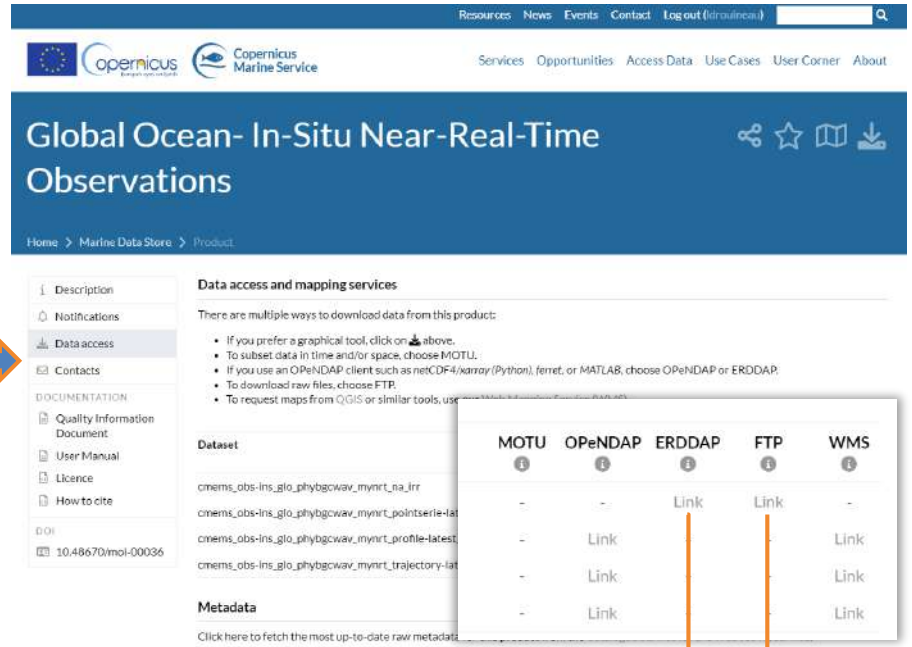


date	time	lon	lat	profiler	cycle	pres	sss	mean_ssps
20/05/2023	03:12:08	9.0297	-17.971	1902327	51	4	35.848	35.8426
21/05/2023	00:11:26	4.6129	-13.4941	1902427	51	4	36.583	36.4743
23/05/2023	05:47:40	-8.4949	-1.4226	6903110	237	4.8	35.444	35.1998

Copernicus Marine Service In Situ TAC



In Situ products: Data Access

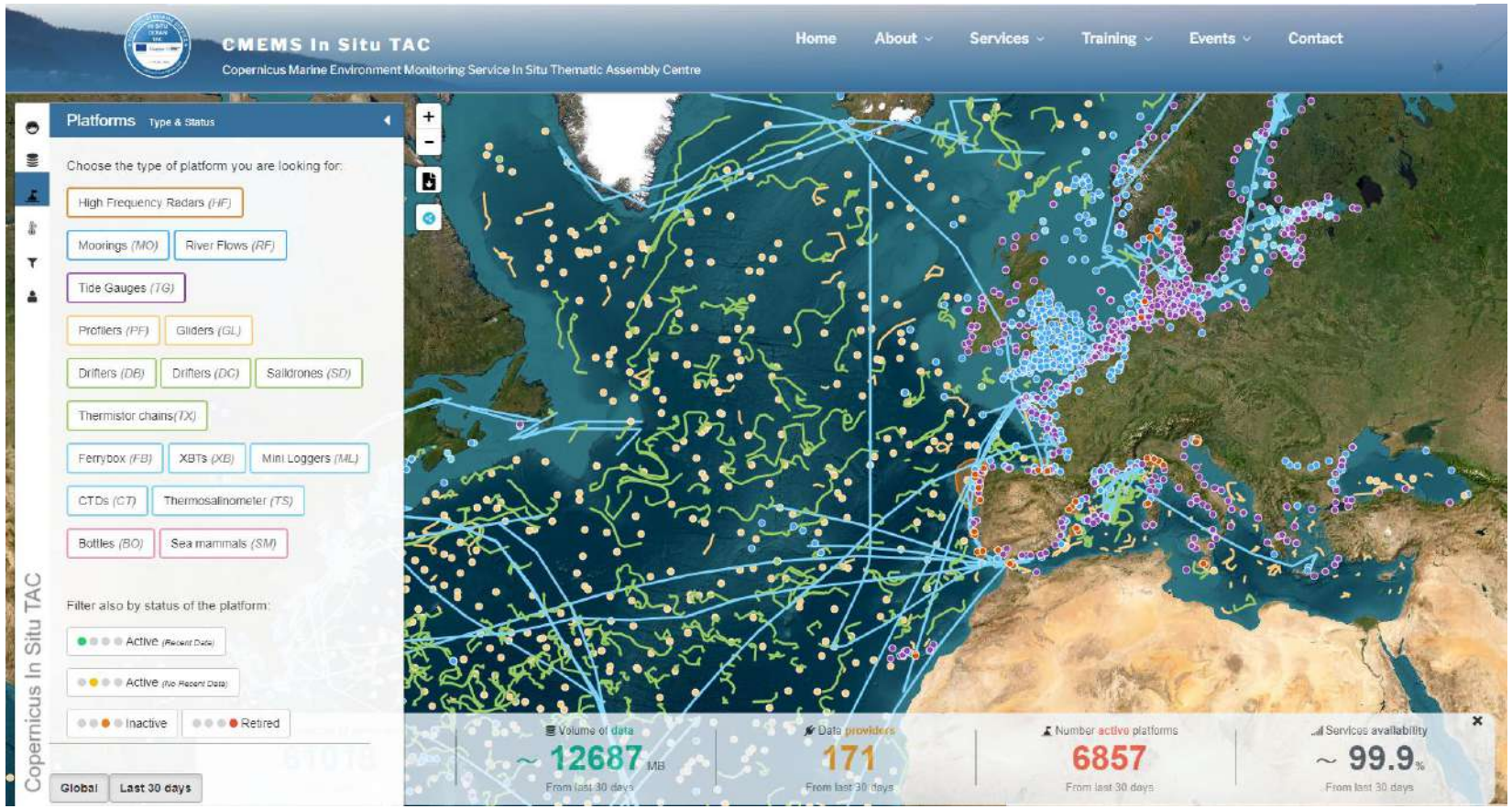


** ERDDAP:* ***only latest*** *files (latest 30 days)*

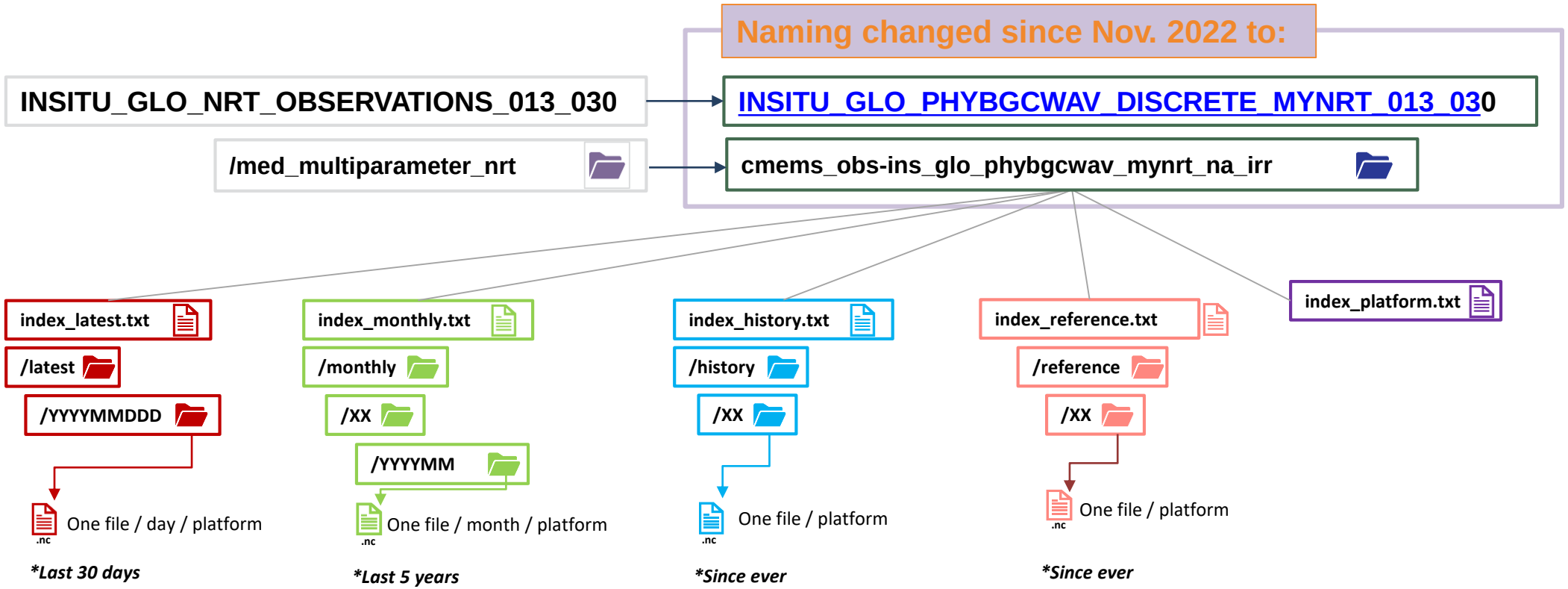
ftp://nrt.cmems-du.eu
/Core/INSITU_GLO_PHYBGCWAV_DISCRETE_MYNRT_013_033/cmems_obs-ins_glo_phybgcwav_mynrt_na_irr/

How to access In Situ Data ? – Copernicus Marine Service In Situ TAC

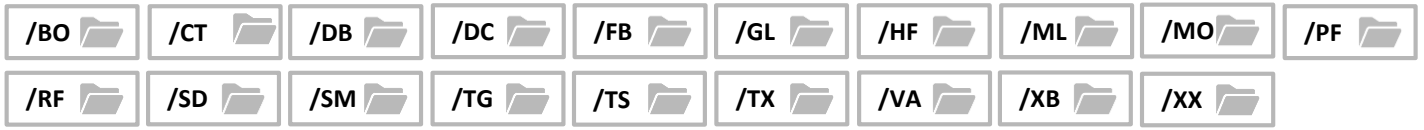
http://www.marineinsitu.eu/dashboard/



In Situ TAC FTP Folder structure



XX =



File naming

Latest directory: RR_XX_YY_CODE_YYYYMMDD.nc
Monthly directory: RR_XX_YY_CODE_YYYYMM.nc
History directory: RR_XX_YY_CODE_ZZZ.nc

Region bigram
- **GL Global**
- AR Arctic
- BO Baltic
- NO North West Shelf
- **IR IBI (Iberia-Biscay-Ireland)**
- **MO Mediterranean**
- BLK Black Sea

File types
- TS Time series
- PR Profile
- TV Total velocities (HF Radars)
- RV Radial velocities (HF Radars)
- WS Wave spectral data

Data types
- PF profiling floats
- GL gliders
- MO fixed buoys, mooring time series
- RF River flow
- HF HF radar
- DB drifting buoys
- DC drifting buoy reporting current
- CT oceanographic CTD profiles
- BO bottle data
- FB ferrybox
- SF towed CTD data (ex: scanfish)
- TG Tide gauge station
- TS thermosalinograph data
- XB XBT or XCTD profiles
- ML mini logger for fishery observing system
- SM Sea mammals data
- XX Not yet identified data type
- SD Saildrones
- VA Vessel mounted ADCPs

YYYYMMDD: year month day of observations
YYYYMM : year month of observations
ZZZ : only if different time samplings
.nc: NetCDF file name suffix

Organisation of the data

Product family	MY	MY-NRT
Update	Every 6 months	Every hour
Scope	Global	Global and regional
Quality Mode	Delayed mode	Delayed mode + Real Time (when delayed mode is not available)
Target	One topic (WAVE/ BGC/ TS/ UV/ Carbon/ SSH)	One or more topics
Products	INSITU_GLO_BGC_CARBON_DISCRETE_MY_013_050 INSITU_GLO_BGC_DISCRETE_MY_013_046 INSITU_GLO_PHY_SSH_DISCRETE_MY_013_053 INSITU_GLO_PHY_TS_DISCRETE_MY_013_001 INSITU_GLO_PHY_TS_OA_MY_013_052 INSITU_GLO_PHY_UV_DISCRETE_MY_013_044 INSITU_GLO_WAV_DISCRETE_MY_013_045	INSITU_ARC_PHYBGCWAV_DISCRETE_MYNRT_013_031 INSITU_BAL_PHYBGCWAV_DISCRETE_MYNRT_013_032 INSITU_BLK_PHYBGCWAV_DISCRETE_MYNRT_013_034 **INSITU_IBI_PHYBGCWAV_DISCRETE_MYNRT_013_033** **INSITU_MED_PHYBGCWAV_DISCRETE_MYNRT_013_035** INSITU_NWS_PHYBGCWAV_DISCRETE_MYNRT_013_036 **INSITU_GLO_PHYBGCWAV_DISCRETE_MYNRT_013_030** INSITU_GLO_PHY_TS_OA_NRT_013_002 INSITU_GLO_PHY_UV_DISCRETE_NRT_013_048 INSITU_GLO_UV_NRT_OBSERVATIONS_013_048

Documents associated with each product:

- **PUM** Product User Manual
- **QuID** Quality Information Document

Other documents (In Situ TAC): Format Manual, Parameters List, Useful Code…
http://www.marineinsitu.eu/documentation/

Let’s play with the data

```
user = 'testactivee' #type CMEMS user name
password = '********' #type CMEMS password

product_name = 'INSITU_GLO_PHYBGCWAV_DISCRETE_MYNRT_013_030' #type aimed In Situ product
dataset_name = 'cmems_obs-ins_glo_phybgcwav_mynrt_na_irr'
host = 'nrt.cmems-du.eu' #type aimed host (nrt.cmems-du.eu or my.cmems-du)
index_file = 'index_history.txt' #type aimed index file

aimed_parameter = 'VHM0'

 #selection criteria: spatial coverage
targeted_geospatial_lat_min = 47.0   # enter min latitude of your bounding box
targeted_geospatial_lat_max =  50.0   # enter max latitude of your bounding box
targeted_geospatial_lon_min = -6.0  # enter min longitude of your bounding box
targeted_geospatial_lon_max =  -2.00  # enter max longitude of your bounding box
targeted_bounding_box = box(targeted_geospatial_lon_min, targeted_geospatial_lat_min, targeted_geospatial_lon_max, targeted_geospatial_lat_max)


date_format = "%Y-%m-%dT%H:%M:%SZ"
ini = datetime.datetime.strptime('2023-03-01T00:00:00Z', date_format)
end = datetime.datetime.strptime('2023-04-30T23:59:59Z', date_format)
```

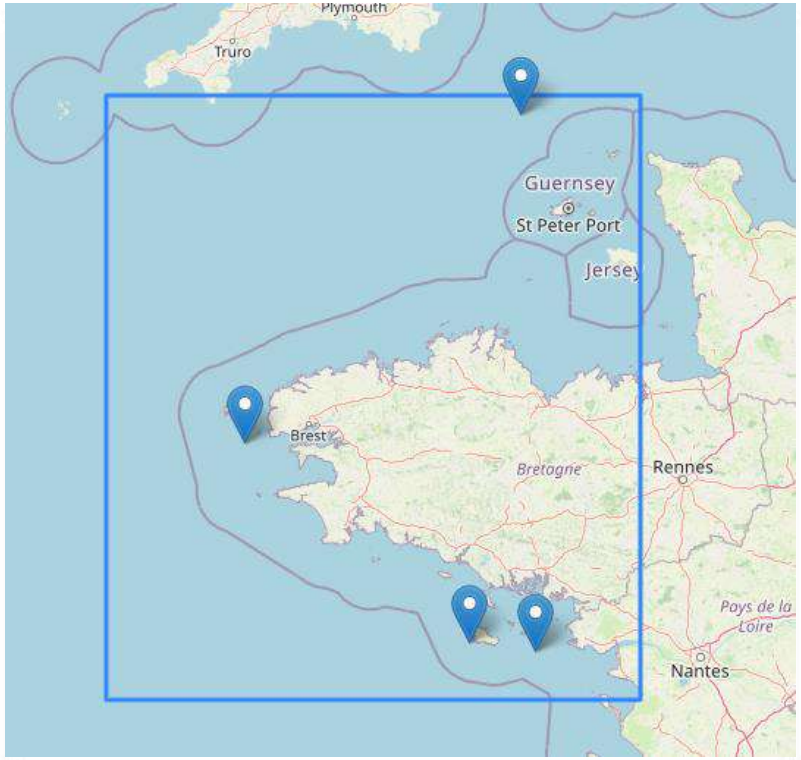
Parameter

Bounding box

Time period

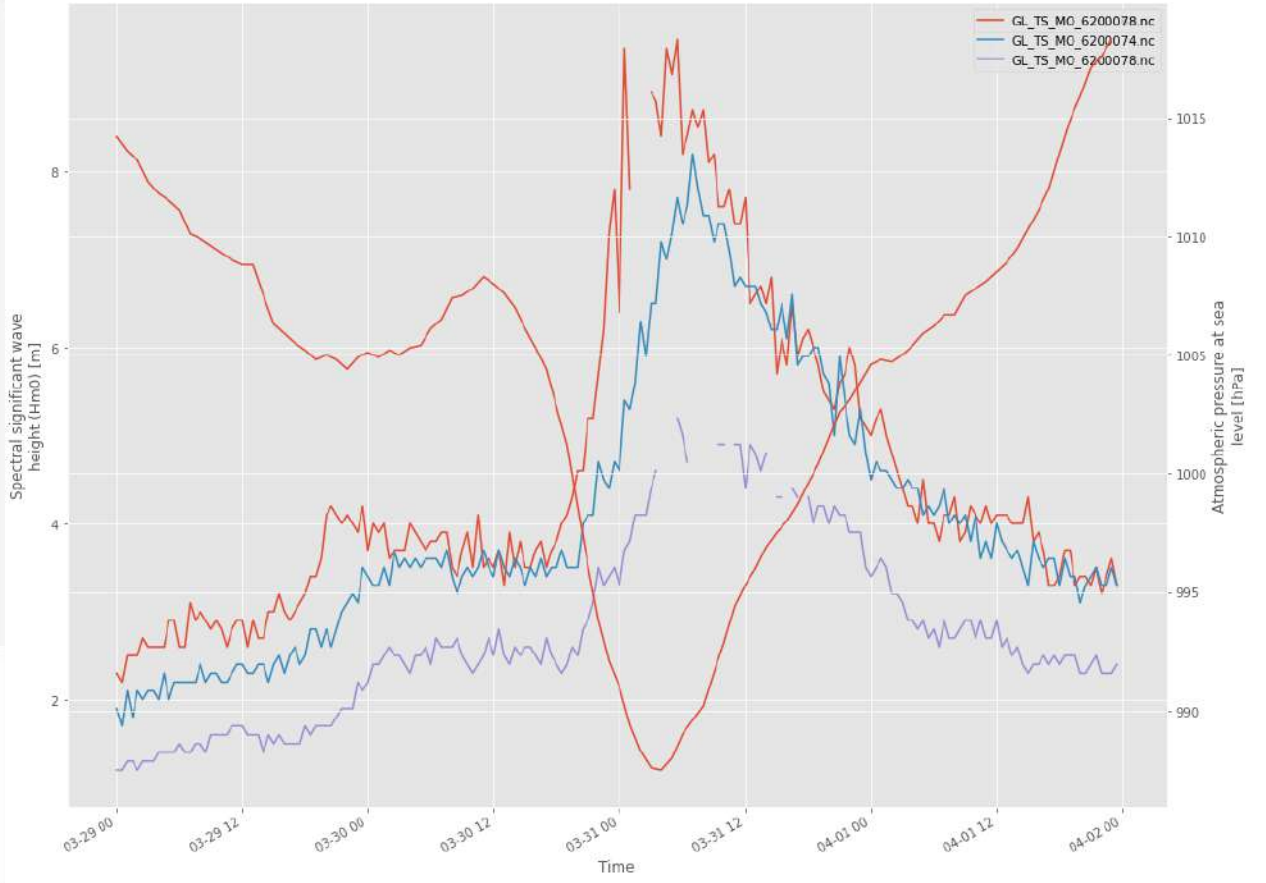
Some subsetting

```
#connect to CMEMS FTP
with ftputil.FTPHost(host, user, password) as ftp_host:

    #open the index file to read
    with ftp_host.open("Core"+'/'+product_name+'/'+ dataset_name + '/' + index_file, "r") as indexfile:

        #read the index file as a comma-separate-value file
        index = np.genfromtxt(indexfile, skip_header=6, unpack=False, delimiter=',', dtype=None, names=['catalog_id', 'file_name','geospatial_lat_min', 'geospatial_lat_max', 'geospat

        map = folium.Map(location=[targeted_bounding_box.centroid.y, targeted_bounding_box.centroid.x], zoom_start=7)
        folium.PolyLine([[targeted_geospatial_lat_min, targeted_geospatial_lon_min],[targeted_geospatial_lat_min, targeted_geospatial_lon_max],[targeted_geospatial_lat_max, targeted_

        #loop over the lines/netCDFs and download the most sutable ones for you
        for netCDF in index:

            #getting ftplink, filepath and filename
            ftplink = netCDF['file_name']
            filepath = '/'.join(ftplink.split('/')[3:len(ftplink.split('/'))])
            ncdf_file_name = ftplink[ftplink.rfind('/')+1:]

            #download netCDF if meeting selection criteria
            time_start = datetime.datetime.strptime(netCDF['time_coverage_start'], date_format)
            time_end = datetime.datetime.strptime(netCDF['time_coverage_end'], date_format)

            #download netCDF if meeting selection criteria
            parameters = netCDF['parameters']

            #download netCDF if meeting selection criteria
            geospatial_lat_min = float(netCDF['geospatial_lat_min'])
            geospatial_lat_max = float(netCDF['geospatial_lat_max'])
            geospatial_lon_min = float(netCDF['geospatial_lon_min'])
            geospatial_lon_max = float(netCDF['geospatial_lon_max'])
            bounding_box = box(geospatial_lon_min, geospatial_lat_min, geospatial_lon_max, geospatial_lat_max)


            geospatial = False
            if geospatial_lat_min >= targeted_geospatial_lat_min and geospatial_lat_max <= targeted_geospatial_lat_max \
            and geospatial_lon_min >= targeted_geospatial_lon_min and geospatial_lon_max <= targeted_geospatial_lon_max:
                geospatial = True

            if aimed_parameter in parameters and time_end > ini and geospatial is True:
                print(ncdf_file_name)
                if ftp_host.path.isfile(filepath):
                    folium.Marker(location = [bounding_box.centroid.y, bounding_box.centroid.x], popup=ncdf_file_name).add_to(map)
                    cwd = os.getcwd()
                    os.chdir(output_directory)
                    ftp_host.download(filepath, ncdf_file_name)  # remote, local
                    os.chdir(cwd)
map
```

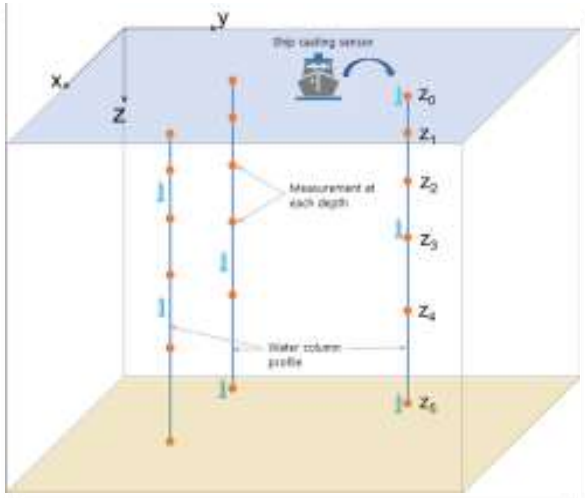


Visualise the data during Mathis storm

```
path = './output_cmems'
file = ['GL_TS_MO_6200069.nc','GL_TS_MO_6200074.nc','GL_TS_MO_6200078.nc']

fig = plt.figure(figsize=(16,12))
ax = fig.add_subplot(111)

for f in file:
    datasets = xr.open_dataset(path + '/' + f)
    vhm0 = datasets.VHM0.isel(DEPTH=0).sel(TIME=slice('2023-03-29', '2023-04-01'))
    #vzmx = datasets.VHM0.isel(DEPTH=0).sel(TIME=slice('2023-03-29', '2023-04-01'))
    vhm0.plot(label=f)
plt.legend()

file = ['NO_TS_MO_6200103.nc']

ax2 = ax.twinx()
datasets = xr.open_dataset(path + '/' + file[0])
atms = datasets.ATMS.isel(DEPTH=0).sel(TIME=slice('2023-03-29', '2023-04-01'))
#vzmx = datasets.VZMX.isel(DEPTH=0).sel(TIME=slice('2023-03-29', '2023-04-01'))
atms.plot(label=f)

plt.legend()
plt.show()
```



Future evolutions

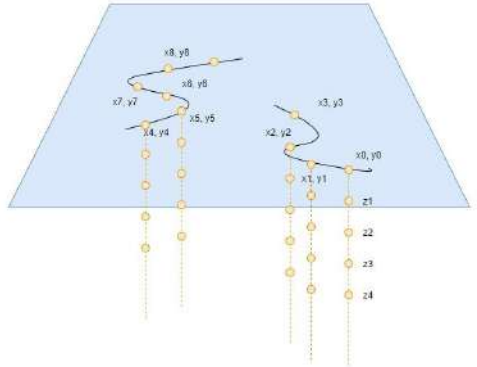


featureType	Description
timeSeries	a series of data points at the same spatial location with monotonically increasing times
trajectory	a series of data points along a path through space with monotonically increasing times
profile	an ordered set of data points along a vertical line at a fixed horizontal position and fixed time
trajectoryProfile	a series of profile features located at points ordered along a trajectory

Simplify access to external tools

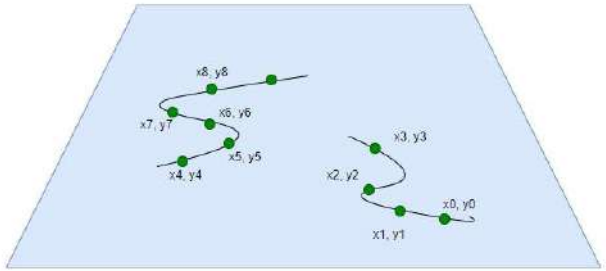
GL_TS_TS_FQBE_changed_traj.nc	Global Ocean - In Situ Observation Coperni…	Local File
DC_REFERENCE	Station/Location unique identifier in data c…	—
DEPH	Depth	1D
DEPH_QC	Depth quality flag	GeoTraj
LATITUDE	Latitude of each location	1D
LONGITUDE	Longitude of each location	1D
POSITION_QC	Position quality flag	GeoTraj
PSAL	Practical salinity	GeoTraj
PSAL_QC	Practical salinity quality flag	GeoTraj
TEMP	Sea temperature	GeoTraj

TrajectoryProfiles



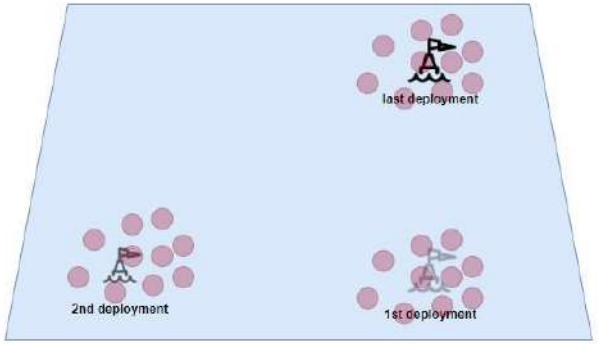
```
dimensions:
    TIME = 186 ;
    DEPTH = 69 ;

variables:
    string TRAJECTORY_NAME ;
        TRAJECTORY_NAME:cf_role = "trajectory_id" ;
        TRAJECTORY_NAME:long_name = "trajectory name" ;

    double TIME(TIME) ;
        time:standard_name = "time";
        time:long_name = "time of measurement" ;
        time:units = "days since 1970-01-01 00:00:00" ;
        time:missing_value = -999.9;
    float LONGITUDE:(TIME) ;
        LONGITUDE::standard_name = "longitude";
        LONGITUDE::long_name = "longitude" ;
        LONGITUDE::units = "degrees_east" ;
        LONGITUDE:axis = "X";
    float LATITUDE(TIME) ;
        LATITUDE:standard_name = "latitude";
        LATITUDE:long_name = "latitude" ;
        LATITUDE:units = "degrees_north" ;
    float DEPH(TIME, DEPTH) ;
        DEPH:standard_name = depth;
        DEPH:long_name = "Depth" ;
        DEPH:units = "m" ;
        DEPH:positive = "up" ;
        DEPH:axis = "Z" ;

    float TEMP(TIME, DEPH) ;
        TEMP:standard_name = "surface_temperature" ;
        TEMP:long_name = "skin temperature" ;
        TEMP:units = "Celsius" ;
        TEMP:coordinates = "TIME LONGITUDE LATITUDE DEPTH" ;

attributes:
  :featureType = "trajectoryProfile";
```

Trajectories




```
dimensions:
    TIME = 42;
    name_strlen = 23 ;

variables:
    char TRAJECTORY(name_strlen) ;
        trajectory:cf_role = "trajectory_id";

    double TIME(TIME) ;
        TIME:standard_name = "time";
        TIME:long_name = "Time" ;
        TIME:units = "days since 1970-01-01 00:00:00" ;
        TIME:axis = "T";
    float LONGITUDE:(TIME) ;
        LONGITUDE::standard_name = "longitude";
        LONGITUDE::long_name = "longitude" ;
        LONGITUDE::units = "degrees_east" ;
        LONGITUDE:axis = "X";
    float LATITUDE(TIME) ;
        LATITUDE:standard_name = "latitude";
        LATITUDE:long_name = "latitude" ;
        LATITUDE:units = "degrees_north" ;
    float DEPH(TIME) ;
        DEPH:standard_name = depth;
        DEPH:long_name = "Depth" ;
        DEPH:units = "m" ;
        DEPH:positive = "up" ;
        DEPH:axis = "Z" ;

    float TEMP(TIME) ;
        TEMP:standard_name = "air_temperature";
        TEMP:long_name = "air temperature" ;
        TEMP:units = "Celsius" ;
        TEMP:coordinates = "TIME LONGITUDE LATITUDE DEPH" ;

attributes:
    :featureType = "trajectory";
```

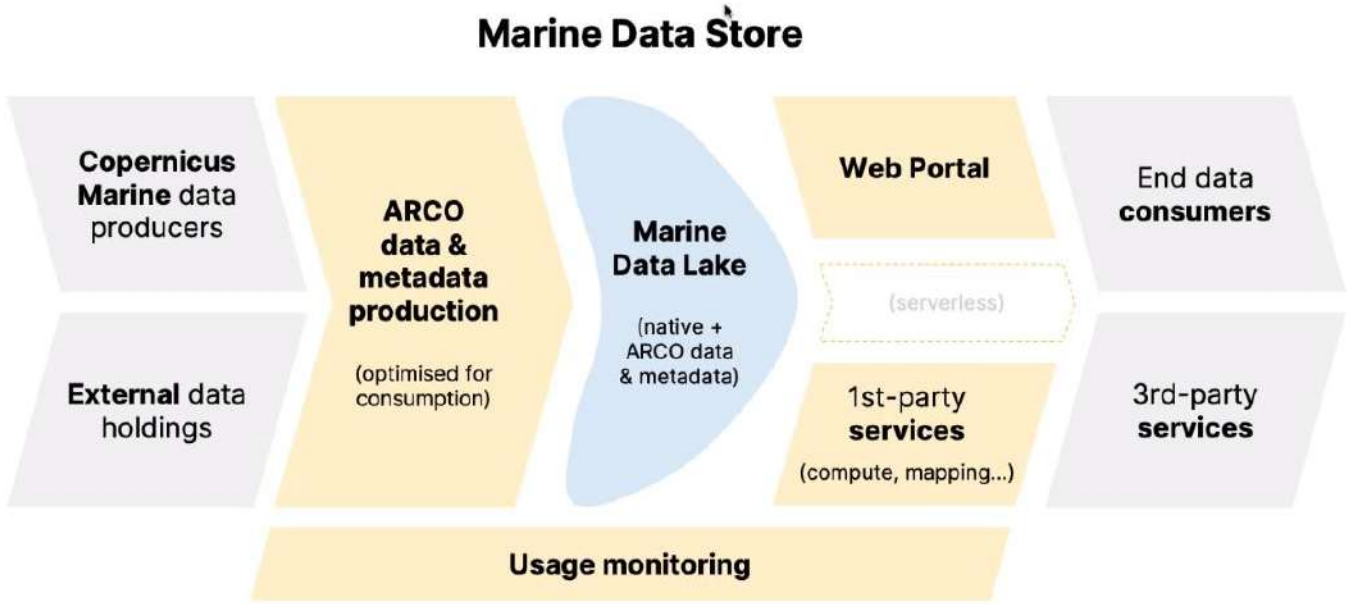
Timeseries

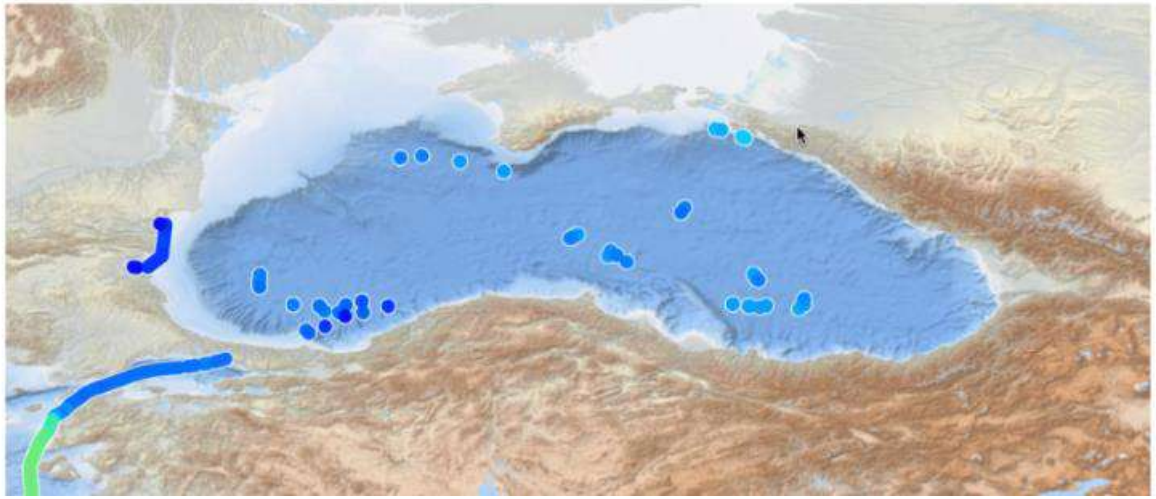


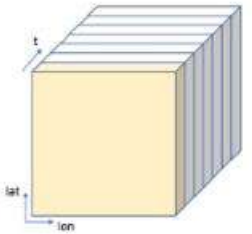

```
dimensions:
  TIME = 1234 ; // aggregated number of times in all timeseries
  DEPLOYMENT = 3 ; // aggregated number of deployments for stations
variables:
  float LONGITUDE ; // resp. LATITUDE
    LONGITUDE:standard_name = "longitude";
    LONGITUDE:long_name = "nominal station longitude";
    LONGITUDE:units = "degrees_east";
    LONGITUDE:axis = "X"; // resp. axis="Y" for LATITUDE
  float DEPLOY_LONGITUDE (DEPLOYMENT=3);
    DEPLOY_LONGITUDE:standard_name = "deployment_longitude";
    DEPLOY_LONGITUDE:long_name = deployment longitude";
    DEPLOY_LONGITUDE:units = "degrees_east";
  float PRECISE_LONGITUDE (TIME);
    PRECISE_LONGITUDE:standard_name = "longitude";
    PRECISE_LONGITUDE:long_name = "precise station longitude";
    PRECISE_LONGITUDE:units = "degrees_east";
  float DEPH(DEPTH=5) ;
    DEPH:long_name = "vertical distance above the surface" ;
    DEPH:standard_name = "depth" ;
    DEPH:units = "m";
    DEPH:positive = "down";
    DEPH:axis = "Z";
  char STATION_NAME(name_strlen) ;
    STATION_NAME:long_name = "station name" ;
    STATION_NAME:cf_role = "timeseries_id";
  int DEPLOYMENT(DEPLOYMENT=3) ;
    DEPLOYMENT:long_name = "index of the first time after (re)deployment" ;
    DEPLOYMENT:compress="TIME";
  double TIME(TIME) ;
    TIME:standard_name = "time";
    TIME:long_name = "time of measurement" ;
    TIME:units = "days since 1950-01-01 00:00:00" ;
    TIME:axis = "T";
  float TEMP(TIME, DEPTH) ;
    TEMP:standard_name = "air_temperature" ;
    TEMP:units = "Celsius" ;
    TEMP:coordinates = "TIME LONGITUDE PRECISE_LONGITUDE DEPLOY_LONGITUDE DEPH STATION_NAME" ;
data:
 STATION_NAME = "44088";
 LONGITUDE = 74.841;
 DEPLOY_LONGITUDE = 74.839, 74.842, 74.841;
 DEPLOYMENT = 24537, 24654, 26691;
```

new standard name

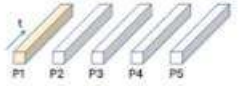
The new Marine Data Store







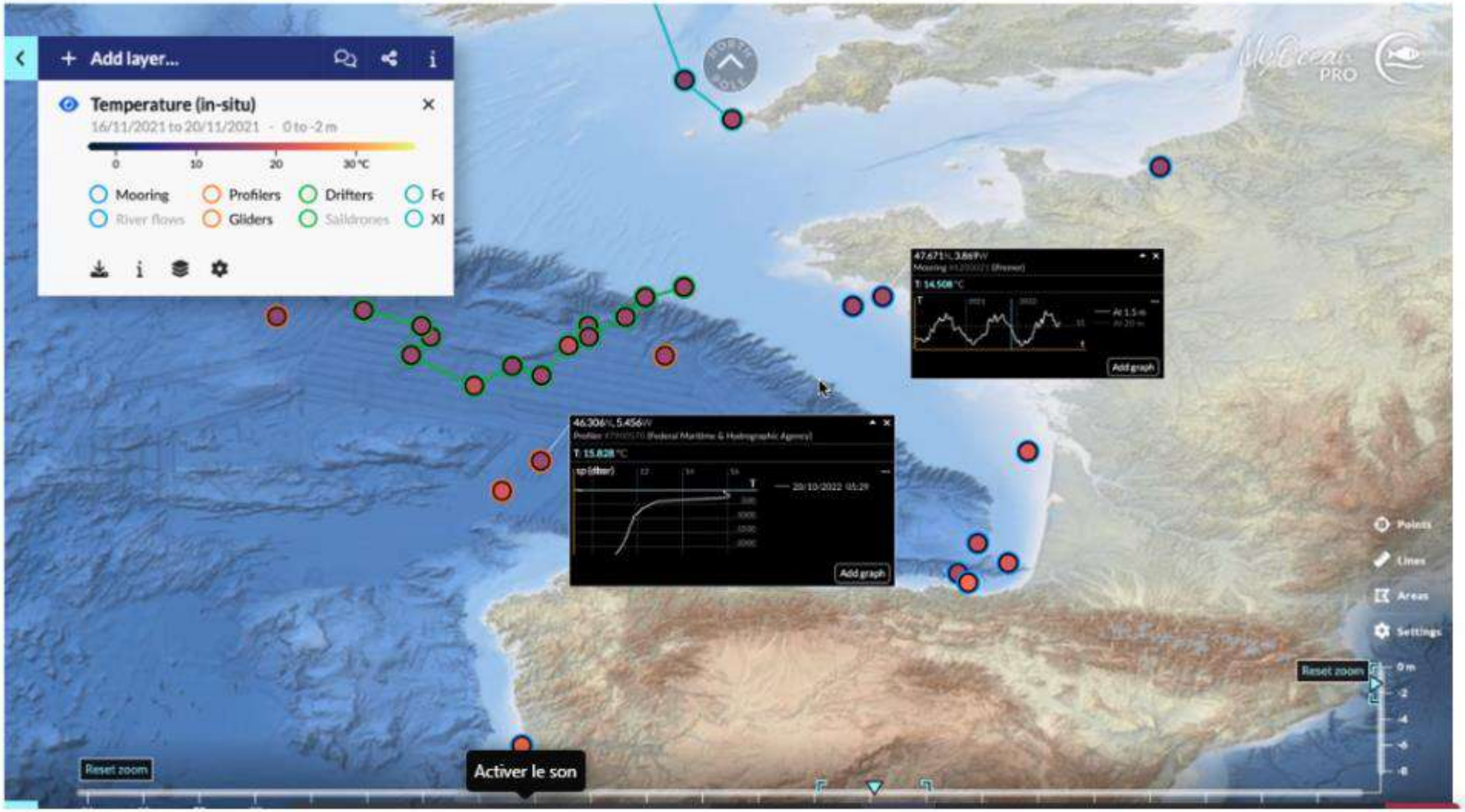
Time-chunked (T)



Platform-chunked (P)

+ platform directory

Visualize In Situ data + Satellite products + Model products



Merci !

Des questions ?