# Euroargodev *Cheat Sheet*

argopy

Join the community at github.com/euroargodev/argopy

## Fetching Argo data  `API`

Import the data fetcher, select an access point (region, float or profile) and trigger data or index download:

### A basic example

```
from argopy import DataFetcher

fetcher = DataFetcher().region([-75, -45, 20, 30,
                                0, 100,
                                '2011-01',
                                '2011-06'])
fetcher = DataFetcher().float([6902746, 6902755])
fetcher = DataFetcher().profile(6902746, [1,12])

fetcher.to_xarray()
fetcher.to_dataframe()
fetcher.data
fetcher.index
```

## User modes  `API`

argopy provides 3 user modes with different level of data post-processing:

- 🏄 **expert** mode: return all the Argo data, without any post-processing,

- 🛶 **standard** mode: simplifies the dataset, remove most of its jargon and return a priori good data, namely: QC=[1,2] & DM=[R,D,A]. This is the default mode.

- 🚣 **research** mode: simplifies the dataset to its heart, preserving only data of the highest quality for research studies, including studies sensitive to small pressure and salinity bias (e.g. calculations of global ocean heat content or mixed layer depth), namely: QC=1 & DM=D.

### Full session
```
import argopy
argopy.set_options(mode='expert')
```

### Temporary context
```
with argopy.set_options(mode='expert'):
    DataFetcher().profile(6902746, 34)
```

### Fetcher option
```
DataFetcher(mode='research').region([-75, -45,
                                     20, 30,
                                     0, 100])
```

## Data manipulation  `API`

Use methods from the **argo** xarray accessor

### Transformation

**Points vs profiles**
```
ds.argo.point2profile()
ds.argo.profile2point()
```

**Interpolation (pressure levels)**
```
std = [0,100,200,500]  # in db
ds.argo.interp_std_levels(std)
```

**Group-by pressure bins**
```
b = np.arange(0., 2000., 250.0)  # in db
ds.argo.groupby_pressure_bins(bins=b,
                              select='deep')
ds.argo.groupby_pressure_bins(bins=b,
                              select='random')
```

### Filters

**QC flags**
```
ds.argo.filter_qc(QC_list=[1,2],
                  QC_fields='all')
ds.argo.filter_qc(QC_list=1,
                  QC_fields='PSAL')
```

**Data modes**
```
ds.argo.filter_data_mode()
```

**OWC variables**
```
ds.argo.filter_scalib_pres(force='default')
```

### Additional variables

Complete your dataset with additional variables using the TEOS-10

```
ds.argo.teos10(['SA', 'CT', 'CNDC'])
```

## Data sources  `API`

**argopy** allows users to fetch Argo data from several sources:

- the **Ifremer erddap**. Updated daily, this database holds the complete dataset and is efficient for large requests

- a **GDAC server**. This could be one of the 2 ftps or the Ifremer http server.

- your **local data** copy of the GDAC. Useful to work offline.

- the **Argovis** server. Updated daily, provides access to QC=1 data only

## Argo meta data  `API`

### Index of profiles

Based on GDAC servers or local file, support: core, synthetic and bio profiles index

```
from argopy import ArgoIndex
ArgoIndex().N_RECORDS
ArgoIndex().to_dataframe()
ArgoIndex().search_lat_lon([-60, -55, 40, 45])
ArgoIndex().search_wmo([1901393, 6902755])
```
🏄

For a more user-friendly API, you can use the index fetcher:

```
from argopy import IndexFetcher
fetcher=IndexFetcher().region([-75, -45, 20, 30])
fetcher.to_xarray()
fetcher.to_dataframe()
fetcher.index
```
🛶

### Reference tables

Based on NERC Vocabulary Server (NVS) managed by the Argo Vocabulary Task Team (AVTT)

```
from argopy import ArgoNVSReferenceTables
ArgoNVSReferenceTables().tbl_name('R01')
ArgoNVSReferenceTables().tbl('R01')
ArgoNVSReferenceTables().all_tbl_name
ArgoNVSReferenceTables().all_tbl
```

### Deployment plan

Based on Ocean-OPS API, retrieve past and future plans

```
from argopy import OceanOPSDeployments
OceanOPSDeployments().to_dataframe()
OceanOPSDeployments([-90, 0,
                     0, 90]).to_dataframe()
OceanOPSDeployments().plot_status()
```

### ADMT Documentation

```
from argopy import ArgoDocs
ArgoDocs().list
ArgoDocs(35385)
ArgoDocs(35385).open_pdf(page=12)
ArgoDocs().search('CDOM')
```

### Full session
```
import argopy
argopy.set_options(src='gdac',
                   ftp='https://...')
```

### Temporary context
```
with argopy.set_options(src='argovis'):
    DataFetcher().profile(6902746, 34)
```

### Fetcher option
```
DataFetcher(src='erddap')
```

# Data visualisation
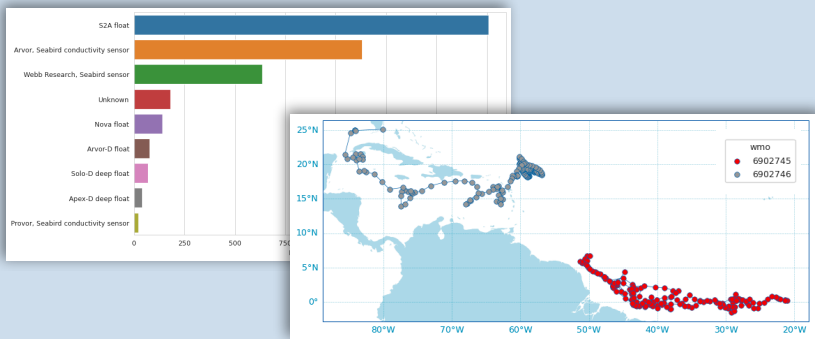
## From a Data or Index fetcher

```python
from argopy import DataFetcher, IndexFetcher
fetcher = DataFetcher()  # or IndexFetcher()
fetcher.region([-75, -45, 20, 30, 0, 100,
                '2015-01', '2020-01']).load()
```

### Trajectories
```python
fetcher.plot()
fetcher.plot('trajectory')
```

### Histograms on properties
```python
fetcher.plot('dac')
fetcher.plot('profiler')
```



## Scatter maps from Datasets

```python
from argopy.plot import scatter_map
scatter_map(ds)
scatter_map(ds, hue='DATA_MODE')
scatter_map(ds.isel(N_LEVELS=0), hue='PSAL_QC')

ds['year'] = ds['TIME.year']  # Add a variable
scatter_map(ds.isel(N_LEVELS=0),
            hue='year',
            cmap='Spectral_r',
            legend_title='Year of sampling')
```
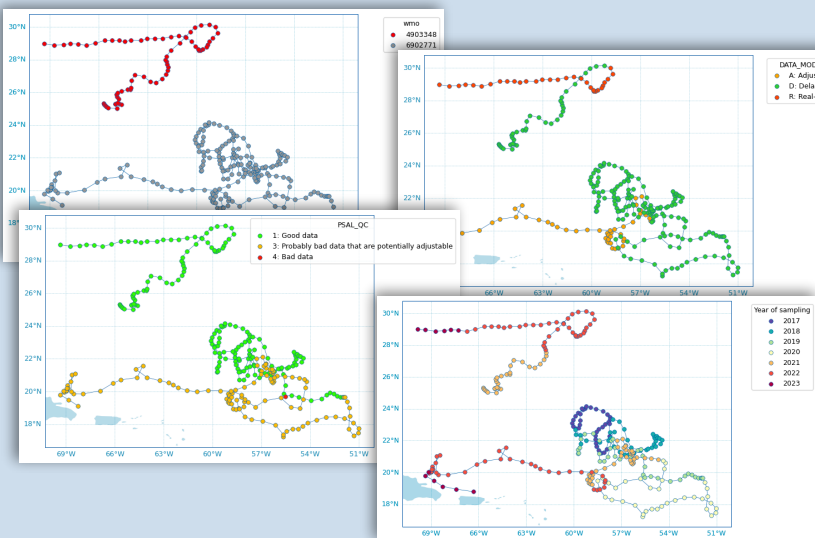


## Dashboards

For a collection of floats or profiles, get an easy and direct access to Euro-Argo, BGC, Ocean-Ops, Coriolis and Argovis dashboards
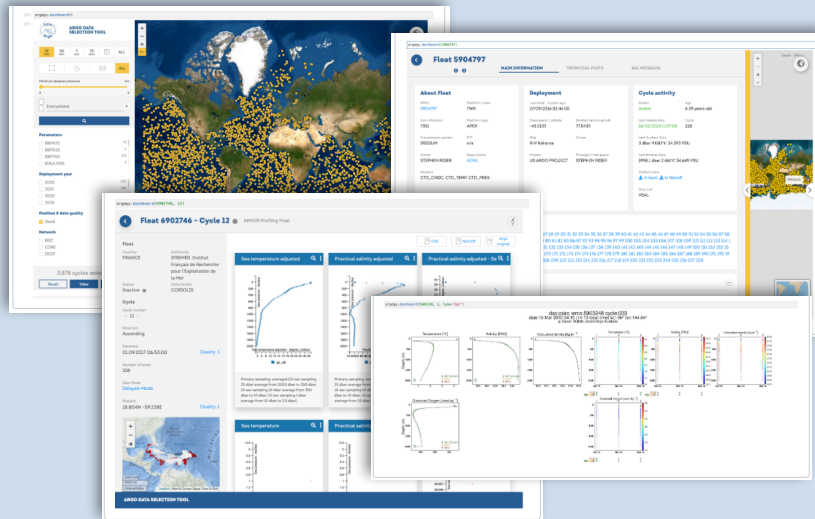
### From a fetcher
```python
DataFetcher().float(6902746).dashboard()
```

### or direct access
```python
from argopy import dashboard
dashboard()
dashboard(6902746)

dashboard(6902746, 12)
dashboard(5903248, 3, type='bgc')
```
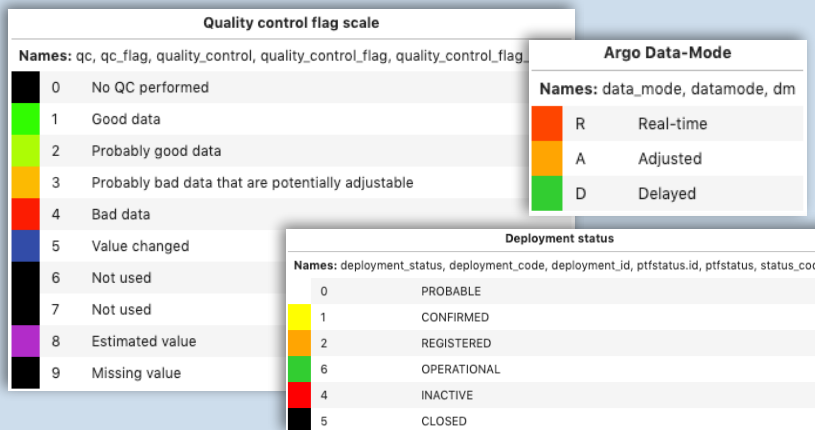
*By default, this will insert the dashboard in a notebook cell, but it can also return the url to open in your browser.*



## Argo color palettes

```python
from argopy.plot import ArgoColors
ArgoColors('data_mode')
ArgoColors('qc_flag')
ArgoColors('deployment_status')
```



# Data quality control

## Topography

Download a regional subset of the GEBCO 15" topography

```python
from argopy import TopoFetcher
ds = TopoFetcher([-65, -55, 10, 20],
                 cache=True).to_xarray()
```

## CLS Altimetry tests

Easily checkout CLS altimetry test figures for one or more floats

```python
from argopy import DataFetcher
fetcher.float([6902745,
               6902746])
fetcher.plot('qc_altimetry')
```

## Data sources for OWC

Prepare Matlab data source files for the OWC analysis.

```python
from argopy import DataFetcher
ds = DataFetcher(mode='expert')
     .float(6902766)
     .load().data
ds.argo.create_float_source('output_folder')
```

## Reference data for core

Using the Ifremer erddap, argopy provides access to the core reference dataset from past Argo profiles as well as from ship-based CTD

### Argo reference profiles
```python
fetcher = Datafetcher(src='erddap', ds='ref')
fetcher.region([-65, -55, 10, 20,
                0, 5000]).load()
ds = fetcher.data
```

### Ship-based reference CTD profiles
```python
from argopy import CTDRefDataFetcher
with argopy.set_options(user='jane_doe',
                        password='****'):
    fetcher = CTDRefDataFetcher([-65, -55,
                                 10, 20,
                                 0, 5000])

    ref_ctd = fetcher.to_xarray()
```