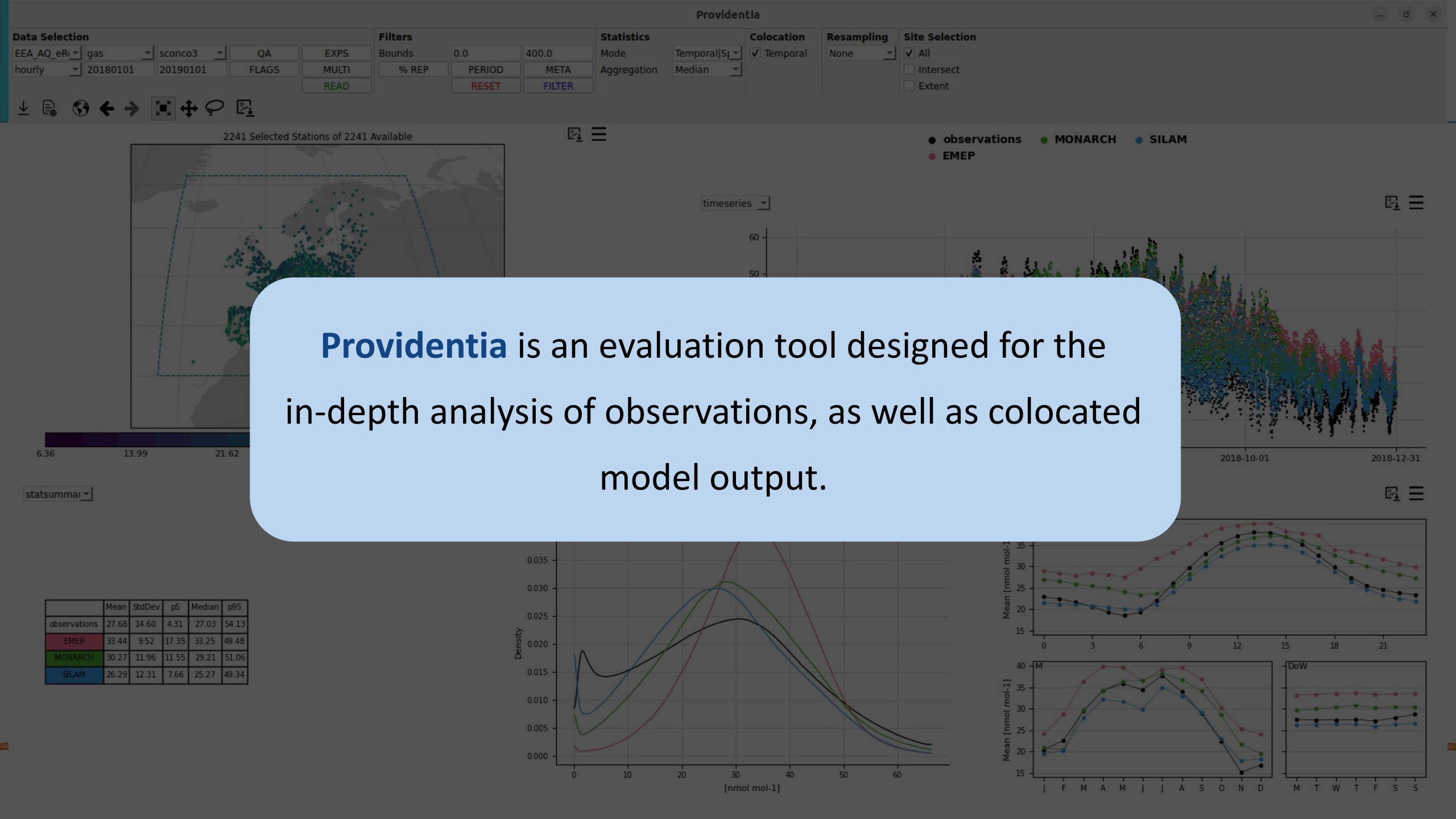
Providentia

Funded by the European Union

GA No 101160258

Alba Vilanova Cortezón, Paula Serrano Sierra

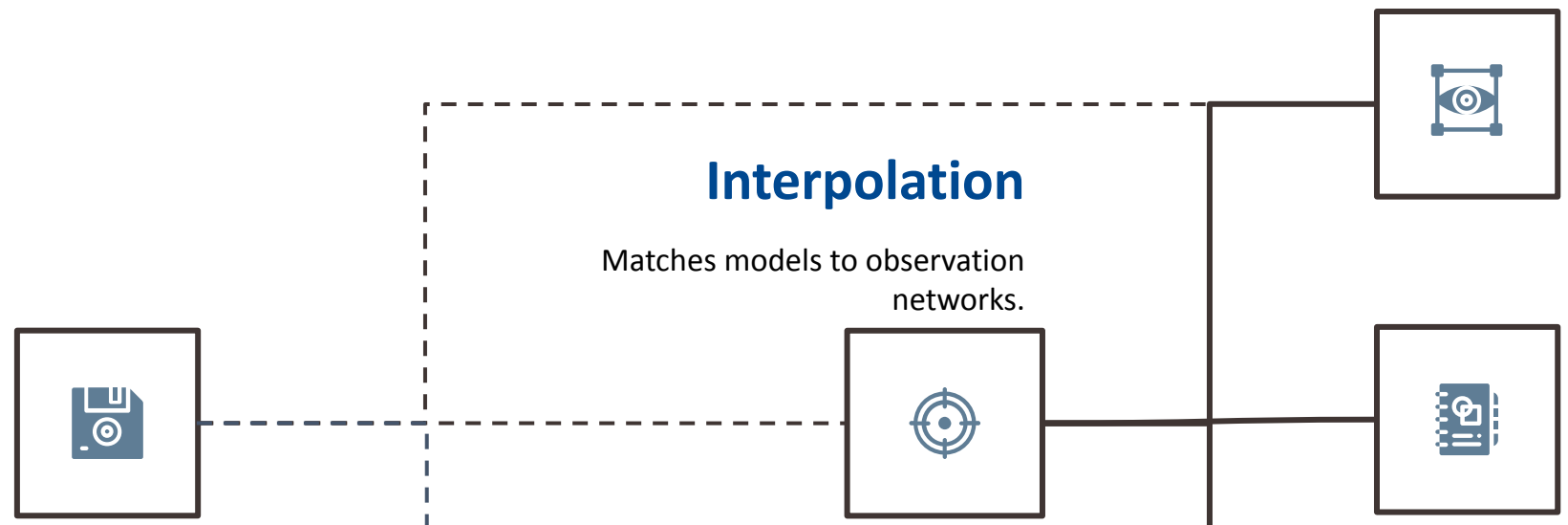
Introduction



Providentia is an evaluation tool designed for the in-depth analysis of observations, as well as colocated model output.

	Mean	StdDev	p5	Median	p95
observations	27.68	14.60	4.31	27.03	54.13
EMEP	33.44	9.52	17.35	33.25	49.48
MONARCH	30.27	11.96	11.55	29.21	51.06
SILAM	26.29	12.31	7.66	25.27	49.34

Structure



Download

Allows you to download the observations and models.

Interpolation

Matches models to observation networks.

Dashboard

To analyse in real time BSC model results, using observational data.

Report

Creates detailed pdf reports for in-depth analysis.

Library

Uses backend functions within Jupyter notebooks.

Set up

Prerequisites

Providentia is stable on macOS and Linux, where you need to install Git and Conda.

The software has not been designed to work on Windows. If you are a Windows user you must first choose one of these options:

- Use Windows Subsystem for Linux (WSL), only for Windows Pro users (**Recommended**). Instructions here.
- Use a virtual machine (e.g. Oracle, VirtualBox). Instructions here.
- Use Git Bash. Instructions here.

For more details, see the official documentation:
https://providentia.readthedocs.io/en/latest/Getting-started.html#prerequisites

Clone Providentia

Enter the project's GitHub page (https://github.com/bsc-es/providentia) and clone Providentia in your working directory:

```
git clone https://github.com/bsc-es/providentia
```

The latest releases are saved in the production branch, but we suggest users to stay in the **master** branch. In order to stay up-to-date just run this line from time to time:

```
git pull
```

Launching Providentia

Enter the project's GitHub page (https://github.com/bsc-es/providentia) and clone Providentia in your working directory:

```
cd providentia
```

The latest releases are saved in the production branch, but we suggest users to stay in the **master** branch. In order to stay up-to-date just run this line from time to time:

```
./bin/providentia
```

Main launch options

	Main purpose	Argument
Dashboard	On-the fly analysis	
Report	Detailed analysis in PDFs format	--report, --reports
Notebook	Open a Jupyter notebook	--notebook, --nb
Interpolation	Interpolate models to observational networks	--interp, --interpolate, --interpolation
Download	Download data to local machine	--download, --dl
Configuration file	Path or name (if in **configurations/** directory)	--conf, --config, --configuration
Section	Specify configuration file section	--section

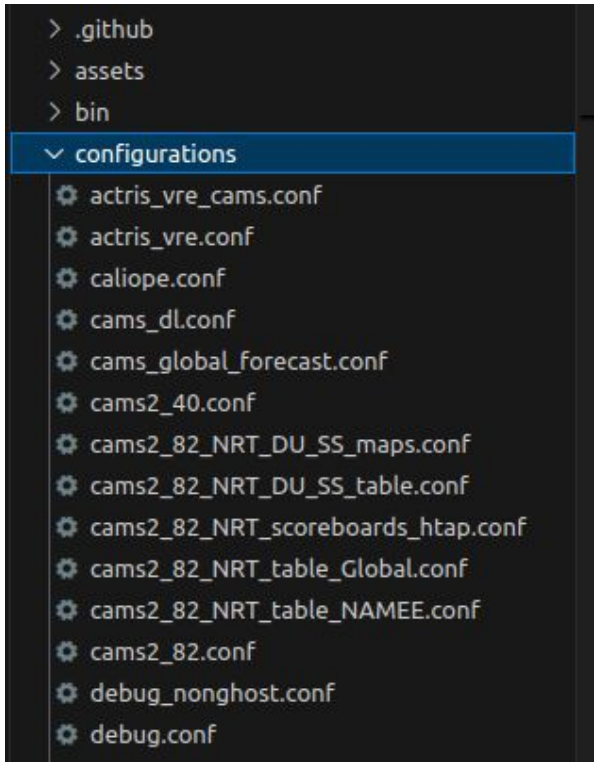
For more details, see the official documentation:
https://providentia.readthedocs.io/en/latest/Command-line-configuration.html

Configuration files

Create a configuration file

Configuration files specify which data to download (download mode), interpolate (interpolation mode), read and plot (dashboard, report and library modes) in the different modes.

By default, Providentia reads these files from the **configurations/** directory:



Structure of a configuration file

Configuration files are composed of:

- Sections (mandatory) act as the parent container for all parameters.
- Subsections (optional), contain metadata to filter data.

When reading and filtering data, subsections inherit all values defined in their parent section.

configurations/configuration_name.conf

Section ([name])

Subsection ([[name]])

```
[PRV_sconco3_CHIMERE]
ghost_version = 1.5
network = EBAS
species = sconco3
resolution = hourly
start_date = 20180101
end_date = 20180601
model = cams61_chimere_ph2-eu-000 (CHIMERE)

  [[Barcelona]]
  latitude = 39.8, 41.8
  longitude = 1.5, 2.5
  area_classification = keep: rural || remove: urban-suburban
```

Additional notes:

- Configuration fields are not order-sensitive.
- When defining a section-subsection pair in the command line, use:

--section=section·subsection

All available fields can be found here:
https://providentia.readthedocs.io/en/latest/Configuration-fields.html#configuration-fields

Main fields in a configuration file

The following fields are commonly defined in configuration files across all the different modes:

	Description
ghost_version	GHOST version used when a GHOST network is selected. (e.g. **1.5**)
network, observation, framework	Observational data. (e.g. **EBAS**)
model, models, experiments, experiment	Model or experiment IDs. Domain and ensemble information can also be included. (e.g. **a4j1-regional-000**)
species	Atmospheric components to analyse in BSC nomenclature. (e.g. **sconco3**)
resolution	Temporal resolution of the observations (e.g. **hourly**)
start_date	Start date. In YYYYMMDD format or YYYYMM for interpolation. (e.g. **20210101**)
end_date	End date. In YYYYMMDD format or YYYYMM for interpolation. (e.g. **20220101**)

Species

- Species refer to atmospheric components such as gases and aerosols.
- Providentia uses the BSC nomenclature defined in GHOST standards.

Chemical formula	Description	BSC parameter name
O_3	Surface concentration of ozone	sconco3
NO	Surface concentration of nitrogen monoxide	sconcno
NO_2	Surface concentration of nitrogen dioxide	sconcno2
SO_2	Surface concentration of sulphur dioxide	sconcso2
CO	Surface concentration of carbon monoxide	sconcco
CH_4	Surface concentration of methane	sconcch4

Model, ensemble and domain

In Providentia, models are defined using three key components:

- **Model**: Model ID (or experiment ID in MONARCH terminology). Examples: cams61_chimere_ph2, a8g1, osuite.
- **Domain**: Geographical region where the model is run. Examples: global, eu, cat.
- **Ensemble**: Simulation of the same model with different initial conditions or configurations to estimate uncertainty. Examples: 000, 001, 002.

You can specify **model, domain, and ensemble** either:

- All in one line.
- Separately in different fields.
- A combination of both.

For more details, check the models section in the official documentation: [Models configuration]

```
[PRV_sconco3_CHIMERE]
network = EBAS
species = sconco3
resolution = hourly
start_date = 20180101
end_date = 20180601
model = cams61_chimere_ph2-eu-000 (CHIMERE)

[PRV_sconco3_CHIMERE]
network = EBAS
species = sconco3
resolution = hourly
start_date = 20180101
end_date = 20180601
model = cams61_chimere_ph2 (CHIMERE)
domain = eu
ensemble = 000
```

Multiple sections in a configuration file

When launching the tool multiple sections, the modes behave differently:

- **Dashboard**: a specific section-subsection pair must be selected, it can be specified with the **--section** argument.
- **Library**: The first section in the configuration file is used by default, unless another section is explicitly specified.
- **Report, Download and Interpolation**: All sections (and subsections) in the configuration file are processed.

  If only one section needs to be used, it can be specified with the **--section** argument.

Section1

Section1·Subsection1

Section1·Subsection2

Section2

Section2·Subsection1

Section3

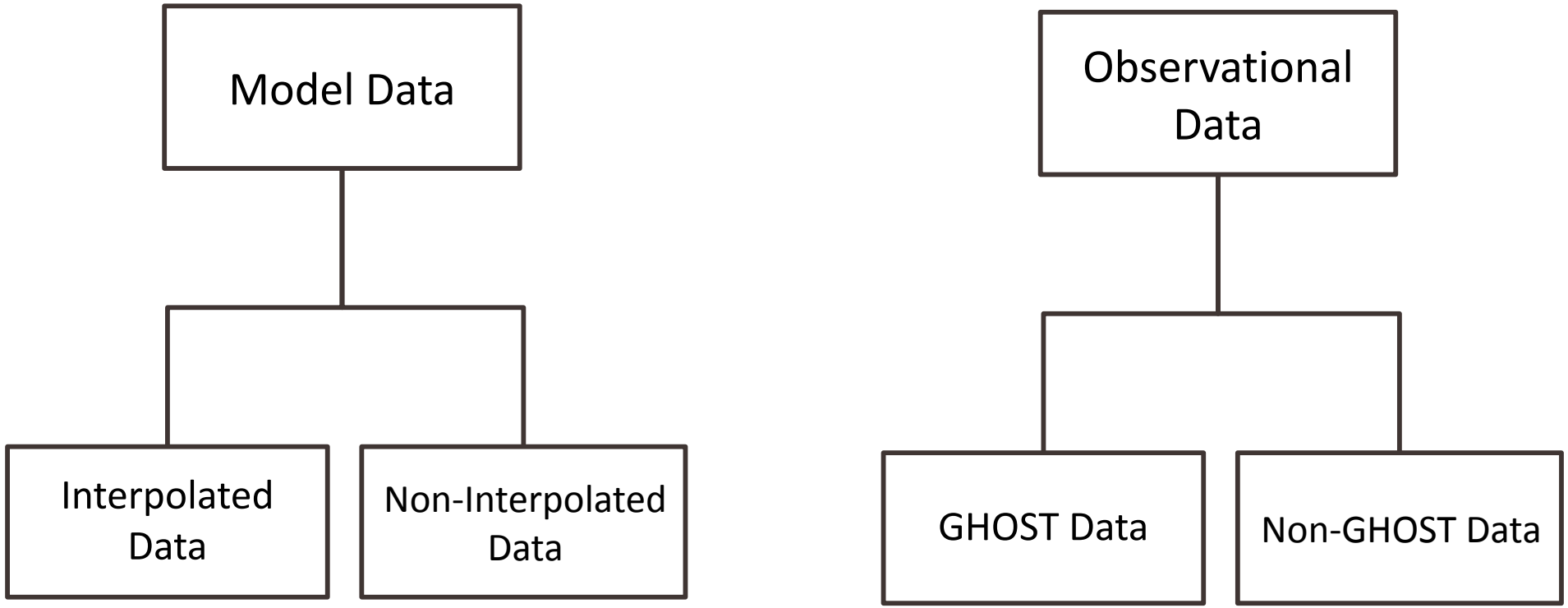
configurations/multiple_configuration_name.conf

```
[PRV_sconco3_CHIMERE]
network = EBAS
species = sconco3
resolution = hourly
start_date = 20180101
end_date = 20180601
models = cams61_chimere_ph2-eu-000 (CHIMERE)

  [[Spain]]
  latitude = 35, 44
  longitude = -9, 2

   [[British_Isles]]
   latitude = 50, 58
   longitude = -12, 2

[PRV_sconco3_EMEP]
network = EBAS
species = sconcno2
resolution = hourly
start_date = 20180101
end_date = 20190101
model = cams61_emep_ph2-eu-000 (EMEP)

  [[Spain]]
  country = keep: Spain ||

[PRV_sconco3_fc]
network = eea/eionet
species = sconco3
resolution = hourly
start_date = 20230101
end_date = 20230201
model = a59g-regional-000 (fc)
```

Data

Classification of data

To help you understand the data used in Providentia, we classify it into the following groups and subgroups:



Data directory format

Depending on the type of data, the directory structure changes:

- **Observational Data**
  - **GHOST**
    - {ghost_root}/{network}/**{ghost_version}**/{resolution}/{species}/{species_YYYYMM.nc}
  - **Non-GHOST**
    - {nonghost_root}/**{institution}/{network}**/{resolution}/{species}/{species_YYYYMM.nc}
- **Model Data**
  - **Non-Interpolated**
    - {mod_to_interp_root}/{model_id}/{domain}/{resolution}/{species}/{species_YYYYMM[DD].nc}
  - **Interpolated**
    - {mod_root}/{ghost_version}/{model_id}-{domain}-{ensemble}/{resolution}/{species}/**{network}**/{species_YYYYMM.nc}

Data formats

Providentia requires observational and model data to be of a defined file format to be able to be read.

Providentia's in-built download mode automatically formats data to be the correct format (more soon), but in case someone wants to read their own data with Providentia, we have substantive guides which aid this process.

For a guide on how to format observational data files to be read by Providentia, see here.

For a guide on how to format model data files to be read by Providentia, see here.

GHOST

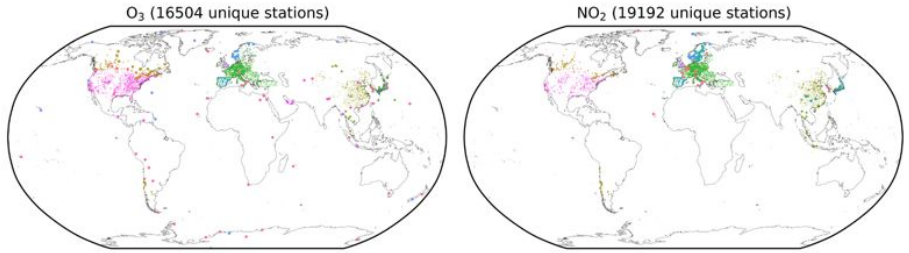
GHOST: Globally Harmonised Observations in Space & Time

N° Processed Measurements: **~7.3 billion**

N° Components: **227**

N° Networks: **38**

Temporal range: **1970-2023**



- ACTRIS
- BJMEMC
- Chile SINCA
- EEA AirBase
- EUSAAR
- Independent (EBAS)
- NADP AMNet
- NOAA-GGGRN
- US EPA AirNow DOS
- WMO GAW WDCGG
- AERONET v3 Level 1.5
- CAMP
- CNEMC
- EEA AQ e-Reporting
- HELCOM
- Japan NIES
- NADP AMoN
- OECD
- US EPA AQS
- WMO GAW WDCRG
- AERONET v3 Level 2.0
- Canada NAPS
- COLOSSAL
- EMEP
- HTAP
- Mexico CDMX
- NILU
- UK AIR
- US EPA CASTNET
- AMAP
- CAPMoN
- EANET
- EUCAARI
- IMPACTS
- MITECO
- NOAA-ESRL
- UK DECC
- WMO GAW WDCA

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Earth Syst. Sci. Data, 16, 4417–4495, 2024
https://doi.org/10.5194/essd-16-4417-2024

Open Access Earth System Science Data

GHOST: a globally harmonised dataset of surface atmospheric composition measurements

Dene Bowdalo[1], Sara Basart[1,a], Marc Guevara[1], Oriol Jorba[1], Carlos Pérez García-Pando[1,3], Monica Jaimes Palomera[4], Olivia Rivera Hernandez[4], Melissa Puchalski[5], David Gay[6], Jörg Klausen[7], Sergio Moreno[2], Stoyka Netcheva[2,b], and Oksana Tarasova[2]

[1]Barcelona Supercomputing Center, Barcelona, Spain
[2]World Meteorological Organization (WMO), Geneva, Switzerland
[3]Catalan Institution for Research and Advanced Studies (ICREA), Barcelona, Spain
[4]Secretaría del Medio Ambiente de la Ciudad de México (SEDEMA), Mexico City, Mexico
[5]Environmental Protection Agency (EPA), Washington, DC, United States
[6]National Atmospheric Deposition Program (NADP), Wisconsin State Laboratory of Hygiene, Madison, WI, United States
[7]Federal Office of Meteorology and Climatology MeteoSwiss, Zurich, Switzerland
[a]currently at: World Meteorological Organization (WMO), Geneva, Switzerland
[b]currently at: Environment and Climate Change Canada (ECCC), Toronto, Canada

Correspondence: Dene Bowdalo (dene.bowdalo@bsc.es)

Received: 7 October 2023 – Discussion started: 27 March 2024
Revised: 30 July 2024 – Accepted: 1 August 2024 – Published: 7 October 2024

Abstract. GHOST (Globally Harmonised Observations in Space and Time) represents one of the biggest collections of harmonised measurements of atmospheric composition at the surface. In total, 7 275 148 646 measurements from 1970 to 2023, of 227 different components from 38 reporting networks, are compiled, parsed, and standardised. The components processed include gaseous species, total and speciated particulate matter, and aerosol optical properties.

The main goal of GHOST is to provide a dataset that can serve as a basis for the reproducibility of model evaluation efforts across the community. Exhaustive efforts have been made towards standardising almost every facet of the information provided by major public reporting networks, which is saved in 21 data variables and 163 metadata variables. Extensive effort in particular is made towards the standardisation of measurement process information and station classifications. Extra complementary information is also associated with measurements, such as metadata from various popular gridded datasets (e.g. land use) and temporal classifications per measurement (e.g. day or night). A range of standardised network quality assurance flags is associated with each individual measurement. GHOST's own quality assurance is also performed and associated with measurements. Measurements pre-filtered by the default GHOST quality assurance are also provided.

In this paper, we outline all steps undertaken to create the GHOST dataset and give insights and recommendations for data providers based on the experiences gleaned through our efforts.

The GHOST dataset is made freely available via the following repository: https://doi.org/10.5281/zenodo.10637449 (Bowdalo, 2024a).

non-GHOST

Providentia can read data non-GHOST observational data, the only limitation is the data will be have limited filtering features, specifically:

- No GHOST QA flags
- No network flags
- No native representativity filters
- No period filters
- Limited metadata

On the dashboard this is reflected via greyed out buttons:

Setting data paths

Providentia knows where data lives via one key file: **settings/data_paths.yaml**

There are multiple headings in the file which set **observational** and **model** data paths on different machines, e.g. **local** and BSC machines (**mn5**, **nord4**, etc.).

Under each heading, the following paths are set:

- **ghost_root :** Observational GHOST data
- **nonghost_root:** Observational non-GHOST data
- **mod_root:** Interpolated model data
- **mod_to_interp_root:** Non-interpolated model data

```
local:
  ghost_root: ~/data/providentia/obs/ghost
  mod_root: ~/data/providentia/mod
  mod_to_interp_root: ~/data/providentia/mod_to_interp
  nonghost_root: ~/data/providentia/obs/nonghost
```

If you are using Linux inside Windows (through WSL, VirtualBox or Git Bash) you will need to modify the paths to point to your Windows paths.

Download

Download

If Providentia is cloned on a local machine, no data is available initially. Data must be downloaded before running any analysis or visualisation.

To download data, add the **--download** or **--dl** option together with a configuration file:

```
./bin/providentia --dl --config=example.conf
```

By default, all sections in the configuration file are processed. If only one section is required, it can be specified with the **--section** argument:

```
./bin/providentia --dl --config=example.conf --section=section1
```

Types of downloads

Providentia supports several types of download:

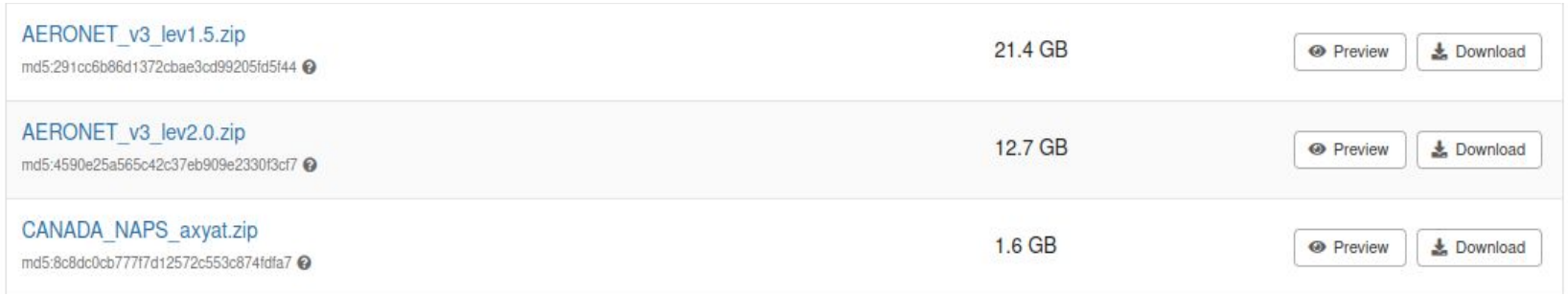
- GHOST observational data from Zenodo
- ACTRIS observational data from NILU Thredds servers
- CAMS non-interpolated model data from the Atmosphere Data Store

GHOST Zenodo Download

Download GHOST observational data from Zenodo

GHOST observational networks uploaded to a Zenodo repositories. An account is not required to use this feature.

Currently available versions are 1.5 and 1.5.1. To see the list of available networks, check the available networks section in the official documentation.



For more details, see the official documentation:
https://providentia.readthedocs.io/en/latest/Zenodo-download.html

Steps to download

Steps to download GHOST observational data:

1. **Include the network in your configuration file**
   - You must include at least one network in your configuration.
   - If the configuration file contains only observation(s) and no model(s) there is no need to set the download mode, network data will be downloaded automatically.
2. **Set the download mode**
   - Answer **“both”/”obs”** to the prompt:
     - *Which type of data do you want to download? Observational, modelled or both?*
   - Or set **dl_mode** to **both** or **obs.**
3. **Set the observational data source**
   - Answer **“n”** to the prompt:
     - *Do you want to download observational data from the BSC remote machine? (Otherwise, GHOST observational data will be retrieved from Zenodo)*
   - Or set **dl_ghost_source** to **zenodo**.

Configuration file

[ZENODO]
ghost_version = 1.5
network = EBAS-EUCAARI
species = pm10
resolution = daily
start_date = 200912
end_date = 201001
dl_ghost_source = zenodo

ACTRIS Download

Download of ACTRIS network data from Thredds

Observational data can be downloaded from the ACTRIS framework through THREDDS. An account is not required to use this feature.

Species names are accepted using both the ACTRIS naming convention and the BSC naming convention. Examples:

- sconco3 → ozone mass concentration
- sconcno3 → aerosol particle nitrate mass concentration

There is a total of 105 available species. Find the whole list here.

For more details, see the official documentation:
https://providentia.readthedocs.io/en/latest/ACTRIS-download.html

Steps to download

Steps to download ACTRIS observational data:

1. **Include the framework in your configuration file**
   - You must include the **actris/actris** framework in your configuration file.
   - If the configuration file contains only observation(s) and no model(s) there is no need to set the download mode, framework data will be downloaded automatically.
2. **Set the download mode**
   - Answer **“both”/”obs”** to the prompt:
     - *Which type of data do you want to download? Observational, modelled or both?*
   - Or set **dl_mode** to **both** or **obs.**
3. **Decide whether to update the Thredds filetrees**
   - Answer **“y” or “n”** to the prompt:
     - *File containing information of the files available in Thredds for {actris_parameter} ({info_path}) already exists. Do you want to update it?*
   - Or set **dl_thredds_update** to either **“True”** or **“False”**

Configuration file

[ACTRIS]
framework = actris/actris
species = ozone mass concentration
resolution = hourly
start_date = 20180101
end_date = 20180201
dl_thredds_update = False

ADS CAMS Download

Download of non-interpolated model data from the Atmosphere Data Store (ADS)

Non-interpolated model output data provided by **CAMS can be downloaded** from the Atmosphere Data Store

An ECMWF account is required to use this feature. For a tutorial on how to create and set up your account, see the Account setup in Atmosphere Data Store page.

For more details, see the official documentation:
https://providentia.readthedocs.io/en/latest/CAMS-download.html

Steps to download

Steps to download CAMS model data:

1. **Include one of the models in your configuration file**
   - You must specify the model as **cams_analysis**, **cams_forecast** or **cams_reanalysis** in your configuration file.
   - If the configuration file contains only observation(s) and no model(s) there is no need to set the download mode, network data will be downloaded automatically.
2. **Set the download mode**
   - Answer **“both”/”mod”** to the prompt:
     - *Which type of data do you want to download? Observational, modelled or both?*
   - Or set **dl_mode** to **both** or **mod.**
3. **Indicate that you want non-interpolated data**
   - Answer **“n”** to the prompt:
     - *Model data was detected in the configuration file. Do you want to download the interpolated version? (Otherwise, the non-interpolated model data will be downloaded)*
   - Or set **dl_interpolated** to **False**.

Configuration file

```
[CAMS]
model = cams_analysis_dehm
species = sconcno2
start_date = 20250701
end_date = 20250702
domain = regional
resolution = hourly
```

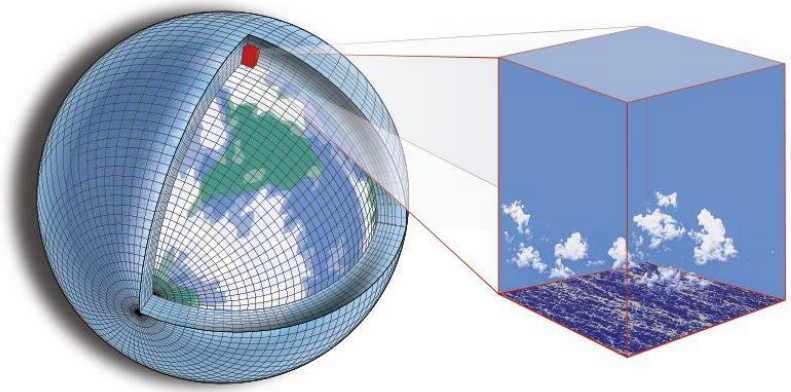
Interpolation

What does the interpolation do?

Observations: Data taken at fixed stations in specific locations.

Model: Data usually provided on a regular grid with fixed spacing.

Interpolation: The mathematical process of estimating the model value at the exact location of an observation station.



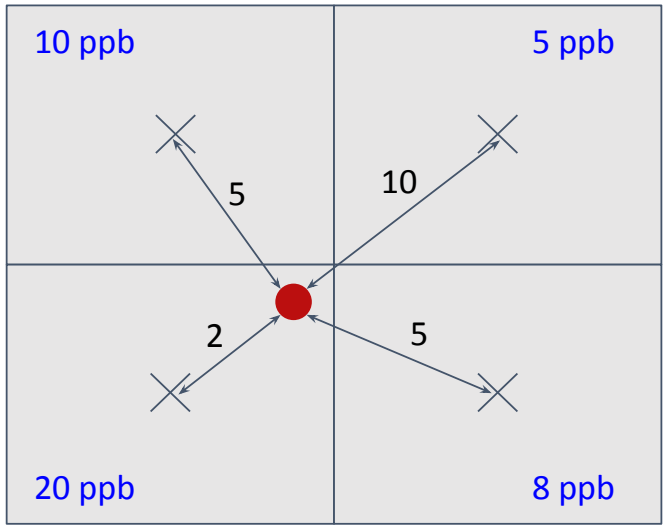
Inverse distance weighted interpolation

Inverse Distance Weighted interpolation estimates the value at an observation station using the **N nearest grid cells**.

Cells that are closer to the station have more influence.

N neighbours can be set via **interp_n_neighbours = N** in the configuration file. The default is **4.**

For 4 closest neighbours (default):



Weights

½, ⅕, ⅕, □
= 0.5, 0.2, 0.2, 0.1

Weighted average = **14.1 ppb**
Nearest neighbour = **20 ppb**

Mismatching observational and model temporal resolutions

Model simulations may have a different time resolution than observations.

Providentia requires both to match for evaluation.

How Providentia handles this:

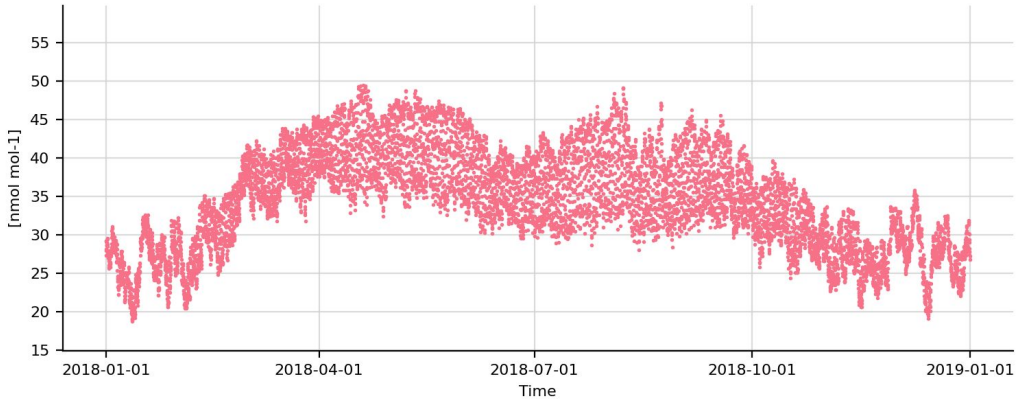
1. If model and observation resolutions already match → use the model data as is.
2. If the model resolution is finer → downsample the model data.
3. If the model resolution is coarser → upsample the model data.

The observational resolution is fixed using the **resolution** argument.

You can manually choose the model resolution using the **model_resolution** variable.

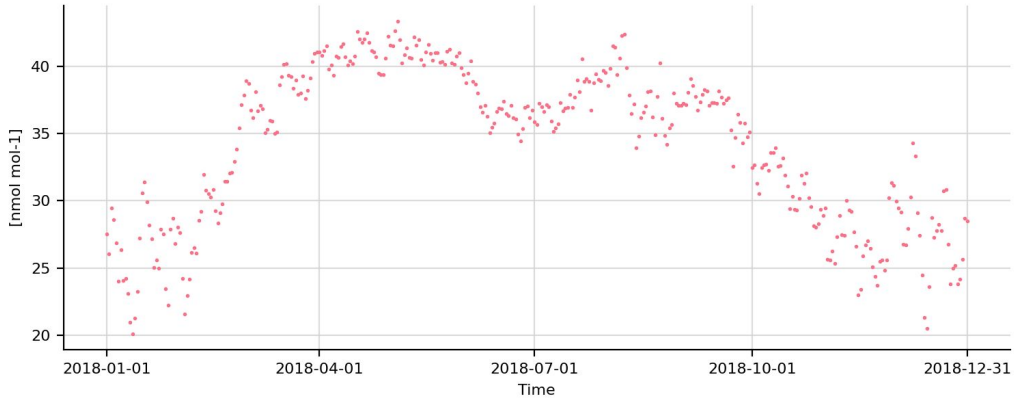
Downsampling model resolution

If we have a model resolution that is finer than our observational resolution, this can be handled via an operation called **downsampling**. This is as simple as averaging the finer model resolution to a coarser resolution. This can be set via **interp_model_downsampling = [option]** in the configuration file, with the two valid options being **mean** and **median.** The default is **mean**.



Hourly

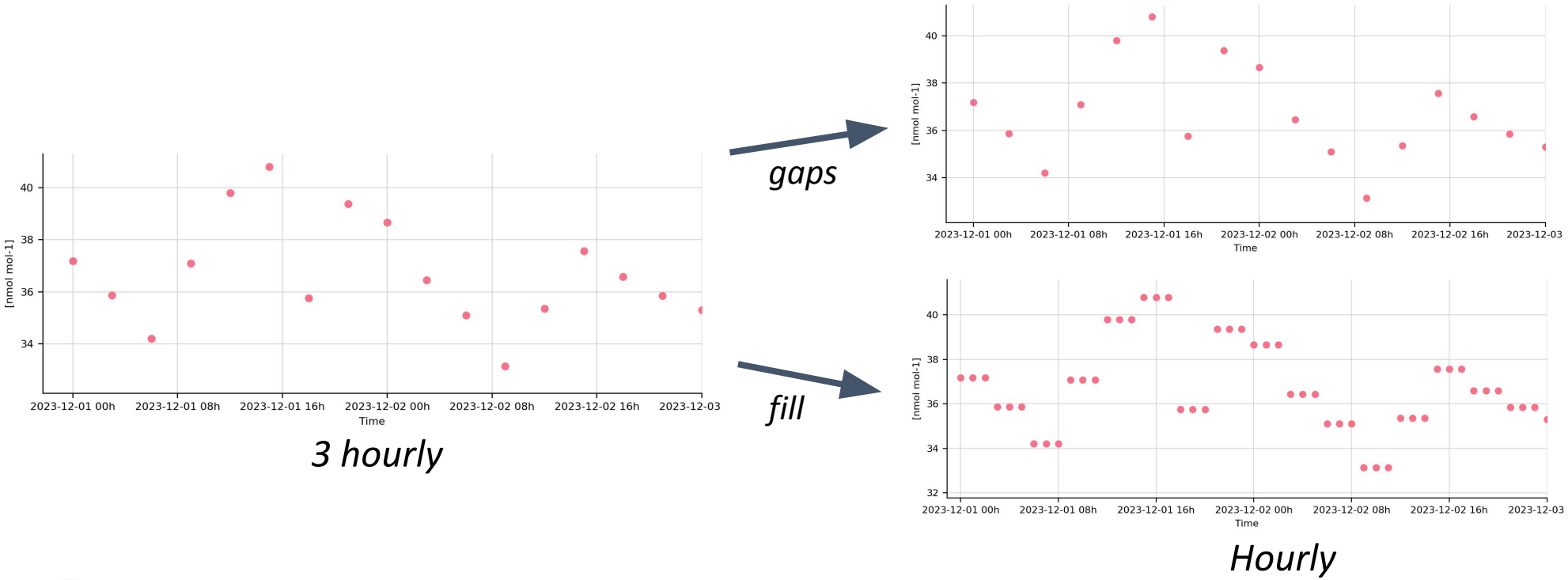
mean



Daily

Upsampling model resolution

If we have a model resolution that is coarser than our observational resolution, this can be handled via an operation called **upsampling**. This can be set via **interp_model_upsampling = [option]** in the configuration file, with the two valid options being **fill** and **gaps.** The default is **fill.**



Mapping species

Sometimes the **component** provided in the **observations** does not exactly match the component given in a **model**. For example a model may provide information in 4D (e.g. time, lon, lat, alt), whereas in the observations it may be given in 3D (e.g. time, lon, lat). For these cases Providentia provides some allowance so that data can be mapped together.

This information can be set in **settings/mapping_species.yaml**, with the syntax for doing so being:

[observational component]

- **[model component]**

The typical use cases for mapping are:

1. 3D observational component to 4D model component. This can be either be through an additional vertical dimension in the model, e.g. sconco3 : vmro3 ; or an additional binned dimension in the model, e.g. vconcaerobin1 : loaddu
2. Observational component to fractional model component, e.g. pm10 : pm10du

settings/mapping_species.yaml

```
sconco3
  - vmro3
vconcaerobin1
  - loaddu
pm10
  - pm10du
```

Run interpolation

If **--interp**, **--interpolate** or **--interpolation** is added as a launch option together with the mandatory configuration file on the command line, the interpolation will begin:

```
./bin/providentia --conf=configuration_name.conf --interp
```

configurations/configuration_name.conf

```
[A005]
start_date = 20180101
end_date = 20190101
resolution = hourly
species = sconco3
network = EBAS
models = cams61_chimere_ph2-eu-000 (CHIMERE)
```

Note:

Each interpolation job covers one full month, thus for the given **start_date** and **end_date** only **YYYYMM** is considered, **DD** will be ignored.

Output logs

When running the interpolation locally, information will be output in realtime to the command line as it goes on. Other logs will be generated in multiple different folders inside the ***logs/interpolation*** folder:

management_logs/ Informs on the output of the overall interpolation. Go here first.

interpolation_logs/ Stores output information of individual interpolations.

For more information about the interpolation, check the official documentation:

https://providentia.readthedocs.io/en/latest/Interpolation.html

Dashboard

Providentia

Data Selection

EBAS | gas | sconco3 | QA | MODS
hourly | 20180101 | 20190101 | FLAGS | MULTI
READ

Filters

Bounds | 0.0 | 400.0
% REP | PERIOD | META
RESET | FILTER

Statistics

Mode: Temporal|S
Aggregation: Median

Colocation

☑ Temporal

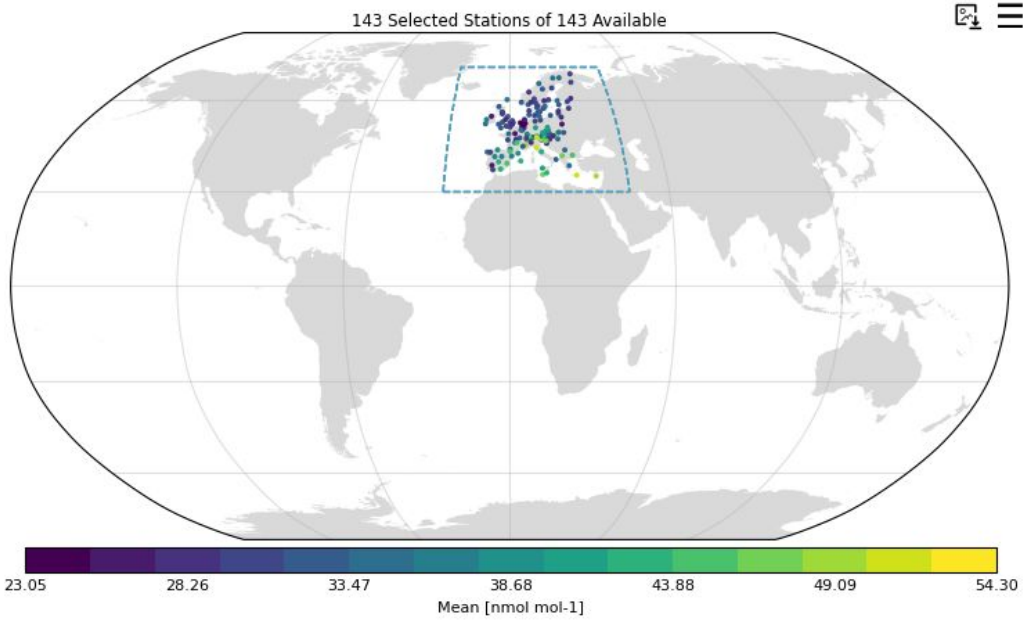
Resampling

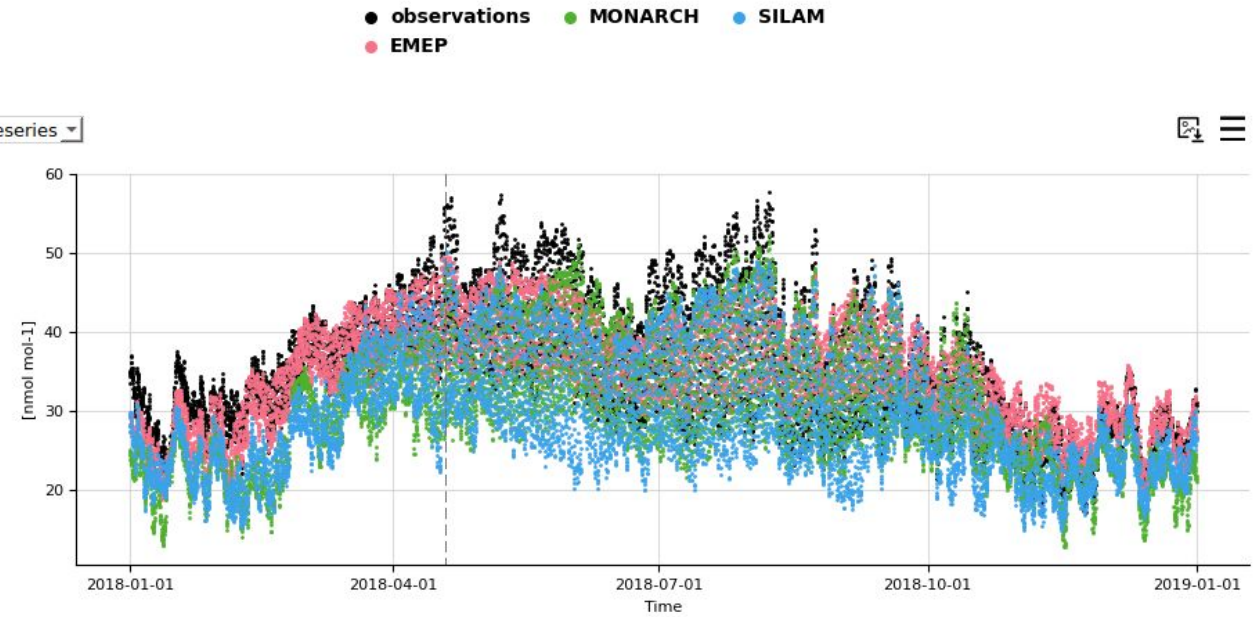
None

Site Selection

☑ All
☐ Intersect
☐ Extent

(x, y) = (2018-04-18, 23.8)

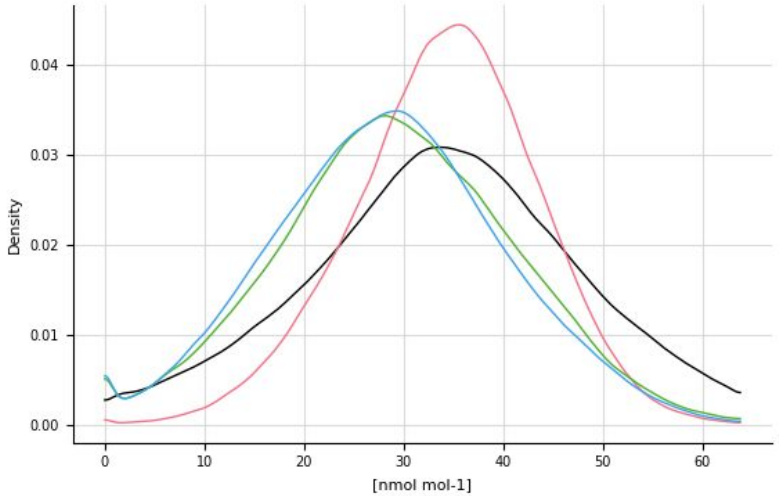




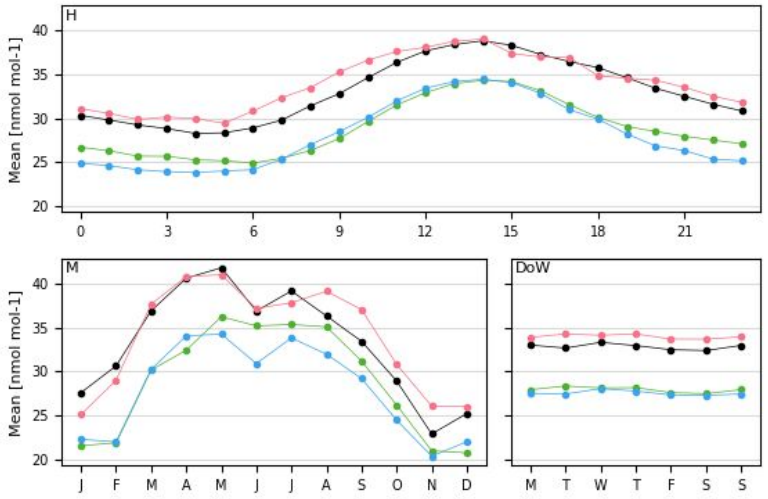
statsumma

	Mean	StdDev	p5	Median	p95
observations	32.91	11.84	13.30	32.80	54.46
EMEP	33.96	8.47	18.77	34.14	47.72
MONARCH	27.88	10.57	12.19	27.44	47.20
SILAM	27.41	10.55	11.02	27.29	47.37

distributior



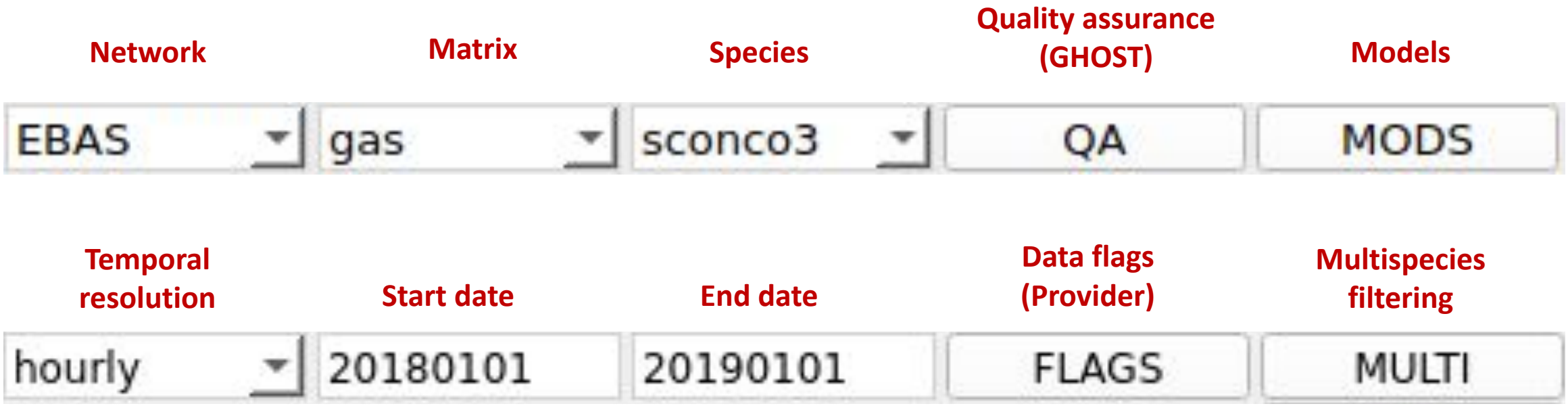
periodic



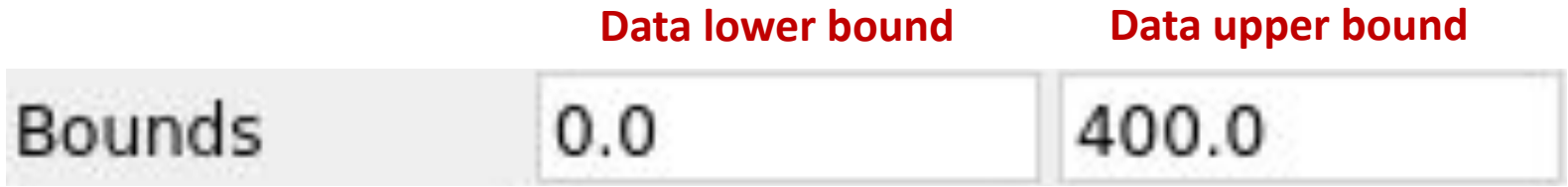
Main menu

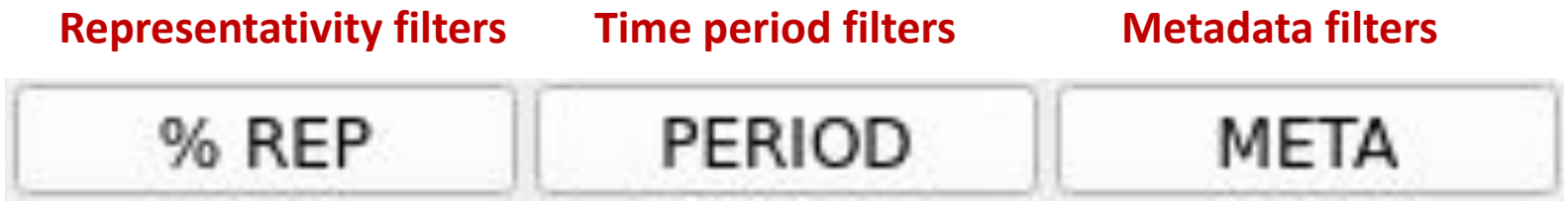

Data Selection
EBAS
gas
sconco3
QA
MODS
hourly
20180101
20190101
FLAGS
MULTI
READ
Filters
Bounds
0.0
400.0
% REP
PERIOD
META
RESET
FILTER
Statistics
Mode
Temporal|S
Aggregation
Median
Colocation
Temporal
Resampling
None
Site Selection
All
Intersect
Extent

Data selection menu

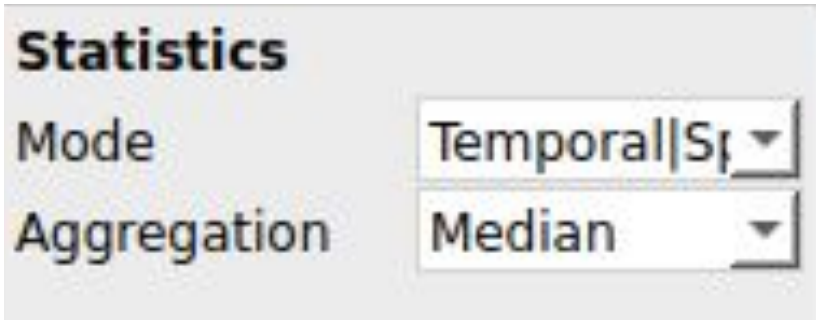


Filters menu





Statistics menu



Statistic mode

Statistic aggregation

Colocation menu

Colocate models vs. observations, removing temporal gaps

Resampling menu

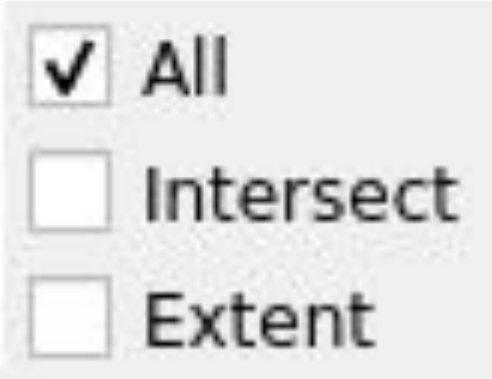
Temporal resolution to resample your data to (always lower than the selected resolution)

Stations selection menu

Select all stations

Select intersecting stations within all model domains

Select stations on current map view



Icons menu


Upload configuration files
Set original view
Forward
Pan
Save canvas
Download data and configuration files
Set world view
Back
Zoom to rectangle
Lasso

Interactive features

Navigation tips

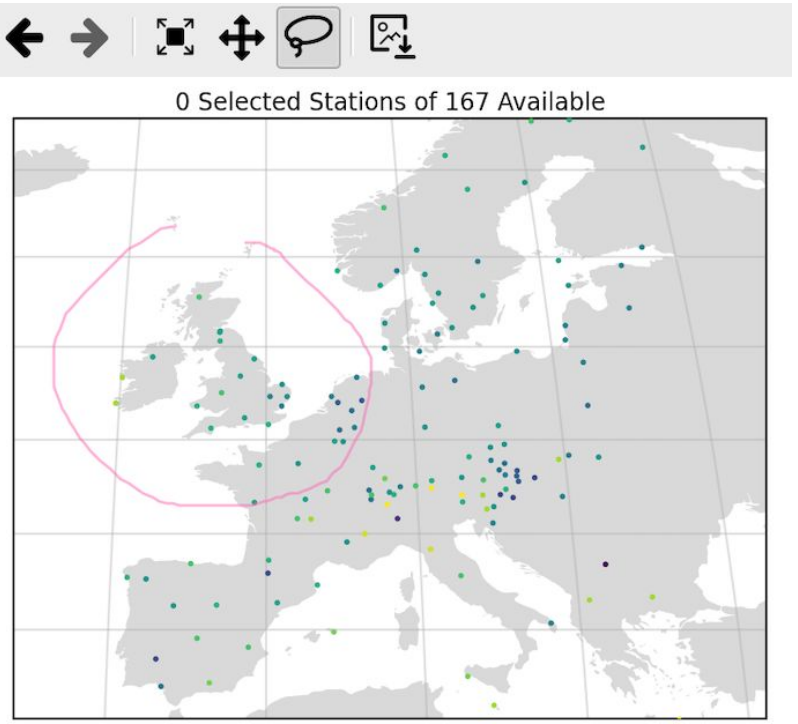
Individual stations can be selected by left clicking on them on the map.

Multiple stations can be selected by activating the lasso, and drawing it round all desired stations.

To unselect stations simply click on an empty space on the map.

When multiple stations are selected, to add or remove any individual station right click on the given station.

To quickly zoom on the map, the mouse or trackpad scroll wheel can be used.

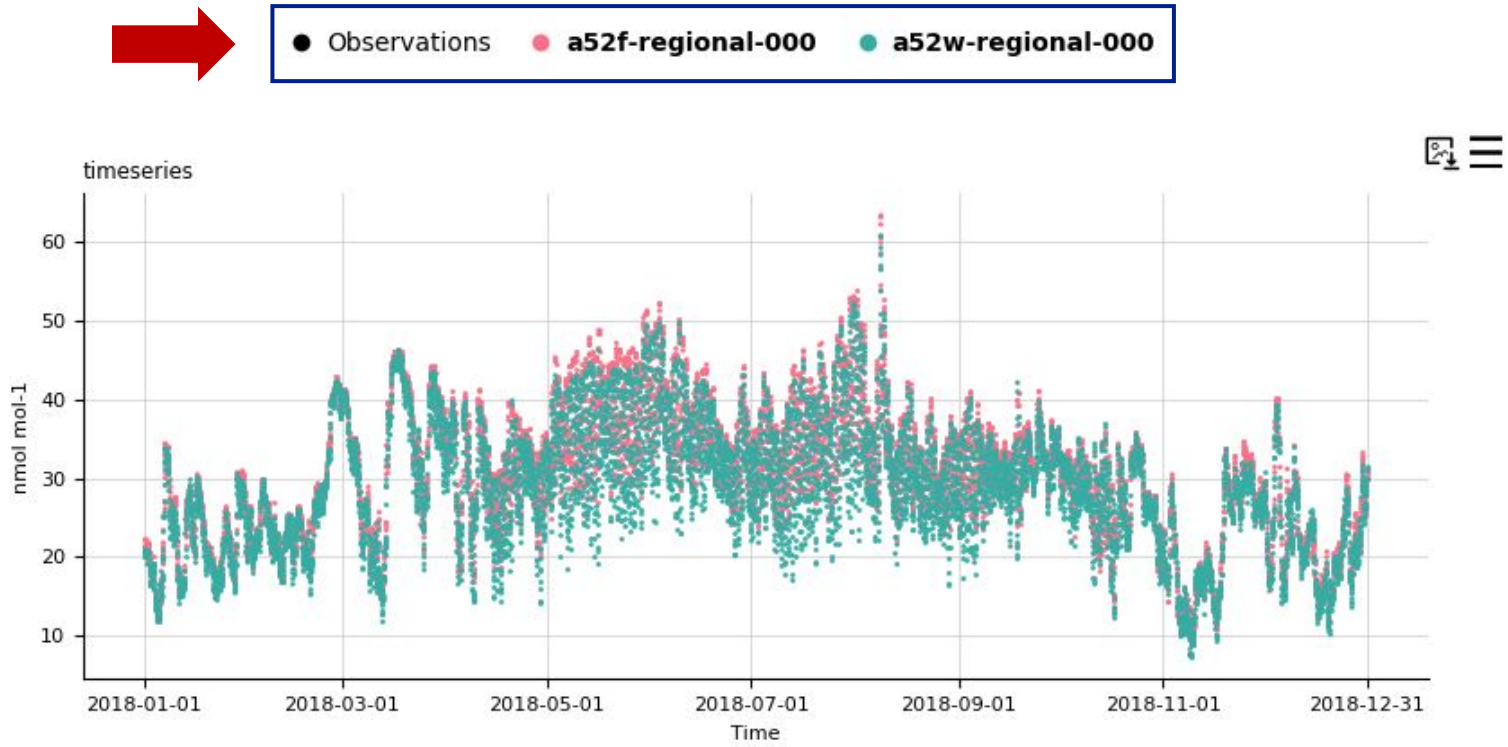


Example of lasso selection

Legend picking

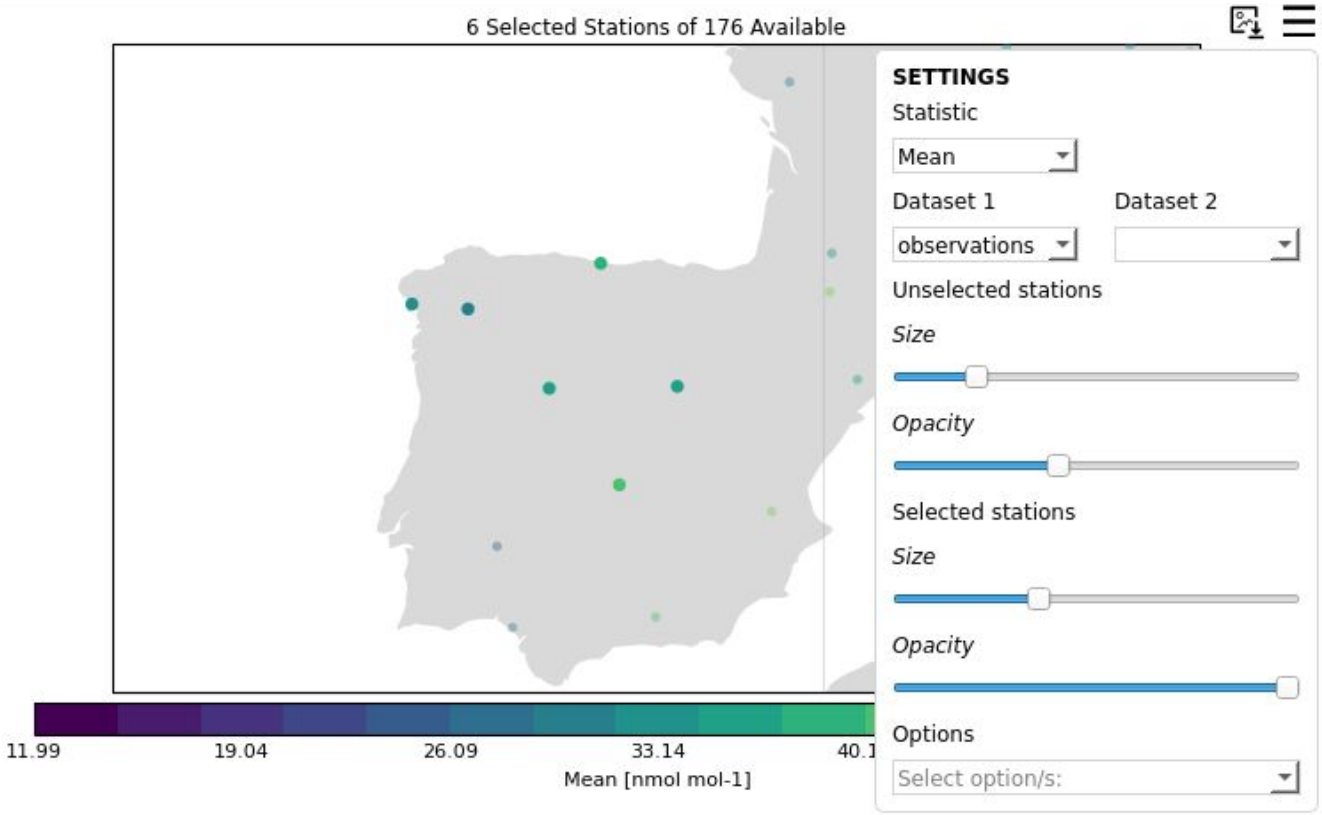
Clicking on the legend labels will remove or add data to each of the plots

Bold = Visible
Roman = Invisible



Customise the plot style

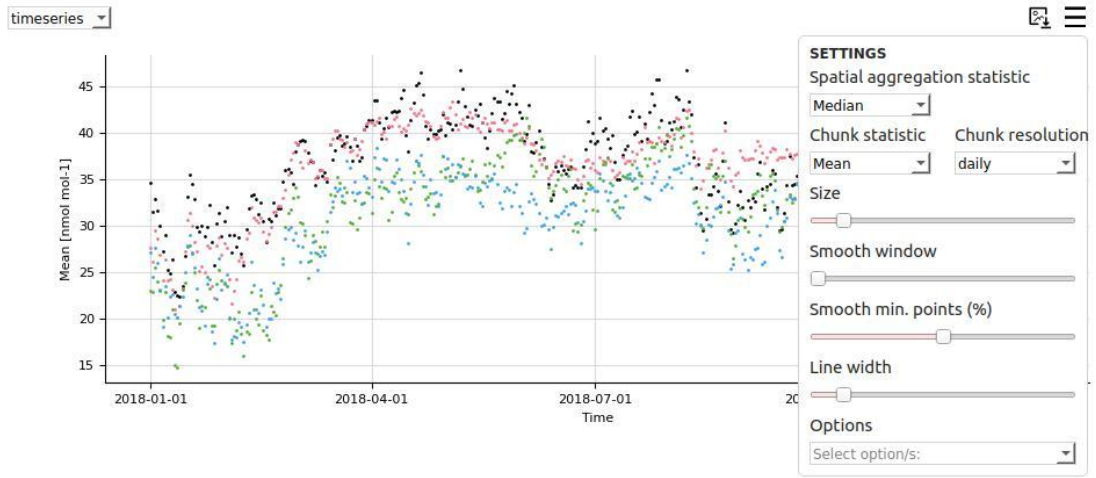
The style of the plots can be edited by clicking on the **burger menus** and changing the settings.



Choosing the statistics

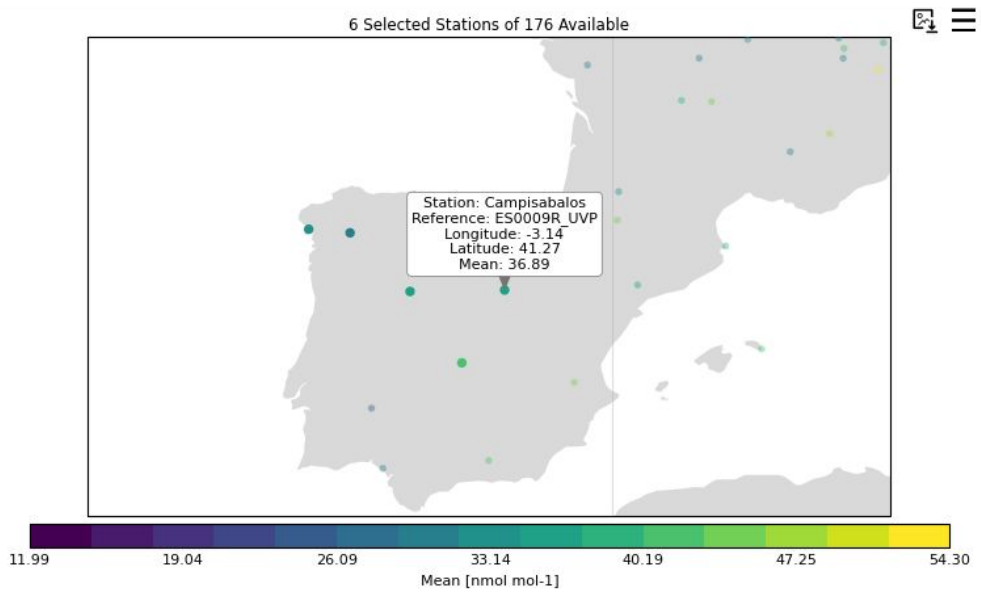
The statistics in the statsummary can be updated from the burger menu.

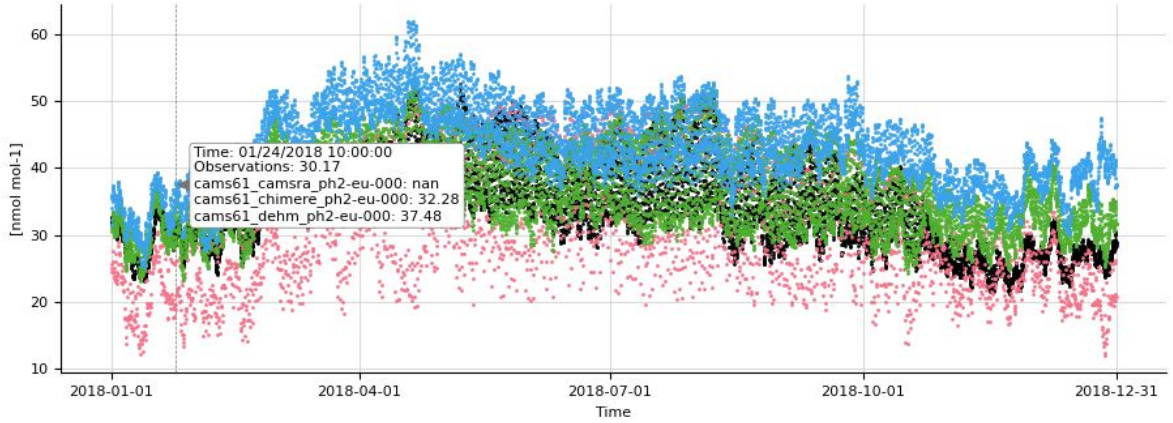
	Mean	StdDev	p5				
Observations	34.01	13.98	10.96				
cams61_camsra_ph2-eu-000	30.21	13.27	7.40				
cams61_chimere_ph2-eu-000	36.24	9.37	20.07	36.76	50.57	100.98	79.17
cams61_dehm_ph2-eu-000	42.40	10.27	24.92	42.74	58.55	117.41	79.17



Information on hover

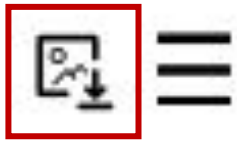
We can see the stations and values details and data by hovering on the map and charts.

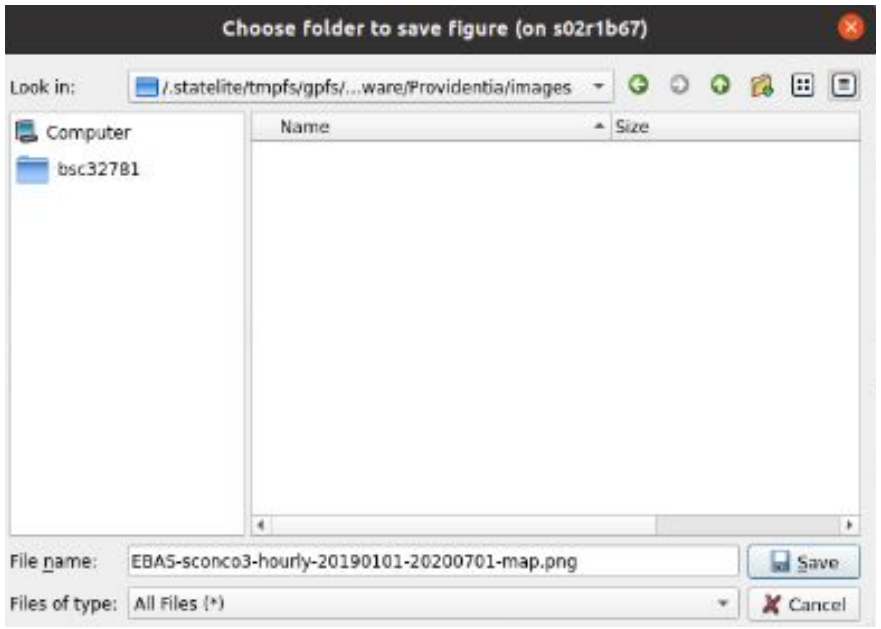


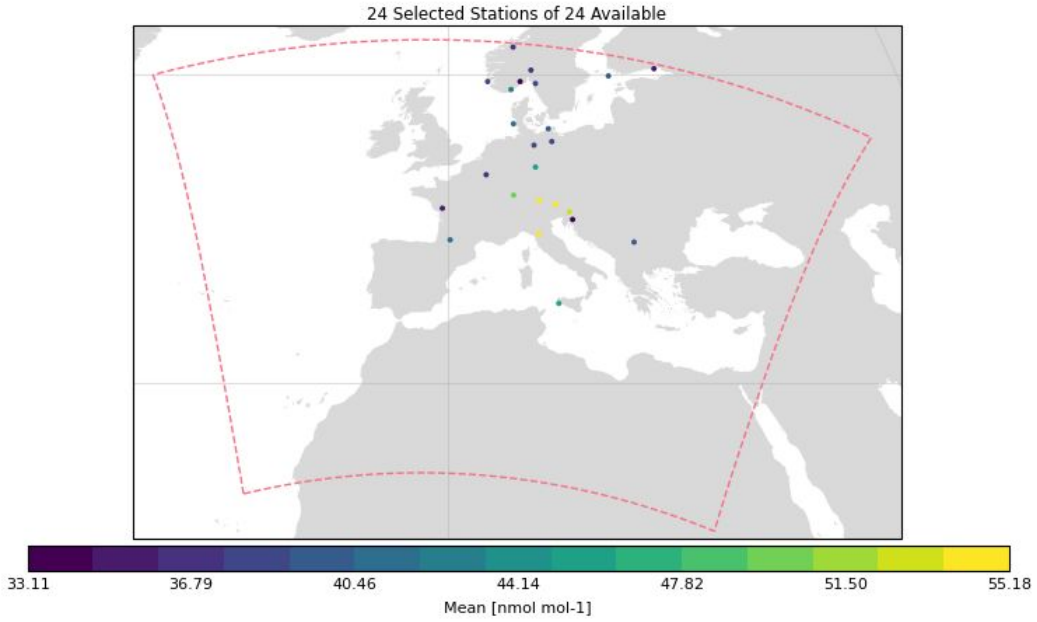


Saving plot figures

Saving plot figures is now possible by clicking on the image icons next to the burger menus

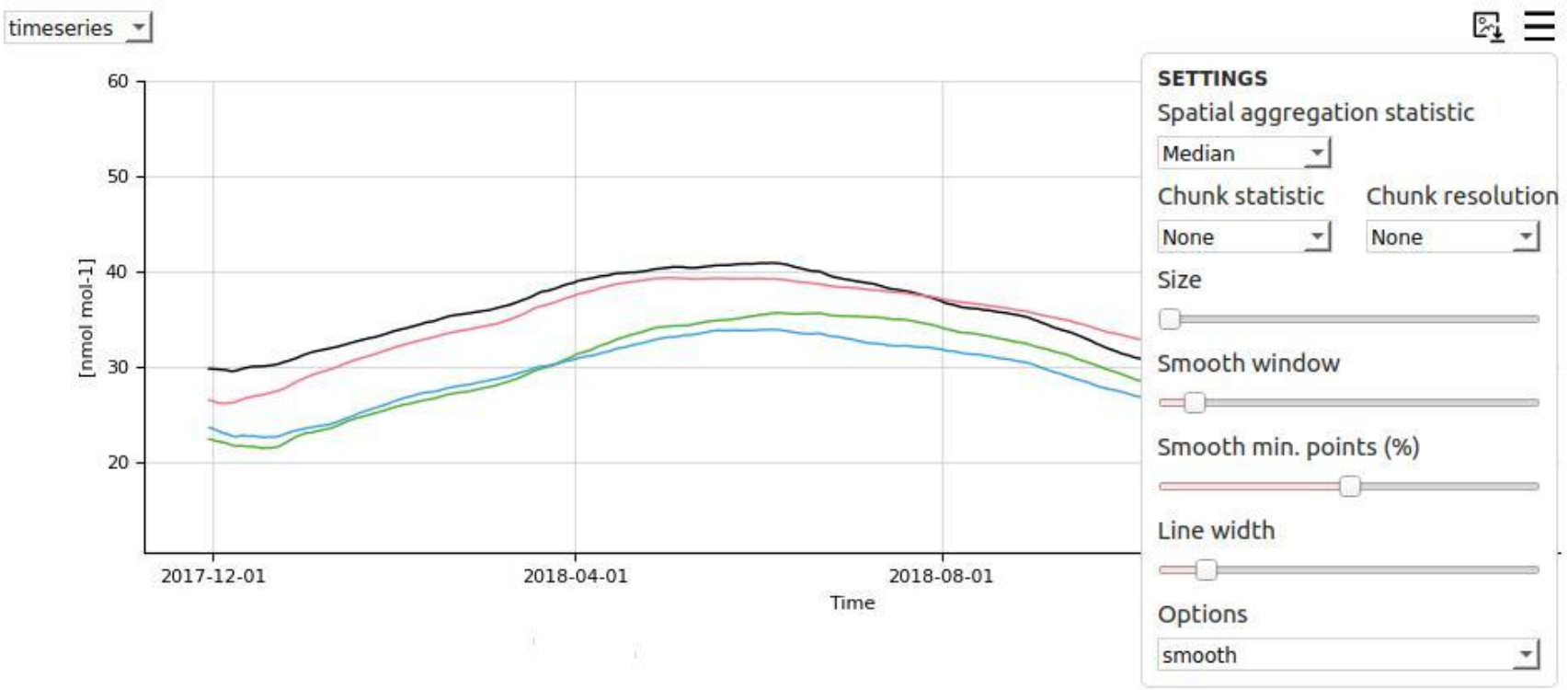






Smoothing

It is possible to add a smooth line in the timeseries plot by editing the smooth window, the data points can be then hidden by reducing their size to 0.



Filtering

Metadata filtering

The most common type of data filtering is by metadata, e.g. for **country**, **longitude**, **altitude** etc.

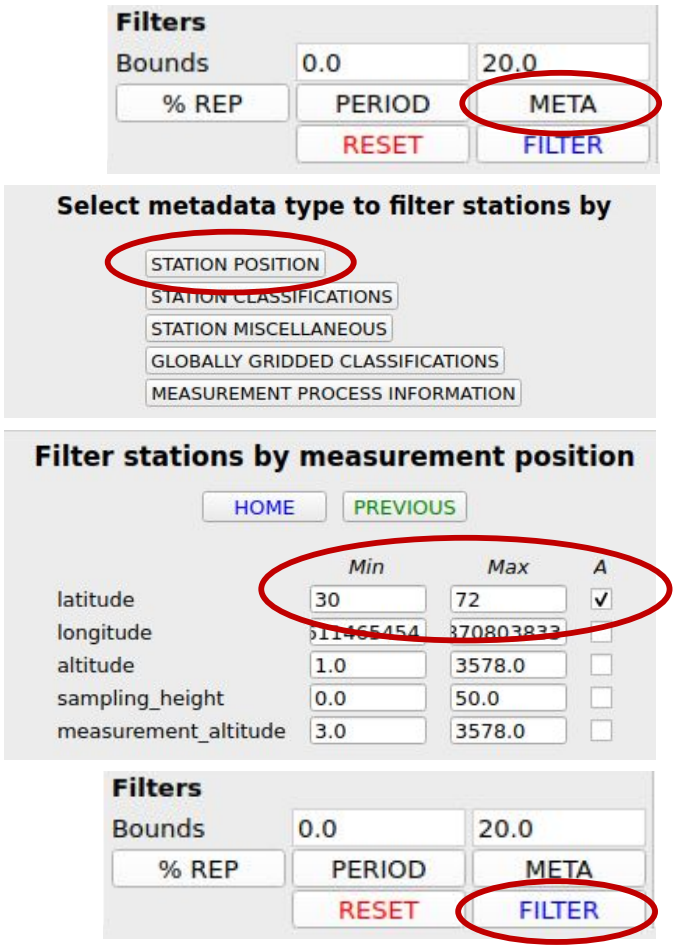
Metadata can either by **numerical** or **text** and the method for filtering for these slightly varies, as will be described.

One common question is what metadata available variables exist that can be filtered by. These can be seen empirically by looking through the sub-menus under the **META** button on the dashboard, with all available variables are organised into 5 categories: **station position**, **station classifications**, **station miscellaneous**, **globally gridded classifications** and **measurement process information**.

Reference: https://providentia.readthedocs.io/en/latest/Metadata-fields.html

Metadata filtering: Numerical data

Dashboard



Configuration file

latitude = 30, 72

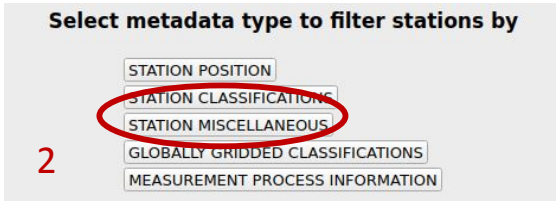
Library

provi.filter('latitude', lower=30, upper=72)

Reference:
https://providentia.readthedocs.io/en/latest/Filtering.html#numerical-metadata

Metadata filtering: Text metadata

Dashboard



Configuration file

```
country = keep: Austria
country = remove: Germany, France
```

Library

```
provi.filter("country", keep="Austria")
provi.filter("country", remove=["Germany",
"France"])
```

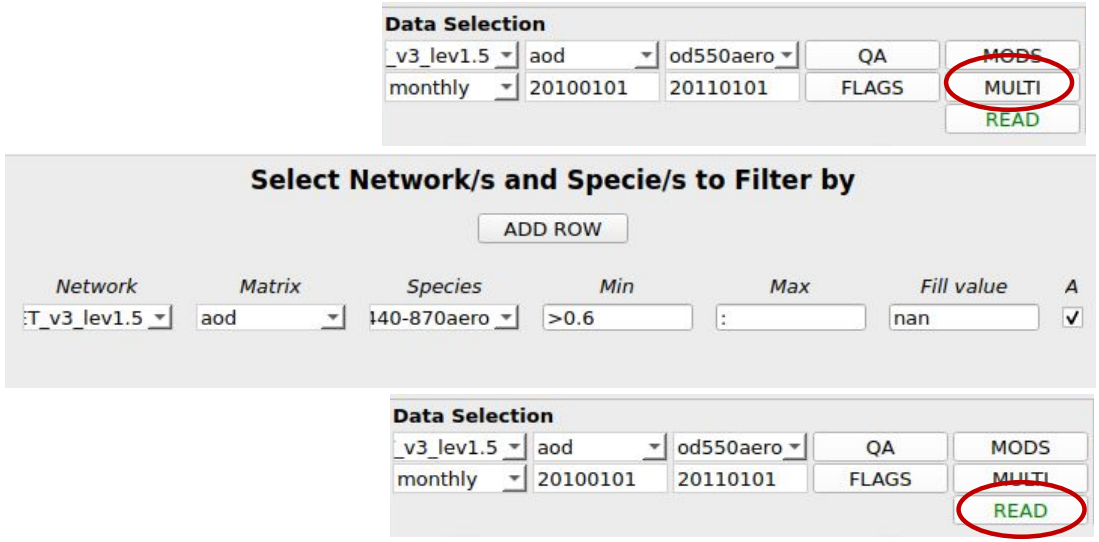
Reference:
https://providentia.readthedocs.io/en/latest/Filtering.html#text-metadata

Multispecies filtering

Multispecies filtering refers to the ability to **filter loaded species data by the values of another species**.

For example, when performing investigations of dust in the atmosphere it is a common practice to filter the AOD by the Angstrom exponent, to isolate values associated with dust.

Dashboard



Configuration file

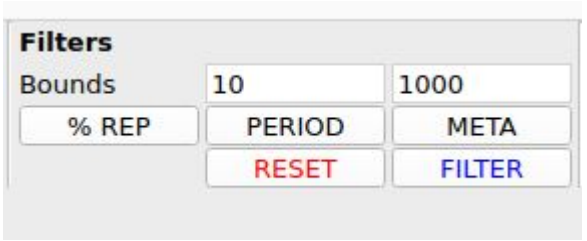
```
network = AERONET_v3_lev1.5
species = od550aero
filter_species = AERONET_v3_lev1.5:ae440-870aero (>0.6, :, nan)
spatial_colocation = True
```

Reference:
https://providentia.readthedocs.io/en/latest/Filtering.html#multispecies-filtering

Bounds

Often it is desired to remove values which exceed certain extreme bounds, as it is known that data should appear at such extremes. These bounds will be **by default active in Providentia**, with extreme bounds associated with a given species taken from definitions in GHOST.

Dashboard




```
'extreme_lower_limit':0.0, 'extreme_upper_limit':400.0,
'extreme_lower_limit':0.0, 'extreme_upper_limit':1200.0,
'extreme_lower_limit':0.0, 'extreme_upper_limit':600.0,
'extreme_lower_limit':0.0, 'extreme_upper_limit':3000.0,
'extreme_lower_limit':0.0, 'extreme_upper_limit':30000.0,
'extreme_lower_limit':0.0, 'extreme_upper_limit':50000.0,
```

Configuration file

```
lower_bound = 10
upper_bound = 1000
```

Link to GHOST Standards

Reference: https://providentia.readthedocs.io/en/latest/Filtering.html#bounds

Colocation

Spatial colocation

When loading more than one species (using **multispecies filtering** or in **multispecies plots** in the report and library modes) you may want to ensure that the available stations measure **data for all species** that are to be loaded. To do this, we need to activate spatial colocation.

After activating spatial colocation, any stations that do not have valid data for any of the loaded species are dropped.

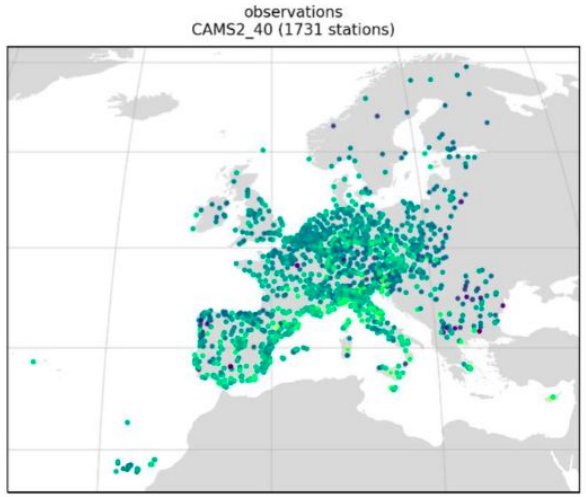
Spatial colocation can be set in the configuration file by setting a boolean as follows (default: True):

```
spatial_colocation = False
```

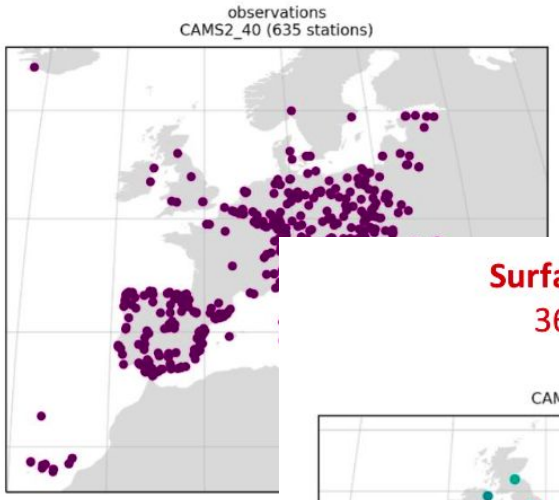
Reference: https://providentia.readthedocs.io/en/latest/Colocation.html

Spatial colocation

Surface O_3 (mean)
1731 stations

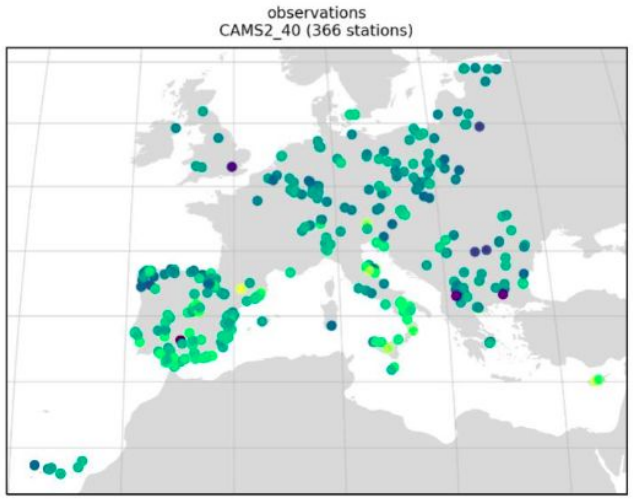


Surface CO (mean)
635 stations

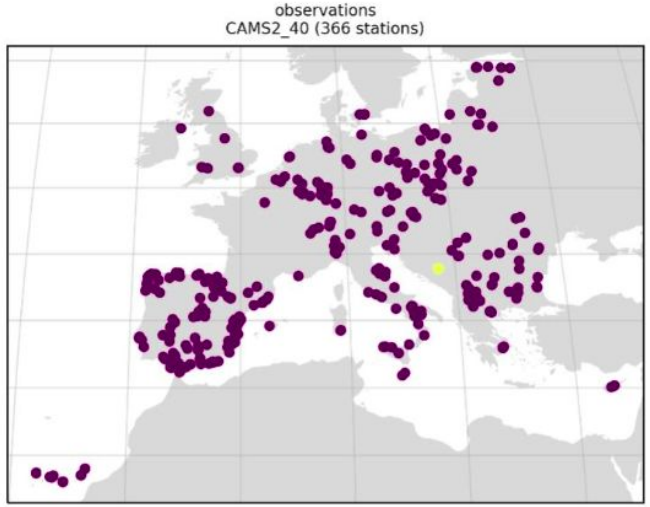


Without spatial colocation

Surface O_3 (mean)
366 stations



Surface CO (mean)
366 stations



With spatial colocation

Temporal colocation

Temporal colocation is used to **temporally pair observations and model data**, with any missing measurements in either the observational or model array, imposing missing measurements on the other.

When temporal colocation is active, you will have access to more plot types (scatter, taylor, fairmode-target, and fairmode-statsummary).

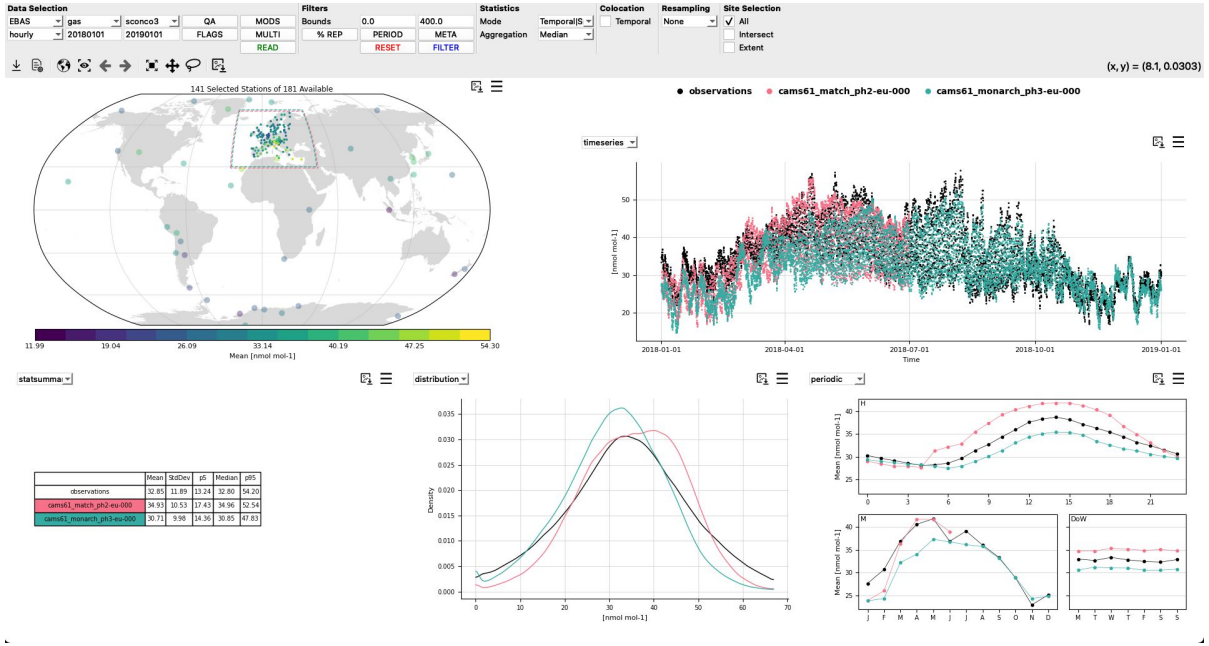
Temporal colocation can be set in the configuration file by setting a boolean as follows (default: True):

```
temporal_colocation = False
```

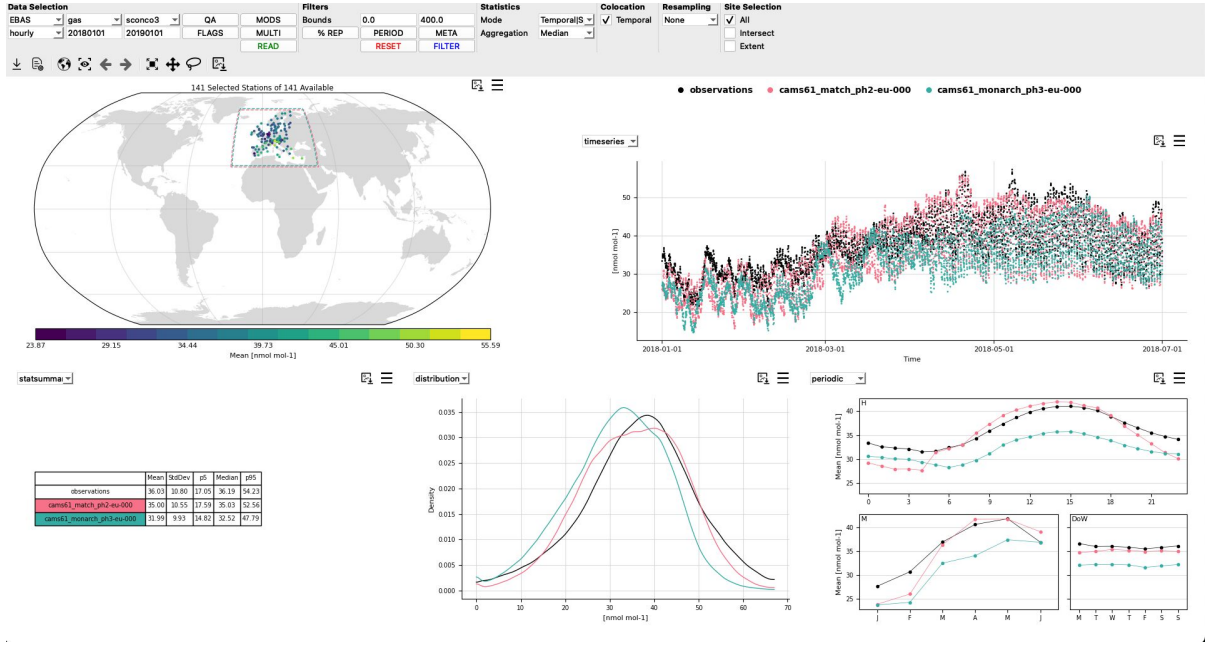
Reference: https://providentia.readthedocs.io/en/latest/Colocation.html

Temporal colocation

Without temporal colocation



With temporal colocation



Statistics

Statistical modes

Most of the plotted data in Providentia depends on certain logical choices made when calculating statistics.

Statistics can be calculated in a variety of ways, and Providentia allows a number of ways to alter the calculation, via 3 statistical modes. These are as follows:

- **Temporal|Spatial (default)**
- **Spatial|Temporal**
- **Flattened**

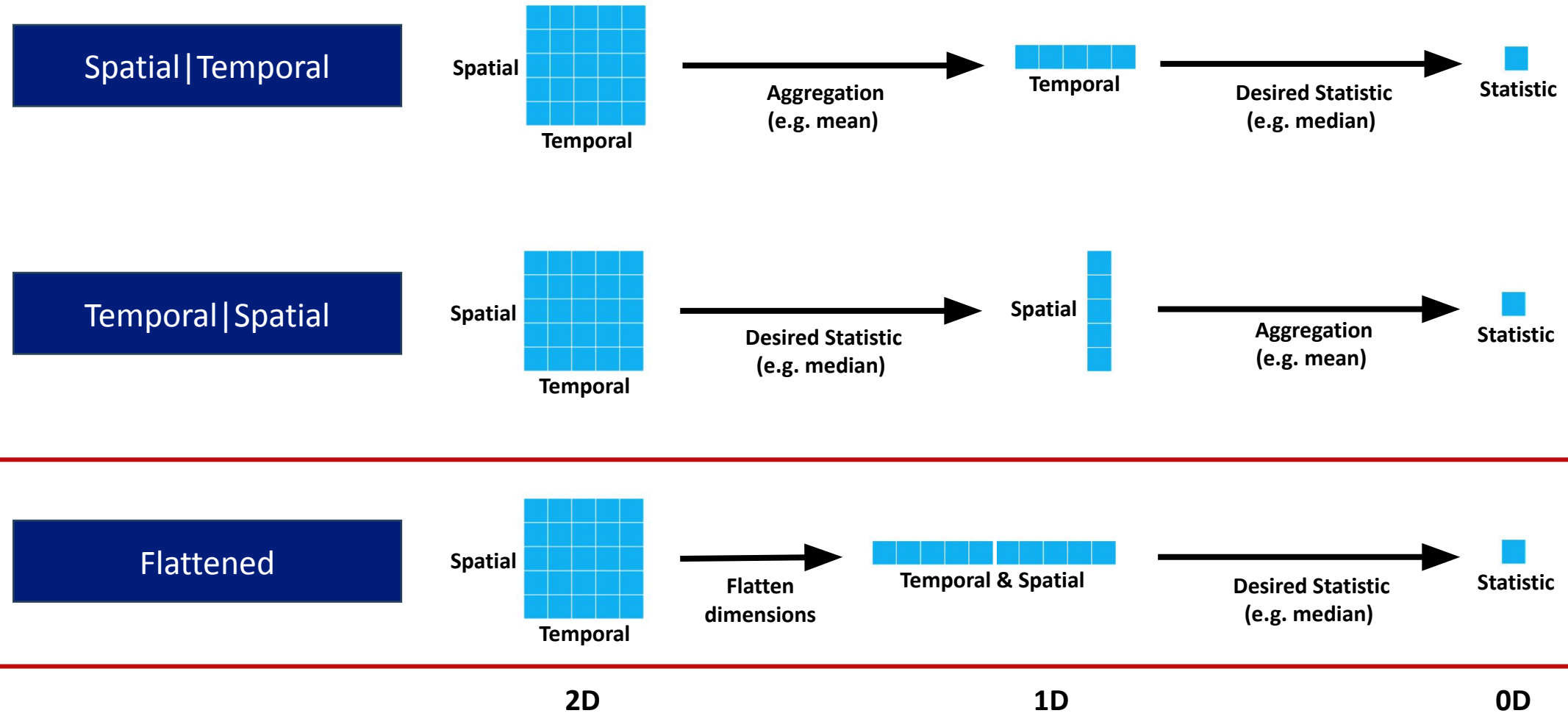
Statistical modes

The name of each statistical mode relates to how the dimensions of the selected data are reduced to calculate the statistical metrics, e.g. mean, median etc, going from 2D to 0D.

When selecting data across multiple stations, it has 2 dimensions: **Spatial** and **Temporal**.

For some plot types the full dimensional reduction is not possible, e.g. map. Therefore, they are locked in certain reduction configurations.

Statistical modes



Statistical modes

On the dashboard, in the statistics menu at the top, the mode and aggregation can be selected via the dropdown menus.

In the configuration files these can be set like so:

```
statistic_mode = Temporal|Spatial
statistic_aggregation = Median
```

Note: For the Flattened mode, there is no aggregation statistic.

Plot types

Plot types

- Map (map)
- Metadata summary (metadata)
- Timeseries (timeseries)
- Periodic plot (periodic)
- Periodic violin plot (periodic-violin)
- Box plot (boxplot)
- Distribution plot (distribution)
- Scatter plot (scatter)
- Heat map (heatmap) - Not available in dashboard
- Table that gives one statistic per subsection per model (table) - Not available in dashboard
- Table that gives multiple statistics per model (statsummary)
- Taylor Diagram (taylor)
- FAIRMODE target plot (fairmode-target)
- FAIRMODE statistics summary plot (fairmode-statsummary)
- Contingency table (contingencytable)

Reference: https://providentia.readthedocs.io/en/latest/Plot-types-and-options.html

Statistics in plots

Statistics in plots

It must be added as -[stat] (report, library) or selected from the burger menus (dashboard) to create **maps, periodic plots, heatmaps, tables and Taylor diagrams**.

Reference: https://providentia.readthedocs.io/en/latest/Statistics.html#available-statistical-metrics

Available statistics

Basic statistics

Statistic	Meaning
Mean	Mean
StdDev	Standard deviation
Var	Variance
Min	Minimum
Max	Maximum
Data%	Data availability
Exceedances	Number of exceedances
p1, p5, p10, p25, p50, p75, p90, p95, p99	Percentiles
NStations	Number of stations
MDA8	Daily maximum 8 hour average

Model bias statistics

Statistic	Meaning
MB	Mean bias
NMB	Normalised mean bias
ME	Mean error
NME	Normalised mean error
MNB	Mean normalised bias
MNE	Mean normalised error
MFB	Mean fractional bias
MFE	Mean fractional error
RMSE	Root mean square error
NRMSE	Normalised root mean square error
COE	Coefficient of efficiency
FAC2	Fraction of model values within a factor of two of observed values
IOA	Index of agreement
R	Pearson correlation coefficient
R^2	Coefficient of determination
UPA	Unpaired peak accuracy

Plot options

Plot options

- Make bias (_bias)
- Show annotate (_annotate)
- Show label plots (_individual)
- Show observation plots (_obs)
- Make multispecies (_multispecies)
- Show logarithmic in x axis (_logx)
- Show logarithmic in y axis (_logy)
- Add regression line (_regression)
- Add smooth line (_smooth)
- Hide points (_hidedata)
- Show domain (_domain)
- Add threshold line (_threshold)
- Normalise y axis (_normalise)
- Show Gerrity scores (_gerrity)

Reference: https://providentia.readthedocs.io/en/latest/Plot-types-and-options.html

Library

Providentia as a module

The library functionality allows Providentia to be used as a Python module, **with access to its backend functions**, allowing different parts of Providentia to be used in scripts, as required. In that way, we can import it:

```
import providentia as prv
```

From a Python script or notebook outside Providentia it is important to add the path to your directory to the system paths before importing Providentia:

```
import sys
sys.path.append("path/to/your/Providentia/directory")
```

Examples: https://github.com/BSC-ES/providentia/tree/master/tutorials

Opening Jupyter notebook session

One typical use case is the interactive evaluation in a Jupyter notebook. Providentia has an in-built manner of opening a Jupyter notebook. If we add --notebook or --nb as a launch option in the command line, a Jupyter session will start:

```
./bin/providentia --notebook
```

Running locally, Jupyter **will open automatically**.

Loading a configuration file

Once imported the first step is calling the class Providentia with the configuration you plan to use.

```
provi = prv.Providentia("interactive_template.conf")
```

This class' methods can be used:

- To access the modes: download, interpolate, report and dashboard.
- To load, filter, save and plot the data.

You can specify the section or subsection you want to work with by. Otherwise, if you have multiple sections, the first one will be selected unless it is explicitly defined.

```
provi = prv.Providentia("interactive_template.conf",
section="SECTIONNAME", subsection="SECTIONNAME·SUBSECTIONNAME")
```

Viewing active configuration

It is possible to view the active configuration variables in two different ways:

- Using the method *print_config*:

```
provi.print_config()
```

  This method can also be used to view another configuration file, by passing the filename:

```
provi.print_config("important.conf")
```

- By printing the active class instance:

```
print(provi)
```

Accessing the modes

The *download*, *interpolate, report* and *dashboard* functionalities can be integrated in workflows with little effort.

provi.download()

provi.interpolate()

provi.report()

provi.dashboard()

Loading the data

When the data is available in your machine, you can load it by doing:

```
provi.load()
```

Once the data has been loaded, it can be easily accessed by these methods:

- Retrieving a specific variable, by providing a variable name to the *variable* method.

```
var_data = provi.variable("network|component_variablename")
```

- Retrieving all metadata / data variables by using the *data* method. The returned data can be in different formats: nc (netCDF), np (numpy) or xr (xarray) by passing the format argument:

```
data = provi.data(format="xr")
```

Applying filters

To apply a filter not set in the configuration file, the *filter* method is used. The fields to filter by can be any data /metadata variable standardised in GHOST, and can be textual or numeric.

- If the field is numeric, use *lower* and/or *upper*, e.g.:

```
provi.filter(field, lower=28, upper=31)
```

- If the field is textual, use *keep* and/or *remove*, e.g.:

```
provi.filter("country", keep="Spain")
```

```
provi.filter("country", remove=["Spain", "France"])
```

- If the field is a representativity field, use *limit*:

```
provi.filter(rep_field, limit=20)
```

Selecting stations

To select station for one or more stations, you can use the function filter_station:

```
provi.filter_station(['AT0002R_UVP','SK0007R_UVP'])
```

Reset filters

If at any time any active data / metadata filters are wished to be reset, this can be done by using the *reset* method.

To reset all filters to the Providentia defaults, including those set from the original read configuration file:

```
provi.reset()
```

To keep the filters defined in the configuration file, but reset the ones added later, pass the *initialise* argument as True:

```
provi.reset(initialise=True)
```

Plotting

You can use the plotting functions of Providentia with the *plot* method. Any of the available plot types in Providentia can be made by passing the desired type.

```
provi.plot(plot_type)
```

Plots can also be saved to file, by passing *save* and a filename with the wanted extension (e.g. png, etc.):

```
provi.plot(plot_type, save="myplot.png")
```

The plotting object can also be returned for additional tinkering, by setting *return_plot* to True:

```
plot_obj = provi.plot(plot_type, return_plot=True)
```

Reference: https://providentia.readthedocs.io/en/latest/Plotting.html

Calculating statistics

All defined basic and model bias statistics defined in Providentia can be calculated from the loaded data, using the *statistic* method:

```
stat_calc = provi.statistic(stat, labela="OBS")
```

where *stat* is the statistic wished to be calculated, and *labela* is “observations”/model name or the alias set in the conf file. If wanting to calculate a bias statistic, you need to set *labelb:*

```
stat_calc = provi.statistic(stat, labela="OBS", labelb="EMEP")
```

For bias statistics for which a subtraction is involved, it is always done as *datab* - *dataa*.

If wanting to calculate statistics per station, *per_station* can be passed as True:

```
stat_calc = provi.statistic(stat, labela="OBS", labelb="EMEP", per_station=True)
```

Saving the data

Save the current configuration:

```
provi.save(format='conf')
```

Save the data as netCDF:

```
provi.save(format='nc')
```

Save the data as numpy:

```
provi.save(format='np')
```

If you want to save the data to a specific location, use the argument fname. By default, the data is saved in the folder **providentia/saved_data**:

```
provi.save(fname=''data/output/config.conf', format='conf')
```

Funded by the European Union

THANK YOU

This project has received funding from the European Union's Horizon Europe Framework Programme under the grant agreement No 101160258. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union.

ERATOSTHENES CENTRE OF EXCELLENCE · BSC Barcelona Supercomputing Center Centro Nacional de Supercomputación · GRASP · ARMINES · MINES PARIS | PSL · Ciências ULisboa · FCiências.ID · pmod wrc