Getting OpenData for SYMPHONIE

The goal of this session is to give some tools to retrieve atmospheric and oceanic forcing files from ECMWF and CMEMS open source dataset.

1 Setting conda environment :

For the training session :

- activate the retrieve_data env :

  conda activate retrieve_data

From a blank environment, on your own system for example, you can create this environment using conda/mamba (or else…) by following the instruction in the annexe

2 Retrieving ECMWF Opendata :

In this training session, you will use a preset credential file. So there is no need to set any credentials here. But on your own system you will need to follow the instructions given in this section in the annexe.

Go to the directory : ~/FORMATION_2026/RETRIEVE_DATA/ECMWF

You will find two files :

- the python script : Get_ecmwf.py
- the parameter file : Param_ECMWF.ini

Here we can only retrieve open data from ECMWF ERA5 reanalyses. That means that you can retrieve a lot of data from the past (forecasts are also available but not from the present script, it will in a futur version) on a 1/4° resolution and with an hourly frequency.

The retrieving script, Get_ecmwf.py, is controlled by a parameter file by default Param_ECMWF.ini if any argument is given when launching the command :

python Get_ecmwf.py Param_ECMWF.ini

or

python -m Get_ecmwf Param_ECMWF.ini

The script will retrieve daily data (as grib formatted file) and store them in daily separated **netcdf** files under a set directory.

Before running the command lest take a look to the parameter file :

[ECMWF]

```
# Source of data
prod_type=reanalysis
# List of Variables : Instantaneous variables, cumulative variables and variables name  in the dataset
instvar_lst=['10m_u_component_of_wind','10m_v_component_of_wind','2m_dewpoint_temperature','2m_temperature','mean_sea_level_pressure','surface_pressure','land_sea_mask']
cumuvar_lst=['total_precipitation','surface_solar_radiation_downwards','surface_thermal_radiation_downwards']
cumuvarnam_lst=['tp','ssrd','strd']
# Time to retrieve
time_lst=[
'00:00','01:00','02:00','03:00','04:00','05:00','06:00','07:00','08:00','09:00','10:00','11:00','12:00',
'13:00','14:00','15:00','16:00','17:00','18:00','19:00','20:00','21:00','22:00','23:00' ]
# Download format
file_format=grib
# ECMWF dataset type
dataset=reanalysis-era5-single-levels
[date_start]
# Start date
year=2025
month=12
day=3
[date_end]
# End date
year=2025
month=12
day=3
[area]
# Area : lat_max, lon_min, lat_max, lon_min
lat_max=6.75
lat_min=3
lon_max=4
lon_min=0
[clean_up]
# Removing "GRIB" attrs from variable attributes
if_rm_grib_attrs=True
pattrn=GRIB
[output]
# Output files
dirout=./
prefix_=Benin_local_
ext=.nc
[dcsapirc]
cdsapirc=.cdsapirc_cds
```

By default you can simply only change the values of **date_start** and **date_end** data, the geographical **area** and the output directory **dirout**.

The script will loop day by day from date_start to date_end dates.

For the purpose of the Training session try to retrieve only 1 or 2 days.
Change the values in the Param_ECMWF.ini if you want (you can make a copy of it if you want before modifying it).
and them run the command :

python Get_ecmwf.py Param_ECMWF.ini

By default you will get some log looking like that :
example of log output :

```
****************************
* Retrieve ECMWF from Param  *
  Param_ECMWF.ini
****************************
Retrieving date : 2025.12.01
2026-08-14 12:07:19,242 INFO Request ID is 7eac79fb-d9b4-4779-a12c-79f5ac59f508
2026-08-14 12:07:20,999 INFO status has been updated to accepted
2026-08-14 12:08:11,351 INFO status has been updated to successful
```

And in the directory “dirout” you should found the downloaded data file :
Benin_local_20251201.nc

The name is composed by : **prefix_** + date + **ext**

The script is formatting the data in such a way that SYMPHONIE will be able to read them directly.

3 Retrieving CMEMS Opendata :

In this training session, you will use a preset credential file. So there is no need to set any credentials here. But on your own system you will need to follow the instructions given in this section in the annexe.

Go to the directory : ~/FORMATION_2026/RETRIEVE_DATA/CMEMS

You will find two files :

- the python script : Get_copernicusmarine.py
- the parameter file : Param_COPERNICUS.ini

Here script can retrieve any open data from CMEMS, but from the actual parameter file, the choice is restricted to Global Analyses or Global Reanalyses :

DATA description for Reanalyses :
https://data.marine.copernicus.eu/product/GLOBAL_MULTIYEAR_PHY_001_030/description

DATA description for Analyses :

https://data.marine.copernicus.eu/product/GLOBAL_ANALYSISFORECAST_PHY_001_024/description

The retrieving script, Get_copernicusmarine.py, is controlled by a parameter file by default Param_COPERNICUS.ini if any argument is given when launching the command :

python Get_copernicusmarine.py Param_COPERNICUS.ini

or

python -m Get_copernicusmarine Param_COPERNICUS.ini

The script will retrieve daily averaged data and store them in daily separated **netcdf** files under a set directory.

Before running the command lest take a look to the parameter file :

```
[COPERNICUS]
## Reanalyses : Form Reanalyses, for a daily averaged data,
##            there is only one file to retrieve with all data.
##             dataset_name list can be set to None (optional).
#dataset_id=['cmems_mod_glo_phy_my_0.083deg_P1D-m']
## List of Variable
#variables=['so', 'thetao', 'uo', 'vo', 'zos']
#dataset_name=None
## Analyses : Form Analyses, for a daily averaged data,
##            there are several files to retrieve :
##            one for each So, thetao and zos. and one for uo and vo.
##            he file pattern looks alike for each with a specific variable names
##            to differentiate them. Those name are set in the dataset_name list.
dataset_id=['cmems_mod_glo_phy','_anfc_0.083deg_P1D-m']
# List of Variable
variables=['so', 'thetao', 'uo', 'vo', 'zos']
dataset_name=['-so', '-thetao', '-cur', '-cur', '']
[date_start]
# Start date
year=2025
month=12
```

```
day=3
[date_end]
# End date
year=2025
month=12
day=3
[area]
# Area : lat_max, lon_min, lat_max, lon_min
lat_max=6.75
lat_min=3
lon_max=4
lon_min=0
[depth]
depth_min=0.49402499198913574
depth_max=5727.9169921875
[output]
# Output file
dirout=./Analyses
prefix_=Benin_local_
ext=.nc
[STATIC]
# Use to retrieve the static bathymetry from COPERNICUS
# If the dataset=None, any file will be retrieved.
# No retrieve bathy
#dataset=None
## Reanalyses
#dataset=cmems_mod_glo_phy_my_0.083deg_static
## Analyses
dataset=cmems_mod_glo_phy_anfc_0.083deg_static
bathyfile=Benin_bathy.nc
```

You can comment/uncomment the **Reanalyses** or **Analyses** section (one at a time) to retrieve the desired dataset. In this example the parameter are set to retrieve the **Analyses** data.
It is then possible to change the date_start and date_end variables, area and depth also.

N.B. The STATIC part is used if the dataset parameter is not set to None . Here also there are two choices for **Reanalyses** or **Analyses** cases.
The bathymetry can be very important for SYMPHONIE to handle the best way the OGCM Open Boundary Conditions forcing data.

The script will loop day by day from date_start to date_end dates.

For the purpose of the Training session try to retrieve only 1 or 2 days.
Change the values in the Param_COPENICUS.ini if you want (you can make a copy of it if you want before modifying it).
and them run the command :

python Get_copernicusmarine.py Param_COPERNICUS.ini

By default you will get some log looking like that :
example of log output :

```
*****************************
* Retrieve COPERNICUS          *
  from Param_COPERNICUS.ini
*****************************
* *******************************
* Retrieving COPERNICUS data from
* from date : 2025-12-01
*  to  date : 2025-12-01
* *******************************
Retrieving date : 2025.12.04
INFO - 2026-08-17T16:41:53Z - Selected dataset version: "202406"
INFO - 2026-08-17T16:41:53Z - Selected dataset part: "default"
INFO - 2026-08-17T16:41:55Z - Total size of the download: 453.79 KB.
INFO - 2026-08-17T16:41:56Z - Selected dataset version: "202406"
INFO - 2026-08-17T16:41:56Z - Selected dataset part: "default"
INFO - 2026-08-17T16:41:59Z - Total size of the download: 453.79 KB.
INFO - 2026-08-17T16:41:59Z - Selected dataset version: "202406"
INFO - 2026-08-17T16:41:59Z - Selected dataset part: "default"
INFO - 2026-08-17T16:42:03Z - Total size of the download: 453.79 KB.
INFO - 2026-08-17T16:42:04Z - Selected dataset version: "202406"
INFO - 2026-08-17T16:42:04Z - Selected dataset part: "default"
INFO - 2026-08-17T16:42:07Z - Total size of the download: 453.79 KB.
INFO - 2026-08-17T16:42:08Z - Selected dataset version: "202406"
INFO - 2026-08-17T16:42:08Z - Selected dataset part: "default"
INFO - 2026-08-17T16:42:09Z - Total size of the download: 22.12 KB.
......
......

Retrieving Static Bathymetry
INFO - 2026-08-17T16:42:10Z - Selected dataset version: "202211"
INFO - 2026-08-17T16:42:10Z - Selected dataset part: "bathy"
INFO - 2026-08-17T16:42:11Z - Total size of the download: 30.93 KB..
```

Under the directory "**dirout**" you will found the data files :

- data file : Benin_local_20251204.nc
  The data files will have a name formatted as : **prefix_** + date + **ext**

- and the bathymetry file : Benin_bathy_analyses.nc

The script is formatting the data in such a way that SYMPHONIE will be able to read them directly.

4 Retrieving GLOFAS Opendata :

In this training session, you will use a preset credential file. So there is no need to set any credentials here. But on your own system you will need to follow the instructions given in this section in the annexe.

The GLOFAS data are distributed by ECMWF (not only) but with a different url for the credentials : cdsapirc_ewds.

Go to the directory : ~/FORMATION_2026/RETRIEVE_DATA/GLOFAS

You will find two files :

- the python script : Get_glofas.py
- the parameter file : Param_GLOFAS.ini

Here we can retrieve open data from GLOFAS Reanalyses and Forecast data. The Reanalyses data do not cover the near realtime period. The data give daily averaged discharge flow (m3/s) over a 0.05° x 0.05° resolution over the globe.

The retrieving script, Get_glofas.py, is controlled by a parameter file by default Param_GLOFAS.ini if any argument is given when launching the command :

python Get_glofas.py Param_GLOFAS.ini

or

python -m Get_glofas Param_GLOFAS.ini

The script will retrieve daily data (as grib formatted file) and store them in a **single** **netcdf** file under a set directory.

The retrieval process will depend on the nature of the dataset chosen. For reanalyses, the script will try to retrieve the data year by year, while for the Forecast case, it will be done month by month.

Before running the command lest take a look to the parameter file :

```
[GLOFAS]
## For Forecast
#dataset=cems-glofas-forecast
#system_version=['operational']
#hydrological_model=['lisflood']
#product_type=['control_forecast']
#leadtime_hour=['24']
## List of Variable
#var_lst=['river_discharge_in_the_last_24_hours']
#fcstname=dis24

# For Reanalyses
dataset=cems-glofas-historical
system_version=['version_5_0']
hydrological_model=['lisflood']
product_type=['consolidated']
timespan=['time_mean']
# List of Variable
var_lst=['average_river_discharge_in_the_last_24_hours']
reaname=avg_dis
```

```
varnamout=discharge
# data format
data_format=netcdf
# download format
download_format=unarchived

[dates]
#year=['2024','2025',]
#month=['10','11','12']
#day=['07','08','09',]
year=['2025',]
month=None
day=None
# If month=None, month set to full year in the script:
#month=['01','02','03','04','05','06','07','08','09','10','11','12']
# If day=None, day set to full month in the script:
#day=['01','02','03','04','05','06','07','08','09','10','11','12','13','14','15','16','17','18','19','20','21','22','23','24','25','26','27','28','29','30','31']
[area]
# Area : lat_max, lon_min, lat_max, lon_min
lat_max=6.75
lat_min=3
lon_max=4
lon_min=0
[output]
# Output file
dirout=./BENIN
prefix_=Benin_reanalyse_
ext=.nc
[clean_up]
# Removing "Grib" attrs from variable attribute
if_rm_grib_attrs=True
pattrn=GRIB
[CDSAPIRC]
cdsapirc=.cdsapirc_ewds
```

You can comment/uncomment the **For Reanalyses** or **For Forecast** sections (one at a time) to retrieve the desired dataset. In this example the parameters are set to retrieve the **Reanalyses** data.
It is then possible to change the dates variables, area also. The dates are given as list. It could contains a single informations or several. Be sure to keep a chronological order.
For month and day if they are set to None the script will try to get all the possible months/days.

For the purpose of the Training session try to retrieve only 3 monthz for example from october to december 2025…
Change the values in the Param_GLOFAS.ini if you want (you can make a copy of it if you want before modifying it).

and them run the command :

```
python Get_glofas.py Param_GLOFAS.ini
```

By default you will get some log looking like that :
example of log output :

```
*****************************
* Retrieve GLOFAS from Param  *
  from Param_GLOFAS.ini
*****************************

Retrieving data from cems-glofas-historical for the year 2025 and month(s) ['10', '11', '12']

2026-08-21 18:44:51,216 INFO [2024-02-01T00:00:00] Please note that accessing this
dataset via CDS for time-critical operation is not advised or supported

2026-08-21 18:44:51,217 INFO [2024-02-01T00:00:00] Please note we suggest checking the
list of known issues on the GloFAS wiki

[here](https://confluence.ecmwf.int/display/CEMS/GloFAS+-+Known+Issues)

before downloading the dataset.

2026-08-21 18:44:51,217 INFO [2026-07-30T00:00:00] Due to changes in the data structure
of the cems-glofas-historical dataset, previous API requests may no longer work. Please
refer to the updated API request and update your scripts accordingly. Please refer to
[GloFAS historical: Changes to the EWDS API
Request](https://confluence.ecmwf.int/pages/viewpage.action?pageId=699325478) for
more information.
2026-08-21 18:44:51,217 INFO Request ID is 5c5d9bf5-3c3e-4039-8c60-a77265b4a0e1
2026-08-21 18:44:51,283 INFO status has been updated to accepted
2026-08-21 18:45:04,818 INFO status has been updated to running
2026-08-21 18:45:12,826 INFO status has been updated to successful
 Saving GLOFAS data in : ./BENIN/Benin_reanalyse_2025.nc
```

Under the directory "**dirout**" you will found the data file : Benin_reanalyse_2025.nc.
For several year requested (2023,2024 and 2025) the output file would have been named: Benin_reanalyse_2023-2025.nc

5 Treating the GLOFAS data file for SYMPHONIE :

SYMPHONIE will need ascii format timeseries to deal with the downloaded netcdf GLOFAS files.
The script **Make_timeseries_glofas.py** will be used to process them.
The script can also create a file that will help to format the **notebook_river** used to set the water discharge in the simulation.

In the directory : ~/FORMATION_2026/RETRIEVE_DATA/GLOFAS

You will find three files :

- the python script : Make_timeseries_glofas.py
- the parameter file : Param_Make_TSGlofas.ini

- a template file : template_atlas.txt

Here we can process several GLOFAS data retrieved previously.

The script, Make_timeseries_glofas.py, is controlled by a parameter file by default Param_Make_TSGlofas.ini if any argument is given when launching the command :

python Make_timeseries_glofas.py Param_Make_TSGlofas.ini

or

python -m Make_timeseries_glofas Param_Make_TSGlofas.ini

The script will :

- read one or several data files,
- show the median values of discharge over time.
- The user is then invited to select one or several points.
- After the selection, depending on the settings in the parameter file the script will show the time series for each selected point one by one and then create the ascii formatted file of those timeseries.
- Finally, if the option is activated, the script will produce a version of the **notebook_atlas** that can be read by SYMPHONIE to help create the **notebook_river**.

Example of output :

```
****************************************
* Making River Timeseries for SYMPHONIE *
  from Param file : Param_Make_TSGlofas.ini
****************************************
... Will turn on interactive mode
```

A window will pop-up :

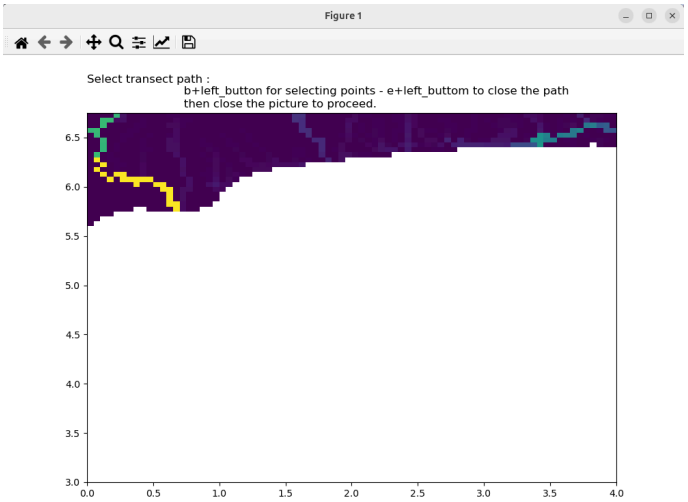
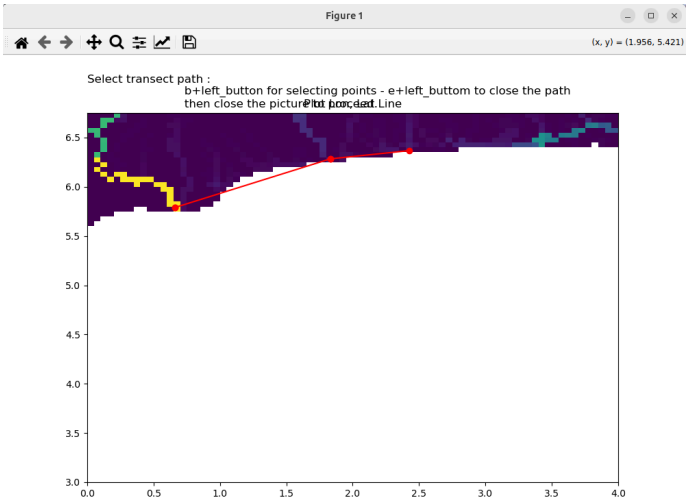
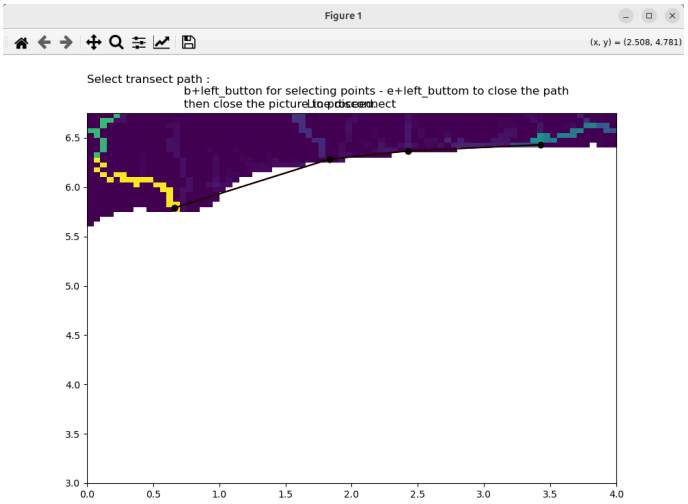


Figure 1
Select transect path :
b+left_button for selecting points - e+left_buttom to close the path
then close the picture to proceed.
6.5
6.0
5.5
5.0
4.5
4.0
3.5
3.0
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0

The user can select a serie of point by using “b+left_click” for each points (the points will be connected by a red line):



Finishing the selection (for the last point) is done by using “e+left_click” :



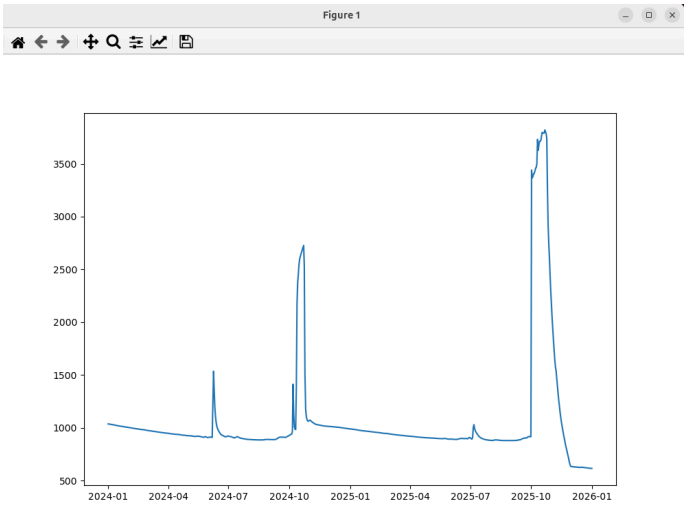
All the points (and lines connecting them) turn from red to black. That means that the selection is over. Then close the image to continue the script.

After several information about the search of the location of point in the dataset.

on_press : b-key + left pressed 1 0.6606451612903231 5.789888682745824
on_press : b-key + left pressed 1 1.8322580645161306 6.283858998144712
on_press : b-key + left pressed 1 2.425806451612905 6.36734693877551
on_stop :e-key+ left pressed 1 3.4270967741935507 6.429962894248608
Last selection OK => Interactive transect on map DISCONNECTED
Disconnecting both mpl events id cid1, cid2
@search_transect_nearest_gridpoint : now get transect indices
@@@ obs 0
3921.2898972464995 2294.077647251595 [2294.07440652]
@@@ obs 0
….
Obs Geographical point 3.4271 6.4300
Matching grid config point 3.4250 6.4250
Matching grid config point 3.4250 6.4250

Grid Indices :
(eta,xi)=(68,6)
(jj,ii) =(68,6)
Distance 598.51 [m]
13 19

If you have select the “if_plot” option, the script will generate a visualisation of each timeseries one by one :



The user will have to close the image to proceed to the next one.

At the same time if you have selected the plot option or not, the script will save ascii formatted files of the timeseries (one per point, numbered from 0 to “N-1”).

After the last image the script ask if the user want to change the selection :
Another transect ? y/n :

If “y” the script will restart at the selection step.
If “n” the script will stop or if the option to create the **notebook_atlas**.

At the end the script give the following message if the script stop properly :

The notebook_atlas : notebook_atlas has been generated

****
* Script : Make_timeseries_glofas.py success
****

If asked, the **notebook_atlas** will look like :

```
*******************************************************************************
riverfile_0
   5.782931354359924  0.676129032258065
      10 ! distance (grid indexes) from the mouth where upwind advection is applied
 -999.000 ! Default Value (if no river file) River discharge  m3/s      RIVERFLUX(:,1)
   10.000 ! In case of an additionnal water way, give
HRIVER,CROSSRESOLRIVER,ALONGRESOLRIVER of the firt point
```

```
   0.000 ! Incoming salinity                                    RIVER_S(:)
   8.000 ! Annual minimum value of the incoming temperature          RIVER_TMIN(:)
      2 ! Month of the annual minimum value of the incoming temperature
  22.000 ! Annual maximim value of the incoming temperature          RIVER_TMAX(:)
      0 ! Boundary condition scheme at river grid input cell         RIVER_OBCTYPE(:)
      1 ! 1 if a data discharge file is available, 0 otherwise       REALRIVER(:)
 24.     ! Discharge data file periodicity (hours)                   RIVERINFO(:)
../../../REGION/RIVER/riverfile_0.txt
2024 01  01   12     00      00  ! Year Month Day Hour Minute Second of the first data
*******************************************************************************
riverfile_1
  5.782931354359924  0.676129032258065
     10 ! distance (grid indexes) from the mouth where upwind advection is applied
 -999.000 ! Default Value (if no river file) River discharge  m3/s      RIVERFLUX(:,1)
  10.000 ! In case of an additionnal water way, give
HRIVER,CROSSRESOLRIVER,ALONGRESOLRIVER of the firt point
   0.000 ! Incoming salinity                                    RIVER_S(:)
   8.000 ! Annual minimum value of the incoming temperature          RIVER_TMIN(:)
      2 ! Month of the annual minimum value of the incoming temperature
  22.000 ! Annual maximim value of the incoming temperature          RIVER_TMAX(:)
      0 ! Boundary condition scheme at river grid input cell         RIVER_OBCTYPE(:)
      1 ! 1 if a data discharge file is available, 0 otherwise       REALRIVER(:)
 24.     ! Discharge data file periodicity (hours)                   RIVERINFO(:)
../../../REGION/RIVER/riverfile_1.txt
2024 01  01   12     00      00  ! Year Month Day Hour Minute Second of the first data
*******************************************************************************
```

The script is controlled by the parameter file : **Param_Make_TSGlofas.ini**

```
[Datain]
dirin=./TEST/BENIN
#pattern=['Benin_test*nc','Benin_forecast*nc']
pattern=['Benin_forecast*nc','Benin_test*nc',]
lonnam=longitude
latnam=latitude
varnam=discharge
[output]
# Output dir
dirout=./BENIN
[plots]
vmin=0
vcenter=50
vmax=500
if_plot=False
[atlas]
make_nbk_atlas=True
notebook_atlas=notebook_atlas
template_file=template_atlas.txt
#template_file=None
path2files=../../../REGION/RIVER/
```

Annexe

1. Python environment, example with conda :

Create and activate a new conda environment :

- *conda create -n retrieve_data*
- *conda activate retrieve_data*

with the following python package :

- *cdsapi*
- *copernicusmarine*
- *xarray*
- *netCDF4*
- *scipy*
- *cfgrib*
- *ipywidgets*
- *scikit-learn*
- *matplotlib*
- *cmocean*
- *cartopy*

Installing the python packages required :

conda install -c conda-forge cdsapi copernicusmarine xarray netCDF4 scipy cfgrib ipywidgets scikit-learn matplotlib cmocean cartopy

2 Getting access to dataset :

ECMWF :

Create an account :

You have to create an account on :
https://accounts.ecmwf.int/auth/realms/ecmwf/login-actions/registration?client_id=cds&tab_id=w8_vpsOWfO4

Then follow some steps :
https://cds.climate.copernicus.eu/how-to-api

Setup the “CDS” Api key

Here is how to setup the CDS Api key:

1. If you do not have an account yet, please register
2. If you are not logged in, please login

3. Once logged in, copy the code displayed below to the file
   $HOME/.cdsapirc_cds
   (in your Unix/Linux environment)

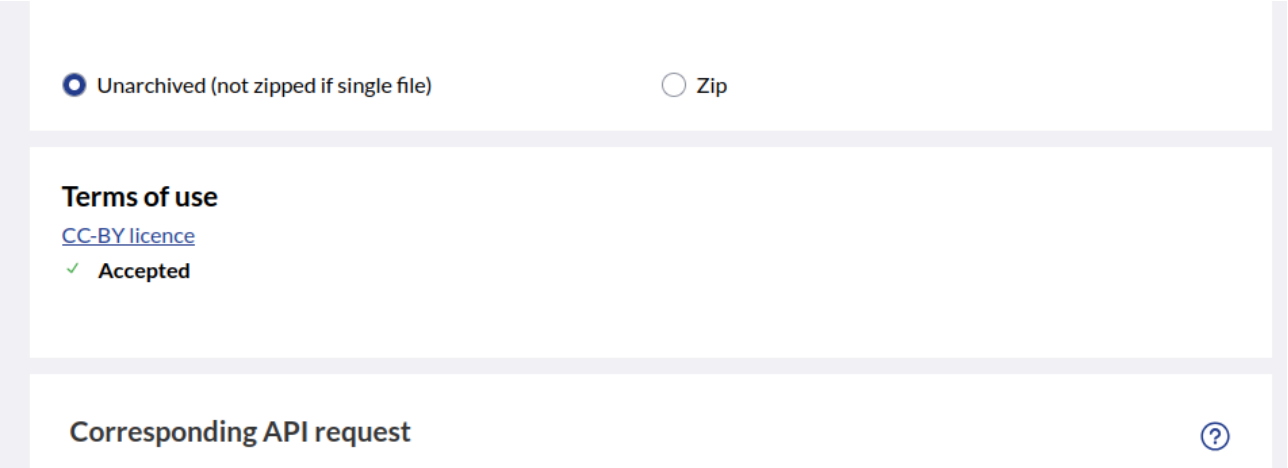
   url: https://cds.climate.copernicus.eu/api
   key: xxxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxx

N.B. Here the file name is changed comparing to the instruction given on the website above.
The reason is that to retrieve ECMWF (ERA5) or GLOFAS data, the python library cdsapi will be using, by default, the same ".cdsapirc" file but with different url :

cds.climate.copernicus.eu/api for ecmwf,
ewds.climate.copernicus.eu/api for glofas

Accept the Terms of use :

!!! Before using the script : Need to accept first on the website to download locally with the API !!!
By making a request here :
https://cds.climate.copernicus.eu/datasets/reanalysis-era5-single-levels?tab=download



If while the script is launch you get the message :
requests.exceptions.HTTPError: 403 Client Error: Forbidden for url: https://cds.climate.copernicus.eu/api/retrieve/v1/processes/reanalysis-era5-single-levels/execution
required licences not accepted
Not all the required licences have been accepted; please visit https://cds.climate.copernicus.eu/datasets/reanalysis-era5-single-levels?tab=download#manage-licences to accept the required licence(s).

You will need to check the terms of use as mentioned :
https://cds.climate.copernicus.eu/datasets/reanalysis-era5-single-levels?tab=download#manage-licences

GLOFAS :

Create an account :

https://ewds.climate.copernicus.eu/

It is the same steps as : Create an account for ECMWF:

You have to create an account on :
https://accounts.ecmwf.int/auth/realms/ecmwf/login-actions/registration?client_id=cds&tab_id=w8_vpsOWfO4

Then follow some steps :
https://cds.climate.copernicus.eu/how-to-api

Setup the “EWDS” Api key

Here is how to setup the CDS Api key:

1. If you do not have an account yet, please register
2. If you are not logged in, please login
3. Once logged in, copy the code displayed below to the file
   $HOME/.cdsapirc_ewds
   (in your Unix/Linux environment)

   url: https://ewds.climate.copernicus.eu/api
   key: xxxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxx

N.B. Here the file name is changed comparing to the instruction given on the website above.
The reason is that to retrieve ECMWF (ERA5) or GLOFAS data, the python library cdsapi will be using, by default, the same “.cdsapirc” file but with different url :

cds.climate.copernicus.eu/api for ecmwf,
ewds.climate.copernicus.eu/api for glofas

Accept the Terms of use :

!!! Before using the script : Need to accept first on the website to download locally with the API !!!
By making a request here :

forecast :
https://ewds.climate.copernicus.eu/datasets/cems-glofas-forecast?tab=download

reanalyses :
https://ewds.climate.copernicus.eu/datasets/cems-glofas-historical?tab=download

CMEMS :

Create an account :

https://data.marine.copernicus.eu/register?redirect=%2Fproducts

https://help.marine.copernicus.eu/en/articles/4220332-how-to-sign-up-for-copernicus-marine-service

Set local credentials :

Under python, you will need your login and password :

```
import copernicusmarine

copernicusmarine.login()
```

Output :

INFO - 2026-08-17T09:48:10Z - Downloading Copernicus Marine data requires a Copernicus Marine username and password, sign up for free at: https://data.marine.copernicus.eu/register

Copernicus Marine username: mylogin

Copernicus Marine password: mypassword

INFO - 2026-08-17T09:48:16Z - Credentials file stored in ~/.copernicusmarine/.copernicusmarine-credentials.

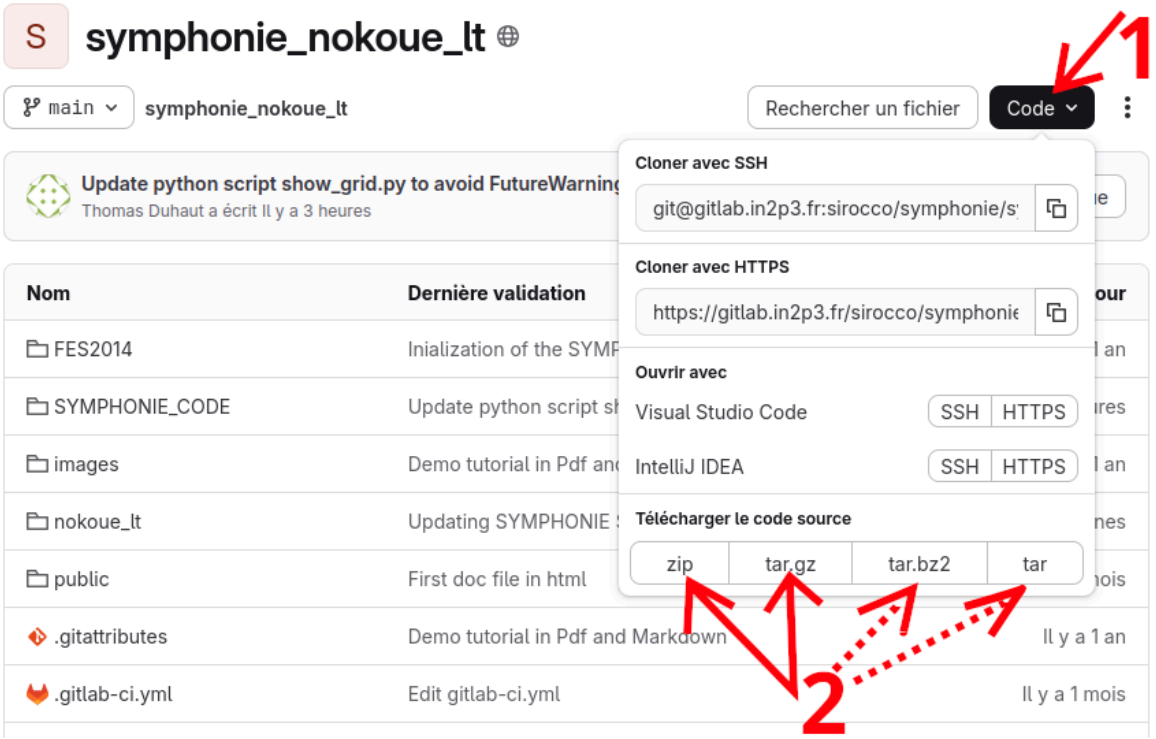
True

Gitlab SYMPHONIE :

It is possible to get a demo version of the latest of SYMPHONIE from the Gitlab Repo :
https://gitlab.in2p3.fr/sirocco/symphonie/symphonie_demo/symphonie_nokoue_lt

Using GIT : from the place you want to set your local repo :
git clone
https://gitlab.in2p3.fr/sirocco/symphonie/symphonie_demo/symphonie_nokoue_lt.git

Or downloading a tarball (or zip etc…) archive :



Then you will have to extract the archive using the appropriate tool : tar, zip …etc