MANUAL FOR DATA_MED TOOLS

Contibutor : Clément Fontana

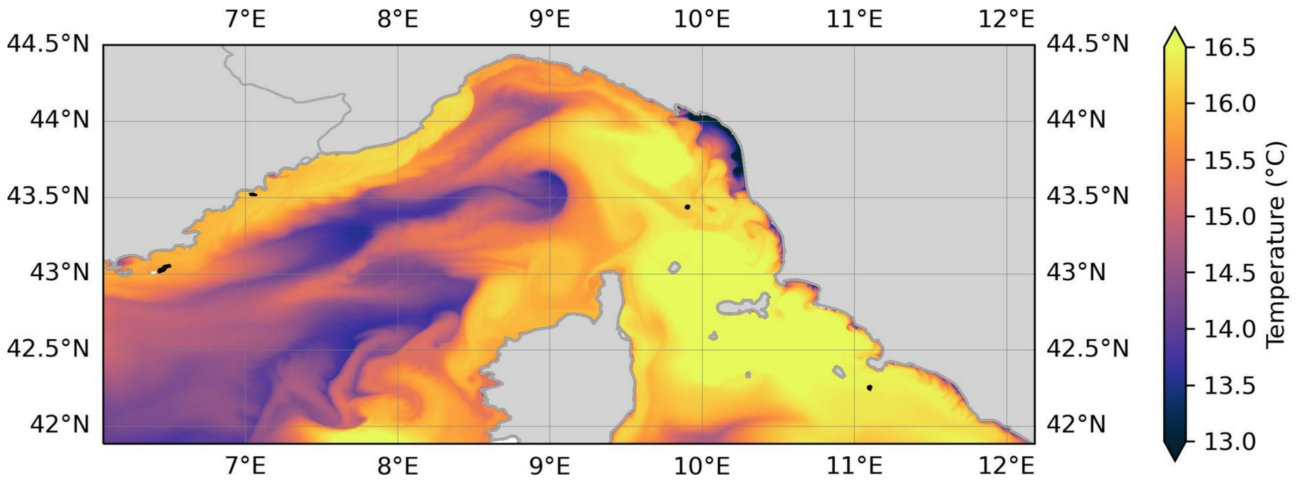


Table of Contents

contact : cfontana [a] ogs.it

Introduction

This document presents the architecture and instructions to process oceanographic model outputs and data in the Mediterranean Sea using the MITgcm model.

Using other models remain possible through simple nomenclature/format conversion. Contact me for any questions.

General arborescence

```
├── config
│   └── config_TEMPLATE.json
├── datasets
│   ├── BOUSSOLE
│   ├── DYFAMED
│   │   ├── dyfamed-buoy
│   │   ├── dyfamed-mooring
│   │   ├── dyfamed-vessel
│   │   └── read_dyfamed.py
│   ├── MY_NEW_DATA
│   │   └── my_new_routine.py
│   └── RIVERS
│       ├── data
│       ├── read_rivers.py
│       └── readyfiles
├── modules
│   ├── fun_gen.py
│   ├── fun_io.py
│   ├── fun_plot_2D.py
│   └── fun_plot_TS.py
└── plot
    ├── make_plot_2D.ksh
    ├── make_plot_2D.py
    └── make_plot_section.py
```

Setup

git clone git@github.com:cfontana/DATA_MED $DIR

Required Python toolbox

numpy, scipy, copernicusmarine, cartopy, matplotlib, xarray

Environment variables

$ set $DIR/modules to your PYTHONPATH in ~/.bashrc

$ export env variable DATA_MED_DIR as /path/to/DATA_MED

$bash

Create your own configuration file

$ cd config

$ cp config_TEMPLATE.json config_MYCONFIG.json

$ cp variables_mit_TEMPLATE.dat variabonfig_MYCONFIG.jsonles_mit_MYCONFYG.dat

(Tag MYCONFIG can be anything)

Parameters description

<u>**Tune your config_MYCONFIG.json file considering parameters below**</u>

```
"_____":"GENERAL PARAMETERS",

"date_ini":"2021-12-11",                      # Initial date of model data
"date_end":"2021-12-12",                      # Final date of model data
"outdir":"/path/to/OUTPUT_SMP",               # Directory containing outputs
"diagdir":"/path/to/diags/dir",               # Directory to save diagnostics files
"river_ini":"2021-01-01",                     # River initial date
"river_end":"2021-12-31",                     # River final date

"_____":"RIVER PARAMETERS",
"french_list":["Argens","Var"],                             # French river list (see README)
"italian_list":["Arno","Magra","Ombrone","Serchio"],    # Italian  river list (see README)

"_____":"DYFAMED PARAMETERS",
"some_par":"some_par",              # To be defined

"_____":"BOUSSOLE PARAMETERS",
"some_par":"some_par",    # To be defined


"_____":"PLOT PARAMETERS",

"resol":"10m",                           # Cartopy coast/land resolution
```

```
"fig_proj":"ccrs.PlateCarree()",           # Cartopy projection
"fig_sx":"8",                              # Horizontal 2D figure size
"fig_sy":"8",                              # Vertical 2D figure size
"fig_fmt":"jpg",                           # Output figure format
"fig_res":"300",                           # Output figure resolution
"tight":"False",                           # Crop white space (time-consuming)
"fig_tck_size":"10",                       # Figure tick size
"fig_tcklbl_size":"10",                    # Figure tick label font size
"fig_lbl_size":"10",                       # Figure label font size
"cb_fraction_2D":"0.02",                   # Python colorbar fraction parameter
"cb_pad_2D":"0.12",                        # Python colorbar pading parameter

"fig_secx":"8",                            # Horizontal in-depth section figure size
"fig_secy":"4",                            # Vertical in-depth section figure size
"sec_lon1":"8.5",                          # Longitude starting point for section
"sec_lat1":"43.",                          # Latitude starting point for section
"sec_lon2":"10.",                          # Longitude ending point for section

"sec_dep":"600",                           # Maximum depth for section
"sec_resH":"0.05",                         # Horizontal resolution for interpolation
"sec_resV":"1",                            # Vertical resolution for interpolation
"cb_fraction_sec":"0.025",                 # Python colorbar fraction parameter for section plot
"cb_pad_sec":"0.06",                       # Python colorbar fraction parameter for section plot



"_____":"OPTIONS",

"itp_meth":"nearest",             # Interpolation method, set nearest for tests or linear
```

Variable options description

```
$ cp variables_mit_TEMPLATE.dat variables_mit_MYCONFIG.dat
```

Description	**Name**	**File tag**	**Colormap**	**Log plot**	**Limit mode**	**Plot min value**	**Plot max value**	**Label**	**Units**
Options				True/ False	set/auto				

Example	Chl	PFTC	cmc.imola	True	Set	0.05	0.25	Chlorophyll	mg.m-3

Plot directory routines

This directory contains routines to perform plots. Diagnostics arborescence is automatically created and name of file saved prompted.

=> *Routine make_plot_2D.py :*

Description : Plot 2D spatial maps on the domain

Arguments : configuration_name variable_name z-level

Bash utilization with *make_plot_2D.ksh*

=> *Routine make_plot_section.py :*

Description : Plot vertical section on the domain

Arguments : configuration_name variable_name direction coordinate_value

direction argument can be horizontal or vertical

If direction is horizontal, routine plots the section from ‘lon1’ to ‘lon2’ at latitudinal ‘coordinate_value’
If direction is vertical, routine plots the section from ‘lat1’ to ‘lat2’ at longitudinal ‘coordinate_value’

Same possible bash utilization as *make_plot_2D.py*

RIVERS datasets routines

This routine create river runoff input files at the MITgcm format

The data can be download from https://www.hydro.eaufrance.fr/ for french river and http://www.sir.toscana.it/consistenza-rete for italian ones.

Get csv files from both sites and save them as river_name.csv in the /datasets/RIVERS/data

```
$ python3 read_rivers.py
===> produces MITgcm ready to read files in /datasets/RIVERS/readyfiles
```

The routine raises an error if data are not continuous. In the case you must pre-process the data (*e.g.* interpolation/climatology) and save the processed file in data/river_name_modif.csv

The routine checks for existence of this file before reading the raw one.

Test data set

You can perform tests using MITgcm NetCDF outputs available in /OUTPUT_SMP. Just set the outdir and diagdir variables in your configuration file.

Create your own directory

You can easily access existing function and create your own directory to process data.
For an example of use :

```
$ cd datasets/MY_NEW_DATA
$ python3 my_new_routine.py MYCONFIG
```

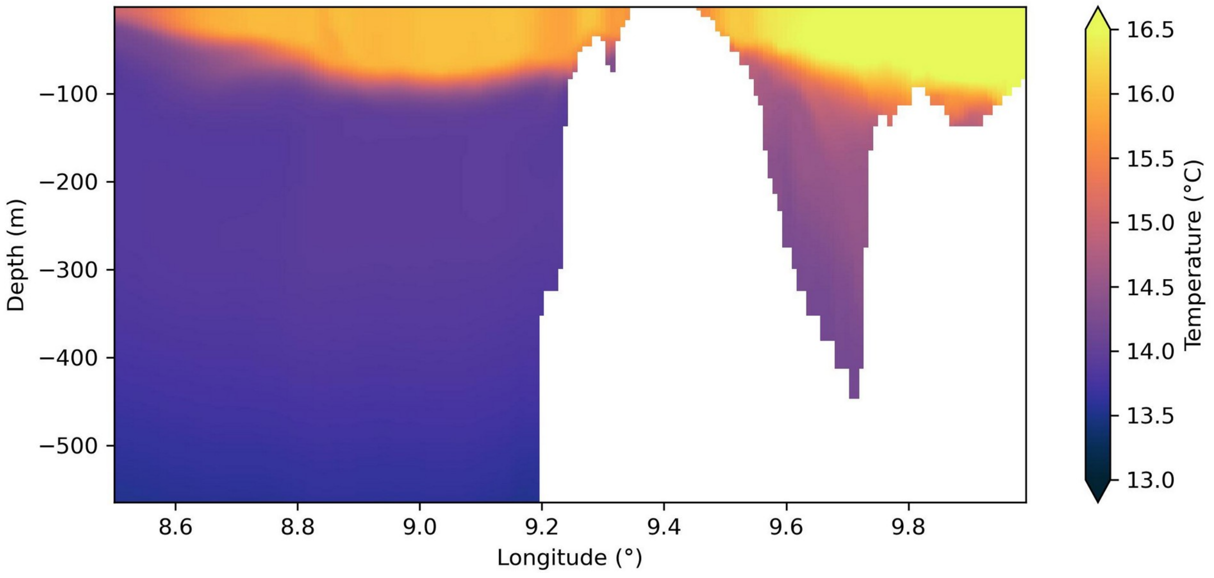
TIPS :

- The configuration is read dynamically and its variables are global inside the system once loaded.
- You can add your own configuration parameter in file /config/config_CUSTOM.json and load it using load_config(CUSTOM) in your routines. We could include them in the general file later.

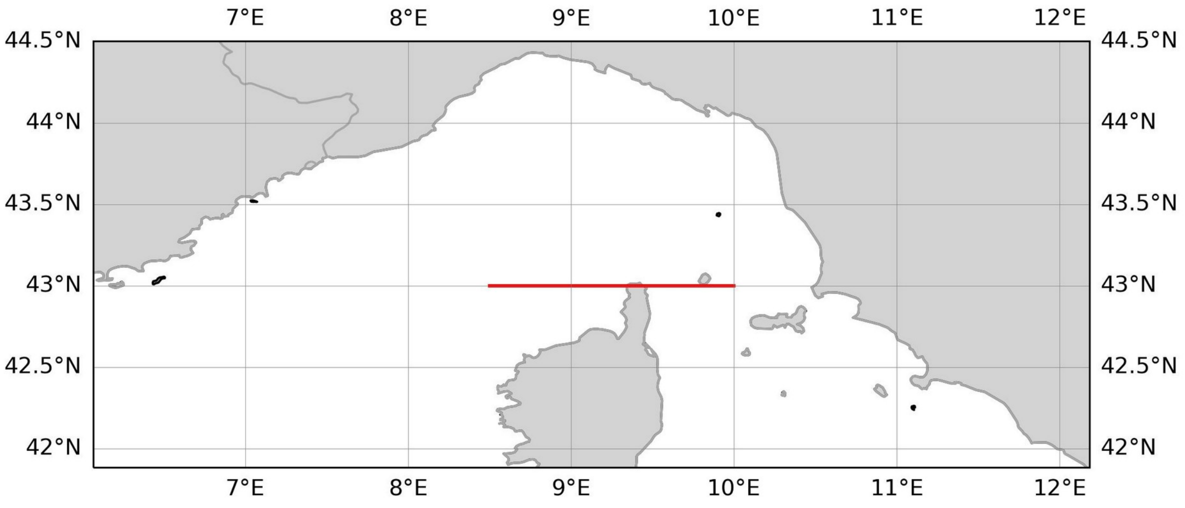
- You can also create your own functions in /modules and load them as others.
- Check for /module/fun_* for already existing functions (e.g. *savefig* in fun_io.py)

ENJOY !

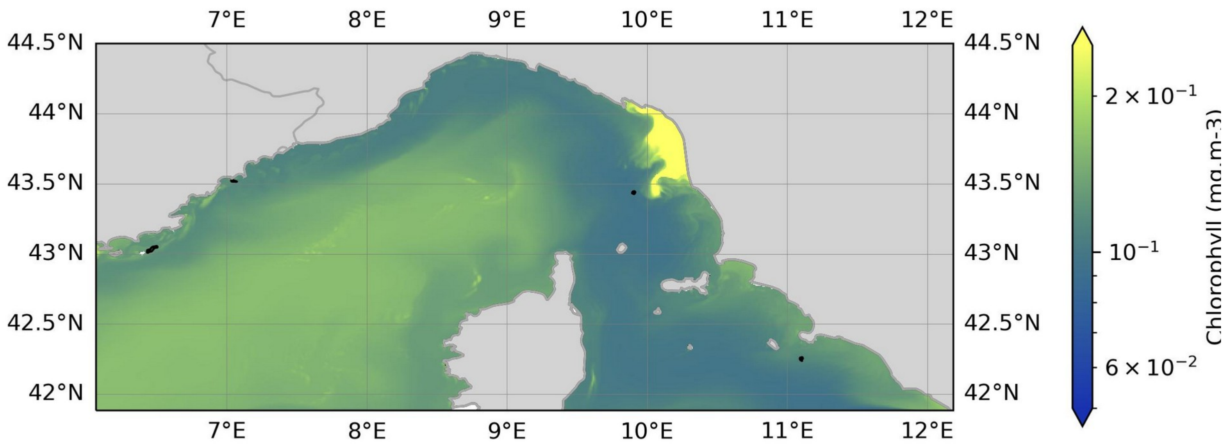
Gallery



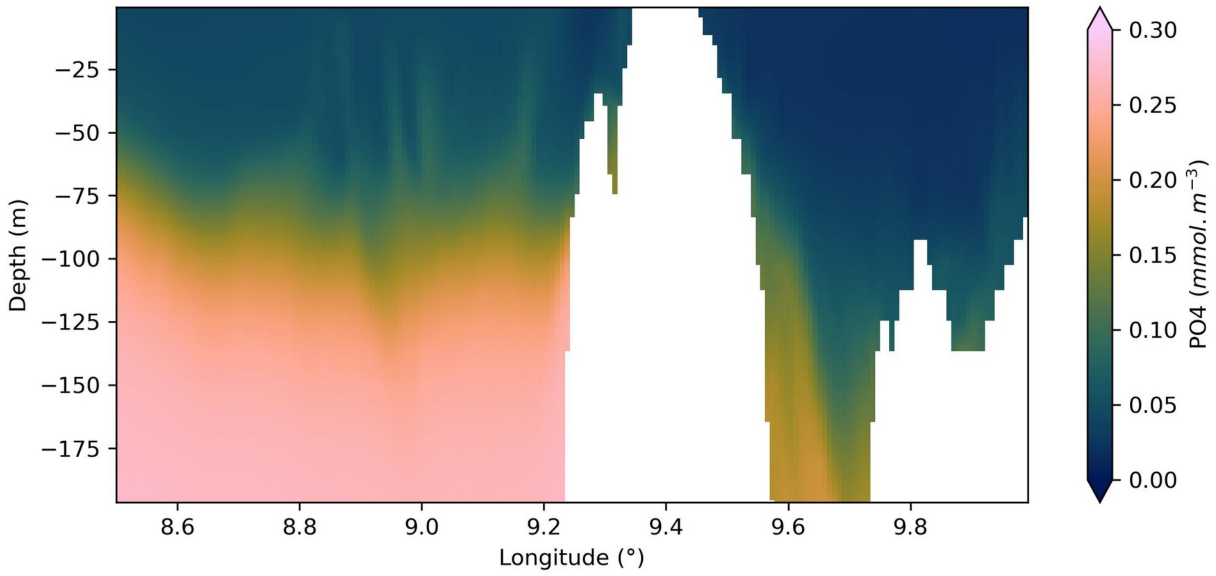
−100
−200
−300
−400
−500
Depth (m)
8.6
8.8
9.0
9.2
9.4
9.6
9.8
Longitude (°)
16.5
16.0
15.5
15.0
14.5
14.0
13.5
13.0
Temperature (°C)



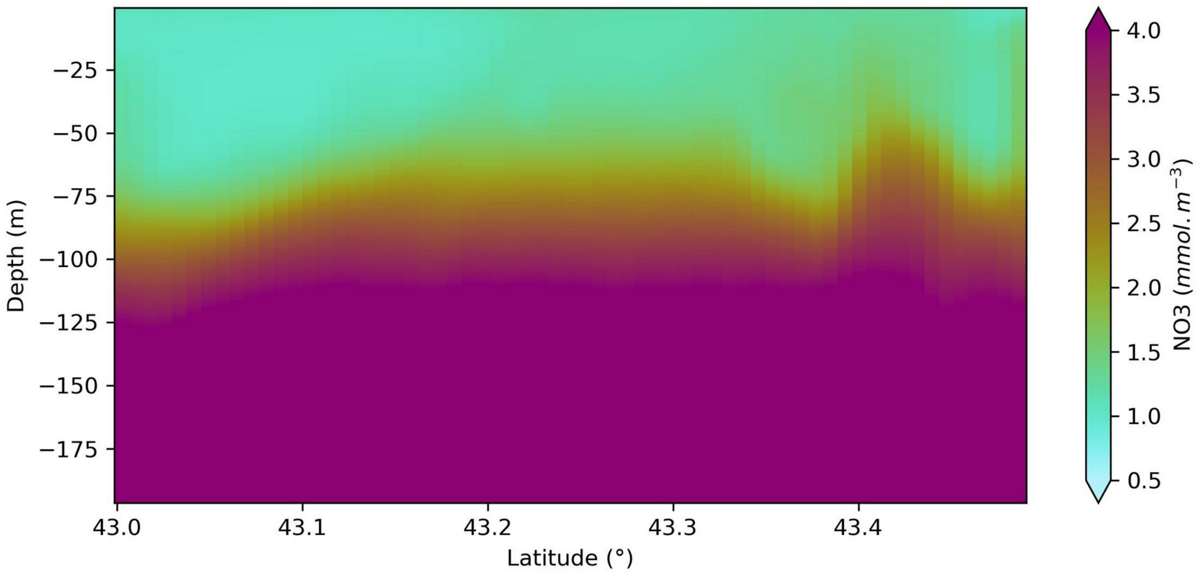
7°E
8°E
9°E
10°E
11°E
12°E
44.5°N
44°N
43.5°N
43°N
42.5°N
42°N



7°E
8°E
9°E
10°E
11°E
12°E
44.5°N
44°N
43.5°N
43°N
42.5°N
42°N
2×10^{-1}
10^{-1}
6×10^{-2}
Chlorophyll (mg.m-3)



−25
−50
−75
−100
−125
−150
−175
Depth (m)
8.6
8.8
9.0
9.2
9.4
9.6
9.8
Longitude (°)
0.30
0.25
0.20
0.15
0.10
0.05
0.00
PO4 ($mmol.m^{-3}$)


Depth (m)
−25
−50
−75
−100
−125
−150
−175
43.0
43.1
43.2
43.3
43.4
Latitude (°)
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
NO3 ($mmol.m^{-3}$)