MMES Documentation

Release 1.0

Amedeo Fadini - CNR ISMAR

Jan 25, 2022

CONTENTS:

1 Introduction **1**

2 Software description **3**
- 2.1 Inputs . . . 3
- 2.2 Directory structure . . . 3
- 2.3 1. Retrieve and download phase . . . 6
- 2.4 2. Processing phase . . . 8
- 2.5 3. Ensemble creation phase . . . 10
- 2.6 MMES Outputs . . . 11

3 Installation **13**

4 Configuration files **15**
- 4.1 General Configuration . . . 15
- 4.2 Suorce Configuration . . . 15
- 4.3 Processing steps Configuration . . . 16

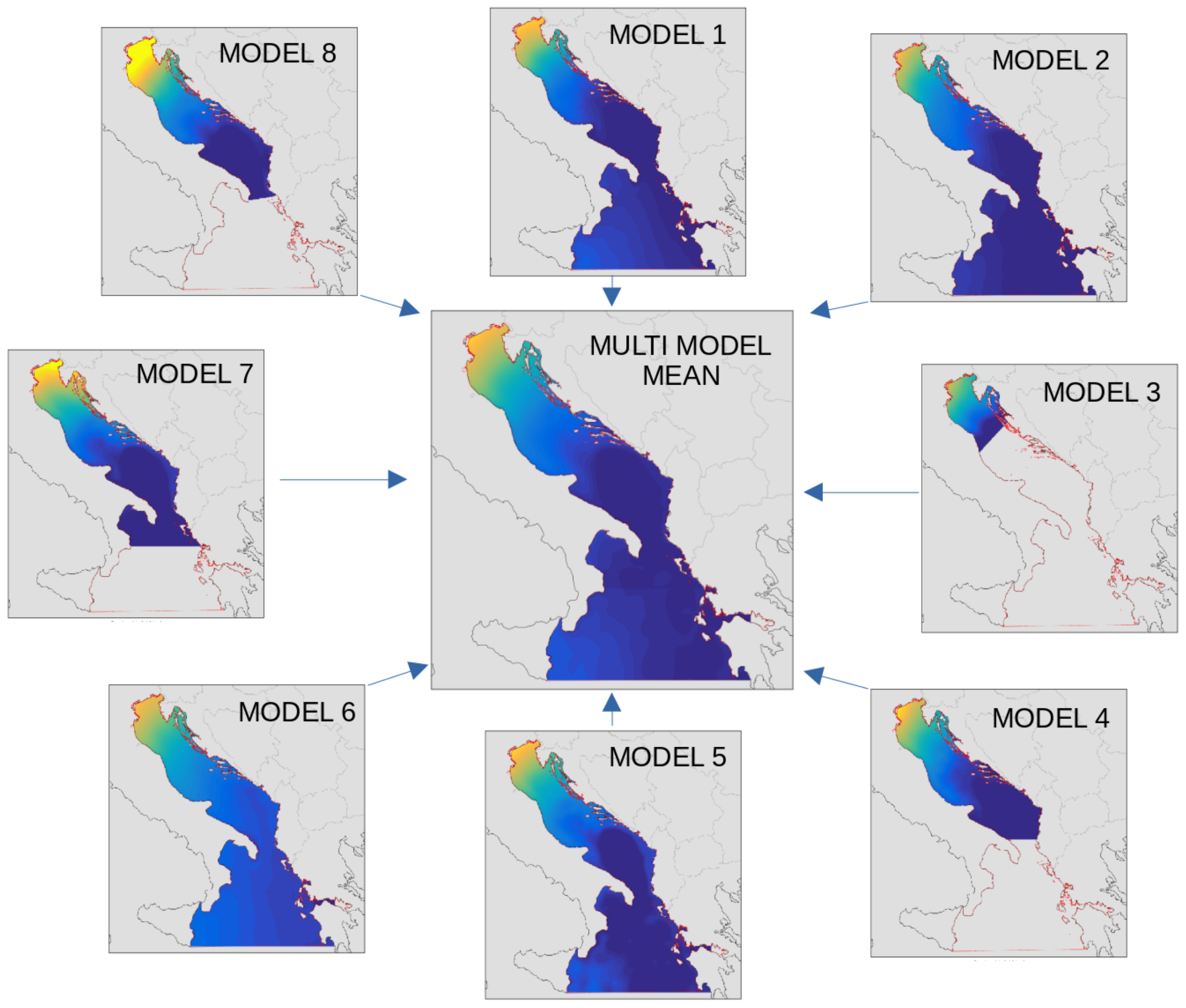
CHAPTER

ONE

INTRODUCTION

Multi model ensemble system (MMES)

This software is intended to collect daily oceanographic forecasts from different sources and create a Multi-model Ensemble. The software is specifically designed to publish NetCDF files about sea-level and waves, but can be configured to manage other variables.



Ocean forecast results are collected by the system every day in the morning: the program contacts each provider of the list, checks if an updated model exists, downloads it and stores it on a local filesystem using one folder for each node with current and historical data. If the updated forecast is not present in the node, the system will pass to the next node and retry later. Once all forecasts available are downloaded, the multi-model builder prepares the data harmonizing all different forecasts. The ensemble creation procedure can be three main task: 1. retrieve and download each single forecast file provided from different sources; 2. process each forecast with appropriate operations; 3. create the ensemble with mean value and estimate error as standard deviation value and archive old ensemble. At the present stage, the download and processing tasks are executed in sequence for each source and the cycle is repeated at specified intervals, including new available resources. The choice of Python scripts instead of bash shells procedure allows to execute more than one task in parallel shortening the time of the whole cycle. The software is capable of downloading data from ftp or http/https servers using credentials stored in the configuration files or using a custom command to be executed on the system shell. The software can retrieve virtually any kind of source.

For more information on the ensemble process see:

Ferrarin, C., Valentini, A., Vodopivec, M., Klaric, D., Massaro, G., Bajo, M., De Pascalis, F., Fadini, A., Ghezzo, M., Menegon, S., Bressan, L., Unguendoli, S., Fettich, A., Jerman, J., Ličer, M., Fustar, L., Papa, A., and Carraro, E.: Integrated sea storm management strategy: the 29 October 2018 event in the Adriatic Sea, Nat. Hazards Earth Syst. Sci., 20, 73–93, doi: 10.5194/nhess-20-73-2020, 2020.

CHAPTER

TWO

SOFTWARE DESCRIPTION

2.1 Inputs

To run the MMES software you need to provide

- a list of sources sources.json: each source can provide one or more models for one or more variable
- a mask file with the final grid you want to get
- [optional] a wheights file for each model that needs to be spatially interpolated to fit the multi-model ansemble
- a small amount of configuration information (directory in which all data will be available, variable to be considered . . .) see [*General Configuration*] section

2.2 Directory structure

The application directory has python scripts and a subdirectory '"scripts"' with bash scripts used to download special sources or called by subprocess during program execution

The data directory should have the following structure:

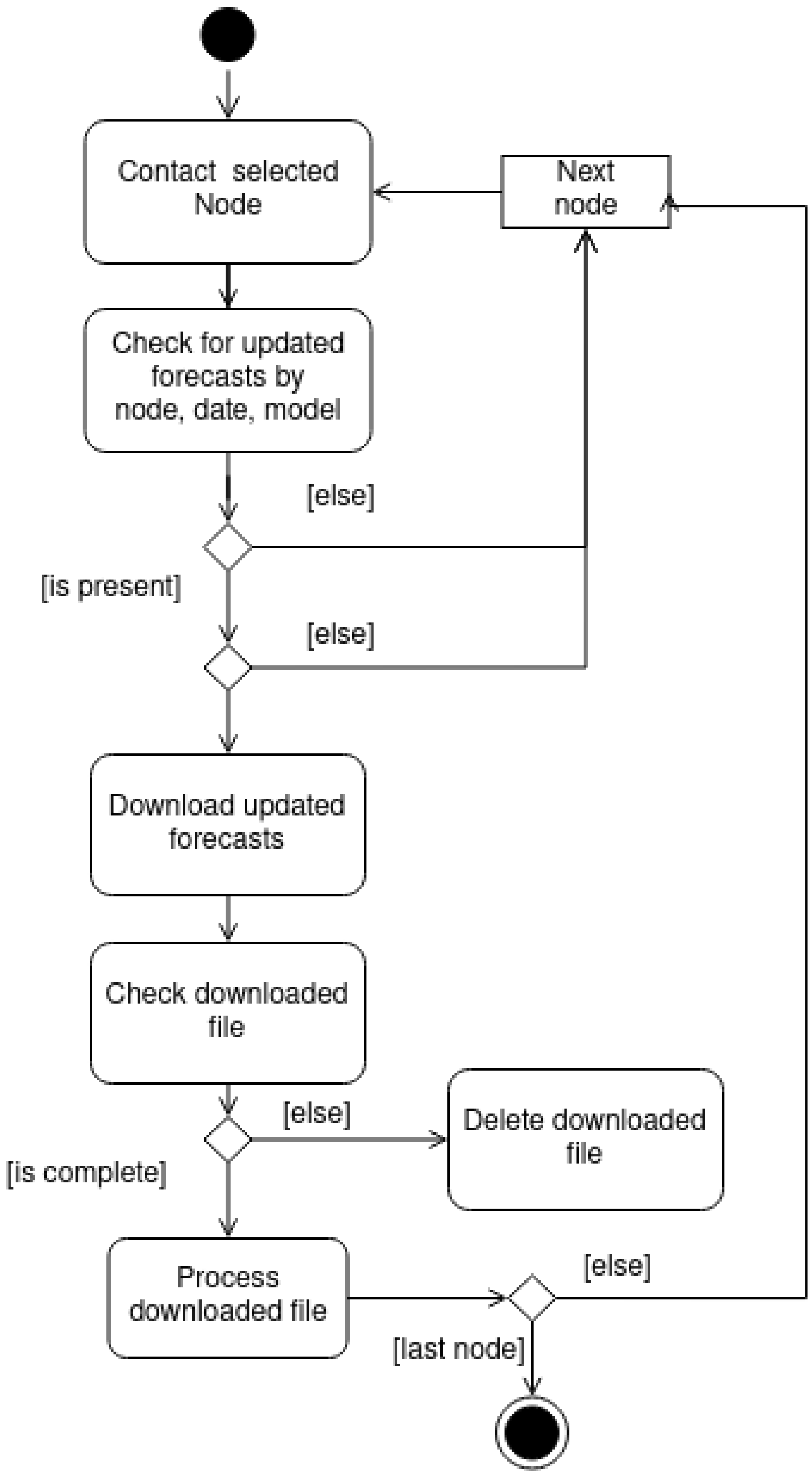
```
data
├── MMES
│ ├── history
├── forecasts
│ ├── SOURCE1
│ ├── SOURCE2
│ ├── SOURCE3
│ ├── . . .
├── mmes_components
│ ├── 20211001
│ ├── 20211002
│ ├── 20211003
│ ├── . . .
├── config
│ ├── mask
│ ├── wheights
├── tmp
```

Inside MMES directory will be placed the daily ensemble produced and inside the MMES/history directory will be stored the old ensamble cutted to first 24 hour The name of the ensemble can be setted in *config.json* file, thus the MMES directory will take the enemble name instead.

If not already presente the directory structure is automatically created by the management command:

```
python manage.py dir
```

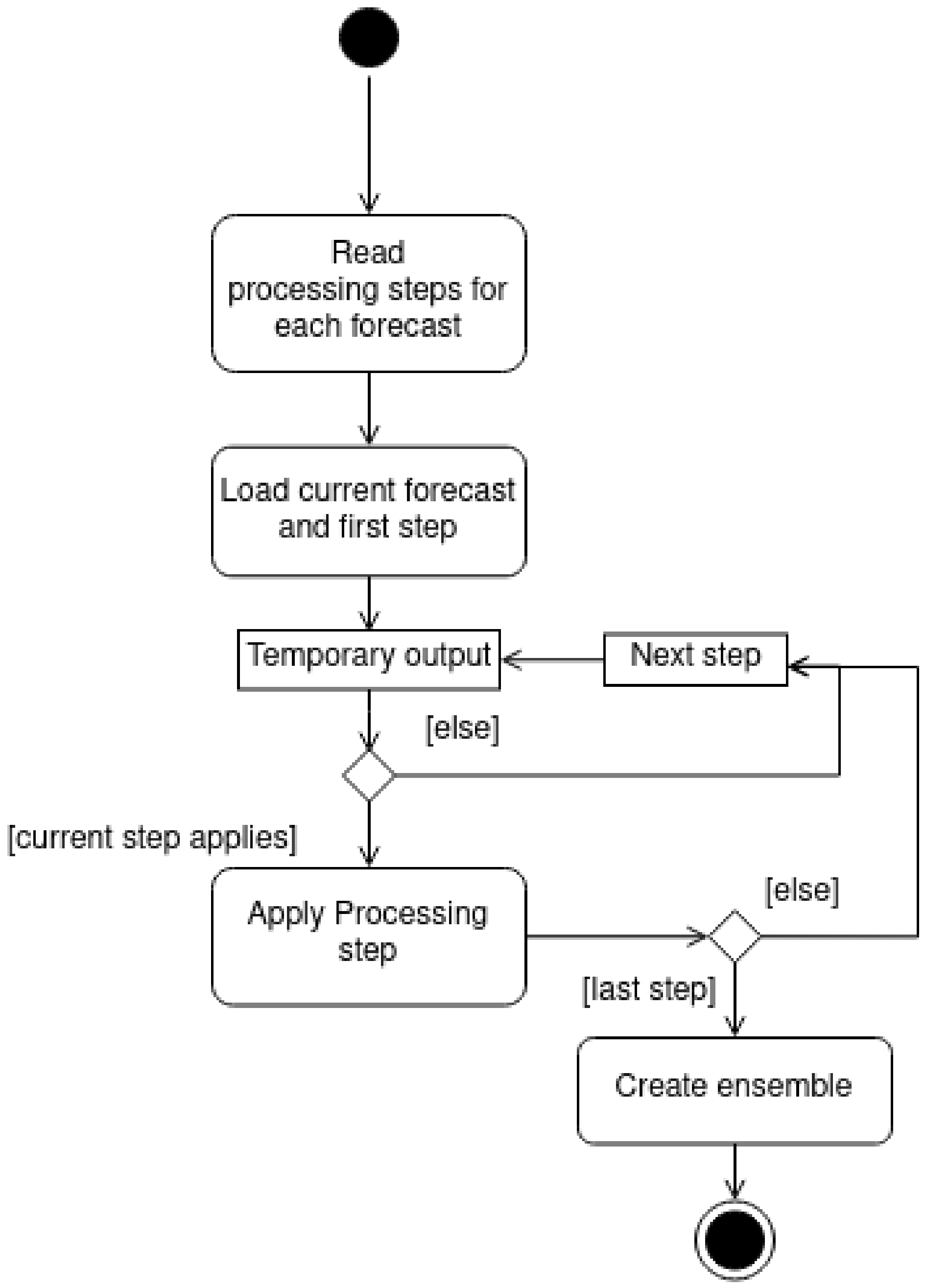
2.3 1. Retrieve and download phase



The source forecasts are provided in different formats and from different types of sources (ftp or http or other). The diagram of the download stage of MMES software is shown in Figure 1. The Daily forecasts are usually published in the morning, but they are not available at the same time and therefore the software contacts all source nodes at regular intervals and checks if the current file is available. If the file is already downloaded and processed the software will pass to the next node. For http sources the exact path of the file to download is needed, for ftp sources the software needs the directory name and filename. The naming schema of the files is different for each provider but usually can be constructed using a constant pattern and current date value.

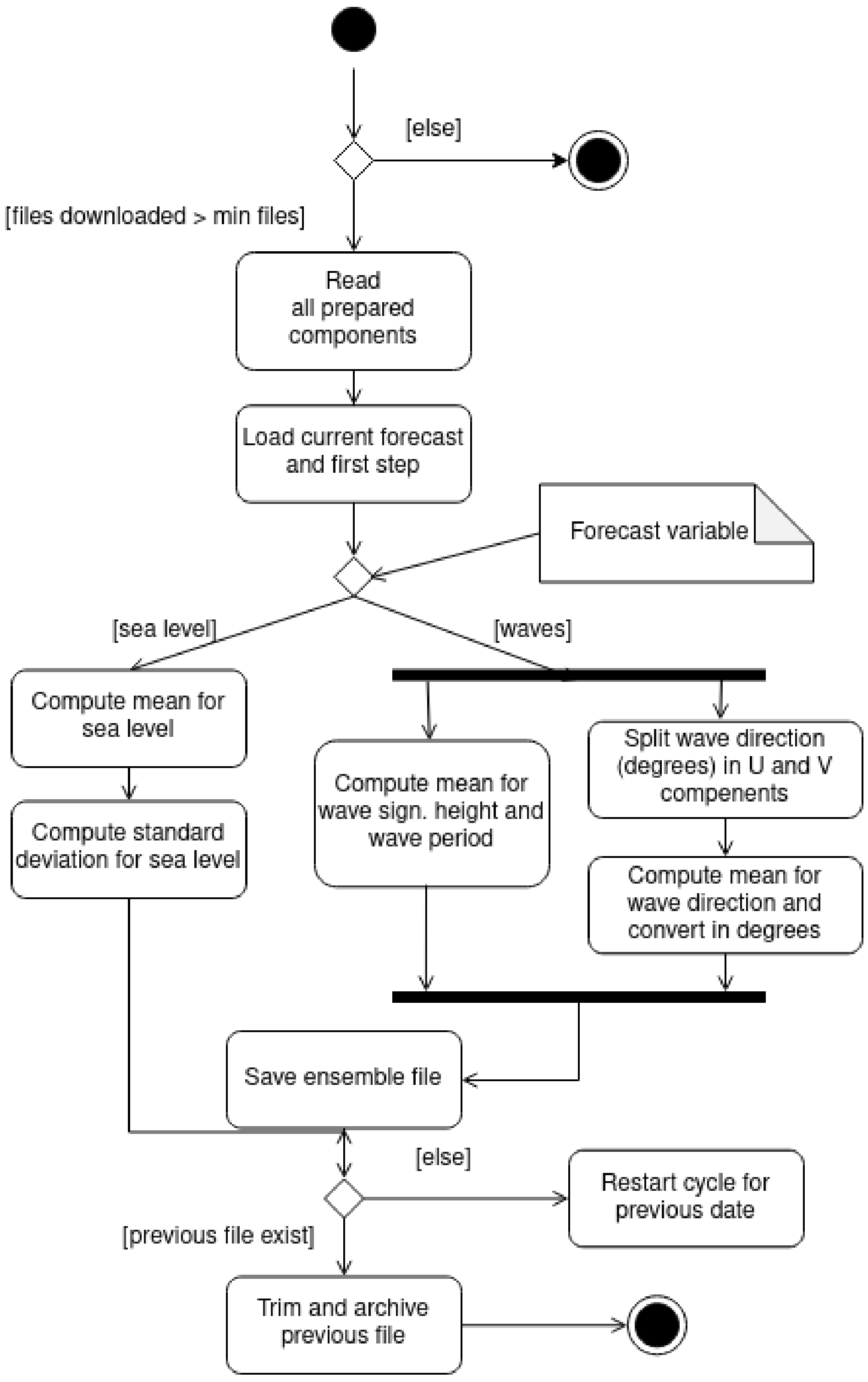
If the download process is interrupted due to network issues or other causes, the file can be incomplete and not suitable to create the ensemble. The forecast duration is checked after download (start time, end time) to ensure that it covers at least the time period of the ensemble (2 days), otherwise the file is deleted so it can be downloaded on the next cycle. If the download process is interrupted due to network issues or other causes, the file can be incomplete and not suitable to create the ensemble. The forecast duration is checked after download (start time, end time) to ensure that it covers at least the time period of the ensemble (2 days), otherwise the file is deleted so it can be downloaded on the next cycle.

2.4 2. Processing phase



If the file is valid, the software will pass to the processing phase. The diagram of the processing stage of MMES software is shown in Figure 2. Each forecast has to be processed in a different way: all the possible steps are implemented in the code (e.g. merge or split variables, rename variables, spatial interpolation on the final grid, temporal interpolation, add tide and offset for sea level, invert wave direction and so on). The processing steps and relative parameters required for each forecast are declared in the configuration files as a JSON object. On each step, a temporary output file is created: the Python cdo wrapper library manages temporary filenames and makes available the data as a Python variable, then clears all temporary files at the end. Then the result is saved as a NetCDF file inside the component's directory. At the end of each processing cycle, the software goes to the ensemble creation phase. The configuration of different processing steps for each variable is the most important part of configuration. See *Processing steps Configuration* for details.

2.5 3. Ensemble creation phase


[else]
[files downloaded > min files]
Read all prepared components
Load current forecast and first step
Forecast variable
[sea level]
[waves]
Compute mean for sea level
Compute standard deviation for sea level
Compute mean for wave sign. height and wave period
Split wave direction (degrees) in U and V compenents
Compute mean for wave direction and convert in degrees
Save ensemble file
[else]
Restart cycle for previous date
[previous file exist]
Trim and archive previous file

The general configuration sets a minimum number of files for the ensemble creation: the ensemble output is overwritten on the next cycles adding more forecasts, when available (last execution is scheduled at 14.00). The diagram of the ensemble precaution stage of MMES software is shown in Figure 3. All numerical model results are interpolated, through a distance-weighted average remapping of the nearest neighbours, on a common regular lat-lon grid covering the Adriatic Sea with a resolution of 0.02 deg. For coastal flooding hazard purposes, the total sea level height must be forecasted. Therefore, the astronomical tidal level values obtained by a specific SHYFEM application over the Mediterranean Sea (Ferrarin et al., 2018) are added to the residual sea level simulated by the operational systems not accounting for the tide (e.g. SHYMED, ISSOS). The obtained sea level heights simulated by the different models are all referred to the geoid.

Figure 3: Diagram of ensemble creation stage of MMES software. The CDO library provides simple commands to compute the mean and standard deviation of a variable. For the wave ensemble we have three different variables, wave significant height, wave period and wave direction: the wave direction is expressed in degrees and must be splitted in the U and V components, then merge the ensembles again. The ensemble forecast duration is 2 days with 48 hourly timesteps, but users can set a different duration in configuration files. When the new ensemble is ready, the previous day is trimmed to the first 24h hours and archived in the history folder: the Thredds data server will publish the whole collection so can be downloaded a subset of custom duration for the past multi-model files.

2.6 MMES Outputs

MMES produces 2-day (duration is defined in mes_functions.py L95)probabilistic forecasts in terms of the ensemble *mean* and *standard* deviation for both the sea level height and wave over the whole Adriatic Sea and part of the Ionian Sea. The spread (i.e. standard deviation) among the operational simulations is expected to represent a measure of the uncertainty of the prediction and should be linked to the forecast error so that cases with the largest spread are those with the highest uncertainty and where a large error of the ensemble mean (and also of the deterministic forecast) is more likely (Flowerdew et al., 2010). It is not straightforward what averaging weights should be used for the multi-model ensemble forecast and therefore we used equally weighted ensemble members, despite the forecasts which are more precise than others should have more importance in the MMES (Salighehdar et al., 2017; Schevenhoven and Selten, 2017). Here we applied a simple average of the forecasts at every timestamp to compute the ensemble mean, but more sophisticated methods based on weighting function determined by comparison of the single model results with near real-time observations will be implemented in future (Di Liberto et al., 2011; Salighehdar et al., 2017). Taking advantage of the near real-time observations acquired by the aggregated monitoring network, the root mean square error of the individual forecast will be next evaluated and stored for long-term statistics. MMES forecasts are produced each day. MMES outputs (in terms of ensemble mean and standard deviation of the sea level and waves) in NetCDF format are available to the end-users and external portals through the CNR-ISMAR Thredds Data Server at the webpage's url https://iws.ismar.cnr.it/thredds/catalog/tmes/catalog.html. The results of the multi-model ensemble system can be visualized via the I-STORMS Geoportal web interfaces (https://iws.seastorms.eu/). The results will be next delivered through the STREAM International Flood Platform.

CHAPTER

THREE

INSTALLATION

The software i sintended to run aona GNU/linux based server

1. Create a python virtualenv and activate it.

```
python3 -m venv /path/to/new/virtual/environment/mmes
source activate /path/to/new/virtual/environment/mmes/bin/activate
```

2. Clone this repository in a convenient location.

```
git clone git@github.com:CNR-ISMAR/mmes.git
```

3. Install requirements with

```
pip install -r requirements.txt
```

4. Prepare the data directory structure according to the *Directory structure* section.

```
python manage.py dir
```

5. Manually edit the general configuration file *config.json* according to your needs (see *General Configuration*) section.

6. From the main directory launch

```
python manage.py new
```

to create a new source config file or

```
python manage.py mod
```

to edit existing source config file. Refer to *Configuration files* section fro detailed information about config files

7. For each source forecast you need to add the model name to required step in *processing.json* *Processing steps Configuration*

CHAPTER

FOUR

CONFIGURATION FILES

4.1 General Configuration

General configuration directives are stored in:guilabel:*config.json* file The file has the following directives:

- **data_dir** root of your data dir see *Directory structure*
- **sources_file** name of sources config file (default: *sources.json*)
- **ensemble_name** name of the enemble yo're going to create, this name will used for filenames
- **ensemble_variables** a JSON objects with the variable of each ensemble file (will be used in filename) and the related array for variable names in the final NetCDF file. Please note that the original variables of each forecast will be renamed following this order according to the processing.json file
- **mask_file** name of final grid of ensemble, all value outside this mask will be set to nodata
- **gap_days** max umber of days that will be attempted to be automatically gap-filled: check the output at least every n days

Default config.json:

```
{
"data_dir": "/usr3/iwsdata",
"sources_file": "sources.json",
"ensemble_name": "TMES",
"ensemble_variables": {"sea_level": ["sea_level"], "waves": ["whs", "wmp", "wmd"]},
"mask_file": "{data_dir}/config/mask/TMES_mask_002_ext.nc",
"gap_days":  "5"
}
```

The config value will be loaded **recursively** by loadconfig() function so in the file you can use previously declared values between curly brackets such as {data_dir} that will be replaced by the actual **data_dir** value.

4.2 Suorce Configuration

Source list are stored by default in the *sources.json* file in the root directory, the filename can be customized in general config *config.json*

Refer to *source_template.json* to create your own configuration file.

If the ensemble creation for sea_level include models that need to add tide to express total sea level we suggest to put the tide source in the first place of config file.

You can also run from command line:

```
` python manage.py new `
```

This will start a *step by step* procedure to create a new sources.json file. If something goes wrong you'll find a convenient backup of your last configuration on sources.json.bak

4.3 Processing steps Configuration

The processing steps for each model are defined in *processing.json* file. You can edit the file and add the model system name to tha step of your interest under each variable group

The processing configuration file is composed by two section: *sea_level_prepare* and *waves_prepare* when modify or adding new models the manage.py script help to add (or remove) the current model to each step.

4.3.1 Sea level prepare

In this table are described the steps for sea level prepare

Table 1: Processing steps sea level

JSON name	Description	Can be skipped if…
variable_selection	Select only sea level variable	source model has only sea level variable: original variable name is setted in *Suorce Configuration*
temporal_interpolation	Temporal interpolation to match start, end and timesteps	source model has exact timesteps of the ensemble
get_48hours	Get first 48hours of forecast model	source model is already 48hours long
add_factor	Add a specific offset factor (setted in *Suorce Configuration* as `sea_level_fact`) for this model to match reference level	the reference level is the same of ensemble
mask_before_interpolation	Add a mask to some part of the model before interpolation (values setted in *Suorce Configuration*)	All values of source model are suitable
spatial_interpolation	Interpolate to match same grid of the ensemble	already match same grid of the ensemble
extrapolate_missing	Extrapolate missing values	No need to fill missing value in source model
mask_after_interpolation	Add a mask to some part of the model after interpolation (values setted in *Suorce Configuration*)	All values of source model are suitable
mask_outside_area	Mask value outside area of interest	The extension of source model is the same of ensemble
add_tide	Add astronomical tide model to source forecast model	Forecast model already integrte tide.

4.3.2 Waves prepare

Table 2: Processing steps waves

JSON name	Description	Can be skipped if. . .
merge_components	Merge multiple files from the source forecast to have allthe waves variables (Wave Significant Height, Wave period and Wave direction) in the same file.	Source forecast model already have the three variables in the same file.
variable_selection	Select from source file only the three variables about waves (variable names setted in *Suorce Configuration*) and rename them according to *ensemble_variables* in *config.json*.	The source file has only the variables about waves and
invert_latitude	Invert latitude direction usng `cdo invertlat` command	The latitude is already coherent with the ensemble
set_miss_value	Fill missing value with `cdo setmissval` . Missing value is setted in *Suorce Configuration* (`miss_value`)	The source forecast
change_int_float	Change data type of variables integer float	Data type for all variables are already float
temporal_interpolation	Temporal interpolation to match start, end and timesteps	source model has exact timesteps of the ensemble
get_48hours	Get 48hours	Step description
set_grid_unstructured	Set grid unstructured	Step description
spatial_interpolation	Spatial interpolation	Step description
extrapolate_missing	Extrapolate missing value with `` `cdo fillmiss` ``	Step description
mask_after_interpolation	Add a mask to some part of the model after interpolation (values setted in *Suorce Configuration*)	All values of source model are suitable
mask_outside_area	Mask value outside area of interest	The extension of source model is the same of ensemble
remove_zero_values	Remove values setted to zero and replace with missing value	No data values are setted correctly