Marine user guide

Release v2.0

Deutscher Wetterdienst (DWD)

Jun 28, 2023

CONTENTS:

CHAPTER

ONE

INTRODUCTION

This project contains the necessary code to produce the data summaries that are included in the Marine User Guide. These helps document the status of the marine in situ data in the CDS after every new data release. The marine data available in the CDS is the result of a series of data releases that are stored in the marine data file system in different directories. This project uses the data in the marine file system, rather than accessing the CDS data.

Additionally, the tools employed to create the individual source deck reports are also available in this project. These can be created for a single data release or for the combination of releases included in a Marine User Guide version.

This manual has two main independent sections dedicated to the Marine User Guide and to the individual source deck reports:

- *Marine User Guide*
- *Individual source-deck reports*

Every new data release can potentially be created with a different version of the marine processing software. The current version of this project is compatible with release v8 of the github marine processing repository (https://github.com/glamod/glamod-marine-processing/).

CHAPTER

TWO

TOOL SET-UP

2.1 Code set up

To clone the latest available version of the Marine User Guide repository:

```
git clone https://github.com/glamod/marine-user-guide.git
```

Build the python environment using the requirements.txt file in marine-user-guide/env. This step is system dependent. The following code block described the steps to followon ICHEC.

```
cd marine-user-guide/env
module load conda/2
conda create -prefix <MUG>/env/env1 python=3.8
source activate <MUG>/pyenvs/env1qqc
pip install -r requirements.txt
```

2.2 Paths setup

Some directory paths and handles are used throughout this document and are summarized in Table 1.

Edit file marine-user-guide/setpaths.sh and modify as needed the following fields:

- code_directory: parent path of the repository installation.
- data_directory: parent path to the data release directories.
- mug_code_directory: marine user guide code directory installation.
- mug_data_directory: marine user guide data directory path.
- mug_config_directory: directory holding the manifold configuration files needed for the marine user guide.

Table 1: Some directory paths and handles used throughout the document

Shorthand	Description	Example
<MUG>	Marine User Guide home directory	/ichec/work/glamod/marine-user-guide.2022/
<MUG_data>	Marine User Guide data directory	/ichec/work/glamod/data/marine/marine-user-guide.2022/
<log_dir>	Directory for log files	<MUG_data>/<version>/level2/log/
<MUG_list>	ascii file of sid-dck partitions to process	<MUG>/config/<release>/mug_list_full.txt
<version>	Tag of MUG version	v456.0
<release>	Tag of data release(s)	release_456
<sid-dck>	Tag combining of sourceID and deck	114-992

CHAPTER

THREE

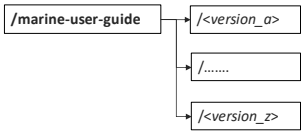
MARINE USER GUIDE

Every C3S Marine User Guide version includes a series of figures that describe the marine in situ data holdings in the CDS. The following sections explain how these figures are created for every new version of the Marine User Guide.

3.1 Initializing a new user guide

The data the tools in this project use and the products created are stored in the marine-user-guide data directory. This directory does not contain the actual data files, but links to the files in the data releases directories. This approach greatly simplifies the configuration of the different scripts and is followed even if a given Marine User Guide version is made up of a single data release.

The marine-user-guide data directory is then split in directories to host subsequent versions of the Marine User Guide.



This general directory needs to be created before starting using the tool.

```
mkdir<MUG_data>
```

Every new version of the Marine User Guide (MUG) needs to be initialized in the tools data directory as shown in the figure.

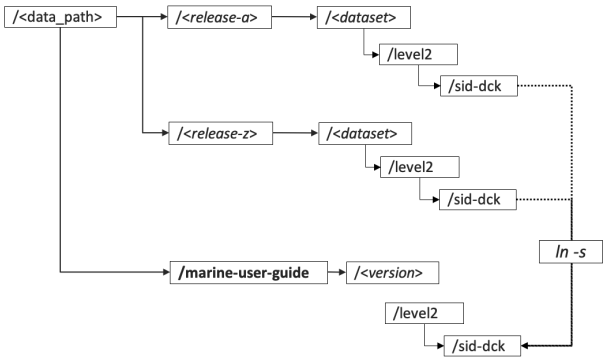


Fig. 1: Marine User Guide data directory and its relation to the individual data releases directories.

These steps initialize a new version:

1. Create the data configuration file (*mug_file*, *Marine user guide data configuration file*) by merging the level2 configuration files of the different data releases included in the new version (*level2 file format*).

```
source <MUG>/setpaths.sh
source <MUG>/setenv.sh
python <MUG>/init_version/init_config.py
```

See Table 1 for the meaning of the <*shorthands*>.

This step creates a *mug_config* file and a *mug_list* file which are used in the following.

An example of this step is a sfollows:

```
$ python init_version/init_config.py
Input name of release (no path): release_5.1
Input name of dataset (no path): ICOADS_R3.0.2T
Input filename with path of level2 configuration file: /ichec/work/glamod/glamod-
marine-processing.2022/obs-suite/configuration_files/release_5.1-000000/ICOADS_R3.0.
↪2T/level2.json
```

2. Create the directory tree for the version in the marine-user-guide data directory.

```
source <MUG>/setpaths.sh
source <MUG>/setenv.sh
python <MUG>/init_version/create_version_dir_tree.py <MUG_data> <version>␣
↪mug_config
```

Note that the first two lines (*setpath.sh/setenv.sh*) do not need to be repeated if these steps are performed in one session. For completeness we will repeat them every time here.

3. Populate it with a view of the merged data releases: rather than copying all the files, this is done by linking the corresponding files from the releases directories to the marine-user-guide data directory. Data linked is the level2 data files and level1a and level1c quicklook json files.

A bash script links each data partition and logs to <log_dir>/*sid-dck*/merge_release_data.*ext*, with *ext* being *ok* or *failed* depending on job termination status.

```
./init_version/merge_release_data.slurm <version> mug_config mug_list
```

where:

- mug_config: path to *mug_config* file as created in step 1.
- mug_list: path to *mug_list* file as created in step 1.

4. Check that the copies really reflect the merge of the releases. Edit the following script to add the corresponding paths and run. If any does not match, it will prompt an error.

```
./init_version/merge_release_data_check.sh
```

3.2 Data summaries

The data summaries are monthly aggregations over all the source-deck ID partitions in the data. These aggregations are on the data counts and observation values and on some relevant quality indicators and are the basis to then create the time series plots and maps included in the MUG.

3.2.1 Monthly grids

Aggregations in monthly lat-lon grids. The CDM table determines what aggregations are applied:

- header table: number of reports per grid cell per month.
- observations tables: number of observations and mean observed_value per grid cell per month.

Each aggregation is stored in an individual netcdf file.

3.2.2 Monthly time series of selected quality indicators

Monthly time series of quality indicators value counts aggregated over all the source-deck partitions. These are additionally, split in counts by main platform types (ships and buoys) and include the total number of reports. They are stored in ascii pipe separated files and are based exclusively on the CDM header table quality indicators.

3.2.3 Running the Code: Data Summaries

Grid and time series aggregations are performed by *monthly_grids.py* and *monthly_qi.py*, respectively. However, to support speed and ease, both of those scripts are configured and launched (in parallel) by *monthly_agg_slurm.py*. They use the common configuration file *monthly_grids.json* (*Monthly grids*). The launcher script configures and queues a single SLURM job in the log directory (*/level2/log/*), named *monthly.slurm* which executes each line of the *monthly.tasks* file in the same directory individually. Depending on the job termination status, each aggregation creates an empty <*aggregation_name*>*.success* or <*aggregation_name*>*.failure* file in the log directory.

The current configuration for the MUG excludes reports not passing all the quality checks. The same tool can be used to produce data summaries with different filter criteria, but modifying the filter values in the configuration file.

```
source <MUG>/setpaths.sh
source <MUG>/setenv.sh
cd <MUG>/data_summaries/
python monthly_agg_slurm.py <verion> ../config/<release>/data_summaries/monthly_grids.
↪json
```

See Table 1 for the meaning of the <*shorthands*>.

3.3 Figures

The data summaries generated are used to create the maps and time series plots included in the Marine User Guide. The following sections give the necessary directives to create them, with references to the configuration files used.

3.3.1 Number of reports time series plot

- Data summary used: *Monthly time series of selected quality indicators* (report_quality counts file: total number of reports field only)
- Configuration file: nreports_ts_plot.json

3.3.2 Duplicate status time series plot

- Data summary used: *Monthly time series of selected quality indicators* (duplicate_status file)
- Configuration file: duplicate_status_ts_plot.json

3.3.3 Report quality time series plot

- Data summary used: *Monthly time series of selected quality indicators* (report_quality file)
- Configuration file: report_quality_ts_plot.json

3.3.4 Number of reports Hovmöller plots

- Data summary used: *Monthly grids* (report counts files: header and observation tables)
- Configuration file: nreports_hovmoller_plot.json

3.3.5 ECV coverage time series plot grid

- Data summary used: *Monthly grids* (report counts files: header and observation tables)
- Configuration file: ecv_coverage_ts_plot_grid.json

3.3.6 Number of reports and number of months maps

- Data summary used: *Monthly grids* (report counts files: header and observation tables)
- Configuration file: nreports_and_nmonths_maps.json

3.3.7 Mean observed value maps

- Data summary used: *Monthly grids* (mean files: observation tables)
- Configuration file: mean_observed_value_maps.json

3.3.8 Running the Code: Figures

The above figures can be created individually or at once with the bash script *<MUG>/figures/plot_all.sh*. For the syntax to run individual plotting scripts we recommend to look into *plot_all.sh*. Each figure requires its own configuration file, located in *<MUG_config>/<release>/figures/* which might need some edits with new versions of the MUG. Where *<MUG_config>* is defined in *<MUG>/setpath.sh*.

Command:

```
source <MUG>/setpaths.sh
source <MUG>/setenv.sh
cd <MUG>/figures/
./plot_all.sh <version> <release> <options>
```

where <options> can be 'grid' or 'ts' to specify to plot only gridded properties ('grid') or only time series type plots ('ts'). If not specified all figures are created.

The bash script executes all plotting scripts in parallel on the login node. We consider this light post-processing which is permitted on login nodes, however, if more data is added it could become necessary to move this to a production node (see individual SID-DCK chapter/code).

Log files are written to the log directory (*<log_dir>*) and are named in accordance with the scripts. Figures are saved to *<MUG_data>/<version>/level2/reports/*. There are no *.success/.failure* files in this case because the presence/absence of figures is already a good indicator of the exit code.

CHAPTER

FOUR

INDIVIDUAL SOURCE-DECK REPORTS

See *<MUG>/config/<version>/data_summaries_sd/* and *<MUG>/config/<version>/figures_sd/* for configuration file and options.

4.1 Reports on a release merge

To create the individual source-deck reports on a merge of data releases, steps in *Initializing a new user guide* first need to be followed, so that the input data and required directory structure is ready in the marine-user-guide data directory.

4.2 Data summaries

The data summaries are monthly aggregations of report counts, observation values and additional CDM fields of the individual source-decks table files.

4.2.1 Monthly grids

Aggregations in monthly lat-lon grids. The CDM table determines what aggregations are applied:

- header table: number of reports per grid cell per month.
- observations tables: number of observations and mean, max and min observed_value per grid cell per month.

Each aggregation is stored in an individual netcdf file.

All the aggregations are configured in a common configuration file. There are currently two configurations that need to be run to create the data summaries needed: one using the full dataset and another one using the optimal dataset (all quality control checks passed).

4.2.2 Monthly time series of selected quality indicators

Monthly time series of quality indicators value counts for every *sid-dck* data partition. These are additionally, split in counts by main platform types (ships and buoys) and include the total number of reports. They are stored in ascii pipe separated files.

4.2.3 Monthly time series with source to C3S IO flow

Collection of monthly time series that describe the main report IO flow, from the initial reports in the source dataset to those finally delivered to C3S for every *sid-dck* data partition.

4.2.4 Running the code: SID-DCK Data Summaries

Grid and time series aggregations are performed by *monthly_grids_sd.py*, *monthly_qi_sd.py* and *report_io_sd.py*, respectively. However, to support speed and ease, all these scripts are configured and launched (in parallel) by *monthly_agg_slurm_sd.py*. This script configures and queues a single SLURM job in the main log directory (<*log_dir*>), named *monthly_sd_all.slurm* which executes each line of the *monthly_sd_all.tasks* file in the same directory individually (the same structure is used for*_optimal* setup as for *_all*).

Two configuration files provide all the necessary input to those scripts and differ only in the data which is used (*report_io_sd.py* is only executed if all data is used). Aggregations including all data are produced with the configuration file called *monthly_grids_sd_all.json* while *monthly_grids_sd_optimal.json* creates aggregations based on data which passed all quality control checks.

Depending on the job termination status, each aggregation creates an empty <*aggregation_name*>*.success* or <*aggregation_name*>*.failure* file in the SID_DCK log directoy. Note that .slurm and .tasks files are located in <*log_dir*>*/*, while aggregated data files, .log as well as .success/.failure files are in the respective <*log_dir*>*/*<*sid-dck*>*/* directories. In this way decks with very different size/requirements can be handled efficiently.

```
source <MUG>/setpaths.sh
source <MUG>/setenv.sh
cd <MUG>/data_summaries_sd/
python monthly_agg_slurm_sd.py <version> ../config/<release>/data_summaries_sd/
↪monthly_grids_sd_all.json mug_list
python monthly_agg_slurm_sd.py <version> ../config/<release>/data_summaries_sd/
↪monthly_grids_sd_optimal.json mug_list
```

where *mug_list* is an ascii file with a list of the *sid-dck* partitions to process, e.g. <*MUG*>*/config/*<*release*>*/mug_list_full.txt*. See Table 1 for the meaning of the <*shorthands*>.

4.3 Figures

4.3.1 ECV reports time series plots

- Data summary used: *Monthly grids* (counts, header and observation tables)
- Configuration files:
  - ecv_reports_ts_plot_grid_sd-all.json
  - ecv_reports_ts_plot_grid_sd-optimal.json
- Plotting script: *ecv_reports_ts_plot_grid_sd.py*

4.3.2 Observed parameters latitudinal time series

- Data summary used: monthly grids (counts, min, max, counts from observation tables). All data and optimal dataset summaries.
- Configuration file: *param_lat_bands_ts.json*
- Plotting script: *param_lat_bands_ts.py*

4.3.3 Duplicate status time series plot

- Data summary used: duplicate_status quality indicators time series.
- Configuration file: *nreports_dup_ts_sd.json*
- Plotting script: *nreports_dup_ts_sd.py*

4.3.4 Report quality time series plot

- Data summary used: report_quality quality indicators time series.
- Configuration file: *nreports_qc_ts_sd.json*
- Plotting script: *nreports_qc_ts_sd.py*

4.3.5 Monthly time series with source to C3S IO flow

- Data summary used: monthly time series with IO flow
- Configuration file: *report_io_plot_sd.json*
- Plotting script: *report_io_plot_sd.py*

4.3.6 Running the Code: SID-DCK Figures

Figures of individual decks are created by the plotting routines stated above, all located in the directory *<MUG>/figures_sd/*. Each of those routines has a configuration file in *<MUG>/config/<release>/figures_sd/*, named accordingly which require adjustments for a new release.

To execute said routines, the script *plot_all_slurm_sd.py* is creating a bash script at *<log_dir>/plot_sd.slurm* and submitts it to the SLURM scheduler. When server time is allocated, each line of the file *<log_dir>/plot_sd.tasks* is executed individually. Log files are created for each plotting routine and deck and can be found at *<log_dir>/<sid-dck>/<routine_name>.log* and upon successful termination figures are saved to *<MUG_data>/<version>/level2/reports/<sid-dck>/*.

Command:

```
source <MUG>/setpaths.sh
source <MUG>/setenv.sh
cd <MUG>/figures_sd/
python plot_all_slurm_sd.py <version> <release> mug_list
```

where *mug_list* is an ascii file with a list of the *sid-dck* partitions to process, e.g. *<MUG>/config/<release>/mug_list_full.txt*

CHAPTER

FIVE

APPENDIX 1. MARINE USER GUIDE CONFIGURATION FILES

The configuration files needed to run this project are maintained in the glamod github repository (https://github.com/glamod/marine-user-guide/config/). Every Marine User Guide version has a dedicated directory within this repository which are further subdivided by data summaries and figures, both for the whole dataset and for individual source-deck combination (*_sd*). The Marine User Guide v8 has been created by v8 of the github Marine User Guide repository.