Fidelity and uncertainty in climate data records from Earth Observations

FIDUCEO WP5.1

MFG/MVIRI SURFACE AND AEROSOL CDR

MVIRI GEDAP-CISAR OPERATION MANUAL

Prepared by : Y. Govaerts, E. Pinat, M. Luffarelli and V. Leroy (Rayference)
Reference : FIDUCEO-GEDAP-OM
Version : MVIRI-BETA
Date : 12/11/2018

Document Change Record

Doc. Version	Date	CISAR Release	Remarks
Version 0.0	November 2017	-	First released version
Version 1.0	October 2018	-	Science release
Version 1.1	November 2018	-	gedap3 release

Table of Contents

List of Acronyms

AOT	Aerosol Optical Thickness
ATBD	Algorithm Theoretical Basis Document
BHR	BiHemispherical Reflectance
BRF	Bidirectional Reflectance Factor
CDR	Climate Data Record
CFC	Cloud Fractional Cover
CISAR	Combined Inversion of Surface and AeRosol
EQMPN	Equivalent Model Parameter Noise
FCDR	Fundamental Climate Data Record
FIDUCEO	Fidelity and uncertainty in climate data records from Earth Observations
GEDAP	GEneric DAta Processing Chain
IFOV	Instantaneous Field Of View
LUT	Look-Up Table
MA	Monochromatic Assumption
MFG	Meteosat First Generation
MVIRI	Meteosat Visible and Infrared Imager
OE	Optimal Estimation
RPV	Rahman - Pinty - Verstraete BRF model
RTM	Radiation Transfer Model
SAF	Satellite Application Facility
SSP	Sub Satellite Point !!
SSR	Sensor Spectral Response !!
TBD	To BE Defined
TOA	Top Of Atmosphere
ZDM	Zero Degree Mission

1 Introduction

1.1 Purpose

The document describes the functionalities and data flow of a Generic Data Processing Chain (GEDAP) used to run the Combined Inversion of Surface and AeRosol (CISAR) algorithm.

The purpose of this first release is to perform the integration of this algorithm in EUMETSAT reprocessing environment, scheduler and distributed computing.

1.2 Scope

GEDAP primary objective is to feed the CISAR with the data needed to perform the inversion of a physically-based Radiative Transfer Model (RTM) and to generate aerosol products based on these retrievals in standard format.

1.3 Overview

GEDAP manages all the I/O operations for the CISAR algorithm. MVIRI images are first decomposed into **Tiles**. A **Tile** is defined by a geographical coverage, spatial resolution and geographical projection. It represents the basic geographical coverage on which the inversion is performed. CISAR requires multi-angular observations in order to perform the inversion. MVIRI data are therefore accumulated in time during a 5-day period, called the **Accumulation Period**. For each **Tile** to be processed, GEDAP prepares the inputs required by the CISAR algorithm for the requested accumulation period. GEDAP also handles the output of the inversion to provide the so called **Solution Tiles** needed to build the final aerosol hourly products.

2 Definitions

Throughout this text, the following formalism is used:

1. **typewriter** type is used for the concepts specifically defined in this Operation Manual;
2. typewriter type is used to indicate commands or instructions passed to GEDAP;
3. `typewriter` type is used to indicate terminal inputs or outputs;
4. typewriter type indicates file or directory names;
5. typewriter type indicates GEDAP configuration file keywords;

This section contains the definition of the main concepts used in this document.

Accumulation Period It consists of a 5-days time interval during which MVIRI observations are accumulated. This allows to generate a multi-angular observation vector from a radiometer that acquires only one viewing direction at a time.

Cold Start First run of GEDAP to process a **Main Task**.

Main Task GEDAP top-level activity defined by:

- the Platform data to be processed, *e.g.*MET7;
- a nominal sub-satellite point longitude;
- the time range to be processed (start and end date);
- the paths to access the data;
- the version of the Easy FCDR file to process;
- the list of **Tiles** to process;

The values of these parameters are defined by a set of keywords stored in a GEDAP json configuration file named common_init.json.

Model Parameters Variables of the radiative transfer model not retrieved during the inversion process.

Prior Information Prior values and uncertainties of the state variable magnitude and temporal behaviour.

Processing Framework Three-layer system used for the processing of MVIRI data with GEDAP.

State Variables Variables of the RTM retrieved during the inversion process.

Tile Tiles are defined as static units of temporal invariant geographical 2D properties (longitude,latitude) forming a geographical grid and are used to store data. Each **Tile** has a unique identification number composed of four digits. The location of the MVIRI **Tiles** for the Zero Degree Mission (ZDM) is shown on Figure 2. Four different types of **Tiles** are used:

- **Static Tile** : contains immutable data such as latitude, longitude, land/sea mask and pixel elevation. All the Static Tiles for the MVIRI disk are provided together with the release.

- **Input Tile** : contains the satellite data, **Model parameters** and cloud mask;
- **Prior Tile** : contains the prior values of the surface and aerosol parameters;
- **Solution Tile** : contains the values of the surface and aerosol parameters retrieved by the CISAR algorithm;

All these **Tiles** consist in netCDF files.

Stacked Tiles List of **Input Tiles** included in an accumulation period.

Sub-Tasks The series of tasks that forms a GEDAP **Main Task**.

Warm Start GEDAP starts running again after an interruption subsequent to a **Cold Start**.

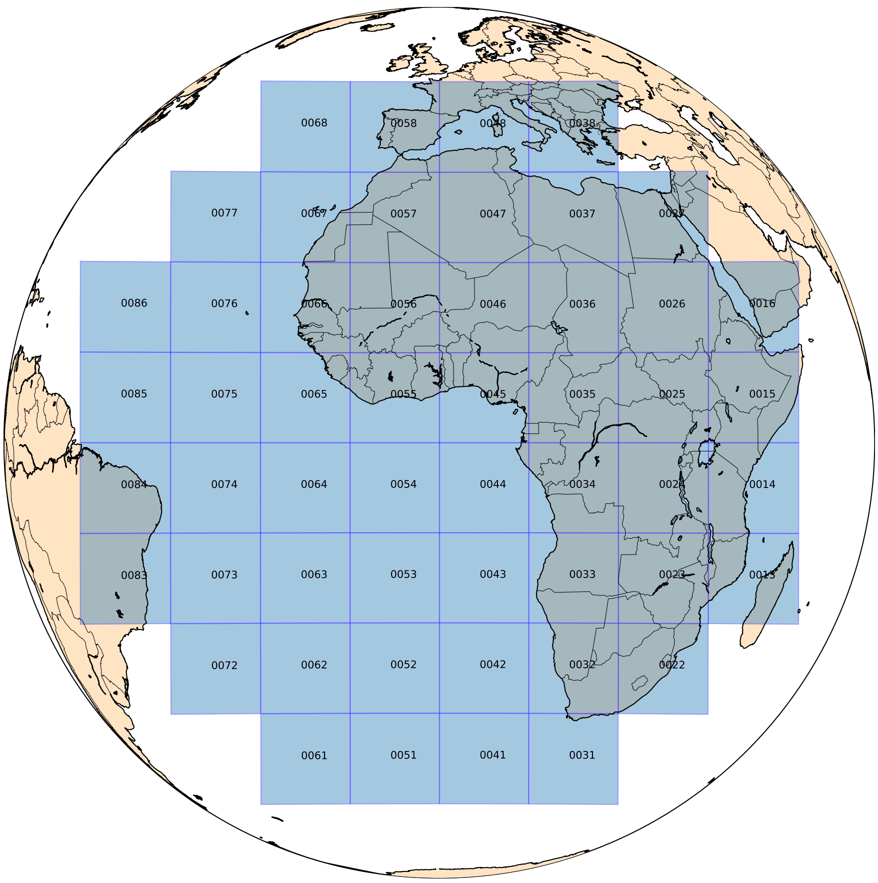

Figure 1: Location of the MVIRI Tiles for the ZDM. The tile id is the same as the STATIC Tiles one. Tiles that have borders outside the Earth disk are not displayed.

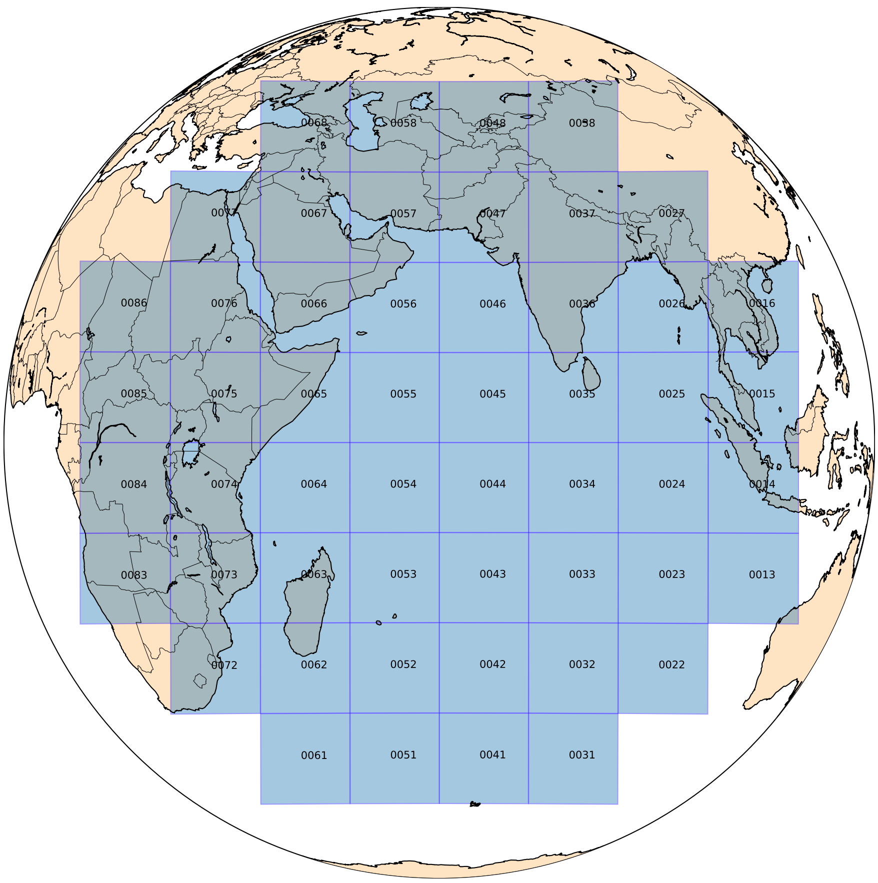

Figure 2: Location of the MVIRI Tiles for the IODC. The tile id is the same as the STATIC Tiles one. Tiles that have borders outside the Earth disk are not displayed.

3 Overall architecture

The overall architecture of the **Processing Framework** is divided into three layers:

1. SCHEDULER The scheduler is responsible for the management of the **Main Task**. It sends commands to GEDAP to perform all the **Sub-Tasks** that form the **Main Task**. These commands are:

- build_list_accumulation_periods: the period to be processed in the **Main Task** is divided into 5-day **Accumulation Periods**, each of which is identified by an index number. These accumulation periods are then written in a list_accumulation_periods.json file. This task is performed only if a **Cold Start** is requested or if the aforementioned file is missing, otherwise the task load_list_accumulation_periods_from_json_file is performed .
- prepare_to_process_first_acc_period: every time GEDAP starts running it is important to identify the index of the first non processed period. In case of a cold start the index of the first period to be processed will always be 0 because the list_accumulation_periods.json is recreated. On the other hand, when a warm start is required, it is possible that periods were already processed, and the index of the *first not processed period* can vary. While this task is performed, the **Static Tiles** and the **Prior Tiles** are created for this *first not processed period* p_0. **Prior Tiles** are populated with default values.
- make_input_tiles: this command triggers the preparation of all the **Input Tiles** for one accumulation period. All these tiles form the **Stacked Tiles** list associated to this accumulation period;
- process_tiles: this is the task where the CISAR inversion takes place. To process one **Tile** few informations are required:
  - a four digits tile identification number t_{ID};
  - the start and end date of the **Accumulation Period** p;
  - the **Static Tiles** and **Prior Tiles** related to the **Accumulation Period** p;

  At the end of the task, **Solution Tiles** storing the aerosol information are created.
- generate_products: this command triggers the generation of the MVIRI CDR product(s) for period p from all the **Solution Tiles**. An example of hourly aerosol product class is given with the release.
- update_accumulation_periods_list: this command triggers the update of the list_accumulation_periods.json file to indicate that period p has been successfully processed.

The reception of a command by GEDAP results in the execution the associated **Sub-Task**. An example that allows the execution of a GEDAP **Main Task** on a single CPU is provided in this release.

2. GEDAP This layer receives commands from the **SCHEDULER** to handle the processing of all the active **Tiles**. The index t It performs all I/O operations to generate the various **Tile** files.

3. CISAR This layer performs the actual inversion of the **Stacked Tiles** on a CPU. All allocated cores of that CPU will be used to perform the inversion. It is not recommended that this allocation exceeds the number of cores.

4 GEDAP Processing Framework overview

4.1 Outline of the retrieval procedure

CISAR algorithm primary objective is to perform the inversion of a forward RTM to derive the **State Variables** describing the observed system. In order to derive these variables and associated uncertainties, the retrieval algorithm requests multiple information:

Satellite Observations: these are TOA BRF and associated ancillary data, *e.g.*acquisition time, geometry and uncertainties. They form the multi-dimensional observation vector containing space, angular and spectral information.

Prior Information: this is the prior knowledge on the retrieved **State Variables** and associated uncertainties.

Model Parameters: belong to the RTM but are not retrieved by the retrieval algorithm.

Static and Dynamic Masks: including cloud mask, land/sea mask, . . .

In addition, CISAR needs also the following:

- Surface type.
- Sun Zenith Angle (SZA), View Zenith Angle (VZA) and Relative Azimuthal Angle (RAA) (rad).
- Acquisition times of each observations (*Modified Julian Date*, MJD).
- Surface Pressure (hPA).
- Total column water vapour concentration (kg/m^2).
- Total column ozone concentration (*Dobson Unit*, DU).
- Aerosol layer height (km).
- Wind speed (m/s).
- Wind direction (°).
- TOA BRF.
- TOA BRF random uncertainty.
- Surface prior values (RPV model).
- Uncertainties associated with surface prior.
- AOT prior value(s).
- Uncertainty(ies) associated with AOT prior.

Running a GEDAP **Main Task** includes the following major **Sub-Tasks**:

1. **Intialization** During this **Sub-Task**, GEDAP performs the following steps:

   (a) loads file common_init.json describing the **Main Task** to be performed and verifies its consistency;

   (b) initialises the log file gedap.log (only in case of **Cold Start**);

   (c) generates the list_accumulation_periods.json file containing the list of **Accumulation Period** to be processed and their processing status (only in case of **Cold Start**);

   (d) creates **Input Tile** and **Static Tile** files associated to the GEDAP **Main Task** (in case of **Cold Start**, if any of these tiles are already present, these tiles are recreated);

   (e) generates of the **Prior Tiles** files with default **Prior Information** for period p_0. The scheduler sends the prepare_to_process_first_acc_period command to GEDAP to perform this **Sub-Task**;

2. Loop over the not processed **Accumulation Periods** p:

   (a) **Input Tiles Preparation**: accumulates the observations and all ancillary information during the 5-day accumulation period, creates the **Input Tiles** files and therefore the **Stacked Tiles** list. The scheduler sends the make_input_tiles command to GEDAP to perform this **Sub-Task**.

   (b) **Stacked Tiles Processing**: loops over the not processed active **Tiles** t. The index t is related to the number of tiles the user decides to process, and can vary from a minimum of one tile to 1728, which is the maximum number of tiles that form the MVIRI grid (Figure 2). The scheduler sends the process_tiles command to GEDAP to perform the processing of one **Tile** t for the **Accumulation Period** p, and this command can be divided into:

      i. inversion of **Stacked Tiles** for index t for **Accumulation Period** p by CISAR;
      ii. generation of the **Solution Tile** for index t for period p;
      iii. generation of the **Prior Tile** for index t for period $p + 1$;
      iv. deletion of the **Input Tiles** for index t for period p (currently not implemented but highly recommended to prevent disk space increase);
      v. deletion of the **Prior Tiles** for index t for period $p - 1$ (currently not implemented but highly recommended to prevent disk space increase).

   (c) **CDR Product Generation**: starting from the **Solution Tiles**, generates the CDR product file(s) for period p.

   (d) Delete the **Solution Tiles** for period p (currently not implemented but highly recommended to prevent disk space increase).

   (e) The list_accumulation_periods.json file is updated to indicate that **Accumulation Period** p has been successfully processed. To do so, the Scheduler sends the update_accumulation_periods_list command to GEDAP.

3. End of GEDAP **Main Task**;

4.2 Data files

The GEDAP **Processing Framework** relies on the following categories of data files:

External data These files are the Easy FCDR image and static files, cloud mask data, aerosol layer height and types climatology and ECMWF data.

Internal data These files are handled by GEDAP.

Product data These data are the MVIRI CDR file generated by GEDAP.

4.2.1 External Data

GEDAP needs to read data from several sources for the preparation of the **Stacked Tiles** files.

4.2.2 MVIRI image and static Easy FCDR

Image MVIRI files should follow EXACTLY this convention:
FIDUCEO_FCDR_L15_MVIRI_METs-rr.r_yyy1m1d1h1m1_yyy2m2d2h2m2_EASY_va.a_fvb.b.nc
where:

- *s* is the MFG satellite number from 2 to 7. This release only accepts *s* = 7.
- *rr.r* is the MFG satellite nominal sub-satellite longitude. This release only accepts *rr.r* = 0.
- *yyy1m1d1h1m1* is the year, month, day, hour and minute of the begining time of the nominal acqusition time.
- *yyy2m2d2h2m2* is the year, month, day, hour and minute of the end time of the nominal acqusition time.
- *va.a_fvb.b* is the MVIRI Easy FCDR image version.

The location of these files is determined in the common_init.json file. It is assumed that these files are organised in the following directory structure: easyfcdrdirnam/yyyy/mm

4.2.3 Model Parameters (ECMWF data)

The **Model Parameters** considered in GEDAP are:

- surface pressure
- total column ozone
- total column water vapour
- wind speed
- wind direction
- aerosol layer height

All these data are taken from ERA-ITERIM files. Since October 2017, daily ERA-INTERIM data can be downloaded only on a per month basis. Therefore, the structure of the "modparams" directory has to be the following: modparamsname/yyyy/mm. The ERA-INTERIM file contains a time dimension that covers, for every day of the month, the 6 hours frequence, i.e. there is a lat/lon frame for 00:00, 06:00, 12:00 and 18:00 hours. These data are interpolated in time to match the time of the MVIRI acquisition. In addition, a spatial interpolation (mean weighted over the distance) is performed over 4 ERA-INTERIM pixels to find the corresponding values of the MVIRI pixels.

4.2.4 Cloud Mask

The cloud mask file associated with MVIRI observations is expected to have EXACTLY this type of file name convention:

prefix W_XX-EUMETSAT-Darmstadt,SING+LEV+SAT,MET0s_CLM_C_EUMS
variable part _yyyymmddhhmmss
suffix _1_OR_FES_E0000_0100.nc

The location of these files is determined in the common_init.json file. It is assumed that these files are organised in the following directory structure: clmdirnam/yyyy/mm

4.2.5 GEDAP internal data

To store data associated with **Tiles** (latitude, longitude, land sea mask, TOA BRF, radiometric noise, **Model Parameters**, cloud mask, prior values, retrieved data), GEDAP uses netCDF files (in netCDF-4 format). GEDAP uses 4 different kinds of **Tiles** files:

STATIC-TILE Names of corresponding netCDF files follow the pattern:
STATIC_TILE_MFG_MVIRI_METs_HIGH_RES_rrrrr_AREA_tttt.nc

INPUT-TILE Names of corresponding netCDF files follow the pattern:
INPUT_TILE_MFG_MVIRI_METs_HIGH_RES_rrrrr_DATETIME_yyyy-mm-dd_hh-mm_AREA_tttt.nc

PRIOR-TILE Names of corresponding netCDF files follow the pattern:
PRIOR_TILE_MFG_MVIRI_METs_HIGH_RES_rrrrr_DATETIME_yyyy-mm-dd_hh-mm_AREA_tttt.nc

SOLUTION-TILE Names of corresponding netCDF files follow the pattern:
SOLUTION_TILE_MFG_MVIRI_METs_HIGH_RES_rrrrr_DATETIME_yyyy-mm-dd_hh-mm_AREA_tttt.nc

where

- *s* is the MFG satellite number;
- *HIGH_RES* is the MVIRIprojection grid identification. Currently, only the processing at the full pixel VIS resolution is foreseen on the MFG standard grid;
- *rrrr* is the nominal sub-satellite longitude in tenth of degrees from 0 to 3600;

- *yyyy-mm-dd_hh-mm* is the beginning of the **Accumulation Period** for the **Prior Tiles** and **Solution Tiles** and image acquisition time for the **Input Tiles**;
- *tttt* is the four digit **Tile** identification number t_{ID}.

The location of these files is determined in the common_init.json file.

In addition to the four **Tile** file types, two additional internal files are used:

Accumulation period file The file list_accumulation_periods.json contains the list of the accumulation periods to be processed with the corresponding status: processed or not processed. This file is generated by the build_list_accumulation_periods command and updated by the update_accumulation_periods_list command.

CISAR LUT CISAR requires information about the gaseous transmittance and aerosol single scattering properties of the processed radiometer. This file is part of the release.

4.2.6 Product data

A simple example of product generation is provided with this release. A basic class to read and write Fiduceo products is provided with the file product_fiduceo.py, and the actual function that will use this class and generate the products is given in generate_products_fiduceo.py. Here the loop over the processed tiles is performed, and an aerosol hourly product is build. Only few basic quantities, as latitude, longitude, time, AOT and its uncertainty and BHR are stored in the products.

4.2.7 CDR

The final CDR will be written using Tom Block's netCDF writing tool. Metadata have been stored in the solution tile, and will have to be passed to the product. These metadata include the list of MVIRI images of the day, the list of cloud masks, the ECMWF data file used, the LUT and all the path of all the ancillary data used to produce the Solution Tile.

5 Running GEDAP

5.1 Pre-requisites

Before beginning to process a GEDAP **Main Task** within the **Processing Framework**, the following check list needs to be performed (It is highly recommended that you prepare a script to automatically perform these sanity checks.)

1. All the Static **Tile** files of the active **Tiles** are present in the specified **Tile** directory.
2. The CISAR LUT file FAST_LUT_MET7_MVIRI.Q08 is present in the specified LUT directory
3. GEDAP and CISAR have been successfully installed according to the installation instruction;
4. The User Limits have been correctly set up with the `ulimit -s 10000` command;
5. All the requested environment variables have been defined, ie:

   GEDAP_DIR : should point to the root of the GEDAP directory;

   OMP_NUM_THREADS : Should be set to the number of CPU CORES on which CISAR will run (not the number of threads);

   OMP_STACKSIZE should be set to 64M (OMP_STACKSIZE=64M);
6. Verify that there is enough space in the **Tile** directory;
7. There is enough coffee;

5.2 GEDAP starting modes

GEDAP has two starting modes, *i.e.*cold and warm, that are described in the following sections. Both modes rely on the same common_init.json file located in the $GEDAP_DIR/resources directory.

5.2.1 Configuration file

The GEDAP configuration file is written json language. This file is named common_init.json and is located in the $GEDAP_DIR/resources. All the fields are mandatory, and define the GEDAP **Main Task** to be processed.

Keyword	field	Example of value	Description
platform		MET7	MFG Satellite ID.
SSP		0	Nominal sub-satellite longitude.
start_date		01-03-2005	Start date of the Main Task in format dd-mm-yyyy.
end_date		25-03-2005	End date of the Main Task in format dd-mm-yyyy.
dir_cisar_lut		(1)	Directory name with the location of the CISAR LUT.
dir_external_data	static	(1)	Directory containing the geopotential file.
dir_external_data	image	(1)	Directory name where the Easy FCDR MVIRI image files are stored. It is assumed that this directory is contains sub-directories /yyyy/mm/.
dir_external_data	mask	(1)	Directory name where the MVIRI cloud mask is stored. It is assumed that this directory is contains sub-directories /yyyy/mm/.
dir_external_data	aot	(1)	Directory name where the AOT prior information is stored.
filename_suffix	image	_EASY_v2.6_fv3.1	
filename_suffix	mask	00_1_OR_FES_E0000_0100.nc	
rootdir_tilemaker			Directory to store the netCDF Tile files. Within this directory, the following subdirectories are automatically created: Input_Tiles, Prior_Tiles, Solution_Tiles. Static Tiles are created inside the Input_Tiles directory.
list_of_tiles	tttt	A	Tile ID (t_{ID} from 0000 to 0097) and activation status (A=active, N=not active).

(1) Valid directory name

An example of input file is provided with the release. In case of missing keywords, a fatal exception is raised with the name of the missing keyword.

5.2.2 Cold start

A **Cold Start** is needed when a GEDAP **Main Task** is performed for the first time. Specifically, a **Cold Start** performs the following operations:

- if any, it recreates Input, Prior and Solution **Tiles** associated to that **Main Task**;
- deletes the gedap.log file;

- recreates the list_accumulation_periods.json file with the list of period to be processed. This file is located in the $GEDAP_DIR/resources directory.
- launches the MVIRI image processing for the specified time range.

5.2.3 Warm start

A **Warm Start** is needed when a GEDAP **Main Task** has been interrupted prior to its completion. Specifically, a **Warm Start** performs the following operations:

- identifies in the list_accumulation_periods.json file the index of the last successfully processed **Accumulation Period**.
- launches the MVIRI image processing from the **Accumulation Period** after the last successfully processed one. Existing Input **Tiles** not yet processed will not be re-created.

5.3 Launching the scheduler from the command line

A simple scheduler is provided with this release to run GEDAP on a single CPU. The entry point to the scheduler is the script `$GEDAP_DIR/SOURCE/PYTHON_SOURCE/gedap/scripts/nodist/main.py` . It can be directly executed using the Python interpreter, but we also added it as an entry point to the GEDAP Python package. In practice, this means that if GEDAP is installed in the Python environment using the provided setup.py script (see installation instructions), then the sample scheduler can run simply by issuing `gedap` in a terminal window:

```
> gedap --help
```

```
usage: gedap [-h] [-b] [-c] [-r RESOURCES_DIR] [-a] [-f] [-m] [-p] [-g] [-u]
             [-i PERIOD_INDEX] [-t TILE_INDEX] [--version]

GEDAP non-distributed application.

optional arguments:
  -h, --help            show this help message and exit
  -b, --batch           execute in batch mode (non-interactive)
  -c, --cold_start      in batch mode, require cold start
  -r RESOURCES_DIR, --resources_dir RESOURCES_DIR
                        path to resources directory to be used (defaults to
                        $GEDAP_DIR/resources)
  -a, --all             pass this argument to perform the non distributed
                        processing on one core
  -f, --process_first   pass this argument to perform only the
                        prepare_to_process_first_acc_period task
  -m, --make_input      pass this argument to perform only the
                        make_input_tiles task
  -p, --tile_processor  pass this argument to perform only the process_tiles
                        task
  -g, --product_generator
                        Pass this argument to perform only the
```

```
                        generate_products task
-u, --update_list       Pass this argument to update the list of accumulation
                        periods
-i PERIOD_INDEX, --period_index PERIOD_INDEX
                        Passing the index of the accumulation period to be
                        processed
-t TILE_INDEX, --tile_index TILE_INDEX
                        Passing the index of the tile to be processed
-n, --monitoring        Launch the monitoring script, to plot product
                        statistics timeseries.
--version               display version information and exit
```

All these options have been implemented to facilitate the parallelisation on EUMETSAT side with PBS.

The `gedap` options are :

- `b` In the interactive (default) mode, interactive safeguards help avoid accidental destructive operations (*e.g.*unwanted cold start). The optional batch mode suppresses these safeguards.
- `c` The cold start flag allows to force a cold start, regardless what is configured in the common_init.json configuration file.
- `h` Displays the `gedap` usage on the terminal.
- `r` The resources directory option allows to specify the resources directory to use (*e.g.*in case a user would want to choose between multiple configuration files).
- `a` The "–all" option performs all the steps of the processing.
- `f` This option to perform only the prepare_to_process_first_acc_period task.
- `m` This option to perform only the make_input_tiles task.
- `p` This option to perform only the tile_processor task.
- `g` Pass this to perform only the generate_products task.
- `u` To update the list of accumulation periods.
- `i` To pass the index of the period to be processed, required by the *p* option.
- `t` Allows to pass the tile index, and can be combined with any of the above options as for example in:

```
gedap -a -t="0002"
```

5.4 Error handling

During the execution of a GEDAP **Main Task**, a log file named gedap.log is generated in the working directory. This log file contains the following category of messages:

Info Informative message on the Main Task processing.

Warning Warning message concerning unexpected events that does not prevent GEDAP to run.

Error Error message concerning the unexpected events that could impact the results nut does not prevent the processing to continue.

Fatal Fatal exception generates the interruption of the Main Task processing. The following list of events can generate a fatal exception:

- the GEDAP configuration file has an erroneous content;
- the CISAR LUT is not correctly loaded. This event generates an exception with numbers between 1110 to 1120. The LUT path and file name should be verified;
- the Static Tiles cannot be found in the directory;
- the aerosol height climatology files cannot be found in the directory;
- the Prior Tile is missing or corrupted;
- the Solution Tile cannot be saved;
- `ulimit` is not set properly;
- when a single tile is to be processed and its Tile ID t_{ID} does not match any of the known tiles GEDAP will issue a ValueError "Empty static tile list." and stop.

6 Limitations and know bugs of the current release

6.1 Limitations

Sub-satellite point This release allows the processing of MET7 data only acquired at the ZDM sub-satellite point.

MVIRI CDR format The MVIRI CDR formats has not yet been determined. A simple Python script is provided to show how to generate such file from the solution tiles (`$GEDAP_DIR/SOURCE/PYTHON_SOURCE/gedap/scheduler/tasks/generate_products_fiduceo.py`).

Scheduler The example of Python scheduler provided allows the processing of the active tiles only on a single CPU (`$GEDAP_DIR/SOURCE/PYTHON_SOURCE/gedap/scripts/nodist/main.py`).

Aerosol class The aerosol class used in CISAR is the same as in GSA.

6.2 Warnings!

- Known issues are currently present in the cloud mask, and a new release from CLM is expected to fix those.
- At the moment only daily cloud masks are being read for the IODC configuration. If the per-slot product will be available at the moment of the IODC processing, few fixes will be required.

7 Scientific release 0.1.1: additional notes

This GEDAP version is meant to handle MVIRI images from version EASY_v2.6_fv3.1 and not previous file versions, as it will be explained here below.

- The longitude has been flipped with respect to the previous image (version EASY_v2.2_fv2.5). In this final version, both longitude and latitude follow the natural scanning order, i.e. the radiometer scans from East to West and from South to North. Figures 3 and 4.

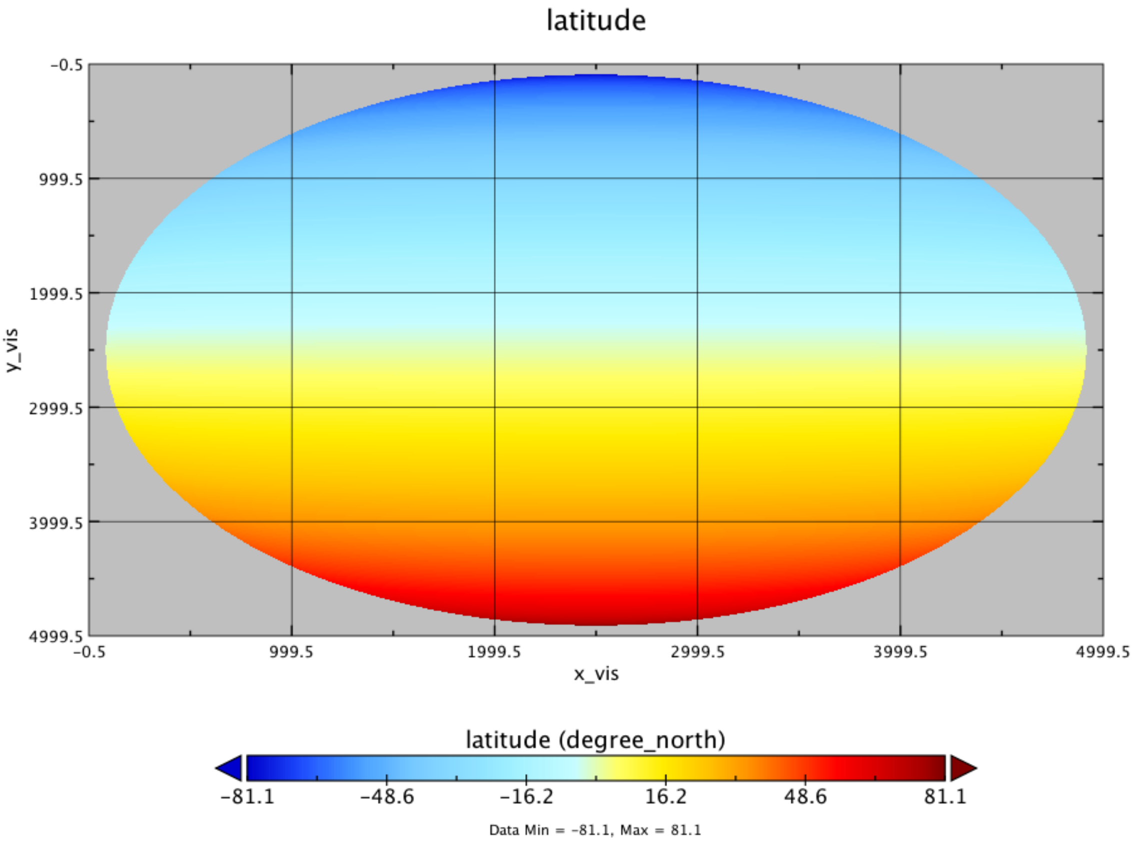

Figure 3: Latitude, natural MVIRI scanning order.

This requires a fix in all the ancillary data orientation and reading of the cloud mask, which is now aligned with the image.

- The new orientation of MVIRI images requires a new set of static tiles, which are numbered differently with respect to the previous list for version EASY_v2.2_fv2.5. The new tiles are shown in figure 2.

- Satellite and Sun angles are now stored in the MVIR EasyFCDR image, at a lower resolution of $x_{tie}, y_{tie} = (500 \times 500)$ (every 10^{th} pixel). A bilinear interpolation has been applied to retrieve these fields at the image native resolution, and no FIDUCEO static file is required any longer.

- An additional mask has been added next to the cloud mask, using the field “data_quality_bitmask". If a pixel is cloud free it will be considered for processing only if the value of this mask is zero, conditions that determines the good quality of the data.

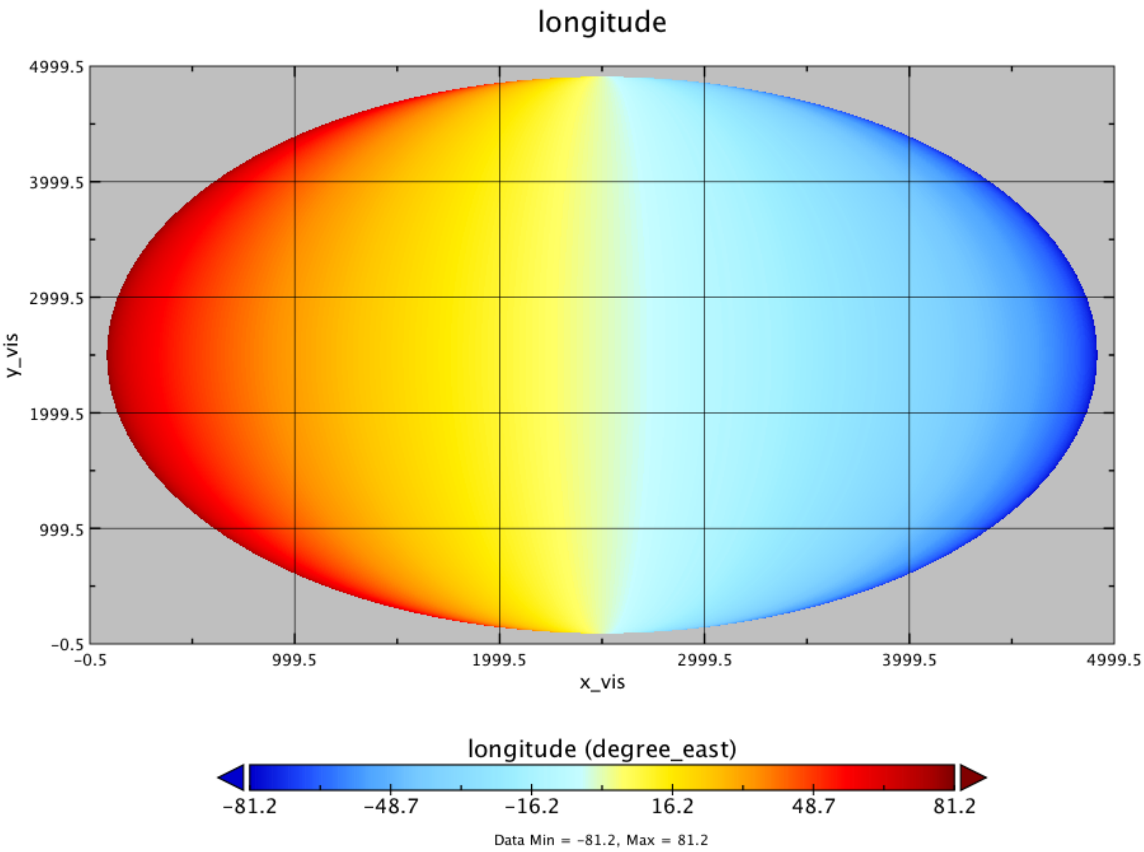

Figure 4: Longitude, natural MVIRI scanning order.

- A misalignment between between the static files (geopotential and land sea mask for example) has been noticed. The routines RefGeo and GeoRef have been taken from Frank's mviri_cruncher and converted to C++ to align the code. This was believed to be the cause of the border on the left of the Earth disk on Figure 6. This however has been discovered being due to a wrong handling of the cloud mask.

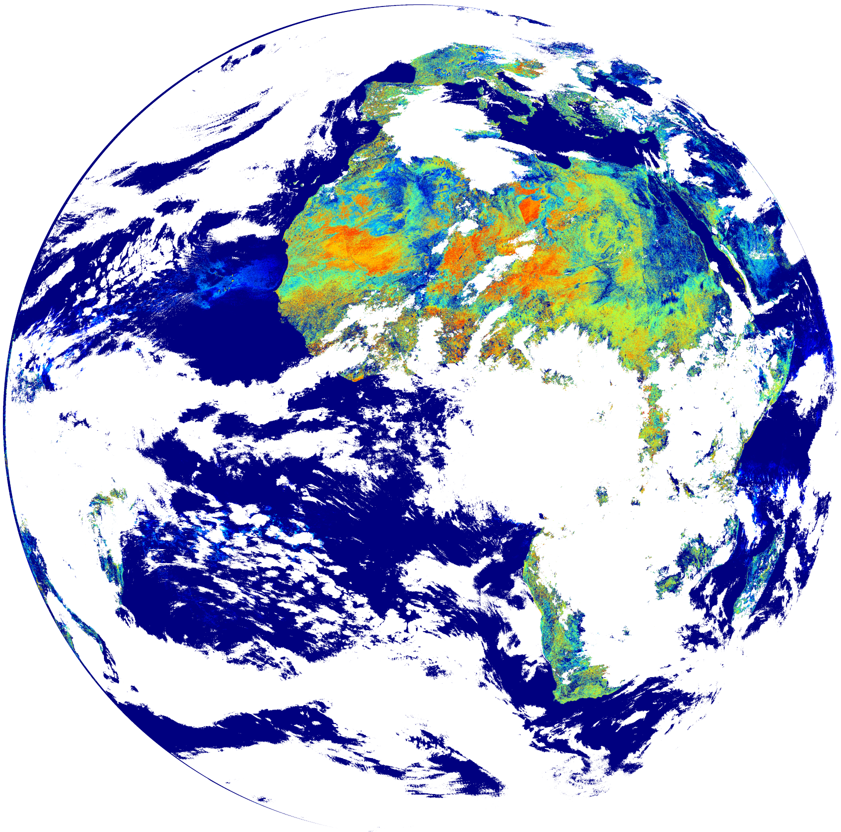

Figure 5: Old full disk product: notice the darker band at the extreme of the Earth disk, resulting from a misalignment.

- The IODC implementation that has been implemented requires at the moment daily cloud masks. If however at the moment of the actual processing the cloud mask will be per-slot, a patch is required (few lines of code, very simple, but required).

- In mid August 2017, when testing the full disk mean daily AOT product on April 4th 2005, the product showed a ladder shaped structure in South Africa that presented higher AOT values with respect to the neighbouring pixels. The shape was so geometrically perfect that the natural origin was immediately excluded. After quite some investigation it has been discovered that this ladder is due to the interpolation of viewing and Sun azimuth angles. These angles are provided in the EasyFCDR at a lower resolution of 500x500, and have to be interpolated to obtain the full MVIRI image resolution of 5000x5000. This ladder shape was the result of the interpolation at the jump 0°-360°. The interpolation adopted is a bilinear interpolation, and few tricks have been implemented to avoid this gray aera of few pixels with faulty values. Tests have been performed, and after introducing these patches the ladder has disappeared. However full disk products have been tested only in April and not in the winter season.

- Spatial interpolation of ECMWF data. A fix in the interpolation of ECMWF data has been performed. The interpolation that was used before was a weighted mean considering the geometrical distance of the point of interest with respect to the four closest ECMWF stations. Now a bilinear interpolation is used instead. A comparison between the old method and the new one is shown in Figure **??** for the total column water vapour parameter for one Input Tile. The bilinear interpolation method is now used, however it presents some features due to the presence of a non uniform gradient. However this behaviour is under investigation.

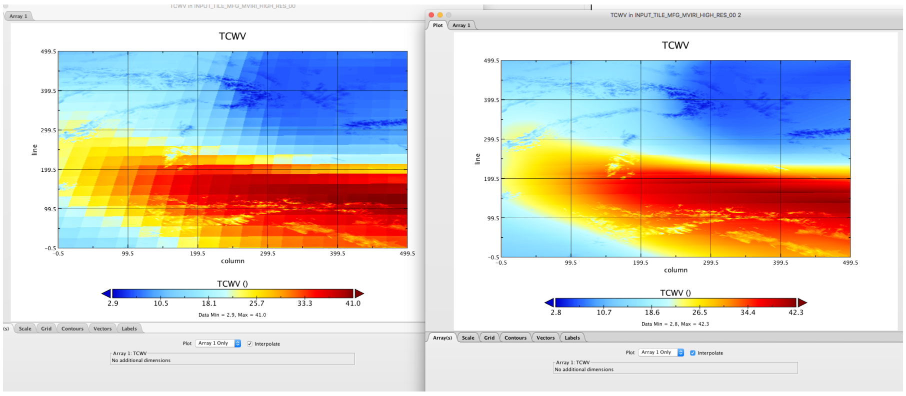

Figure 6: Comparison between the old and new interpolation methods for ECMWF data. The bilinear interpolation used on the right presents however some features, due to the presence of a non uniform gradient in the image.

- **IF ANY IMAGE OR CLOUD ORIENTATION CHANGES IN THE FUTURE, PLEASE BARE IN MIND TO WARN US!**
- **IODC runs at the moment with daily cloud masks!**

8 gedap3 release

The gedap3 code is a “standalone" version of gedap that runs on python3. The only main scope of this release is to:

1. allow the generation of the FIDUCEO CDRs. The FCDRTool library written at Brockmann requires indeed Python3.6, while the scientific release of GEDAP was still relying on Python2.7.

2. monitoring option.

1) The environment required to run gedap3 is the same as the one generated for the previous release, except for Python, which now is Python3.6. In addition, the fiduceo library needs to be installed in the same environment. As it was before, the GEDAP_DIR environment variable should point to the root of the gedap3 directory. To launch the generation of the CDRs, it is necessary that the conda environment where gedap3 is installed is active, and to be located in the directory where the processing has happened (i.e. where the subdirectories Input_Tiles or Products are located). The command to issue is

```
gedap -b -g -i=0
```

or passing the optional *-r* argument to point to a specific resources folder. Gedap will then look for the list_accumulation_periods.json file present in the resources folder and will start generating the products for the periods that have to be marked as “NOT PROCESSED" . The list of tiles that will be processed is the same as in common_init.json.

2) To launch the monitoring one should be located always in the processing directory (i.e. where the subdirectories Input_Tiles or Products are located) with the environment loaded. The command to generate the timeseries is then

```
gedap -b -n
```

this will open two pop up windows with their plots. One plot is about the ratio of processed pixels over total number. Despite for AOT this value is roughly 0.5 (i.e. about 50% of the pixels are processed on average for a full disk for one day) the values that will be displayed will not reflect this estimate, as in the total number of pixels there are also the borders of the image which are outside the Earth Disk and falsify the result. Additionally, for the surface albedo, the sea pixels are not processed, and the ratio is therefore lower than for AOT. The important thing to monitor is therefore the trend of the timeserie, to eventually spot a faulty processing.