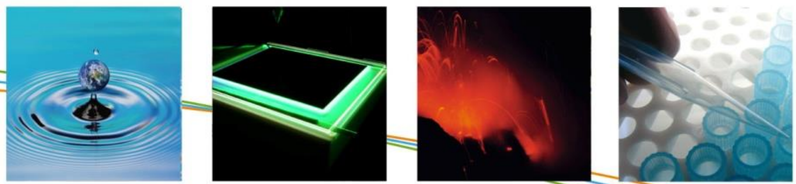
MSG Toolbox

Software User Guide

Date:	2015-06-16	Version:	1.0
Author(s):	T. Jacobs, D. Haesen, B. Smets, H. Eerens (VITO)		
Dissemination level:	Public		



Change Record

Issue	*Date*	*Page(s) / Section(s)*	*Description of Change*
1.0 rc1	2014-02-03	All	Initial draft for test version (rc1)
1.0	2015-06-16	All	First release of software and user manual

Table of Contents

1 BACKGROUND AND SCOPE 9

1.1 References and acknowledgements 9

1.2 Structure of this document 10

1.3 Contact 10

2 INSTALLATION AND REQUIREMENTS 11

2.1 Requirements 11

2.2 Download and Installation 11

2.3 Upgrading from a previous version 12

2.4 Software packages included in the installation 12

2.4.1 PDF Renderer 12

2.4.2 jCalendar and (part of) jGoodies 13

2.4.3 Jbzip2 13

2.4.4 Hdf-java 13

2.4.5 Glimpse 13

2.4.6 GDAL – Geospatial Data Abstraction Library 14

3 MAIN PROCESSING WORKFLOWS 15

3.1 Processing flow schematics 15

3.1.1 Main processing flow: producing daily composites from sub-daily input products 16

3.1.2 n-day composites from sub-daily and daily input products 16

3.1.3 Processing flow without temporal compositing 17

3.1.4 Secondary ROI preparation flow 17

3.1.5 Limitations in the flows 18

3.1.6 Intermediate data outputs for quality control 18

3.1.7 Optional processing steps 18

3.2 Overview of the processing steps 19

4 RETRIEVAL AND MANAGEMENT OF INPUT PRODUCTS 20

4.1 Retrieval of LSA-SAF products 20

4.2 File naming convention 21

4.3 Products supported in the toolbox 21

4.4 Working with a large number of products 23

4.5 Data management software 25

5 BASIC USAGE 27

5.1 Startup 27

5.2 Preparing Regions of Interest 28

5.2.1 Bounding box 29

5.2.2 Selection of bounding box on a map 30

5.2.3 Pixel size 30
5.2.4 Advanced remapping 31
5.2.5 Grid settings 35
5.2.6 Launching the ROI preparation workflow 35
5.3 Producing composites 37
5.3.1 Input data folder 38
5.3.2 Scenario 41
5.3.3 Workspace 48
5.3.4 Output Folder 48
5.4 Monitoring processing tasks and progress 50
6 TROUBLESHOOTING 52
6.1 File organization in the workspace 52
6.1.1 Import sub-flow 53
6.1.2 Daily compositing step 55
6.1.3 Remapping step 56
6.2 Viewing intermediate results 57
7 OUTPUT FORMATS 58
7.1 Default IDL-ENVI format 58
7.2 Additional GeoTIFF and ILWIS formats 58
ANNEXES 60
Annex 1. Example configuration for the GEONETCast Data Manager software 60
Annex 2. Sample source code for reading outputs in IDL-ENVI format 64
In regular C programming language 64
In Fortran90 programming language 71

List of figures

Figure 1 Main processing flow from sub-daily input to daily composite 16

Figure 2 Flows for producing n-day composites from sub-daily (top) or daily (bottom) input products 16

Figure 3 Processing flow without temporal compositing........ 17

Figure 4 Secondary ROI preparation flow 17

Figure 5 Product download page of the LSA-SAF web site........ 20

Figure 6 Warning messages informing users that the compositing task may take long to complete. 25

Figure 7 Questionmark icon for help (left) and *Advanced* button for experienced users (right) 27

Figure 8 *Welcome* tab of the *MSG Toolbox* 28

Figure 9 *Regions of Interest* tab........ 29

Figure 10 *Bounding box coordinates* sub-panel on the *Regions of Interest* tab 29

Figure 11 Bounding box selection via a map, with part of Europe selected 30

Figure 12 *Pixel size* sub-panel in the *Regions of Interest* tab 31

Figure 13 Advanced remapping configuration dialog window........ 31

Figure 14 Difference between 5% (default) and 30% (most severe) threshold on nominal/effective resolution fraction 32

Figure 15 Land-sea mask from GLC2000 for entire MSG disc (left) and zoomed in to Italy - Sicily (right) 34

Figure 16 Examples of mask interpolation threshold: 10% land (left), 50% (default, mid) and 90% (right), with overview of Italy (top) and zoom to Southern Italy – Sicily, with sub-nadir pixel size........ 34

Figure 17 *Grid settings* sub-panel of the *Regions of Interest* tab 35

Figure 18 *Build grid* dialog showing progress of the ROI preparation workflow 35

Figure 19 Information dialog upon successful completion of the ROI flow 36

Figure 20 Confirmation to over-write an existing ROI........ 36

Figure 21 *Compositing* tab........ 37

Figure 22 Buttons for submitting compositing workflow tasks........ 37

Figure 23 *Input data folder* sub-panel on the *Compositing* tab 38

Figure 24 Warning when recursively searching the input data folder 38

Figure 25 Date picker for entering the start or end date 39

Figure 26 Warning when processing more than one month of sub-daily input products in one go 39

Figure 27 Example relevant time period for regular daily compositing........ 39

Figure 28 Example relevant time period for daily compositing with start at 6 UTC 40

Figure 29 Example relevant time period for dekad compositing 41

Figure 30 *Scenario* sub-panel on *Compositing* tab. The previously saved scenario with the name “ET-daily-sum” and description “Sum of daily ET” is selected........ 41

Figure 31 *Scenario settings* window 42

Figure 32 *Temporal compositing* sub-panel of the *Scenario settings* window 42
Figure 33 Drop-down selection of composite period, with *Day* option disabled 43
Figure 34 *Advanced temporal compositing* dialog window 44
Figure 35 *Quality filtering* sub-panel of the *Scenario settings* window 45
Figure 36 *Remapping to Region of Interest* sub-panel.. 46
Figure 37 *Composite files* sub-panel in the *Scenario settings* window 47
Figure 38 *File formatting* sub-panel in the *Scenario settings* window... 47
Figure 39 *Scenario Information* sub-panel in the *Scenario settings* window.. 47
Figure 40 Warning when storing intermediate data files 48
Figure 41 *Output folder* sub-panel on the *Compositing* tab.. 48
Figure 42 Example of logging messages showing the output files produced by dekad compositing............. 49
Figure 43 Queue and progress of submitted tasks on the *Processing* tab... 50
Figure 44 Example of the workspace file and folder organization... 53

List of tables

Table 1 Overview of supported LSA-SAF input products 22

Table 2 Bitwise quality filtering 46

1 Background and scope

In late June 2011, the EUMETSAT Council gave the green light for the Satellite Application Facilities (SAF) network to receive funding for and thus enter the second part of the Continuous Development and Operations Phase (CDOP-2). This phase brings continuity to the existing and operational SAF products, as well as a wide range of further developments. One of those SAFs is the Land Surface Analysis SAF (LSA-SAF or L-SAF), headed by the Portuguese Institute for Sea and Atmosphere (IPMA, formerly known as the Portuguese Meteorological Institute).

Soon after, VITO signed the contract to participate in and contribute to the LSA-SAF in its CDOP-2 phase. This effectively marked the first time VITO joined EUMETSAT's SAF network.

VITO's contributions to LSA-SAF are two-fold: operational production and delivery of 10-daily vegetation indicators based on MetOp-AVHRR and a user tool to aid exploitation of LSA-SAF products, called the *MSG Toolbox*.

For the preparatory work on the *MSG Toolbox*, VITO received further funding from the Belgian Science Policy Office (BELSPO) through its PRODEX Programme with the European Space Agency (ESA). This preparatory work, the first step in the toolbox development, includes the analysis of requirements, comparison of existing software solutions that may contribute to the toolbox and the initial software design (i.e. chosen concept, structure). To ensure later take-up in the LSA-SAF context, this initial work was reviewed by IPMA.

This document is the user manual of the *MSG Toolbox* and is included in the software installation. It describes the software components, how they work together and how users can configure and use the toolbox.

1.1 References and acknowledgements

Please see the web site of the LSA-SAF, http://landsaf.ipma.pt, for more details on LSA-SAF work and products, or the EUMETSAT web site for more details on the related SAF networks.

From its inception, the *MSG Toolbox* was inspired by and re-uses parts of other, existing software (executables, graphical user interface code, design concepts, etc), in particular from:

- The LSA-SAF product processing chains developed and operated by VITO on behalf of, and delivering its output to, the European Commission, DG Joint Research Centre (JRC), Unit Monitoring of Agriculture through Remote Sensing (MARS, http://mars.jrc.ec.europa.eu/, http://www.marsop.info). These chains include a number of modules (executables) from the Global Image Processing Environment or "Glimpse" library (developed by VITO).
- The Time Series analysis software called "Software for Processing and Interpretation of Remotely sensed Image Time Series", or SPIRITS in short, that VITO developed for the MARS unit at the JRC.
- VITO's VGTExtract utility for integrating VEGETATION products into various commonly-used GIS and remote sensing software.

And, to less extent, the "Georeferencer" utility developed by Ricardo Da Silva and LSA-SAF and the free ILWIS software with its GEONETCast toolbox extension, available from University Twente, faculty ITC and 52North.

1.2 Structure of this document

Following this first, introductory chapter, chapter 2 focuses on the requirements and procedure for installing the *MSG Toolbox* software.

Chapter 3 details the main processing capabilities and workflows, essential to the understanding of how the toolbox works.

Chapter 4 provides information on the products from LSA-SAF that are used as inputs to the *MSG Toolbox*.

Chapter 5 describes how the toolbox can be configured and used to process LSA-SAF products.

Chapter 6 provides tips and techniques for troubleshooting and improving toolbox results.

Finally, Chapter 7 provides more details on the formats of the files produced by the toolbox and how these can be further read or analysed in GIS and Remote Sensing software.

Some information is provided in Annexes, for convenience.

1.3 Contact

For technical assistance in using *MSG Toolbox* software, please contact VITO's helpdesk via email to helpdeskticket@vgt.vito.be (preferably) or by calling +32 (0)14 336855.

For questions related to the characteristics and retrieval of LSA-SAF products, please contact the LSA-SAF Helpdesk service at helpdesk.landsaf@ipma.pt or call +35 1218447098.

2 Installation and requirements

2.1 Requirements

MSG Toolbox is written in the platform-independent Java language, and is designed as a shell that launches a few internal applications (written in C++). The *MSG Toolbox* software is distributed in binary form, without source code and **only Microsoft Windows platforms** (Windows XP or more recent, both 32 and 64 bit) are currently supported.

In order to run *MSG Toolbox* properly, a **Java Virtual Machine (JVM)** must be installed first. A Java Virtual Machine is typically part of a larger Java Development Kit (JDK), intended for software developers, or **Java Runtime Environment (JRE).** Though most Windows systems offer a Microsoft JVM, the use of this JVM has not been tested. It is therefore recommended to use Java from Oracle, which can be downloaded and installed from http://java.com free-of-charge.

The toolbox works on **Java version 7 (JRE 1.7) or higher**. As it is possible to install multiple Java versions alongside each other, and newer versions offer backward compatibility, installing the latest Java should not cause any problems. To check the version, please type in *java –version* at the command prompt, or check Control Panel – Java.

When processing, *MSG Toolbox* will create files in temporary folders that may be located in its installation path. For this reason, it is required that there is **sufficient free disk space** (100MB or more) on the hard disk where *MSG Toolbox* is installed. And if it is installed in a protected system path, for instance in a subfolder of *c:\program files* or *c:\ program files (x86)*, **administrator privileges** may be required for both the installation and for running the software. To avoid this, install *MSG Toolbox* in an un-protected path, such as c*:\MSGToolbox* (making sure to avoid spaces in folder names).

Beyond the set of input products, potentially large in volume and/or number of files, the *MSG Toolbox* can write a very large number of temporary files to the user-configurable *workspace folder*. By default, these are automatically removed, unless the *Keep intermediate results* checkbox is checked on the *Compositing* window (see Chapter 3 for details). It is therefore recommended to foresee significant free hard disk space (e.g. 1GB) for the workspace.

2.2 Installation

The current version of *MSG Toolbox* can be installed via self-extracting archive (*MSGToolbox.exe)*. The following files and folders are included in the main folder:

- The main program, in the form of a Java archive, *MSGToolbox.jar*
- A related example batch routine (.bat file) for launching the application
- The LSA-SAF program icon, iconLSASAF.ico
- This user manual
- The release notes
- The legal terms and conditions

The sub-folder *MSGToolboxData* contains

- Pre-defined bounding boxes for countries and economic regions covered by MSG (Africa, Europe, part of South America, Middle East), in the *roi* folder.
- Pre-defined list of data types in the *outputType* folder.
- Pre-defined list of output formats, in the *outputFormat* folder.
- Various images and logos used by the toolbox in the *images* folder.
- External software libraries and programs, in the *libs* folder.
- Static input grids, in the *libs\GLIMPSE\REFGRIDS*

The MSGToolboxData sub-folder furthermore contains three empty folders:

- *Temp:* to be used as default workspace (can be overridden by the user)
- *Scenarios:* for storing processing scenarios (configuration files)
- *Grids*: for storing the outputs of the ROI preparations (inverted grids and land-sea masks) for use in the remapping.

A set of ancillary, static data files is to be downloaded separately (see above link), so that users can update them independently of the software in case of (infrequent) changes. These data files include:

- LSA-SAF's input latitude and longitude data with 4 byte precision for the centre of each pixel and filenames following the pattern RRRR_XXX.img, where:
  RRRR = LSA-SAF region NAFR, SAFR, SAME and EURO and XXX being LON for longitude and LAT for latitude.
- Static IGBP mask, named RRRR_MSK.img with RRRR again referring to the region (NAFR,…).
- GLC2000-derived land-sea mask for the entire MSG disc, with the filename GLC2K_LS.img

These ancillary data files are available in ENVI format, ready for use in the toolbox, in an archive. It suffices to decompress this archive into the *MSGToolboxData\libs\GLIMPSE\REFGRIDS* folder.

2.3 Upgrading from a previous version

As this is the first official release version (1.0), there are no previous versions. For test-users who were using the beta-release (1.0rc1), it should suffice to update the *MSGToolbox.jar* file and *MSGToolboxData\libs\gdal* sub-folder.

2.4 Software packages included in the installation

The *MSG Toolbox* relies on several software components (libraries, programs) provided by third parties, i.e. beyond LSA-SAF or VITO. These are all free-of-charge and publicly available, but may be subject to specific licensing terms and conditions such as open source license. For informational purposes only, and without intending to provide an exhaustive list, here is a list of these third-party software components:

2.4.1 PDF Renderer

Location in the installation: MSGToolboxData\libs\PDFRenderer\
Source: http://java.net/projects/pdf-renderer

PDF Renderer is an all-Java library that renders PDF documents to the screen using Java2D. PDF Renderer is licensed under the GNU Lesser Public License, version 2.1 (LGPL-2.1) More information on LGPL-2.1 can be found at http://opensource.org/license/lgpl-2.1.php

2.4.2 jCalendar and (part of) jGoodies

Location in the installation: MSGToolboxData\libs\jcalendar\
Source: http://www.toedter.com/en/jcalendar/
JCalendar is a Java date chooser bean for graphically picking a date. It is provided as free software under the terms of the GNU Lesser Public License (LGPL). See http://opensource.org/license/lgpl.html for details.

2.4.3 Jbzip2

Location in the installation: MSGToolboxData\libs\jbzip2\
Source: http://code.google.com/p/jbzip2/

Jbzip2 is a Java-based compression/decompression library for bzip2 archives. It is provided with the following, open source license according to the MIT model (see http://www.opensource.org/licenses/mit-license.php):

2.4.4 Hdf-java

Location in the installation: MSGToolboxData\libs\HDF5\
Source: http://www.hdfgroup.org/downloads/

It is Java library for accessing, reading and writing files in the Hierarchical Data Format (HDF), notably version 5 (HDF5). It is provided with a BSD-style open source license, as specified on http://www.hdfgroup.org/products/licenses.html.

2.4.5 Glimpse

Location in the installation: MSGToolboxData\libs\Glimpse\
Source: in-house developed by VITO NV, the developer of the *MSG Toolbox*.

The GLobal Image Processing SoftwarE (Glimpse) is a set of command-line driven image processing routines developed since 1990, notably as part of several contracts for the European Commission, DG Joint Research Centre's unit on monitoring of agriculture via remote sensing (MARS, http://mars.jrc.ec.europa.eu). The

MSG Toolbox contains Glimpse programs for grid inversion (GRIDinvM.exe), remapping (GRIDmapM.exe) and mask interpolation (INTERPOL.exe) in particular. The GRIDinvM and GRIDmapM were modified/rewritten to serve the toolbox purpose.

2.4.6 GDAL – Geospatial Data Abstraction Library

Location in the installation: to be included in future.
Source: http://www.gdal.org

GDAL is a translator library for raster geospatial data formats. GDAL is distributed under an X/MIT license reproduced below. The OGR Simple Features Library is an C++ open source library (and command line tools) providing read (and sometimes write) access to a variety of vector file formats including ESRI Shapefiles, S-57, SDTS, PostGIS, Oracle Spatial, and Mapinfo mid/mif and TAB formats. OGR and GDAL libraries are provided as an integrated package.

The GDAL licensing terms are intended to give you permission to do whatever you want with the GDAL source code: download, modify, redistribute as you please, including building proprietary commercial software, no permission from Frank Warmerdam, OSGeo Foundation or anyone else is required.

Copyright (c) 2000, Frank Warmerdam

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be include in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

A more complete collection of license terms for GDAL/OGR and subcomponents will be included within the software distribution in the LICENSE.TXT file.

3 Main processing workflows

The *MSG Toolbox* operates the following four main image processing workflows, using operationally provided LSA-SAF products as input:

- A flow for the production of daily composites from sub-daily (i.e. 15 min, 30 min, hourly) frequency input products. This is the primary objective of the toolbox, hence the most important workflow.
- Two workflows to produce n-daily or periodic (10-daily or dekadal, monthly and yearly) composites from respectively sub-daily and daily frequency input products.
- A fourth flow to remap input products on file-per-file basis, without any temporal compositing operation.

The above four processing flows all share a remapping step, that uses grids and a mask as input. These are prepared separately in a specific workflow, called “ROI preparation flow” (see section 3.1.4). The grids and mask prepared by this separate flow can be re-used in any subsequent run of four main workflows. The processing flows do not mix different types of input LSA-SAF products (radiation flux DSSF, vegetation cover FVC, etc.).

This chapter provides an overview of these workflows and the individual processing steps that they are composed of. Knowledge of these flows and components is vital to understanding how the *MSG Toolbox* works and what it aims to accomplish.

The initial step in all of the flows, being the retrieval and management of LSA-SAF products, is not included in the toolbox software. As these are important pre-requisites to exploiting the toolbox properly, good practices and examples of this are provided in the next chapter.

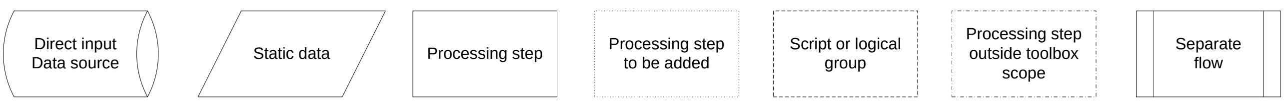
For the proper understanding, some prior knowledge of the LSA-SAF products and their formatting is recommended. While this is described on the LSA-SAF website, http://landsaf.ipma.pt and the product documentation provided there, it is good to know that LSA-SAF products are organized as Bzip2-compressed archives of data files in HDF5 format. Each data file in turn includes one or (more likely) several data layers. For example, one layer with the main variable, one layer with uncertainty estimates and one layer with bitwise-encoded quality information (often called quality flags). The latter uses specific bits to indicate, for instance, cloud or snow occurrence that obscure the measurement and hence decrease the quality of the data value or may even prevent the value from being computed (retrieved) at all.

LSA-SAF products are typically made available for four specific regions, called NAFR (northern Africa), SAFR (southern Africa), EURO (Europe) and SAME (South America), with fixed sizes in terms of numbers of lines and columns and in the native, geostationary projection.

The default output format is IDL-ENVI, with some extensions to the text header (.hdr) file to allow exploitation in SPIRITS. Additional outputs in GeoTIFF and ILWIS format are available as well. These output files all contain a single data layer, resulting from the processing of the main variable layer of the inputs (with e.g. optional filtering on quality, remapping, temporal compositing).

3.1 Processing flow schematics

The flows below are all illustrated using the following legend:



The arrows indicate the logical flow of data, from one processing step to the next. Configuration files, optional input data (e.g. land-sea mask for quality filter) and optional (e.g. intermediate) output data are not shown for simplicity.

3.1.1 Main processing flow: producing daily composites from sub-daily input products

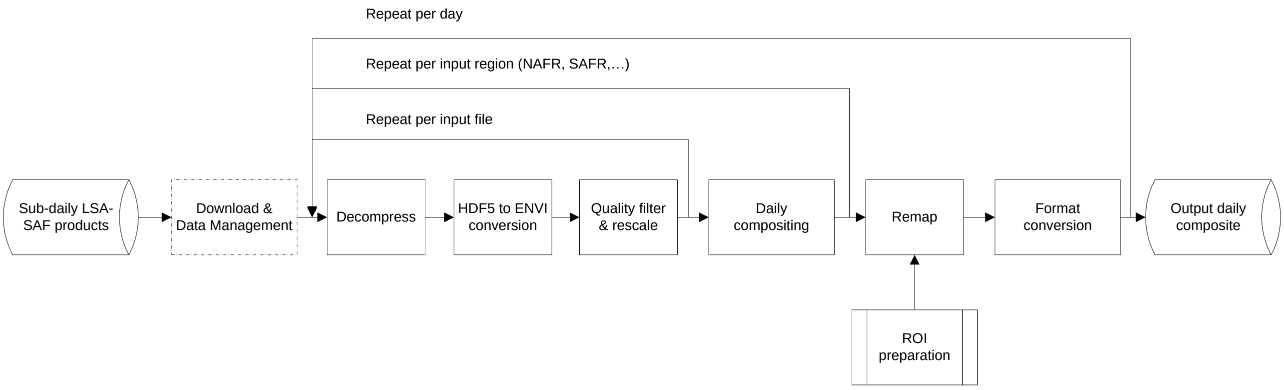


Figure 1 Main processing flow from sub-daily input to daily composite

After optional decompression and conversion to the internal ENVI-like file format of each relevant file, the main data layer is re-scaled (implying a filter on data value range) and optionally filtered using thresholds on the error margin layer and the bitwise quality flag information, where these are available in the input products. These resulting imported (filtered) values, still per time slot (15, 30 minutes or hours), are fed into the daily compositing to derive the daily minimum, maximum, average and/or sum, optionally accompanied by the number of observations used. Each daily composite image is then remapped to Geographic lat/lon, using a separately prepared region of interest (ROI). And finally, the format conversion step adds optional outputs in other formats (e.g. GeoTIFF).

3.1.2 n-day composites from sub-daily and daily input products

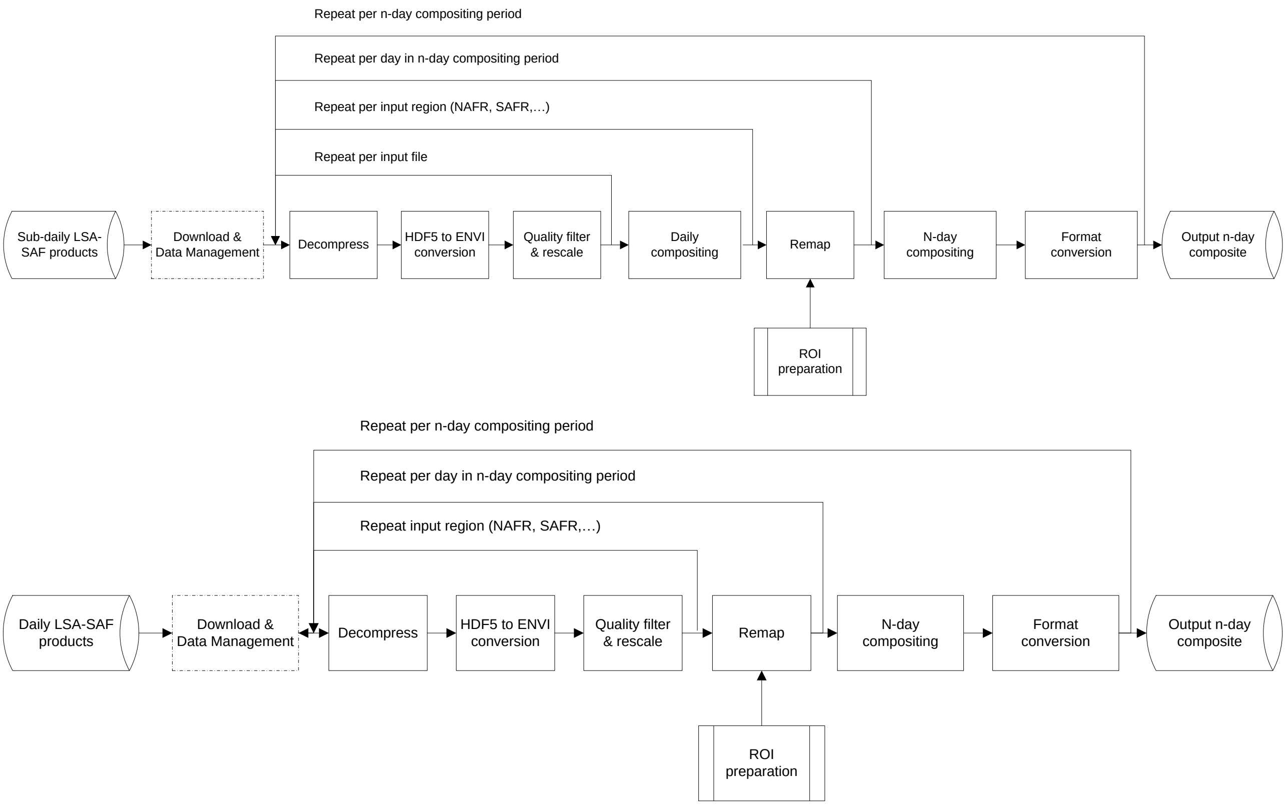


Figure 2 Flows for producing n-day composites from sub-daily (top) or daily (bottom) input products

Compared to the above first workflow, there are two additions:

- a processing step, called "N-day compositing" to compute the n-day values from daily ones, immediately following the remapping;
- the ingestion of daily frequency input LSA-SAF products (e.g. Leaf Area Index), therefore omitting the daily compositing step (bottom figure).

The resulting composites over dekads (periods of 10 days), months or years, called n-day or periodic composites, comprise the minimum, maximum, average and/or sum, again optionally accompanied by the number of observations. By convention, the dekads are defined as days 1-10, 11-20 and 21-end of month.

3.1.3 Processing flow without temporal compositing

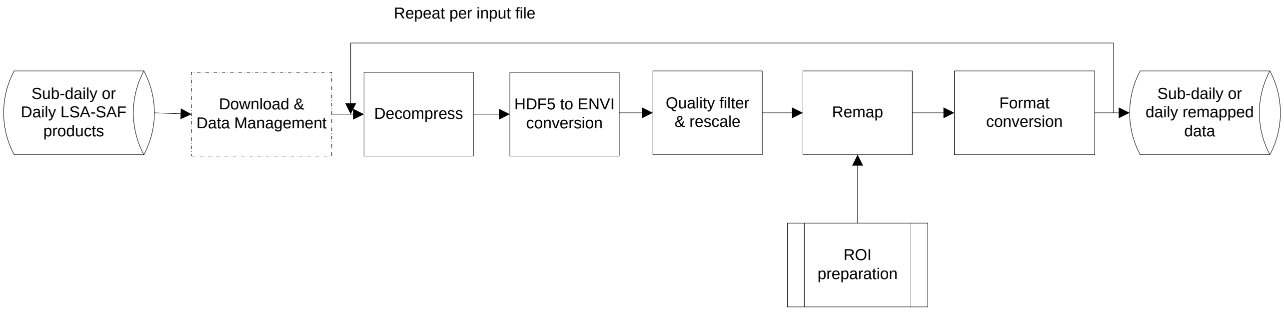


Figure 3 Processing flow without temporal compositing

In this fourth flow, the temporal compositing steps - both daily and n-day - are removed altogether. This flow works on a file-per-file basis and can thus ingest sub-daily, daily and ten-daily (D10) frequency input products.

3.1.4 Secondary ROI preparation flow

In all of the above main processing flows, the remapping step uses grids and masks prepared by the same, secondary flow called "ROI preparation", depicted here. It is intended to be run first, for each study area and thus independently of the available input LSA-SAF products.

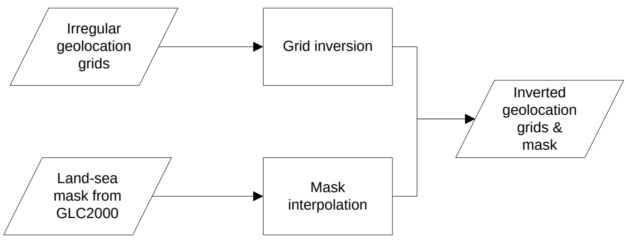


Figure 4 Secondary ROI preparation flow

Through respectively the grid inversion and mask interpolation (resampling) steps, static input grids and land-sea mask are processed into inverted grids and a mask, that are spatially congruent to the user-defined region of interest (ROI), in terms of spatial extent (bounding box), pixel size (spatial resolution) and spatial reference system (currently fixed to geographic lat/lon with WGS84).

This ROI preparation flow is started independently from the main flows and only executed when the user defines another target ROI (i.e. study area or resolution). It thus allows the prepared, ROI-specific grids and masks to be re-used for subsequent runs of the main processing workflow(s). For example, the same grids can be re-used in subsequent processing of composites of different product types (Leaf Area Index, Fire Risk Map,…) and/or temporal intervals (days, dekads, months) over the same study area.

The irregular geo-location grids were downloaded from LSA-SAF website and prepared for use in the toolbox in advance (see section 2.2). When updates of the grids are made available, suitable versions will be posted on the LSA-SAF website so that users may update them without having to re-install the toolbox software completely. A land-sea mask is derived from GLC2000 (resolution of 112 pixels/degree) beforehand as well and and is similarly provided.

3.1.5 Limitations in the flows

Since all the proposed processing flows include either remapping, temporal compositing or both, they cannot provide layers with discrete values such as classified snow cover as output. Such flags are thus used only for optional quality filtering of the input products.

The remapping currently assumes that the input products are provided in the four typical regions: NAfr, SAfr, Euro and SAme and in geostationary view (GEOS projection), thus preventing the preventing the processing of in particular Fire Radiative Power (FRP) products.

There is currently no workflow to produce monthly or yearly composites from ten-daily (D10) input products. These products can however by remapped by the fourth workflow that omits temporal compositing.

While LSA-SAF aims to back-process products in order to provide a consistent time series (same version of algorithm), users may still have products from different versions, which often should not be combined into the same temporal composite. A check on the product version is in the roadmap for the next version of the *MSG Toolbox* software.

3.1.6 Intermediate data outputs for quality control

As the “de-compression” (incl. file copy) and “HDF5 to ENVI conversion” steps are preparatory as they do not affect the data values, their intermediate outputs are not made available to the user. In other words, the “de-compression”, “HDF5 to ENVI conversion” and “Quality filter & rescale” steps can be considered together as an atomic “import” sub-flow, repeated for each relevant input file. And first intermediate output files are only available at the end of this sub-flow, effectively the output of the “Quality filter & rescale” step.

To allow users to verify and visualize the processing performed, the outputs of this import sub-flow, “daily compositing”, “Remapping” and “n-day compositing” steps may be optionally retained on disk in simple flat binary (ENVI with text header extensions) format. See chapters 5.3.3 and 6 for more information.

3.1.7 Optional processing steps

The de-compression (un-zipping) of input LSA-SAF products (.bz2) is only performed when the inputs files were not already decompressed prior to their ingestion in the *MSG Toolbox*.

When not filtering on the error margin or the bitwise quality flag layers, the “Quality Filter & Rescale” step falls back to re-scaling the ENVI input file, implying a filter on the range of pixel values for the main variable layer, and adjusting the intermediate ENVI header file.

The temporal compositing steps, both daily and n-day, are optional.

When the output format is ENVI-format (with extensions for use in SPIRITS software), the format conversion step does not perform any operation except moving the resulting files from the workspace to the final output folder.

3.2 Overview of the processing steps

Processing step	Main function(s)
Decompress	Removes the bzip2 compression from input .bz2 files.
HDF5 to ENVI conversion	Converts the files to ENVI as internal working format of the *MSG Toolbox*. Also extracts the HDF5 file attributes.
Quality filter & rescale	Optionally filters the data layer based on the associated bitwise quality flag layer or error margin layer. Re-scales the main layer of input data, which implies the use of a pre-configured data value range.
Daily compositing	Combines time slots within a day into a daily minimum, maximum or average.
Remap	Remaps to geographic lat/lon and spatially assembles the NAFR, SAFR, SAME and/or EURO regions, with optional filtering on spatial resolution. Masking with land-sea mask is also part of this step.
Format conversion	Translates the default ENVI-format outputs to additional output format(s) for subsequent file read/import in a specific Remote Sensing/GIS software.
Grid inversion	Inverts the fixed geolocation grids to provide column/record lookups for each pixel in the output region-of-interest (regular grid).
Mask interpolation	Resamples the land-sea mask, making it spatially congruent with the output region-of-interest.

4 Retrieval and management of input products

4.1 Retrieval of LSA-SAF products

LSA-SAF products can be retrieved both via the web (FTP) and via the EUMETCast satellite broadcast.

For web access, navigate to the LSA-SAF web site, http://landsaf.ipma.pt, register and login. The Download option should then appear in the menu on the left and allow ordering of specific time slots (periods) of the various types of products. Bulk orders of large quantities are possible as well, however only with manual intervention by IPMA staff, as indicated on the web site. An automatic dissemination can be provided as well. The data can then be accessed in a routinely updated FTP folder. FTP client software, such as the free FileZilla (with graphic user interface) or GNU wget (see here and here) and LFTP (for automatic, command-line based downloading) are freely available.

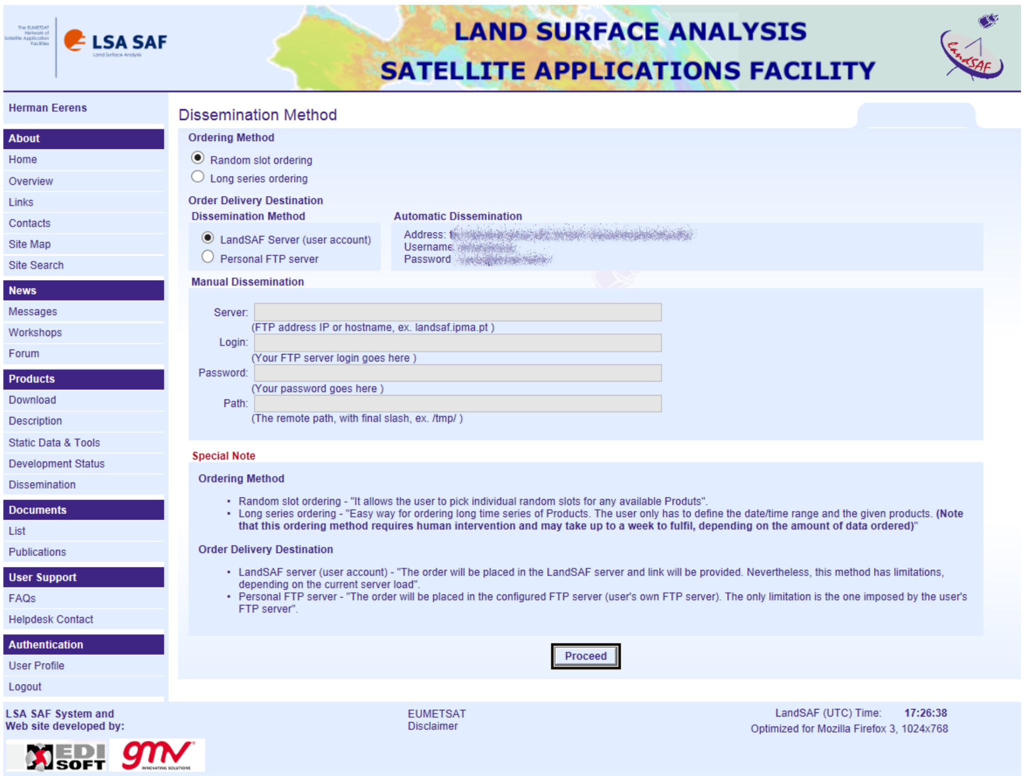


Figure 5 Product download page of the LSA-SAF web site

For access via the EUMETCast satellite broadcast, and after setup of the receiving station (antenna etc.), it suffices to register for LSA-SAF products via the EUMETSAT EO Portal and enabling the proper channels on your receiving station. European Ku-band receivers can receive all three channels - SAF-Africa, SAF-Europe or SAF-Americas. EUMETCast users in Africa and Americas C-band beams can only receive the respective SAF-Africa and SAF-Americas channels offering fewer products.

For the latest updates on the dissemination of LSA-SAF products, you can consult the EUMETSAT Product Navigator. This also provides sample file names, typical file sizes and delivery frequencies. Simply use the "Extended search" option and select "LSA-SAF" as provider.

4.2 File naming convention

LSA-SAF products use the following file naming convention.

For online delivered products:
HDF5_LSASAF_SENSOR_TYPE_REGION_YYYYMMDDhhmm(.extension)
Whereby

- The literal characters "HDF5_LSASAF" indicate the data format (HDF5) and source (LSA-SAF)
- SENSOR is an identifier representing the sensor, e.g. MSG for Meteosat Second Generation or MO1/MO2 for MetOp 1 and 2;
- TYPE is an identifier for the type of product, which references the purpose or the main data variable (e.g. Leaf Area Index product provides LAI data values).
- REGION represents one of the four regions – NAFR, SAFR, EURO or SAME.
- YYYYMMDDhhmm is the nominal date and time for the product. It is usually the start of the time period (time slot) covered by the product. The time slot can cover 15 minutes (mm=00,15,30,45), 30 minutes (mm=00 or 30), one hour (mm=00), a full day (hhmm=0000) or a 10-day period.

Products disseminated via the EUMETCast satellite broadcast are in addition prefixed with "S-LSA_-":
S-LSA_-HDF5_LSASAF_MSG_var_region_YYYYMMDDhhmm

The filename extension is typically .bz2 for the bzip2 compressed archives (products) and omitted or set to .h5 for the HDF5 data files contained in those archives. The toolbox automatically decompresses the .bz2 files as needed, but can also work with the HDF5 files (with or without the .h5 extension) and supports also the S-LSA_- prefixed file names. It can also search through sub-folders recursively, with a speed optimization.

In particular, the *MSG Toolbox* searches for input files in following order of precedence:

- .bz2 compressed products without S-LSA_ prefix
- .bz2 compressed products with S-LSA_ prefix
- .h5 de-compressed products without S-LSA_ prefix
- .h5 de-compressed products with S-LSA_ prefix
- HDF5 files without filename extension and without S-LSA_ prefix
- HDF5 files without filename extension and with S-LSA_ prefix

Other formats of LSA-SAF products or related information (e.g. files starting with "ASCII_...") are not supported.

4.3 Products supported in the toolbox

The *MSG Toolbox* aims to support all products from the LSA-SAF that are operationally produced and thus routinely (and openly) available. The first release of the toolbox will however not support products derived from MetOp polar orbiting satellite and is thus limited to products derived from Meteosat Second Generation (MSG).

The LSA-SAF is performing a number of changes to their products during the second Continuous Development and Operations (CDOP) phase, such as the addition of new products and generating existing products at different time intervals. As these products are not yet available, they will only be considered in a future release of the toolbox.

The following table provides an overview of the support for the products that are currently operational in this first release of the toolbox.

Category	Product type	Frequency	Code in the filename	Comments
Fire	Fire Radiative Power – per fire pixel	15 min	FRP-PIXEL-ListProduct and FRP-PIXEL-QualityProduct	Not supported yet
Fire	Fire Radiative Power - grid	Hour	FRP-GRID	Not supported yet
Fire	Fire Risk Map	Daily	FRM-F024	All layers except the risk classes and reference temperature (TRef)
Vegetation	Fraction of Vegetation Cover	Daily	FVC	
Vegetation	Leaf Area Index	Daily	LAI	
Vegetation	Fraction of Absorbed Photosynthetic Active Radiation	Daily	FAPAR	
Vegetation	Fraction of Vegetation Cover	10-daily	FVC-D10	
Vegetation	Leaf Area Index	10-daily	LAI-D10	
Vegetation	Fraction of Absorbed Photosynthetic Active Radiation	10-daily	FAPAR-D10	
Snow	Snow cover	Daily	SC2	Not supported yet
Temperature	Land Surface temperature	15 min	LST	
Temperature	Land Surface temperature	Daily		Not available yet
Radiation fluxes	Down-welling Surface Short-wave Radiation Flux	30 min	DSSF	
Radiation fluxes	Down-welling Surface Long-wave Radiation Flux	30 min	DSLF	
Radiation fluxes	Down-welling Surface Short-wave Radiation Flux	Daily	DIDSSF	
Radiation fluxes	Down-welling Surface Long-wave Radiation Flux	Daily	DIDSLF	
Surface Albedo	Surface Albedo, broadband	Daily	ALBEDO	
Surface Albedo	Surface Albedo, broadband	10-daily	ALBEDO-D10	
Surface Albedo	Surface Albedo, spectral	Daily	AL-C1, AL-C2, AL-C3	Supported, except for BRDF model parameters (K012) and covariance matrix (CK).
Evapo-transpiration	Evapo-transpiration	30 min	ET	
Evapo-transpiration	Evapo-transpiration	Daily	DMET	

Table 1 Overview of supported LSA-SAF input products

Note: the *MSG Toolbox* does not include a workflow that produces monthly/yearly composites based on the 30-day synthesis products, ALBEDO-D10, LAI-D10, FAPAR-D10, FVC-D10, that are delivered with 10-daily sliding window intervals.

4.4 Working with a large number of products

The relatively course spatial resolution of MSG-derived products means that the file size per product is modest. The high temporal frequency, the main advantage of using a geostationary satellite as source, can however make for a considerable number of files that need to be stored, ingested and processed and a considerable total disk space volume occupied.

Of course, this depends on the type of application, with for instance near-real time monitoring typically requiring limited amounts of data (e.g. the most recent, plus long-term statistics) and other applications requiring longer time series.

The amount of required LSA-SAF products can easily amount to hundreds or even thousands files. Just imagine an application that requires spatial coverage of the full MSG disc (four regions), using a product with high temporal frequency (e.g. every 15 minutes), while producing composites over large time spans (e.g. yearly composite, but for instance also daily composites for several months in one go).

As the *MSG Toolbox* needs to produce, per input product, several intermediate files for each of the processing steps, in particular when the user asks to keep those files around for troubleshooting (using the *Keep Intermediate data* checkbox on the *Compositing* tab), it is clear that the software needs to organize those files carefully. See section 6.1 for more information on how this is accomplished.

But before the input LSA-SAF products can be ingested and processed, the toolbox needs to locate the relevant files on the hard disk first. **Proper management of the input data,** including **organization** in folders and **archiving** of un-used data, is an clear **MUST-HAVE** to keep the toolbox performance reasonable in terms of time duration, hard disk space and memory consumption.

Computers organize files on a hard disk using a so-called filesystem (e.g. NTFS, FAT32,…). With most common filesystems, file searching takes considerably longer when a large number of files and/or sub-folders (e.g. thousands) are located in the same folder.

HINTS:

- Make sure to **avoid** putting **unnecessary files** in the same folders as input products (keep the folders "tidy").
- **Re-structure** the folders whenever you see or expect **large numbers of files or sub-folders in the same folder**.

The set of input products that the *MSG Toolbox* considers and searches for is determined by:

- The data layer being composited (fAPAR, LAI, DSSF,…) and thus the product type they are part of (e.g. FWI layer is part of the Fire Risk product FRM);
- The Region of Interest specified, as this determines which of the four input regions are relevant;
- The relevant time period, which is compared against the nominal date and time of the product (e.g. : HDF5…**_201201010000**);

HINTS:

- The toolbox never mixes different types of products. So when working with multiple types of products, a **separation of product types** (LAI, FVC-D10,…) should be done **high up in the folder structure** (e.g. at first or second level).
- Keep **only the relevant input regions** (EURO, NAFR,…). When in doubt on the regions needed, prepare your ROI and do a small test run to see which regions are searched for. Checkboxes in the Scenario configuration narrow these down further.
- When **multiple input regions** are used (e.g. when looking at full MSG disc coverage), the **division by region** should be **low** rather than high up **in the folder structure**.

The considered time period, checked against the datetime in the filename, is in turn determined by

- The start and end dates specified via the text boxes or date pickers on the *Compositing* tab.
- The composite time period specified in the selected scenario.
- The maximum consecutive gap defined for the temporal compositing (if any).
- The UTC starting time (hour) in the daily compositing (if any).

And, of course, the folders where the toolbox will try to locate the relevant products are configured through:

- The top-level input data folder
- The checkbox to enable/disable searching subdirectories

HINTS:

- Make sure to read section 5.3 as it provides more information on how the toolbox determines if a product is relevant for the requested composite.
- **For management, keep in mind in particular the lead/trail gap in the compositing as well as the correct specification of the start/end date.**

Example 1: a near-real time production of dekadal composites for monitoring
The production of the first dekad of February, requires the following input data to be present:

- The products for those 10 days, as many time slots as possible
- The products for the preceding days (last dekad of January), at least up to the number of days specified as maximum gap
- The products for the next 10 days (2nd dekad of February) are not available due to near-real time application, so the “trail” will be empty anyhow.

For this type of application, it can be envisioned to set up automatic data retrieval via FTP or EUMETCast in combination with data management that archives input data over e.g. one month old. With this archiving in place, the number of days is limited to approximately 20. The folder structure can thus be simplified to two-levels:

<product type>
    <day 1>
    <day 2>
    …

Example 2: producing monthly composites covering a full year, for a scientific study
To produce monthly composites for the year 2013, the following input products are required:

- All the time slots of products for 2013
- Products from the last days of 2012, up to the number of days specified as maximum gap (note that this can be up to a full year extra!).
- Products from the first days of 2014, also up to the number of days specified as maximum gap.

Keeping in mind that this type of application will likely want to produce composites for several years, which makes it tempting to store the full multi-year archive in one spot, it is not recommended to have the toolbox search through so many files.

A better approach could be to work with the following structure:
```
<product type>
    <year 2010>
    <year 2011>
    <year 2012>
    <year 2013>
        <month January>
            <day 1>
        <month February>
        <Duplicate of 1 gap at end of 2012, e.g. full month of December>
        <Duplicate of 1 gap at start 2014>
```

This allows you to use the “year 2013” folder as starting point for the search (top-level input data folder), thereby avoiding searching the multi-year archive (irrelevant years 2010, 2011,…). As such studies are not regularly repeated, the lead/trail data (from previous and subsequent year) can be duplicated manually (once-off) and the archive only contains the “clean” folder structure, i.e. without the duplicates.

The toolbox will warn users when they enable the search of sub-directories, when the intermediate results are kept and when they start up the compositing of large periods of sub-daily frequency input products. Users can still continue the processing after this warning, however with the risk that it can take a long time to complete.

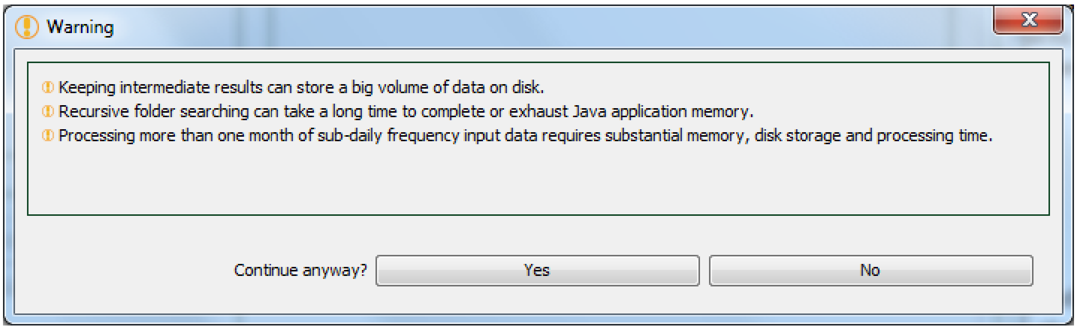


Figure 6 Warning messages informing users that the compositing task may take long to complete.

4.5 Data management software

The ILWIS and its various toolbox extensions, for instance for dealing with EUMETCast data (so-called GEONETCast toolbox), includes a software that helps to automate data management. This software, the GEONETCast Data Manager, can be downloaded and installed separately from the 52North Earth Observation Community web site, which also provides more a manual for its installation.
http://52north.org/communities/earth-observation/reception-stations/data-manager-software

A sample configuration (to be saved in plain text file, hence easy to edit) provided in Annex 1 defines:

- One group 'LSA-SAF_FTP' for managing LSA-SAF products downloaded via the website / FTP (i.e. without the S-LSA_- filename prefix
- Different groups (e.g. LSA-SAF_GNC_Europe) for managing the LSA-SAF products received via the SAF-Europe, SAF-Americas, SAF-Africa and SAF-Global channels on EUMETCast. Depending on the reception beam (e.g. C-band reception in Africa or Americas, Ku-band in Europe/Northern Africa) users may or may not be able to receive all of these channels.

The sample configuration further defines one input folder for retrieved LSA-SAF products and one output folder per group, with one week of data being stored (archived) in each group. It handles only a few types of LSA-SAF products (ET, LST, etc) but can be easily extended by replicating the Items in the respective groups.

For more details on the Data Manager configuration, please see the installation manual provided on the above web site.

The latest version of the Data Manager (2.0.1) is updated to allow an automatic restart, rather than an interactive one, which is convenient to restart after e.g. computer reboot.

5 Basic usage

The expected usage of the toolbox includes the following steps, with further details in the sections below:

- First prepare the Regions of Interest (ROI). This is to be repeated for each study area. It can be useful to try different pixel sizes (spatial resolutions or levels of detail) and, for advanced users, remapping settings.
- Once satisfied with the ROI that have been created, define one or more scenarios for the compositing workflows (e.g. daily, 10-daily or monthly, different data layers,…).
- Try out the scenarios on a limited time period and verify results. For such try-out, it can be useful to keep the intermediate results on disk.
- Now you are ready to start producing composites regularly, or try out larger time periods (larger sets of input products).

A command-line (batch) mode will be added in a later release to further facilitate and automate the production of similar composites at regular time intervals (e.g. every day in near-real time).

HINT:
Throughout the entire toolbox:

- the questionmark icon (figure below, left) opens up the relevant section or chapter in this manual
- and *Advanced* buttons (figure below, right) are intended only for experienced users only.

Figure 7 Questionmark icon for help (left) and *Advanced* button for experienced users (right)

5.1 Startup

To start up the *MSG Toolbox*, double click on the main Java archive (.jar) file in the top-level folder of the installation, called *MSGToolbox.jar*. This starts the application with default Java Virtual Machine (JVM) settings. Alternatively, a simple Batch file (.bat) file can be created, as per the provided example, to act as an application launcher. This offers the advantage of setting specific JVM settings, such the minimum and maximum available memory (respectively through the -Xms and –Xmx options).

While the available options depend on the provider of the JVM, more information on available options can be found through:

- Typing “java” (basic options) and “java –X” (extended options) at the Command Prompt.
- Searching the web, for instance http://www.oracle.com/technetwork/java/javase/documentation/index.html for Oracle JVM

When the *MSG Toolbox* starts up for the first time, it will open up the *Welcome* tab (figure below). Other main tabs include *Regions of Interest* (see section 5.2), *Compositing* (see section 5.3) and *Processing* (see section 5.4).

On the *Welcome* tab’s left-hand side, the *About* sub-panel provides a short description of the toolbox, acknowledgements of the developers and in particular the version number. The version number is particularly relevant when communicating on software issues or questions.

On the right-hand side, the *Getting started* sub-panel describes the regular steps in using the *MSG Toolbox*, notably some preparatory reading, the definition of ROIs, the compositing and monitoring of the processing tasks.

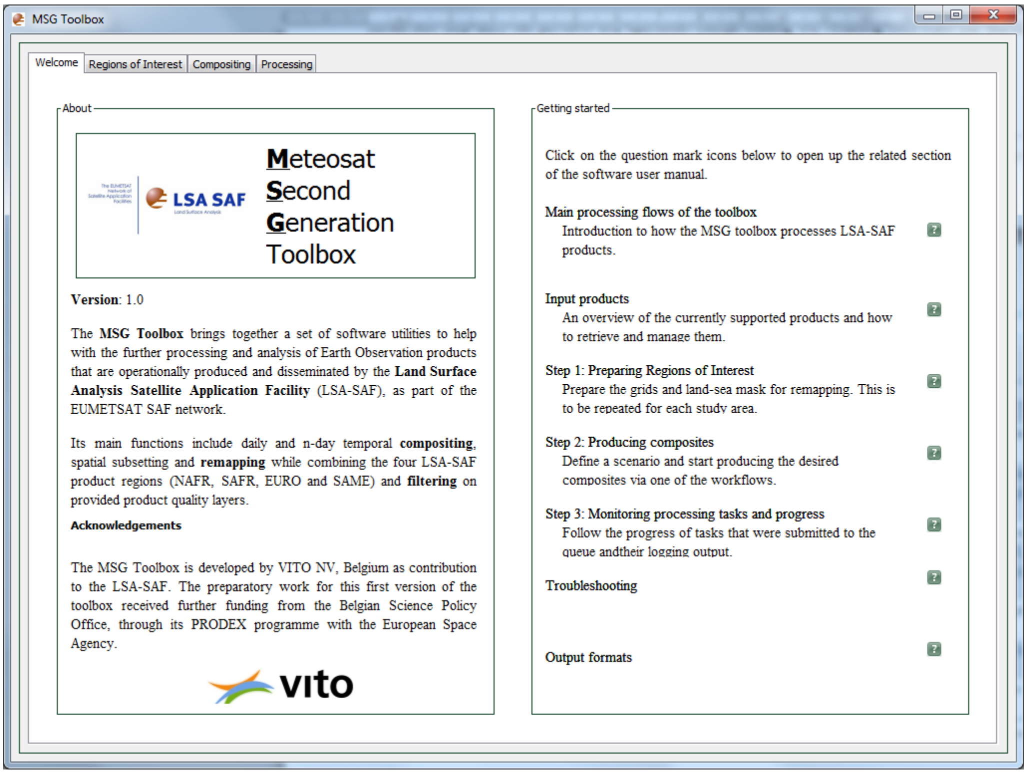


Figure 8 *Welcome* tab of the *MSG Toolbox*

5.2 Preparing Regions of Interest

A separate workflow is to be run to prepare grids and land-sea mask from static data, as input to the remapping. This is to be repeated for each study area, and potentially also for different pixel sizes (spatial resolutions, levels of detail) and with different advanced remapping configurations.

This remapping is needed to go from the native, geostationary view (projection) to a regular, geographic lat/lon spatial reference system, using the WGS84 datum. Reprojection or warping to other spatial reference systems is not foreseen in the toolbox. As the toolbox outputs are designed to fit into a variety of commonly used remote sensing and GIS or Remote Sensing software for further analysis, those software packages can be used for the reprojection.

The preparation of Regions of Interest is done via the *Regions of Interest* tab, shown in the below figure. This tab is further divided into the following sub-panels:

- For defining the rectangular bounding box coordinates: *Bounding box coordinates*;
- For defining the pixel size (spatial resolution): *Pixel size*;
- For advanced remapping configuration: *Remapping*;
- For additional information, notably on the spatial reference system used: *Additional information*;
- The *Grid settings* that contain in particular a name and short description that allow the computed ROI to be selected for use in the subsequent compositing workflows.
- And finally, the button *Start Task* to check the configuration and launch the secondary ROI preparation workflow that actually produces the grid and mask files.

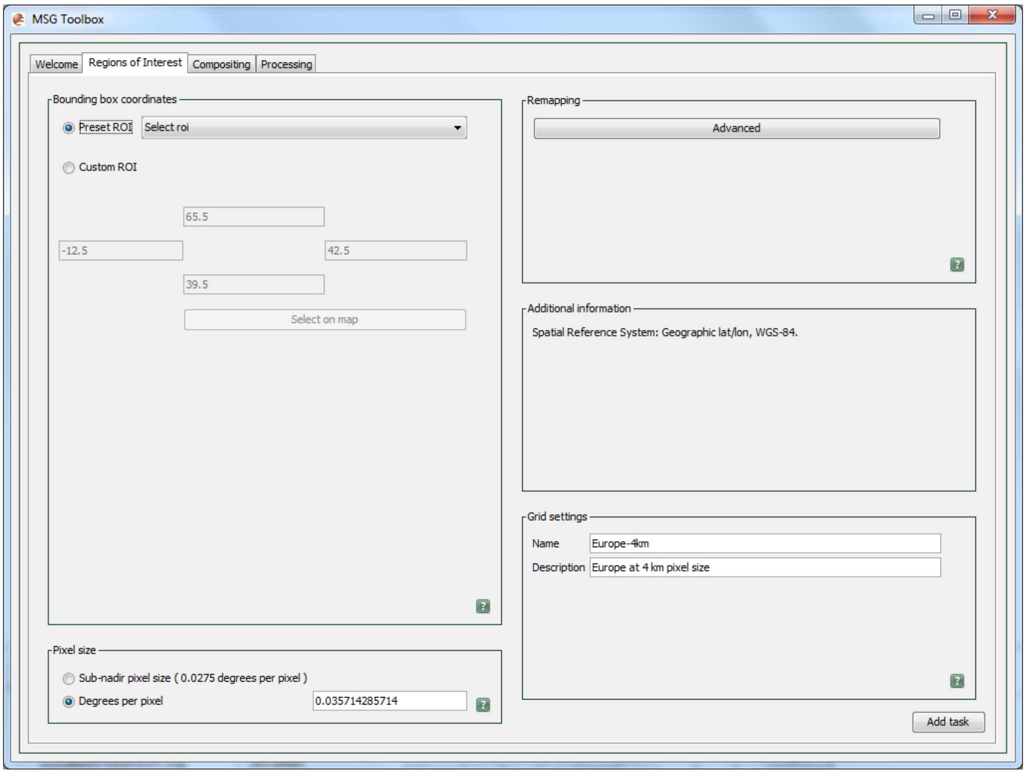


Figure 9 *Regions of Interest* tab

5.2.1 Bounding box

In the *Bounding box coordinates* sub-panel, the rectangular bounding box coordinates can be defined in decimal degrees with positive values for North and East hemispheres as illustrated below.

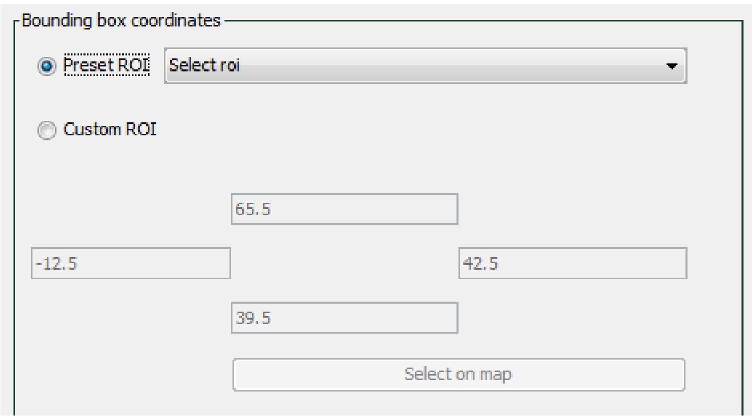


Figure 10 *Bounding box coordinates* sub-panel on the *Regions of Interest* tab

The coordinates can be defined in three ways:

- By selecting a pre-defined bounding box from the drop-down list.
- By selecting *Custom ROI* and entering the coordinates manually in the text fields.
- By selecting *Custom ROI* and the *Select on Map* button to interactively draw the bounding box on a map (see next section).

The pre-defined bounding boxes in the drop-down list represent the coordinates for all the countries (partly or entirely) in the MSG disc, hence all of Europe, Middle East, Africa and parts of South America.

These bounding boxes are rounded to integer degrees, so they may be larger than expected in particular for small islands and city-states (e.g. Vatican City). Upon selection from the drop-down list, the corresponding coordinates are copied to the text fields below. Switching to *Custom ROI* via the radio button then allows the user to modify them.

5.2.2 Selection of bounding box on a map

The *Select on Map* button on the *Bounding box* coordinates sub-panel, opens up a simplified, low resolution map, that is limited to (approximately) the coverage of the MSG disc and uses different colours for the various countries to facilitate selection (see figure below).

To select any area on the map, simply draw a red rectangle by clicking and dragging the mouse cursor (with left mouse button). The mouse scroll button can be used to zoom in and out. To pan the map, drag the mouse while pressing the right mouse button. Double-clicking resets the map to the central position. In case the window is resized, the map itself will be scaled while retaining the aspect (height – width) ratio.

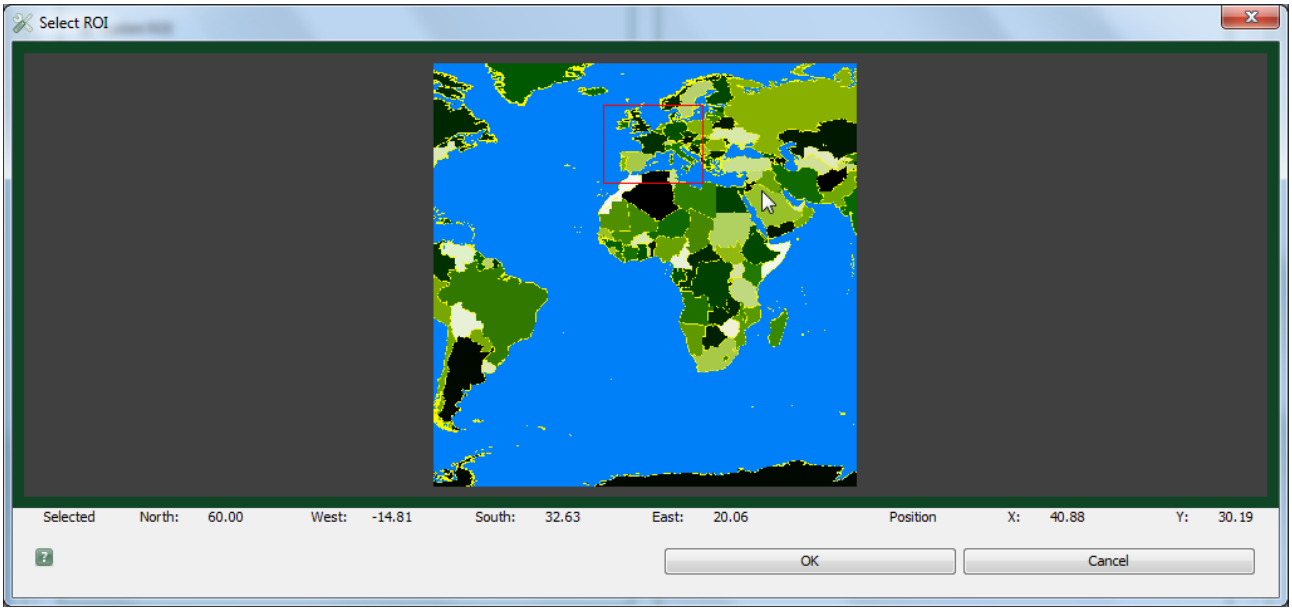


Figure 11 Bounding box selection via a map, with part of Europe selected

The information bar at the bottom informs the user of the currently selected coordinates (Selected – North/West/South/East) and the coordinate the mouse cursor is currently point to (Position – X and Y).

The *OK* button confirms the selection and returns to the *Regions of Interest* tab with the coordinates copied to the text fields for *Custom ROI*. The *Cancel* button returns to the same tab, however without storing the coordinates that were selected on the map.

5.2.3 Pixel size

In the *Pixel size* sub-panel, the user can choose between using the sub-nadir pixel size (0.0275 degrees per pixel), which is the finest level of spatial detail (highest resolution) achievable, or entering a custom pixel size, expressed as decimal degrees per pixel.

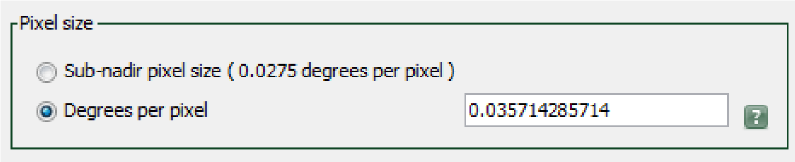


Figure 12 *Pixel size* sub-panel in the *Regions of Interest* tab

Due to the curvature of the Earth, the pixel size increases (resolution becomes coarser, fewer spatial detail) towards the edges of the MSG disc. Fixing the pixel size to the sub-nadir one for those areas is thus not recommended, as it will introduce a lot of interpolated values in-between actually measured values. The Advanced remapping configuration however allows to filter pixels on their size, hence avoiding the coarsest resolution pixels near the edge of the MSG disc.
For example: a pixel size of approximately 4km for most of Africa and 5km for most of Europe is fine.

5.2.4 Advanced remapping

HINT:
This window is intended for experienced users only. The default configuration should suffice for common applications.

The *Advanced* button in the *Remapping* sub-panel on the ROI tab opens up the *Advanced remapping settings* dialog, illustrated in Figure 13.

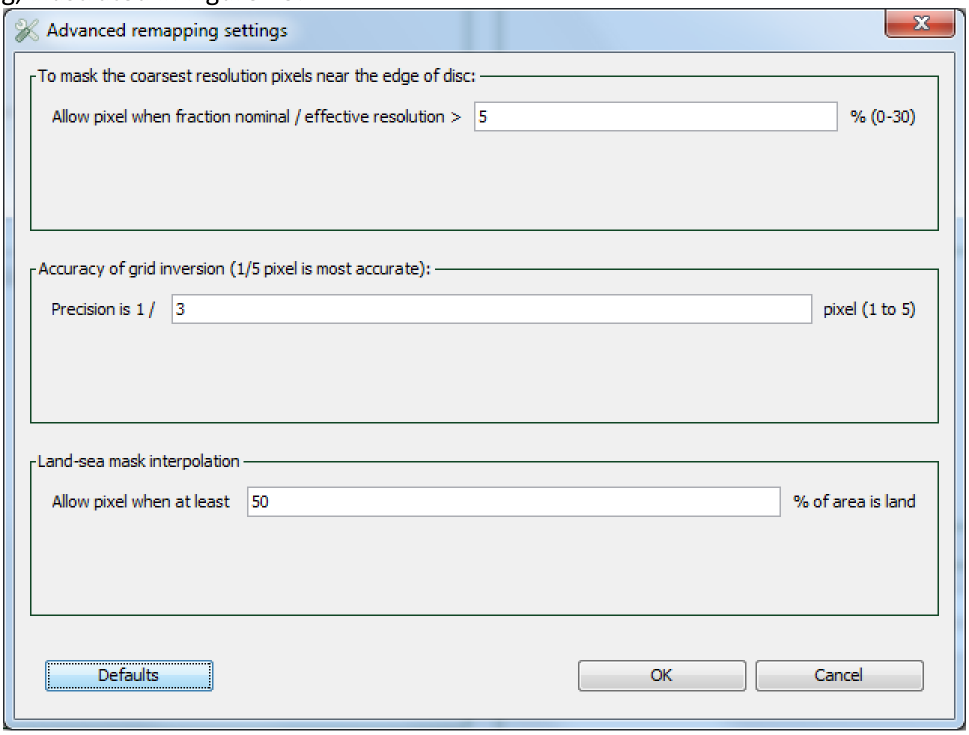


Figure 13 Advanced remapping configuration dialog window

This dialog is further divided into three (3) sub-panels. The first two, configuring the masking of coarse resolution pixels and the accuracy of the grid inversion, are related to the grid inversion step in the ROI preparation workflow (see section 3.1.4). The latter has to do with the land-sea mask interpolation step in the same workflow.

For the masking of the coarsest resolution pixels, near the edge of the MSG disc, a threshold is used on the fraction of the nominal and the effective resolution using the following formula:

Ratio 100 * 'Nominal Resolution' / 'Effective Resolution' < Threshold

Whereby

- The nominal resolution is hereby defined as 3km
- The Effective Resolution is computed as the angular distance between each input cell and its most removed neighbour in its 3x3-environment.

This threshold is set to 5% by default, 0% disables it completely and 30% is the most severe masking.
The below figures show the effect of changing this parameter from its default value of 5% to 30%, with green colour highlighting the pixel values that were masked.

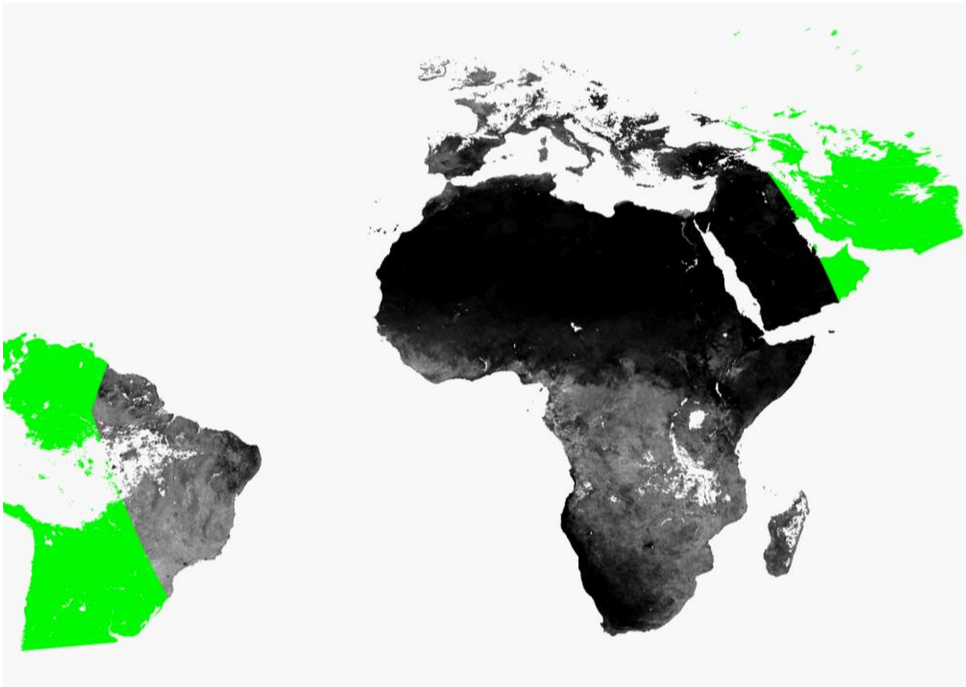


Figure 14 Difference between 5% (default) and 30% (most severe) threshold on nominal/effective resolution fraction

The accuracy of the grid inversion can be specified as a fraction of a pixel. The 1/1 or full pixel precision should be limited to values that may not be interpolated. When using sub-pixel precision, from half pixel (1/2) to one fifth of a pixel (1/5), the higher the accuracy, the longer the grid preparation will take. One fifth (1/5) pixel is the most accurate, but also the slowest. As most applications require only infrequent updates to the grids, a high accuracy is recommended.

To get an idea of the effect of changing the accuracy setting, here are extracts from the histograms (up to 99.99% of pixels) for different values, each time combining the four regions (NAFR, SAFR, SAME and EURO) into one grid (±75°) with pixel size of around 4km (0.035714285714 degrees per pixels to be exact). The Y column hereby represents the error in distance between estimated and true value, expressed in kilometre. Pixels in space or over ocean are flagged with digital value V = -2. As you can see, the accuracy is in any case high, with between 98.5 and 99.5% of pixels with an error smaller than or equal to the nominal resolution (3km).

a) Half pixel (1/2), the fastest sub-pixel precision

BIN	V	Y	Npix	Npix%	CUM%
2	-2	FLAG	10010255	56.74748	56.74748
4	0	0	4530018	25.68037	82.42785
5	1	1	2205364	12.50206	94.92991
6	2	2	468849	2.65787	97.58779
7	3	3	198835	1.12718	98.71497
8	4	4	97989	0.55549	99.27046
9	5	5	51680	0.29297	99.56344
10	6	6	27671	0.15687	99.72030
11	7	7	14666	0.08314	99.80344
12	8	8	8987	0.05095	99.85439
13	9	9	6382	0.03618	99.89057
14	10	10	4848	0.02748	99.91805
15	11	11	3761	0.02132	99.93937
16	12	12	2965	0.01681	99.95618
17	13	13	2183	0.01238	99.96855
18	14	14	1598	0.00906	99.97761
19	15	15	1204	0.00683	99.98444
20	16	16	954	0.00541	99.98985

b) 1/3 pixel, the default

BIN	V	Y	Npix	Npix%	CUM%
2	-2	FLAG	10011914	56.75688	56.75688
4	0	0	6159323	34.91680	91.67368
5	1	1	1040011	5.89575	97.56943
6	2	2	238126	1.34992	98.91935
7	3	3	91647	0.51954	99.43889
8	4	4	39651	0.22478	99.66367
9	5	5	19061	0.10806	99.77173
10	6	6	11417	0.06472	99.83645
11	7	7	7697	0.04363	99.88009
12	8	8	5667	0.03213	99.91221
13	9	9	4243	0.02405	99.93626
14	10	10	3144	0.01782	99.95409
15	11	11	2281	0.01293	99.96702
16	12	12	1755	0.00995	99.97697
17	13	13	1302	0.00738	99.98435
18	14	14	936	0.00531	99.98965

c) 1/5 pixel, the most accurate

BIN	V	Y	Npix	Npix%	CUM%
2	-2	FLAG	10013429	56.76547	56.76547
4	0	0	6955261	39.42892	96.19439
5	1	1	452741	2.56656	98.76095
6	2	2	113979	0.64614	99.40709
7	3	3	40944	0.23211	99.63920
8	4	4	20313	0.11515	99.75435
9	5	5	12192	0.06912	99.82346
10	6	6	8204	0.04651	99.86997
11	7	7	5781	0.03277	99.90274
12	8	8	4303	0.02439	99.92714
13	9	9	3239	0.01836	99.94550
14	10	10	2468	0.01399	99.95949
15	11	11	1885	0.01069	99.97018
16	12	12	1484	0.00841	99.97859
17	13	13	1096	0.00621	99.98480
18	14	14	861	0.00488	99.98968

The interpolation (resampling) creates a mask spatially congruent to the desired output region from a static land-sea mask derived from GLC2000 (1km resolution). The parameter defines the threshold for regarding a pixel as land: by default, at least 50% of the pixels in the GLC2000-based land-sea mask that overlap with the output pixel need to be indicated as a “land” class. To illustrate the effect of this threshold setting, let’s first examine the original land-sea mask derived from GLC2000 (figure below).

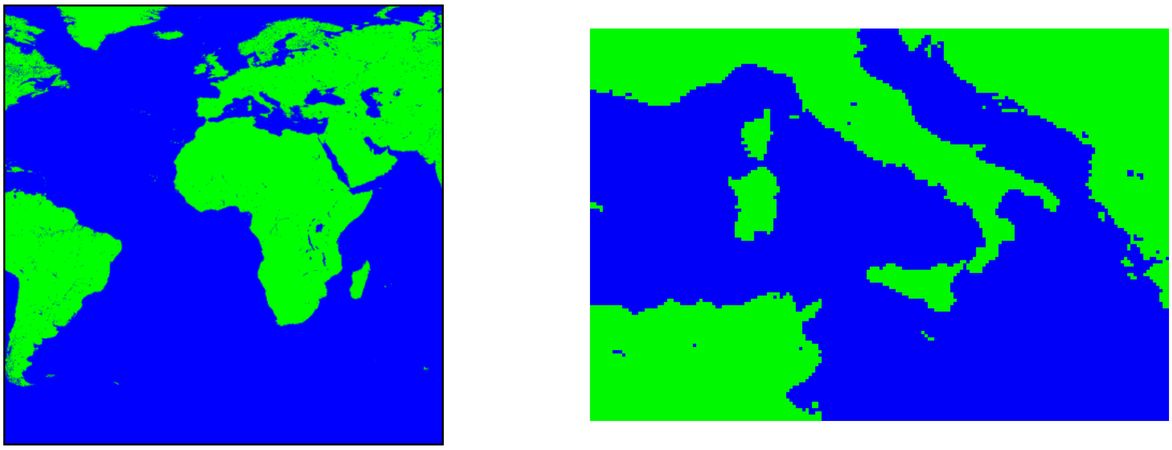


Figure 15 Land-sea mask from GLC2000 for entire MSG disc (left) and zoomed in to Italy - Sicily (right)

The following figures illustrate what happens when the setting is changed to 10%, as extreme low value (figure below, left), kept at its default of 50% (middle) or set to an extreme high value of 90% (right). A low setting will of course increase the number of pixels identified as “land” in the created land-sea mask and thus also the number of pixels considered for the remapping in the main workflows.

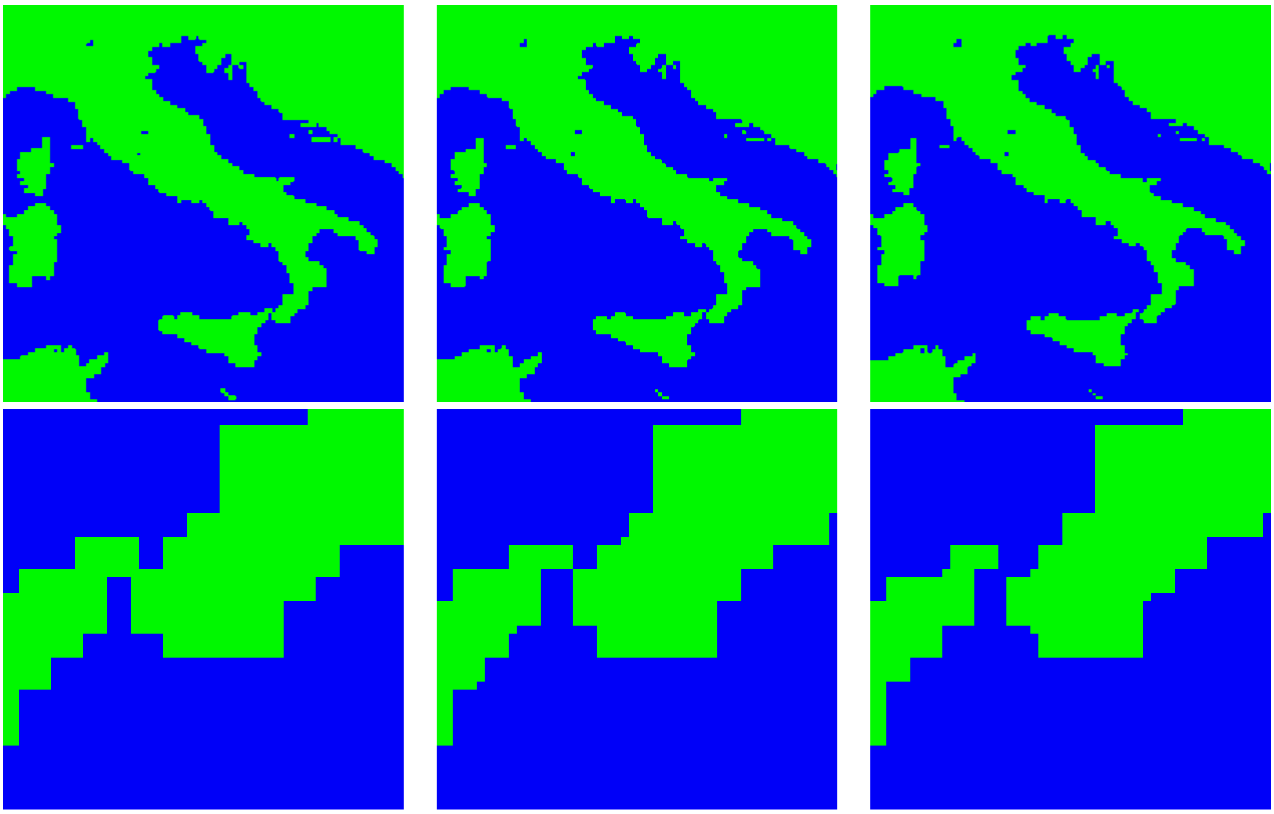


Figure 16 Examples of mask interpolation threshold: 10% land (left), 50% (default, mid) and 90% (right), with overview of Italy (top) and zoom to Southern Italy – Sicily, with sub-nadir pixel size

5.2.5 Grid settings

The *Grid settings* sub-panel of the *Regions of Interest* tab allows to specify a unique name and short description that identify the ROI that is computed. After the ROI computation task is submitted via the *Add Task* button, the newly defined ROI will appear in the drop-down list of ROIs that appears in the compositing scenario definition window (see section 5.3.2) and can then be used for starting compositing tasks. Note however that the ROI computation task, albeit very short in duration, is added to the task queue. It needs to have completed successfully before the new grids and land-sea mask are ready for use in compositing tasks (so be careful when disrupting the task order in the queue via e.g. removal of tasks).

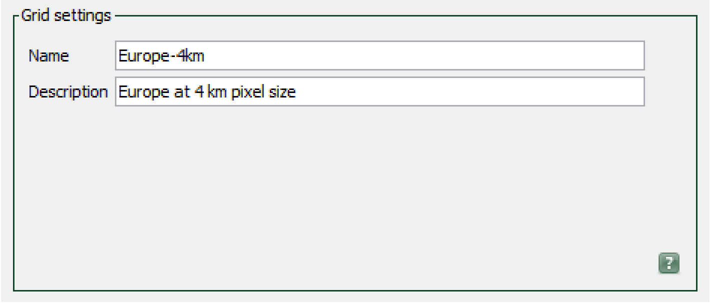


Figure 17 *Grid settings* sub-panel of the *Regions of Interest* tab

The specified *Name* is translated into a machine-usable file name (e.g. allowing alphanumeric characters, hyphens and underscores, but no whitespace).

5.2.6 Launching the ROI preparation workflow

After clicking the *Start Task* button on the *ROI* tab, the ROI preparation workflow (see 3.1.4) is launched. It is run in its own processing thread, not added to the task queue (section 5.4). While this ROI preparation is performed, a simple dialog window shows the progress.

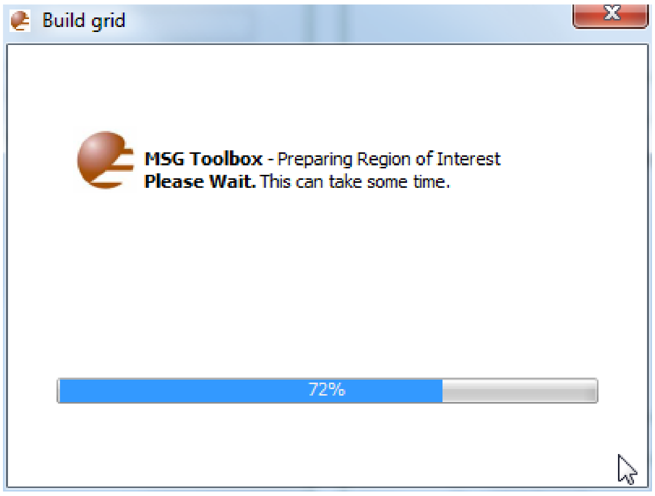


Figure 18 *Build grid* dialog showing progress of the ROI preparation workflow

Upon completion of the ROI preparation flow, an information dialog is shown summarizing the (start/end) of the different steps involved as well as the name, description and location of an XML file. This XML file contains the grid settings (name, description, etc.) and in turn refers to the location of the grid and mask files. Users should not edit the XML file directly, it is better to re-launch the workflow.

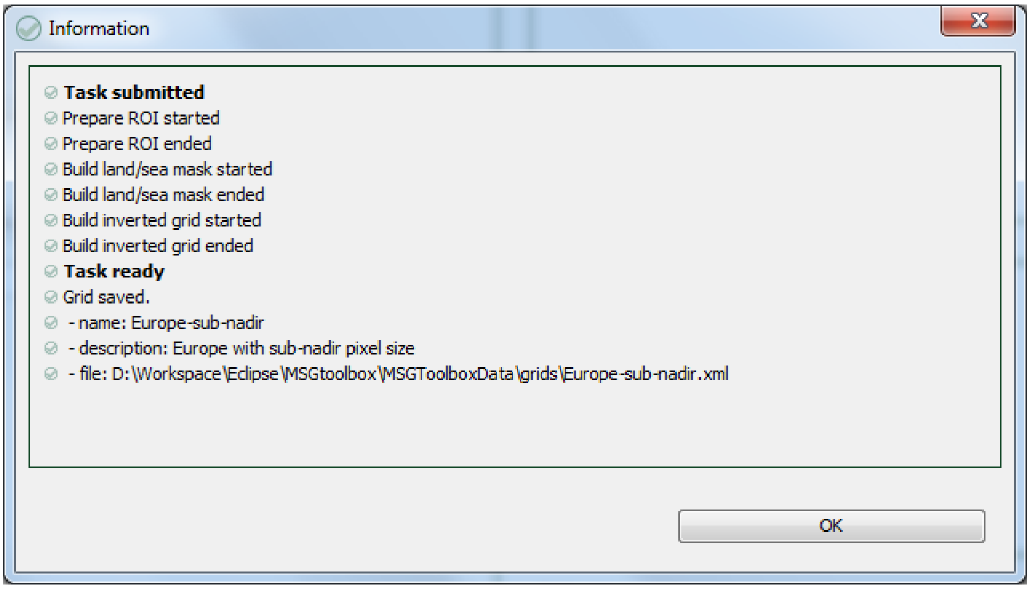


Figure 19 Information dialog upon successful completion of the ROI flow

The user is warned when trying to overwrite an already existing ROI, as this may be used in the on-going and/or future scheduled tasks in the task queue. When confirmed, the overwritten ROI could disrupt these submitted tasks. It is therefore recommended to first check the task queue, cancel or pause relevant tasks before proceeding.

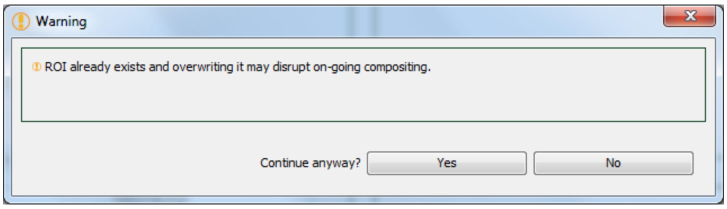


Figure 20 Confirmation to over-write an existing ROI

5.3 Producing composites

Following the preparation of one or more *Regions of Interest* (ROI) for the study area(s), the *Compositing* tab (figure below) allows to produce different types of composites using one of the main workflows that are described in chapter 3.

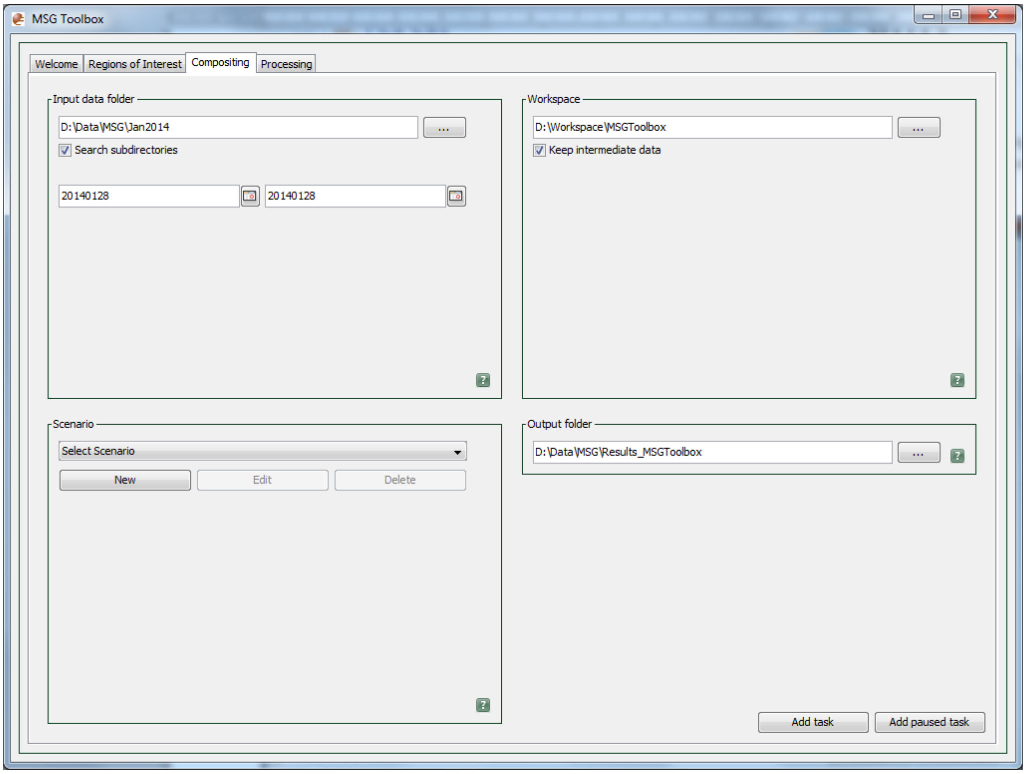


Figure 21 *Compositing* tab

The main configuration to the workflows is done through the definition of processing *scenarios*. These are managed via the *Scenario* sub-panel on the bottom-left. The other sub-panels define the input data folder and date range (*Input data folder* sub-panel, top-left), the *workspace* for storing intermediate results (*Workspace* sub-panel, top-right) and the output folder (*Output folder* sub-panel, bottom-right).

In the bottom right corner, two buttons allow the user to add the task at the end of the processing queue. The *Add task* button foresees that the task is scheduled to run when previous tasks (that were already on the queue) are complete.

For longer processing tasks, in particular for execution outside business hours, the task can be added and directly paused via the *Add paused task* button. The task is then queued like a normal task, but it will not be started until the user chooses to either *Resume* this task, or *Resume all* tasks via the respective buttons on the *Processing* tab (see section 5.4). It is however recommended to try running similar tasks via the *Add task* button first until the scenario is fine-tuned to provide satisfactory results.

Figure 22 Buttons for submitting compositing workflow tasks

5.3.1 Input data folder

After defining the necessary scenario(s), these can thus be re-used on input data located in different folders or different time periods (specified as days) via the *Input data folder* sub-panel.

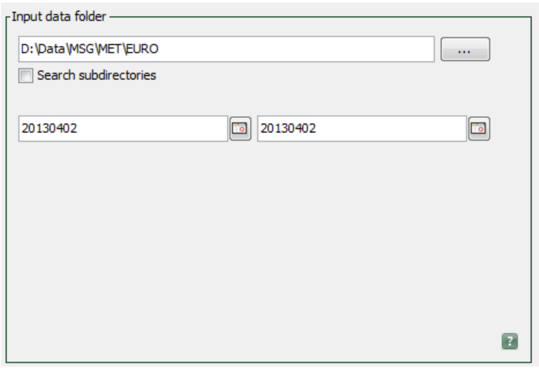


Figure 23 *Input data folder* sub-panel on the *Compositing* tab

The selection of the top-level folder can be done by either typing in the path into the text field or via the folder browse ... button. The user can opt to recursively search through sub-folders via the checkbox. When this sub-folder searching is enabled, the user will however receive a warning message on potential slow performance and be asked to confirm upon submission of a task.

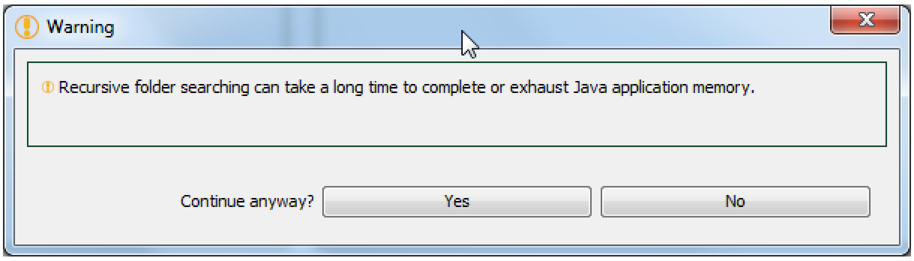


Figure 24 Warning when recursively searching the input data folder

The set of input products that the *MSG Toolbox* considers and searches for in the input data folder (and possibly its sub-folders) is determined by several factors:

- The data layer being composited (fAPAR, LAI, DSSF,…) and thus the product type they are part of (e.g. FWI layer is part of the Fire Risk product FRM and stored in files called HDF5…**_FRM**…, DSSF data layer is located in product files called HDF5…**_DSSF**… and so on);
- The specified Region of Interest, as this determines the relevant input regions (e.g.: HDF5…**_Euro**…, HDF5…**_NAfr**…);
- The relevant time period, which is compared against the nominal date and time of the product (e.g. : HDF5…**_201201010000**).

The relevant time period is in turn determined through a combination of

- The start and end date;
- The composite time period (None, Day, Dekad, Month, Year) selected in the scenario;
- The maximum consecutive gap, which may be defined separately for daily and n-day compositing;
- The hour (UTC) when the 24h period of the day starts (0 UTC by default).

The data layer, the (previously created) Region of Interest, the composite time period, maximum consecutive gap and starting time of the day are all specified in the scenario (see next section for more details).

The start and end day can be entered, as days, via the text fields or date pickers (figure below) in the *Input data folder* sub-panel so that the same processing scenario can be easily repeated for different time periods of input data (e.g. per month). The specified dates are “rounded” to the start or end of the day, dekad, month or year, depending on the composite period selected in the scenario. Please note that this also implies that, for composite period type “None” (the fourth workflow without temporal compositing), always a full day of input data is processed. It is not possible to limit the processing to a single 15/30min or hourly time slot.

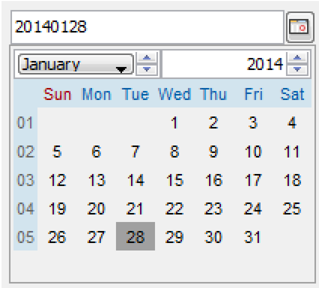


Figure 25 Date picker for entering the start or end date

For instance, when specifying dates 20130101 and 20130201 with dekad composite periods selected in the scenario, four composites will be created: one for each dekad in January (days 01-10, 11-20 and 21-31), and one for the first dekad (01-10) of February.

When the user specifies a date range of more than one month, with sub-daily frequency input images, the toolbox will warn about potential effect on performance and ask for confirmation upon submission of the processing task, as illustrated in the below figure.

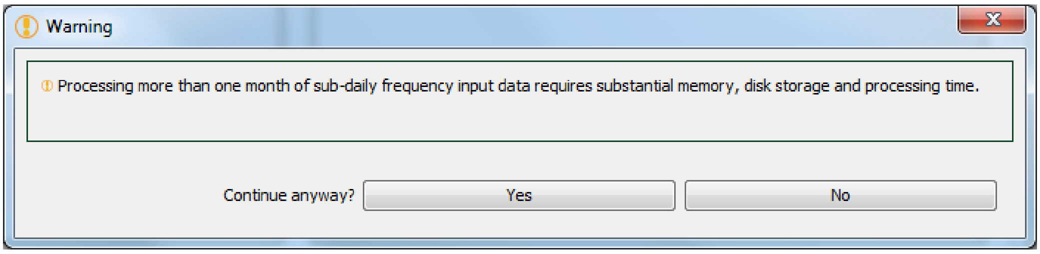


Figure 26 Warning when processing more than one month of sub-daily input products in one go

The below timelines help to illustrate how the relevant time period is determined for a) regular daily compositing, b) daily compositing with starting time and c) n-day (dekad in this example) compositing.

a) Example of time period considered for regular daily compositing:

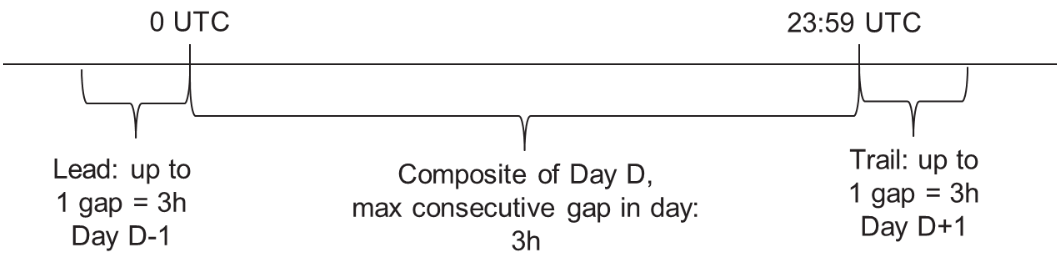


Figure 27 Example relevant time period for regular daily compositing

By default, the compositing for day D is defined from 0 UTC to 23:59 UTC and thus considers the time slots from 0 UTC (included) up to and including

- The 23:00 UTC time slot for hourly frequency products.
- The 23:30 UTC time slot for products with 30 minutes frequency.
- Or the 23:45 UTC time slot for products with 15 minutes frequency.

The subsequent 0 UTC slot is considered as part of the next day (D+1).

The maximum consecutive gap setting is used to determine how many consecutive data values (time slots) may be missing for a given pixel, 3 hours in the above diagram. It is configured as a percentage of available time slots (which basically comes down to a % of the day), which themselves can have durations of 15min, 30min or hourly depending on the layer used in the compositing. A configuration of 25% would thus allow up to six consecutive hours of missing data, which can thus be 6 time slots of one hour, 12 slots of 30min or 24 slots of 15min.

When the first or last slot (0 UTC and the slot just before 23:59 UTC) are missing, then these are interpolated (not extrapolated) using the "lead" and "trail" data of respectively the preceding and the next day. The amount of lead and trail data is limited to the maximum consecutive gap length.

In the above example timeline:
If the 0 UTC slot has no valid data for the given pixel, the *MSG Toolbox* will search backwards in the data of the preceding day (D-1), for the same pixel and not going beyond the last three hours of that day. In case these last three hours also do not contain a valid value, then the pixel is flagged as "missing data". The effective time period considered is thus from the 21:00 UTC time slot on day D-1 up to 02:59 UTC time slot on day D+1.

b) Example of time period considered for daily compositing with start at 6 UTC:

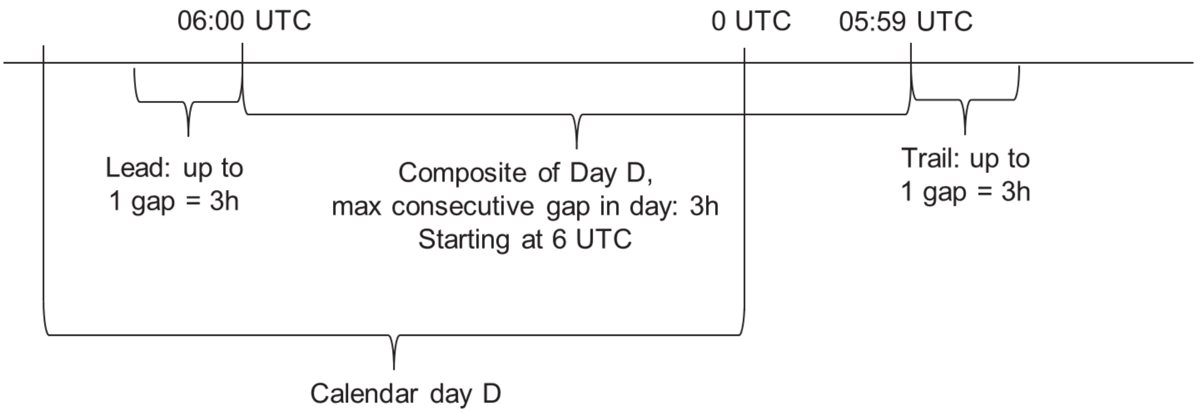


Figure 28 Example relevant time period for daily compositing with start at 6 UTC

When the daily composite starts at 6 UTC, keeping the three hours maximum gap (and thus also the length of the lead and trail data) of the above example, the input data considered effectively ranges from the 3 UTC time slot on day D (included) to the last time slot before 9 UTC (e.g. 08:00 for hourly, 08:30 for 30 minutes, 08:45 for 15 minutes) on day D+1.

Setting the starting time ('delayed start' or 'time shift') is particular useful:

- For composites of specific products such as Land Surface Temperature
- To better capture the diurnal cycles for users in the South American area or with significant time-zone difference with UTC.

c) Example of time period considered for n-day (dekad) compositing:

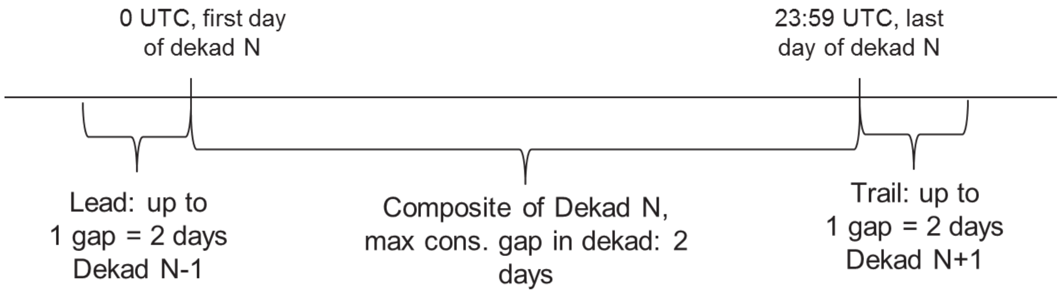


Figure 29 Example relevant time period for dekad compositing

For n-day compositing, such as dekad (10-daily), monthly or yearly compositing, a maximum consecutive gap can also be specified as a number of missing days (not hours). Again, the lead and trail data used for interpolating the missing first and last time slots is limited in amount to one gap. In the above example, all input products are considered within the effective time range from the last two days of dekad N-1 (included) up to the first two days of dekad N+1.

HINTS:

- When producing yearly composites with a one year maximum consecutive gap, the toolbox will effectively search through three (3!) years of input data and may thus produce a lot of "missing input file" warnings when those are absent.
- In near-real time applications, it is normal for the "trail" data to be absent. The toolbox will nevertheless search for it.

5.3.2 Scenario

Using the *Scenario* sub-panel on the *Compositing* tab, the user can define a new processing scenario (*New* button) and edit (*Edit* button) or delete (*Delete* button) a previously saved scenario. Via the drop-down list, any previously saved scenarios can be selected through their name and description.

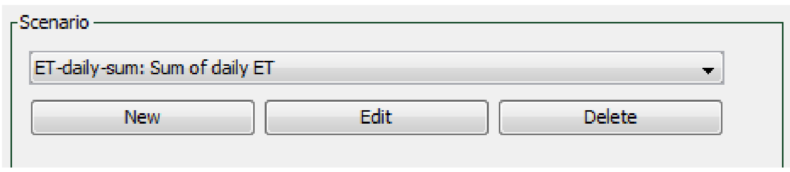


Figure 30 *Scenario* sub-panel on *Compositing* tab. The previously saved scenario with the name "ET-daily-sum" and description "Sum of daily ET" is selected.

When adding a new scenario or editing an existing one, the *scenario settings* or scenario definition window will appear. When adding a new scenario, the name will be changeable. When editing an existing one, the name is greyed out to indicate it cannot be changed, as illustrated in Figure 31.

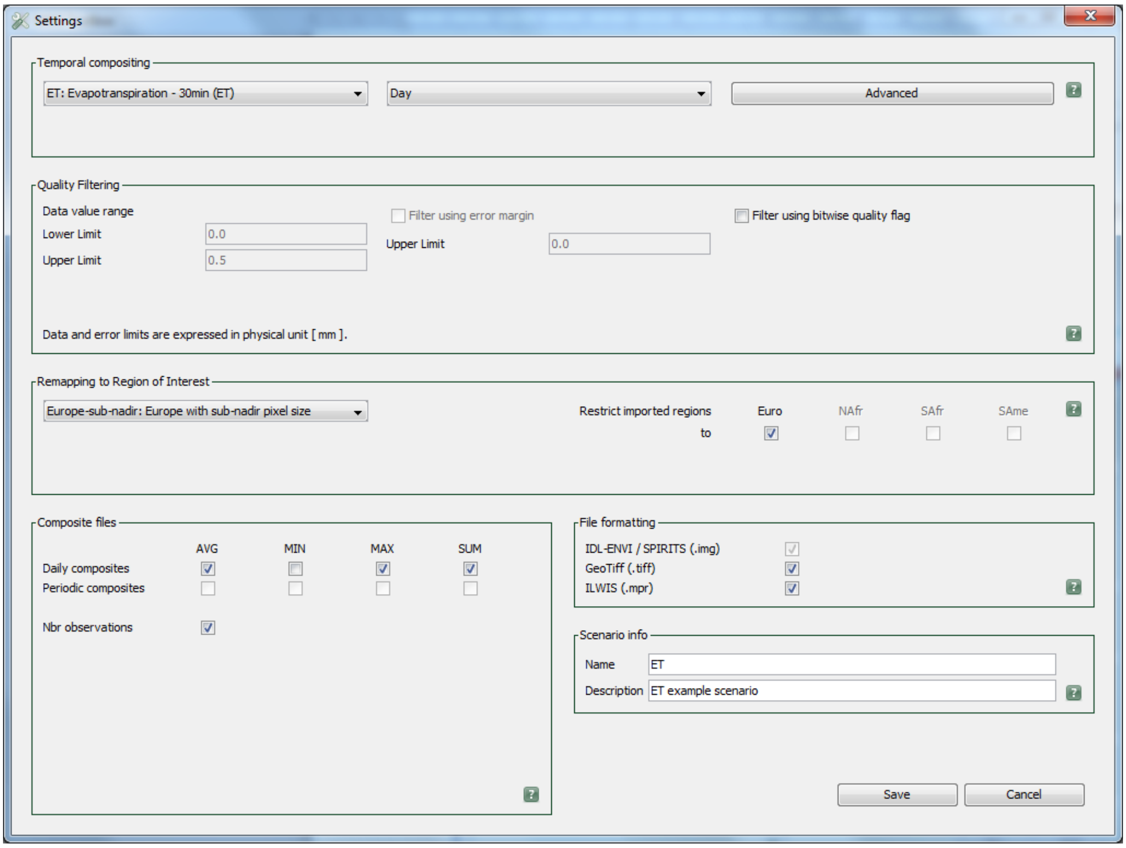


Figure 31 *Scenario settings* window

The *scenario settings* window itself can be split up into several sub-panels, as described in the following sections:

- *Temporal compositing* on top for selecting the layer and composite period,
- *Quality filtering* for optional filtering on error (uncertainty) values and bitwise quality flags,
- *Remapping to Region of Interest* for selecting previously created ROI in the remapping and fine-tune the input regions (NAfr, etc) to be considered;
- *Composite files* to define the compositing rules (minimum, maximum, average, sum) per day or per period and optional number of observations
- *File formatting* to select additional output file formats
- *Scenario info* to give a name and a description to the scenario.
- *Save* and *Cancel* buttons in the bottom-right corner to save the changes or exit the window without saving.

5.3.2.1 Temporal compositing

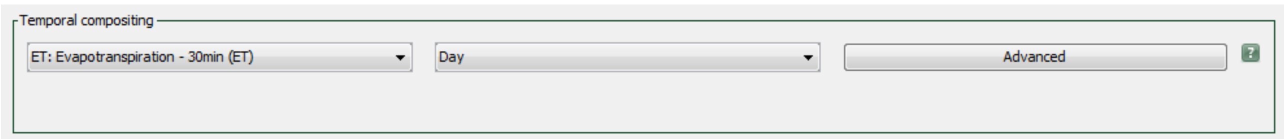


Figure 32 *Temporal compositing* sub-panel of the *Scenario settings* window

On the *Temporal compositing* sub-panel, the user can, from left to right:

- Select the layer to be processed
- Define the composite period type (depending on the selected layer)
- Use the *Advanced* button to open the advanced temporal compositing settings (for experienced users only).

The layer selection is done via a drop-down list. Each entry in the list corresponds to a currently configured and supported layer in a specific input product. The entries are formatted as
<product identifier>: <layer description> (layer identifier)

Whereby

- <product identifier> is a code identifying a specific type of input LSA-SAF product, as it is reflected in the file names (e.g. HDF5_...**FAPAR**_...). See the column "code in the filename" in Table 1 of section 4.3.
- <layer description> is a short description of the main data layer in that product (e.g. for LAI product, the main layer contains the Leaf Area Index values)
- (layer identifier) is the identifier code of the dataset within the HDF5 file that contains the data.

Depending on the selected layer, the *scenario settings* window will be updated as follows:

- The *Quality Filtering* sub-panel will reflect the proper data value range and unit, as well as the presence/absence of an error margin (uncertainty value) layer and bitwise quality layer in the same product;
- For daily frequency input products, the *Day* compositing is disabled (see Figure 33), which in turn affects the content of the *Advanced temporal compositing* dialog and the possibility to calculate daily minima, maximum, average and sum values in the *Composite files* sub-panel.
- For the 30-day synthesis products delivered with 10-daily sliding window intervals (ALBEDO-D10, LAI-D10, FAPAR-D10, FVC-D10), the temporal compositing drop-down selection is fixed to *None* and unusable, the *Advanced temporal compositing* dialog is inaccessible and the selections on the *Composite files* sub-panel are disabled.

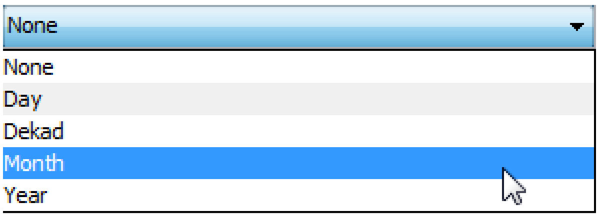


Figure 33 Drop-down selection of composite period, with *Day* option disabled

The second drop-down list selects one of the compositing periods and thus one of the corresponding workflows. It contains the following entries:

- *Day*: for the workflow that produces daily composites from sub-daily inputs.
- *Dekad*: for n-day compositing from sub-daily or daily input products with 10-day (dekadal) compositing windows (days 01-10, 11-20 and 21 to end of month);
- *Month:* for n-day compositing from sub-daily or daily input products with monthly compositing windows (days 01 to end of month);
- *Year:* for n-day compositing from sub-daily or daily input products with yearly compositing windows (01 January to 31 December);
- *None*: to select the workflow without temporal compositing, available to all (sub-daily, daily, 10-daily) inputs.

5.3.2.2 Advanced temporal compositing settings

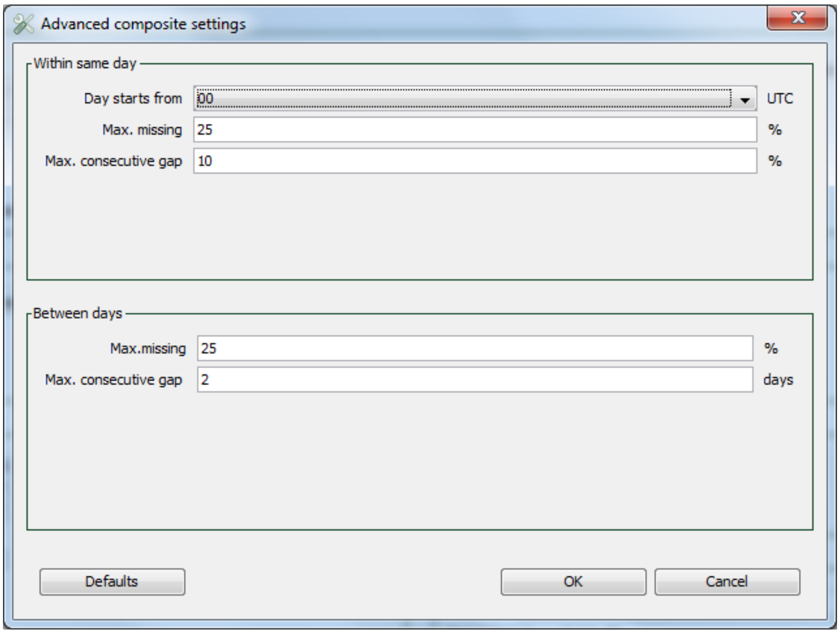


Figure 34 *Advanced temporal compositing* dialog window

If the selected input layer has sub-daily frequency and depending on the selected composite period, the dialog will allow to specify the daily compositing settings (*Within same day* sub-panel at the top), the n-day compositing settings (*Between days* sub-panel, bottom) or both. For instance, for *Dekad* composites of the sub-daily input layer DSSF, all the text fields will be available for updates, as shown in Figure 34. For *Day* composites or a daily frequency input layer such as LAI, the text fields in the “*Within same day”* part will be disabled.

For the daily temporal compositing, the user can specify:

- The start time within the day, which defaults to 0 UTC to 23:59 UTC (see also section 5.3.1).
- The maximum number of allowed missing data values, for any given pixel, expressed as % of available time slots within that day. These data gaps may be spread throughout the day.
- The maximum consecutive gap, also expressed as % of time slots within the day, for which the pixel does not have a valid value.

For the n-day composites, the *Between days* sub-panel likewise allows to define a % of maximum missing days (spread across the n-day period) or the number of consecutive days (not expressed as percentage).

Whenever the % missing or the maximum consecutive gap are exceeded, the corresponding pixel is flagged as ‘missing data’. Smaller gaps are interpolated where needed.

Products that are constructed - by the provider - as daily integrations of other products, notably the daily integrated DSLF/DSSF (DIDSLF and DIDSSF) and daily ET (DMET) products, can contain the following data layers:

- Max_nslots_missing: maximum consecutive gap encountered during production,
- Missing_values_percent: total percentage of missing values encountered during production
- and Weight_missing_values_percent: sum of cosine of the solar zenith angle, thereby giving higher weight to daytime missing values.

These layers are – at present – not considered in the *MSG Toolbox* and therefore unrelated to the above *Advanced temporal compositing* settings.

5.3.2.3 Quality Filtering

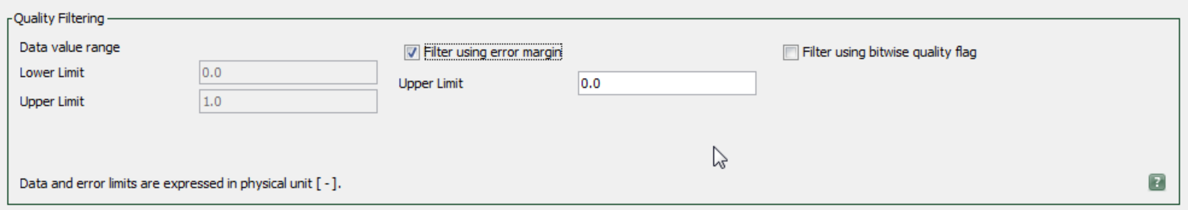


Figure 35 *Quality filtering* sub-panel of the *Scenario settings* window

Early on in the processing workflows, before the main temporal compositing and remapping steps, the main data layer can be optionally filtered on quality. This is threefold:

- The rescaling of the data values themselves imposes a range (lower and upper thresholds);
- A filtering using an upper-limit threshold on the error margin (uncertainty values);
- A filtering using the bitwise quality flag information.

The re-scaling is always performed and the corresponding data value range is shown on the left-hand side, aiming to maintain the data range of the original input products as much as possible. The other filters are optional and subject to the availability of such layers in the input product. The error margin filtering currently requires **an absolute error** value to be provided.

For example, the fAPAR product contains the following three layers:

- “FAPAR” as main data layer;
- “FAPAR err” as layer providing the error margin (uncertainty);
- “FAPAR QF” as the bitwise quality information layer, whereby individual bits indicate if the pixel is land or see, affected by clouds or snow, etc.

As indicated by a small text at the bottom, the data value range and the threshold for the error margin filtering are to be specified in physical values (not digital numbers), expressed in the given unit.

The above figure provides the example of FAPAR, which is dimensionless, as indicated with the [-] unit. As marked in the Product User Manual for the LAI, FAPAR and FVC products (“VEGA suite”), the uncertainty should be limited to 0.2 for FVC and FAPAR, and 1.5 for LAI, beyond which use should be restricted.

Regarding the filtering on the bitwise quality flag, the currently implemented filtering is provided in the following table.

Please note that unprocessed pixels (e.g. marked as Ocean or Space, Algorithm failed) are always filtered, either by the data-range filter or the quality flag filter. When troubleshooting via the logging panel (see 5.4), it is thus important to consider the number of pixels filtered by either filtered as well as the number of pixels converted (remaining) to assess the impact of the filtering.

Product / Layer	Filter removes
MET / ET	Poor quality (decimal values 581, 645, 709, 800)
DMET / ET	None
ALBEDO / any daily & 10-daily, broadband and spectral	Snow
DSSF / DSSF	Cloud filled, cloud contaminated, snow
DIDSSF / DSSF	None
DSLF / DSLF	Cloud filled and cloud undefined/uncertain Algorithm quality below nominal
DIDSLF / DSLF	None
LST / LST	Corrupted satellite image, suspect inputs Snow/ice contaminated Algorithm quality below nominal
FAPAR / FAPAR	Traces of inland water
LAI / LAI	Traces of inland water
FVC / FVC	Traces of inland water
FRM / FWI	Mode 0 / Initialization
FRM / DSR	Mode 0 / Initialization

Table 2 Bitwise quality filtering

For the ALBEDO-D10, FAPAR-D10, LAI-D10 and FVC-D10 products, the same filtering configuration is used as for their daily equivalents.

HINT:

- The configuration of the bitwise quality filtering proved particularly tricky and is thus likely to change in future versions of the software. Make sure to check results carefully.

5.3.2.4 Remapping to Region of Interest

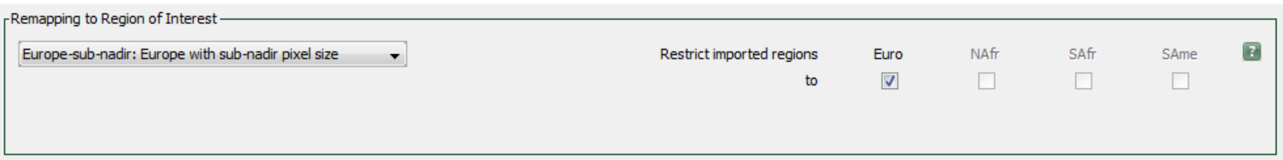


Figure 36 *Remapping to Region of Interest* sub-panel

In the *Remapping to Region of Interest* sub-panel, a simple drop-down list allows to select the *Region of Interest (ROI)* that was previously prepared via the ROI preparation workflow. The entries on the list are formatted as *name : description*, using the name and description provided on the *Regions of Interest* (ROI) tab.

When the ROI is created, it stores the required input regions – NAfr, SAfr, Euro and/or SAme – that could be used to make up the selected bounding box. When a selection is then made in the drop-down, the checkboxes on the right reflect these required regions. The user can then de-select the checkboxes for the regions that were not downloaded, to reduce the number of warnings/errors produced. This is particularly useful for study areas near the borders between regions, considering the remapping.

5.3.2.5 Composite files

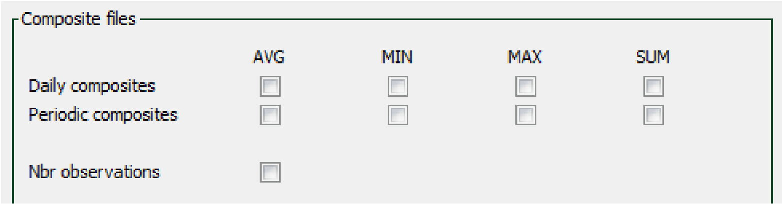


Figure 37 *Composite files* sub-panel in the *Scenario settings* window

In this sub-panel, it can be specified if the toolbox needs to calculate minimum, maximum, average or sum values per day (in the row of checkboxes labelled *daily composites*) or in the n-day compositing period (*periodic composites* row). The checkboxes for the daily composites are disabled in case the selected input layer is not provided with sub-daily (15/30 min, hourly) frequency. All these checkboxes are disabled in case of input products delivered at 10-daily (D10) intervals, or when the compositing periodicity is set to *None* (workflow without temporal compositing).

When checkboxes on both rows are enabled, the toolbox will calculate all possible combinations. For instance, it is possible to compute minimum and maximum daily values, and average those daily minima and maxima across a month.

The *Nbr observations* checkbox allows to output an additional file with the number of observations used (per pixel). For daily composites, this is the number of time slots with usable observation. For periodic composites, the number of days in the period with usable observation.

HINT:
When calculating periodic composites from input products with sub-daily frequency, the daily composites are intermediate results. Therefore, they are stored in the *Workspace* folder and automatically removed unless *Keep Intermediate data* checkbox is enabled (see below). The number of observations used in each daily composite is not provided.

5.3.2.6 File formatting

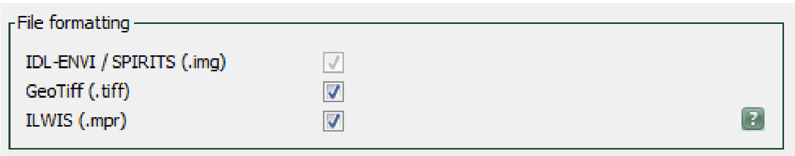


Figure 38 *File formatting* sub-panel in the *Scenario settings* window

In the *File formatting* sub-panel, the user can choose to produce output files in GeoTIFF and ILWIS formats, in addition to the standard IDL-ENVI format. For more information on these formats, please see section 7.

5.3.2.7 Scenario info

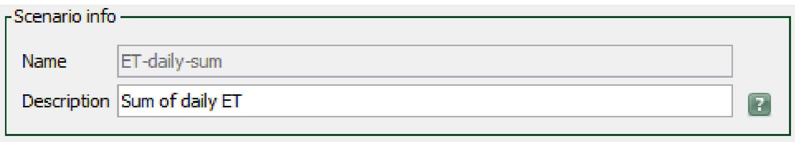
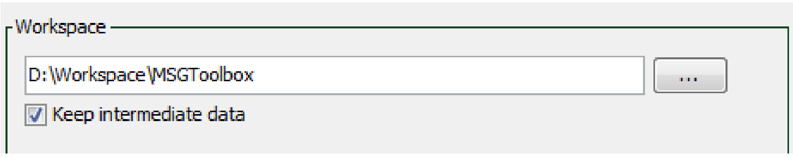


Figure 39 *Scenario Information* sub-panel in the *Scenario settings* window

Using the text field(s) in the *Scenario info* sub-panel, the user can specify a short name and easy-to-read description for the scenario that is being created or edited. When editing an existing scenario, the *Name* cannot be changed.

The provided name will be translated, e.g. removing whitespace, to a name suitable for a file on the hard disk, as the scenario is saved as an XML file. The saved XML scenario files are automatically loaded when the *MSG Toolbox* starts up.

5.3.3 Workspace



The *Workspace* is specified as a top-level folder via the text field or the folder browse button. It represents the folder where all the intermediate results are stored. Those are automatically deleted during the workflow execution, unless the *Keep intermediate data* checkbox is checked. As keeping the intermediate data around is only suitable for troubleshooting, with typically short processing time and limited amount of input data, the user will receive a warning in case this checkbox is checked upon submission of the task.

The files and folders in the workspace are organized by the toolbox, typically in sub-folders per processing step and then further per day. For more details on the organization of the workspace, please refer to Chapter 6 on Troubleshooting.

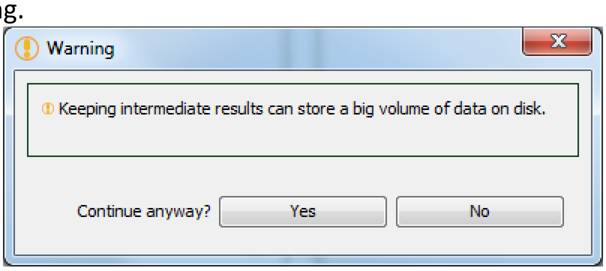


Figure 40 Warning when storing intermediate data files

5.3.4 Output Folder

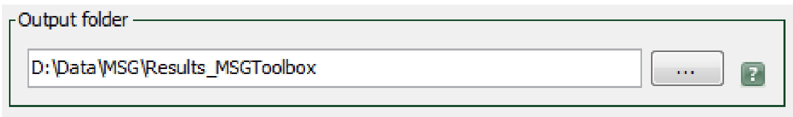


Figure 41 *Output folder* sub-panel on the *Compositing* tab

The *Output folder* sub-panel is used to define the output folder via the text field or folder browse button. In this folder, the output files from the main compositing workflows are stored. The file names vary depending on the workflow performed and its configuration, such as the option to include the format conversion step.

Each produced image holds a single data layer. This corresponds to the main data layer selected in the compositing scenario configuration’s drop-down list, as output by the processing (quality filtered, remapped, etc.). Moreover, the value range is largely kept the same as the input products as much as possible.

Exceptions are

- The additional image for the “number of observations” used, which contains the number of time slots in the day (up to 96 for products with 15min frequency) or number of days in the compositing window (up to 366 for yearly).
- The daily or periodic “sum” composites, which have a larger data range or even a change in data type (e.g. conversion from integer to floating point pixel values).

For the flow performing only remapping and no temporal compositing, the files are named:
var_YYYYMMDDHHmm.extension (for sub-daily inputs)
var_YYYYMMDD.extension (for daily inputs)
Whereby

- **var**: the product identifier code (variable), as listed in Table 1 or in the drop-down list selecting the layer to be composited.
- **YYYYMMDD(HHmm)**: the nominal date and/or time, as copied from the input file.
- Extension: file name extension depending on the format. For ENVI format, a binary data file (.img) and a text header file (.hdr) are provided.

Example: FAPAR_20130722.img

For the workflow that produces daily composites from sub-daily frequency input products:
var_YYYYMMDD_**dct**.extension
Whereby the additional **dct** signifies the type of daily composite and is one of *avg*, *min*, *max, sum* or *cnt* (for number of observations).
Example: DSLF_20130401_avg.img

For the workflows producing periodic (n-day) composites, the file names are composed as follows:
var_per_pct_YYYYMMDD for daily frequency input files
var_per_pct_YYYYMMDD_**dct** for sub-daily frequency input files
Whereby

- **per:** composite period type, which is one of *Dekad*, *Month*, or *Year*;
- **pct:** periodic (n-day) composite type, one of *avg*, *min*, *max*, *sum* or *cnt* for number of observations.

Example: the file DSSF_Dekad_avg_20130721_max.img is the 10-daily average of the daily maximum DSSF values.

Output files in GeoTIFF format are additionally prefixed with *GTIFF_* and have the *.tiff* as filename extension. ILWIS output files are all prefixed with *ILWIS_* and have their standard extensions (*.mpr, .grf, .csy, .mp#*).

When in doubt, the user can check the task logging messages for the output file naming, as in the below example for a dekad processing step.

```
Task Id  : 103
Task name: Composite of daily max for Dekad 2013-07-21.

2015-06-16 13:43:35 START
   INFO: Composite start date: 201307210000
   INFO: Composite end date  : 201307310000
   INFO: Max missing values  : 3
   INFO: Max allowed gap size: 2
   INFO: AVG output file     : D:\Data\MSG\Demo\Output\DSSF_Dekad_avg_20130721_max
   INFO: MAX output file     : D:\Data\MSG\Demo\Output\DSSF_Dekad_max_20130721_max
   INFO: SUM output file     : D:\Data\MSG\Demo\Output\DSSF_Dekad_sum_20130721_max
```

Figure 42 Example of logging messages showing the output files produced by dekad compositing

For the further interpretation of the files produced, for instance reading the ENVI .hdr file to discover the characteristics (data range, flagged values, etc) of the produced files, please see Chapter 7.

5.4 Monitoring processing tasks and progress

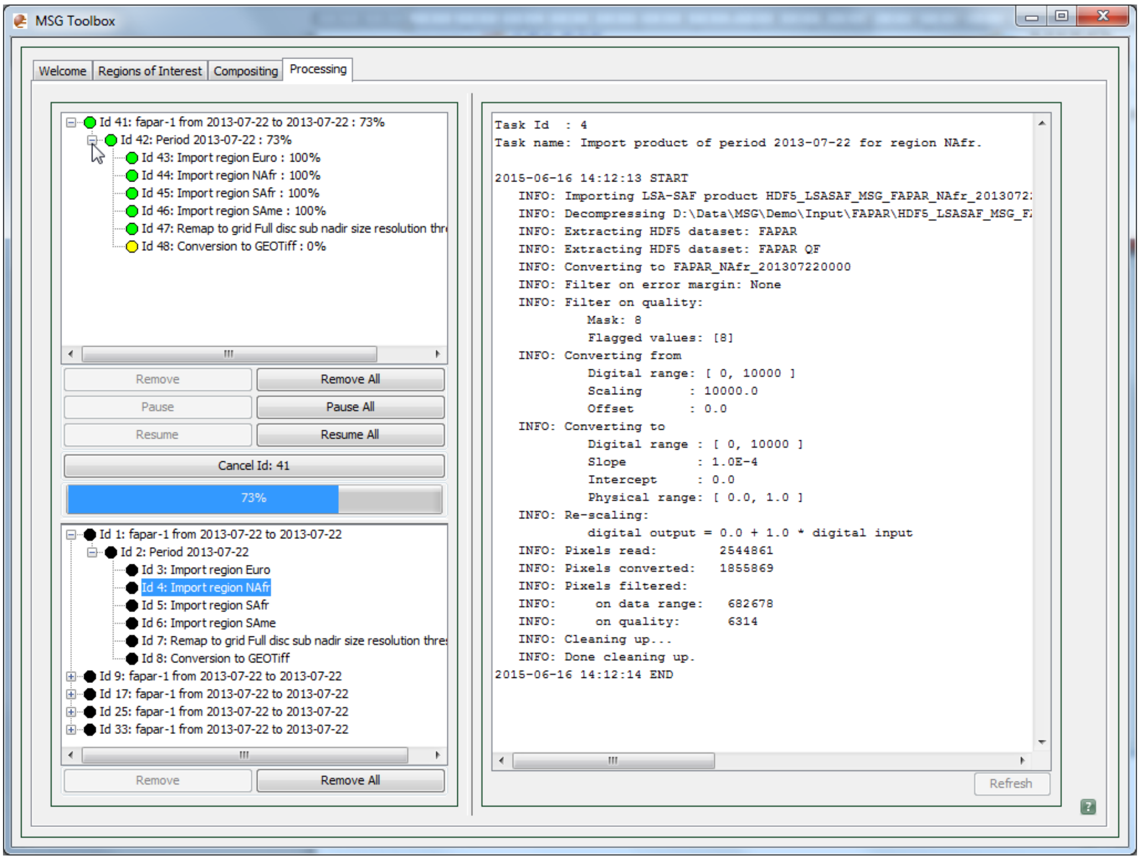


Figure 43 Queue and progress of submitted tasks on the *Processing* tab

After clicking on the *Add task* or *Add paused task* buttons on the *Compositing* tab, a task is added to a waiting list, called the task queue. This queue is depicted as a hierarchy on the top-left of the *Processing* tab, with usual tree navigation (+/- buttons). Lower levels in the tree are for sub-tasks (child tasks). Green circles for successful tasks, orange ones for tasks with warnings, red for tasks with errors, yellow for tasks waiting to be started and white for paused tasks. Warnings and errors are escalated to higher-level (parent or ancestor) tasks, up to the top-level. Even when the majority of sub-tasks was successful, a single warning or error will hence make the parent task get warning or error status (orange or red circle). Error status takes precedence over warning status when both occur.

The buttons just below allow the user to remove tasks, pause them and resume then when they are paused. The *Remove*, *Pause* and *Resume* buttons each affect only a single, selected top-level task (or the currently active one if none are selected). The *Remove All*, *Pause All* and *Resume All* affect all top-level tasks in the queue. The *Cancel* button, sometimes with a reference to the Task Id (number) of the currently selected task, will cancel this task.

These buttons are followed by a blue progress bar, indicating task completion in % of the currently selected, top-level task.

HINT:
The toolbox will wait for a running task (or sub-task) to finish before cancelling, removing or otherwise disrupting it.

When the task is completed, it moves to the bottom left sub-panel that provides a similar view. The attached *Remove* and *Remove All* buttons can then remove those finished tasks that are no longer of interest. The colours of the circles indicate the status: black means finished successfully, red means failed, orange is for warnings. A cross over the circle indicates a cancelled task.

Both for the on-going tasks selected from the queue (top-left) as well as for the finished ones (bottom-left), the right panel will provide detailed messages on progress. These are particularly useful for troubleshooting and can be easily copy-pasted, via the Clipboard, to a text file or emailed when asking for support.

6 Troubleshooting

Beyond the regular monitoring of the progress and status of processing tasks (see section 5.4), whereby the detailed information messages may be copied and emailed for support, the more experienced user can also investigate the problem in more detail.

This is done by:

- Enabling the storage of intermediate results via the *Keep intermediate data* checkbox on the *Compositing* tab (see section 5.3.3);
- Starting compositing tasks, preferably with a small set of input data (e.g. a short time period) so they can be easily repeated with minor changes;
- Visualizing the intermediate data to troubleshoot the workflow step-by-step.

For this visualization, it is of course important to first have a basic understanding of the workflows, the various processing steps involved and the organization of the intermediate data in the *Workspace*.

6.1 File organization in the workspace

Intermediate results of each processing step (except the last one) are temporarily written to the specific folders, called the workspace. They are cleaned up automatically, unless the user specifically opts to keep them. To this end, the user configures a single top-level workspace folder. Within this top-level folder, the toolbox determines the names of the intermediate files and organizes them into **subfolders per variable and processing step** that yields intermediate output data in order to:

- limit the number of files stored in each folder, for performance reasons;
- and to facilitate experienced users in quickly locate the intermediate files for inspection.

When working with **sub-daily** frequency input LSA-SAF products, the **import** sub-flow in particular works with high amounts of files, up to hundreds per day. For this reason, the related import folder in the workspace is further **sub-divided per day**.

The **daily** images that result from import of daily LSA-SAF products, daily compositing of sub-daily inputs and the remapping are **organized per year**.

The **10-day and monthly** composites are organized **per year**, whereas the **yearly** ones are simply stored **in one folder**.

This is illustrated in the below figure.

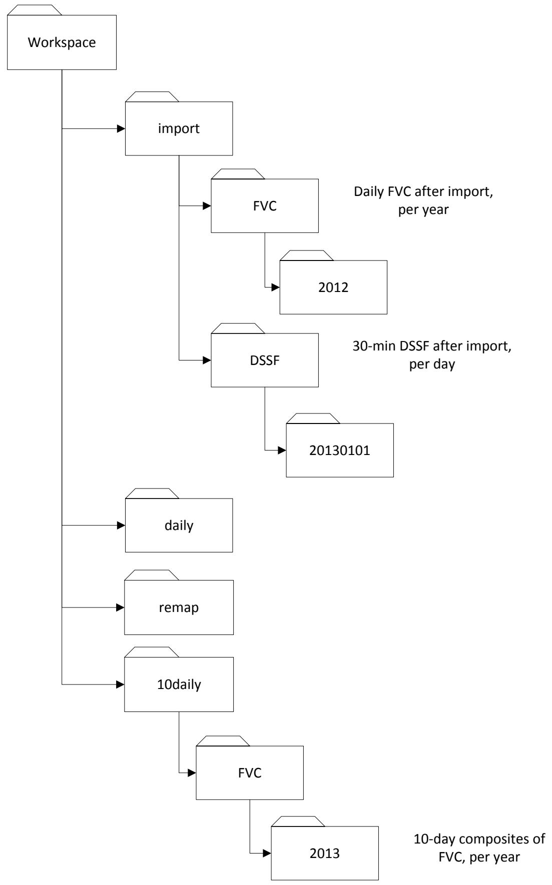


Figure 44 Example of the workspace file and folder organization

6.1.1 Import sub-flow

Input files with a .bz2 file name extension are considered to be bzip-2 compressed product archives containing an hdf5 file. These files will be decompressed, and the resulting files will be placed in the *Import* sub-folder of the Workspace folder. The following naming convention will be used:

input file	decompressed file
HDF5_LSAF_MSG_**var_reg**_YYYYMMDDhhmm.bz2	**var_reg**_YYYYMMDDhhmm.h5

Whereby:
var: variable or product identifier code in the filename (ALBEDO, FARAR,…)
reg: region (one of Euro, NAfr,SAfr, SAme)

Example: input file: HDF5_LSASAF_MSG_FAPAR_NAfr_201206240000.bz2
decompressed file: FAPAR_NAfr_201206240000.h5

Input files with an .h5 extension or no extension at all, are considered to be HDF5 data files that were already decompressed prior to ingestion in the *MSG Toolbox*.

In case the user selected the option to keep the intermediate data, these files will be copied from the input data folder to the *Import* sub-folder of the workspace folder, with the following naming convention:

input file	copied file
HDF5_LSAF_MSG_**var_reg**_YYYYMMDDhhmm	**var_reg**_YYYYMMDDhhmm.h5
HDF5_LSAF_MSG_**var_reg**_YYYYMMDDhhmm.h5	**var_reg**_YYYYMMDDhhmm.h5

Var, reg: same as above

Example: input file: data folder/HDF5_LSASAF_MSG_FAPAR_SAfr_201206240000
copied file: workspace folder/import/…/FAPAR_SAfr_201206240000.h5

Following the above, optional decompression and file copy, the the main data layer ('product layer'), and if applicable, the 'error margin' and 'bitwise quality flags' layers (HDF5 datasets) are extracted from the HDF5 file and converted to the internal ENVI-formatted files. These files will be placed in the *Import* sub-folder of the workspace folder and use the following naming convention:

input file	dataset	extracted dataset files
HDF5_LSAF_MSG_**var_reg**_YYYYMMDDhhmm or HDF5_LSAF_MSG_**var_reg**_YYYYMMDDhhmm.h5 or **var_reg**_YYYYMMDDhhmm.h5	product	**var_reg**_YYYYMMDDhhmm_**prod**.img/hdr
	error	**var_reg**_YYYYMMDDhhmm_**err**.img/hdr
	quality flags	**var_reg**_YYYYMMDDhhmm_**qflags**.img/hdr

Whereby
var: variable or product identifier code in the filename (ALBEDO, FARAR,…)
reg: region (one of Euro, NAfr,SAfr, SAme)
prod: HDF5 dataset name for the main data layer,
err: HDF5 dataset name for the error margin layer,
qflags: HDF5 dataset name for the bitwise quality information layer

Example: input file: workspace folder/import/…/ALBEDO_Euro_201310010000.h5
extracted files: workspace folder/import/…/ALBEDO_Euro_201310010000_AL-BB-BH.img/hdr
workspace folder/import/…/ALBEDO_Euro_201310010000_AL-BB-BH-ERR.img/hdr
workspace folder/import/…/ALBEDO_Euro_201310010000_Q-Flag.img/hdr

Remarks:

- The data type of the ENVI files will be selected automatically to match that of the extracted dataset as close as possible.
- ENVI does not support all HDF5 data types. Example: HDF5 supports a signed byte type, ENVI does not, a 16-bit integer will be used instead (ENVI data type 2).
- Not all HDF5 and/or ENVI types are implemented. Following ENVI types are supported:
    - 1=8 bit byte
    - 2=16-bit signed integer
    - 3=32-bit signed long integer
    - 4=32-bit floating point
    - 5=64-bit double precision floating point
    - 12=16-bit unsigned integer
    - 13=32-bit unsigned long integer
    - 14=64-bit signed long integer

The extracted 'main data layer', and if applicable, the 'error margin' and 'bitwise quality flags' ENVI files are combined into a single 'imported' file. These files will be placed in the *Import* sub-folder of the workspace folder and use the following naming convention:

input file	imported (filtered and rescaled) files
var_reg_YYYYMMDDhhmm_**prod**.img/hdr and (if applicable) **var_reg**_YYYYMMDDhhmm_**err**.img/hdr and (if applicable) **var_reg**_YYYYMMDDhhmm_**qflags**.img/hdr	**var_reg**_YYYYMMDDhhmm.img/hdr

Whereby
var: variable or product identifier code in the filename (ALBEDO, FARAR,…)
reg: region (one of Euro, NAfr,SAfr, SAme)
prod: HDF5 dataset name for the main data layer,
err: HDF5 dataset name for the error margin layer,
qflags: HDF5 dataset name for the bitwise quality information layer

Example: input files: workspace folder/import/…/ FAPAR_Euro_201201020000_FAPAR.img/hdr
workspace folder/import/…/ FAPAR_Euro_201201020000_FAPAR QF.img/hdr
imported file: workspace folder/import/…/ FAPAR_Euro_201201020000.img/hdr

Remarks:

- The data type, scaling and the flag values of the imported files is determined by layer specified in the selected scenario.

6.1.2 Daily compositing step

In case of sub-daily input products in the daily or n-day composites processing flow, the imported (sub-daily) files are composited into daily files. These files will be placed in the *Daily* sub-folder of the workspace folder. Following naming convention will be used:

input files	daily composite files
var_reg_YYYYMMDDhhmm.img/hdr (collection of all files 'DDhhmm', relevant for the day to be composited)	**var_reg**_YYYYMMDD_**dct**.img/hdr

Whereby
var: variable or product identifier code in the filename (ALBEDO, FARAR,…)
reg: region (one of Euro, NAfr,SAfr, SAme)
dct: daily composite type (avg, min, max, sum). Note that the number of observations (cnt) is not provided.

Example: input files: workspace folder/import/…/ DSSF_Euro_201201312200.img/hdr
workspace folder/import/…/ DSSF_Euro_201201312230.img/hdr
workspace folder/import/…/ DSSF_Euro_201201312300.img/hdr
'heads' from previous day workspace folder/import/…/ DSSF_Euro_201201312330.img/hdr

workspace folder/import/…/ DSSF_Euro_201202010000.img/hdr
workspace folder/import/…/ DSSF_Euro_201202010030.img/hdr

workspace folder/import/…/ DSSF_Euro_201202010130.img/hdr
…
workspace folder/import/…/ DSSF_Euro_201202012330.img/hdr

workspace folder/import/…/ DSSF_Euro_201202020000.img/hdr
'tails' from next day
workspace folder/import/…/ DSSF_Euro_201202020030.img/hdr
workspace folder/import/…/ DSSF_Euro_201202020100.img/hdr
workspace folder/import/…/ DSSF_Euro_201202020130.img/hdr

daily composites:
workspace folder/daily/…/DSSF_Euro_20120201_avg.img/hdr
workspace folder/daily/…/DSSF_Euro_20120201_avg.img/hdr
workspace folder/daily/…/DSSF_Euro_20120201_min.img/hdr
workspace folder/daily/…/DSSF_Euro_20120201_min.img/hdr
workspace folder/daily/…/DSSF_Euro_20120201_max.img/hdr
workspace folder/daily/…/DSSF_Euro_20120201_max.img/hdr

Remarks:

- The 'first' file of a day (hhmm), and the 'heads' and 'tails' range is determined by the start of the day and the maximum consecutive gap settings, specified in the selected scenario.

6.1.3 Remapping step

The imported files for the different regions (Euro, NAfr, SAfr, SAme) are remapped to geographic lat/lon, and combined according to the grid specified in the selected scenario. Following naming convention will be used:

input files		remapped files
daily products	**var**_Euro_YYYYMMDDhhmm.img/hdr and/or **var**_Nafr_YYYYMMDDhhmm.img/hdr and/or **var**_SAfr_YYYYMMDDhhmm.img/hdr and/or **var**_SAme_YYYYMMDDhhmm.img/hdr	**var**_ YYYYMMDD.img/hdr
sub-daily products in the 'no composite' flow	**var**_Euro_YYYYMMDDhhmm.img/hdr and/or **var**_Nafr_YYYYMMDDhhmm.img/hdr and/or **var**_SAfr_YYYYMMDDhhmm.img/hdr and/or **var**_SAme_YYYYMMDDhhmm.img/hdr	**var**_ YYYYMMDDhhmm.img/hdr
composited sub-daily products	**var**_Euro_YYYYMMDD_**dct**.img/hdr and/or **var**_NAfr_YYYYMMDD_**dct**.img/hdr and/or **var**_SAfr_YYYYMMDD_**dct**.img/hdr and/or **var**_SAme_YYYYMMDD_**dct**.img/hdr	**var**_ YYYYMMDD_**dct**.img/hdr

Whereby
var: variable or product identifier code in the filename (ALBEDO, FARAR,…)
reg: region (one of Euro, NAfr,SAfr, SAme)
dct: daily composite type (avg, min, max, sum)

Examples:	sub-daily product, 'no composite' flow, grid using Euro, NAfr and SAfr
input files:	DSSF_Euro_201202010130.img/hdr
	DSSF_NAfr_201202010130.img/hdr
	DSSF_SAfr_201202010130.img/hdr
remapped file:	DSSF_201202010130.img/hdr
input files:	sub-daily product, 'daily composites' flow, , grid using Euro, NAfr and SAfr
	DSSF_Euro_20120201_avg.img/hdr
	DSSF_NAfr_20120201_avg.img/hdr
	DSSF_SAfr_20120201_avg.img/hdr
remapped file:	DSSF_20120201_avg.img/hdr

Remarks:

- In the cases 'no composite flow' and 'daily composite from sub-daily products', where no additional format conversion step is specified, the remapped files will be placed in the output folder. In all other cases, the remapped files will be placed in the "remap" sub-folder of the workspace folder.

6.2 Viewing intermediate results

The intermediate results are all stored in an ENVI file format with some extensions to the text header (.hdr) to make it compatible with the Glimpse and SPIRITS software.

To visualize them, it suffices to download the free SPIRITS software from http://spirits.jrc.ec.europa.eu web site that also offers a tutorial and manual to get started.

Using the menu *Analysis* – *Maps*, it is possible to create a single quicklook image (visualization with colours, legends, titles, logos and so on). When satisfied, the quicklook template can be saved and re-used to create maps for larger time series of files (e.g. all intermediate results of different days).

7 Output formats

This chapter briefly describes the various output formats that are provided (extended IDL-ENVI format, GeoTIFF, ILWIS) and their usage.

7.1 Default IDL-ENVI format

The default output format is compatible to IDL-ENVI's standard file format, that consists of a flat binary data file (.img) and a plain-text header file (.hdr). For further exploitation in Glimpse and SPIRITS software, developed by VITO for the European Commission's Joint Research Centre, some additional information has been added to the header.

For more details, please see the User Manual (extensive description) and/or the Tutorial (summary description) of the SPIRITS software and ENVI header format, available on http://spirits.jrc.ec.europa.eu. The IDL-ENVI web site, http://www.exelisvis.com/docs/ENVIHeaderFiles.html, describes the standard IDL-ENVI format, without the Glimpse/SPIRITS extensions.

Note that the *MSG Toolbox*, and hence this manual, does not use all aspects of the format. For instance, the classification keywords (class names etc.) are beyond the scope of the *MSG Toolbox* and the map info is constrained to the simple Geographic Lat/Lon spatial reference system.

Key information in the header file includes:

- The data type
- The number of lines and samples (bands is always 1)
- The mapping information (magic point, pixel size)
- The Values tag that details the variable, physical unit, value ranges and slope/intercept for the conversion from digital to physical values.
- The Flags tag that indicates pixels which contain specific values for e.g. missing data

Sample code, written in ANSI-C and Fortran programming languages, for reading the images is provided in Annex 2. The sample code in C in addition parses the textual header information (.hdr file). Sample code in the object-oriented Java programming language can be provided on request.

7.2 Additional GeoTIFF and ILWIS formats

In addition to the default IDL-ENVI output format, the *MSG Toolbox* can provide copies of the output files, converted into the GeoTIFF and/or ILWIS formats using the standard gdal_translate utility of GDAL.

GeoTIFF is an extension of the standard TIFF image format, adding in the geo-information (geolocation, spatial reference system, etc) in the form of metadata tags or attributes. GeoTIFF format is widely used in GIS and remote sensing software. The toolbox adds the filename extension *.tiff* and filename prefix *GTIFF_* to the GeoTIFF-formatted output files it produces.

For example, the output file GTIFF_DSSF_20130408_avg.tiff is a daily composite of the shortwave flux DSSF, containing the average value for the 8th of April 2013. It is the GeoTIFF equivalent of the DSSF_20130408_avg.img (and .hdr) files in IDL-ENVI format.

The Integrated Land and Water Information System or ILWIS, is a GIS and remote sensing software that was initially developed by ITC, currently a faculty of the University of Twente. Since a few years, it became *ILWIS Open*, an open-source software driven by 52North's ILWIS community, with popular extensions on data access (incl. LSA-SAF product support) developed in the EO community.

Output files in ILWIS format are actually composed of a group of files, varying only by the filename extension. For example, the file ILWIS_DSSF_20130401_max.*mpr* is the main file. This small text file contains references to the other, related files: ILWIS_DSSF_20130401_max.*mp#* (the binary file containing the actual pixel values) and ILWIS_DSSF_20130401_max.grf (describing the geographic mapping), which in turn further refers to the file ILWIS_DSSF_20130401_max.csy (describing the spatial reference system).

Annexes

Annex 1. Example configuration for the GEONETCast Data Manager software

Save the below configuration in a plain text file, e.g. DataManager_LSASAF.txt and select it when running the Data Manager, either interactively, or by adding it as command-line parameter (automatic start).

```
# Note:
# This file will be automatically overwritten when you run the Data Manager!
# You can make edits to this file when the program is not running,
# but it is no use to change the layout of this file or add your own comments.
# Lines starting with # are comments and default values (reset by the software).
# Uncomment them (delete #) to make them take effect.

Title: Data Manager sample configuration for LSA-SAF
Source folder:      \\pc*******\received
Unmatched files folder:    \\pc*******\received\unmatched
# Copy files:no
# Autostart delay:  10
Columns:     3

Group Name:  LSA-SAF_FTP
Description: LSA-SAF via FTP
# Date position:    46
# File id position: 36
Destination folder: \\myServer\LSA-SAF
Dated folders:      no
Missing data log:   \\myServer\LSA-SAF\missing\missing-euro.log
Duration of storage:       ONE_WEEK

Item Name:   MSG-HDF5
Pattern:     *HDF5_LSASAF_MSG*
# Process:   yes
Times per day:      8
# Times to store:   all
Expected segments:  1
# Segments to store:       all

Item Name:   EPS-MetOp
Pattern:     *HDF5_LSASAF_EPS*
# Process:   yes
Times per day:      8
# Times to store:   all
Expected segments:  1
# Segments to store:       all

Item Name:   MSG-ASCII
Pattern:     *ASCII_LSASAF_MSG*
# Process:   yes
Times per day:      8
# Times to store:   all
Expected segments:  1
# Segments to store:       all
```

```
Group Name:  LSA-SAF_GNC_Europe
Description: LSA-SAF Euro region on SAF-Europe channel
# Date position:    46
# File id position: 36
Destination folder: \\myServer\LSA-SAF\EURO
Dated folders:      no
Missing data log:   \\myServer\LSA-SAF\missing\missing-euro.log
Duration of storage:      ONE_WEEK

Item Name:    MSG-ET_Euro
Pattern:      *S-LSA_-HDF5_LSASAF_MSG_ET_Euro_*
# Process:    yes
Times per day:      8
# Times to store:   all
Expected segments:  1
# Segments to store:      all

Item Name:    MSG-DSLF_Euro
Pattern:      *S-LSA_-HDF5_LSASAF_MSG_DSLF_Euro_*
# Process:    yes
Times per day:      8
# Times to store:   all
Expected segments:  1
# Segments to store:      all

Item Name:    MSG-LST_Euro
Pattern:      *S-LSA_-HDF5_LSASAF_MSG_LST_Euro_*
# Process:    yes
Times per day:      8
# Times to store:   all
Expected segments:  1
# Segments to store:      all

Group Name:  LSA-SAF_GNC_Americas
Description: LSA-SAF SAme region on SAF-Americas channel
# Date position:    46
# File id position: 36
Destination folder: \\myServer\LSA-SAF\SAME
Dated folders:      no
Missing data log:   \\myServer\LSA-SAF\missing\missing-same.log
Duration of storage:      ONE_WEEK

Item Name:    MSG-ET_SAme
Pattern:      *S-LSA_-HDF5_LSASAF_MSG_ET_SAme_*
# Process:    yes
Times per day:      8
# Times to store:   all
Expected segments:  1
# Segments to store:      all

Item Name:    MSG-DSLF_SAme
Pattern:      *S-LSA_-HDF5_LSASAF_MSG_DSLF_SAme_*
# Process:    yes
Times per day:      8
# Times to store:   all
Expected segments:  1
# Segments to store:      all

Item Name:    MSG-LST_SAme
Pattern:      *S-LSA_-HDF5_LSASAF_MSG_LST_SAme_*
# Process:    yes
Times per day:      8
# Times to store:   all
Expected segments:  1
# Segments to store:      all
```

```
Group Name:  LSA-SAF_GNC_Africa
Description: LSA-SAF NAfr and SAfr regions on SAF-Africa channel
# Date position:    46
# File id position: 36
Destination folder: \\myServer\LSA-SAF\NAfr
Dated folders:      no
Missing data log:   \\myServer\LSA-SAF\missing\missing-nafr.log
Duration of storage:       ONE_WEEK

Item Name:   MSG-ET_NAfr
Pattern:     *S-LSA_-HDF5_LSASAF_MSG_ET_NAfr_*
# Process:   yes
Times per day:      8
# Times to store:   all
Expected segments:  1
# Segments to store:       all

Item Name:   MSG-ET_SAfr
Pattern:     *S-LSA_-HDF5_LSASAF_MSG_ET_SAfr_*
# Process:   yes
Times per day:      8
# Times to store:   all
Expected segments:  1
# Segments to store:       all

Item Name:   MSG-DSLF_NAfr
Pattern:     *S-LSA_-HDF5_LSASAF_MSG_DSLF_NAfr_*
# Process:   yes
Times per day:      8
# Times to store:   all
Expected segments:  1
# Segments to store:       all

Item Name:   MSG-DSLF_SAfr
Pattern:     *S-LSA_-HDF5_LSASAF_MSG_DSLF_SAfr_*
# Process:   yes
Times per day:      8
# Times to store:   all
Expected segments:  1
# Segments to store:       all

Item Name:   MSG-LST_NAfr
Pattern:     *S-LSA_-HDF5_LSASAF_MSG_LST_NAfr_*
# Process:   yes
Times per day:      8
# Times to store:   all
Expected segments:  1
# Segments to store:       all

Item Name:   MSG-LST_SAfr
Pattern:     *S-LSA_-HDF5_LSASAF_MSG_LST_SAfr_*
# Process:   yes
Times per day:      8
# Times to store:   all
Expected segments:  1
# Segments to store:       all

Group Name:  LSA-SAF_GNC_Global
Description: LSA-SAF Full disc on SAF-Global channel
# Date position:    46
# File id position: 36
Destination folder: \\myServer\LSA-SAF\Global
Dated folders:      no
Missing data log:   \\myServer\LSA-SAF\missing\missing-global.log
Duration of storage:       ONE_WEEK
```

```
Item Name:    MSG-FRP-GRID
Pattern:      *S-LSA_-HDF5_LSASAF_MSG_FTA-FRP-GRID*
# Process:    yes
Times per day:      8
# Times to store:   all
Expected segments:  1
# Segments to store:       all
```

Annex 2. Sample source code for reading outputs in IDL-ENVI format

These below source code samples are provided for illustrative purposes only, without guarantee or even compilation or testing.

In regular C programming language

In the below code, ERROR (str1, str2, exit_value) function prints an error message composed of str1 (function name), str2 (actual error message) and stops the program with exit_value.

```
#include <stdio.h>
#include <stdlib.h>
#include <errno.h>
#include <math.h>
#include <float.h>
#include <string.h>
#include <ctype.h>
#include <time.h>

#include <io.h>
#include <sys\types.h>
#include <sys\stat.h>
// or include <curses.h>, <sys/types.h> and <sys/stat.h> for UNIX/LINUX

#if !defined _MAX_PATH
	#define 	_MAX_PATH 260 // max length for windows path names
#endif

// for UNIX/Linux equivalent string functions:
// #define stricmp   strcasecmp                                  // unknown by UNIX
// #define strnicmp  strncasecmp
#define HDRlmax 10000// max length of ENVI header file lines

/* Structures */

struct MAPINFO
{
	char    name[101];               // Entry in ENVI-file map_proj.txt
	short   zone;                    // only for UTM
	char    ns[11];                  // only for UTM
	double  mc, mr;                  // magic pixel: column/record
	double  mx, my;                  // magic pixel: X/Y
	double  dx, dy;                  // pixel X/Y-size (resolution)
	double  xmin, xmax;              // X/Y of absolute edges of image
	double  ymin, ymax;
	char    datum[101];              // optional
	char    units[21];               // optional ("degrees" or "meters")
};

struct ENVIHDR
{
	char    description[257];
	char    file_type[101];          // 'ENVI standard' or 'ENVI classification'
	long    offset;                  // Byte-offset in data file
	long    samples;                 // Nbr pixels (columns) per line
	long    lines;                   // Nbr lines
	long    bands;                   // Nbr of layers (typically 1)
	short   interleave;              // For multi-layer images: 1=BSQ, 2=BIL, 3=BIP
	short   data_type;               // 1=byte, 2=short, 3=long, 4=single prec float
	short   bpp;                     // Bytes-per-pixel, respectively
	                                 // 1/2/4/4 for data_type = 1/2/3/4
	short   byte_order;              // 0=Big-Endian, 1=Little-Endian
	short   flip;                    // Flip byte-order needed for actual system
	__int64 pixels;                  // Total pixels in image = samples * lines
	struct MAPINFO mi;               // Regular Map info
	struct MAPINFO mit;              // Map info for Pseudo-Images
	                                 // (not used in MSG Toolbox)
	char    Vname[201];              // Meaning of physical value Y
```

```
        char     Vunit[51];                  // Units of Y
        double   Vlo;                        // Vlo/hi: range of significant digital values
        double   Vhi;
        double   Vmin;                       // Vmin/max: actual limits as observed in image
        double   Vmax;                       // Note : Vlo <= Vmin <= Vmax <= Vhi
        double   Vint;                       // Linear relation: Y = Vint + Vslo * V
        double   Vslo;                       // where V = digital value, Y = physical value
        long     date;                       // YYYYMMDD = Image registration date, or
                                             // startdate for composites/syntheses
        Short    days;                       // Periodicity in days: 1,7,10,30,60,90,180,360
                                             // -1(momentaneous),0=unknown/irrelevant
        char     flags[257];                 // Eg. flags={254=out, 255=sea}
        char     comment[257];               // Optional comment string
        char     program[101];               // Name and version number of creating program
        char     sensor[51];                 // Sensor name/type (optional)
        char     projection[257];            // Projection info (optional)
        char     coordsys[1001];             // Coordinate System String (optional)

        // only for file_type = 'ENVI classification' (not used in MSG Toolbox)
        short    classes;                    // Nbr. of classes (consecutive, starting at 0)
        char     **cnames;                   // Class names
        unsigned char   **colors;            // Per class: RGB-colors
};

/*****************************************************************************
Routine to read relevant part of plain-text header files (.hdr) and store the parsed information
into a ENVIHDR memory structure
Parameters:
    *img: pointer to the name of the image file for which the corresponding header is retrieved and
        parsed
    *h: pointer to header structure
    tis: non-zero checks the image's binary data size (nbr of bytes) with the expected value from
        the header information (e.g. number of lines x number of columns x number of bytes per
        pixel)
    d3_ok: allow multi-layer file (e.g. 3-band RGB) or require single-layer
    DT_min, DT_max: range of ENVI data types that are supported
    offset_ok: require header offset of 0 or not
    flip_ok: allow byteswap or not
*****************************************************************************/
void envi_hdr_read(char *img, struct ENVIHDR *h, short tis, short d3_ok, short DT_min, short DT_max,
short offset_ok, short flip_ok)
{
        Char          hdr[_MAX_PATH], str[HDRlmax+1], s[501], fun[25], *item, m[2];
        short         yyyy,  yy,  mm,  dd,  dd_st,  ttm,  tty,  dpm,  doy;
        short         i, j, k, multi;
        double        f, f1, f2;
        FILE          *fp;

        strcpy(fun, "ENVI_HDR_READ");

        // Retrieve name of header file from related image file's name.
        img_name(img, hdr, 2);
        // Open header file for text-read, print error if file-open fails
        if((fp=fopen(hdr, "rt"))==NULL) {
                sprintf(ERRmess, "Opening HDR %s", hdr);
                ERROR(fun, ERRmess, 1);
        }
        // Check first line to see if it is an ENVI header file
        str[0]=0;
        fgets(str, HDRlmax, fp);
        sprintf(ERRmess, "Line 1 in HDR (%s) should be 'ENVI'", hdr);
        if(str == NULL) { ERROR(fun, ERRmess, 1); }
        strtrim(str, 2);
        if(stricmp(str, "ENVI")) { ERROR(fun, ERRmess, 1); }
        strcpy(ERRmess, "");

        // Read standard IDL-ENVI parameters into structure
        strHDR("description", str, fp, &multi, 1, 0);
        if(strlen(str) > (sizeof(h->description)-1)) {
                str[sizeof(h->description)-1] = 0;
        }
        strcpy(h->description, str);
        // only support "ENVI standard" file type, not e.g. ENVI Classification
        strHDR( "file type" , str, fp, &multi, 1, 0);
        strcpy(h->file_type, str);
```

```
if(strlen(h->file_type)==0) { strcpy(h->file_type, "ENVI standard"); }

h->offset  = atol(strHDR( "header offset", str, fp, &multi, 1, 0));
if(h->offset>0 && !offset_ok) {
        sprintf (ERRmess, "OFFSET (=%ld) should be 0 in HDR %s", h->offset , hdr);
        ERROR(fun, ERRmess, 1);
}
h->samples = atol(strHDR( "samples"      , str, fp, &multi, 1, 1));
if(h->samples<1) {
        sprintf (ERRmess, "Samples=%ld in HDR %s", h->samples, hdr);
        ERROR(fun, ERRmess, 1);
}
h->lines = atol(strHDR( "lines"        , str, fp, &multi, 1, 1));
if(h->lines  <1) {
        sprintf (ERRmess, "Lines=%ld in HDR %s", h->lines  , hdr);
        ERROR(fun, ERRmess, 1);
}
h->bands = atol(strHDR( "bands"        , str, fp, &multi, 1, 1));
if(h->bands  <1) {
        sprintf (ERRmess, "Bands=%ld in HDR %s", h->bands  , hdr);
        ERROR(fun, ERRmess, 1);
}
h->pixels = (__int64)h->samples * h->lines;
if(h->bands  >1 && !d3_ok) {
        sprintf (ERRmess,  "Bands (%ld) should be 1 in HDR %s", h->bands  , hdr);
        ERROR(fun, ERRmess, 1);
}
strHDR("interleave", str, fp, &multi, 1, h->bands>1);
if(strlen(str)==0) { strcpy(str, "bsq"); } // default: band-sequential (bsq)
switch( sgn(stricmp( str, "bip")) )
{
        case -1: h->interleave = 2; break;   // bil
        case  0: h->interleave = 3; break;   // bip
        case  1: h->interleave = 1; break;   // bsq
}
i = atoi(strHDR("data type", str, fp, &multi, 1, 1));
if (i<1 || i>4 ) {
        sprintf(ERRmess, "Data_type (%d) beyond allowed range [1-4] in HDR %s", i,
        hdr);
        ERROR(fun, ERRmess, 1);
}
if (i<DT_min || i>DT_max) {
        sprintf(ERRmess, "Data_type (%d) beyond range [%d ... %d] in HDR %s"  , i, DT_min,
        DT_max, hdr);
        ERROR(fun, ERRmess, 1);
}
DTextremes(i, &h->bpp, &h->Vlo, &h->Vhi);
h->data_type  = i;
//optional Byte order. If missing, assume 0 as default
h->byte_order = atoi(strHDR("byte order", str, fp, &multi, 1, 0) );

h->flip = flip_def(h->byte_order, h->bpp);
if(h->flip && !flip_ok) {
        sprintf(ERRmess, "Byteswap needed but not allowed in HDR %s", hdr);
        ERROR(fun, ERRmess, 1);
}

// Map-Info & Geo-location (optional)
strHDR("map info", str, fp, &multi, 1, 0);
if(!strlen(str)) {
        // defaults for map info
        sprintf(str, "%s, %.9g, %.9g, %.9g, %.9g, %.9g, %.9g", "arbitrary", CRmin, CRmin, 0.,
        (double)h->lines, 1.0, 1.0);
}
// Copy from header string to MAP INFO structure (&h->mi) within header structure (h)
map_find(str, &h->mi, 0, h->samples, h->lines, 1);
// Optionally read projection info and coordinate system string. For MSG Toolbox, we only
// support Geographic lat/lon in map info, so don't really need this
strHDR("projection info", str, fp, &multi, 1, 0);
if(strlen(str) > (sizeof(h->projection)-1)) {
        str[sizeof(h->projection)-1] = 0;
}
strcpy(h->projection, str);
strHDR("coordinate system string", str, fp, &multi, 1, 0);
```

```
	if(strlen(str) > (sizeof(h->coordsys)-1)) {
		str[sizeof(h->coordsys)-1] = 0;
	}
	strcpy(h->coordsys, str);

	// Glimpse / SPIRITS header extensions:
	//	Values
	strHDR("values", str, fp, &multi, 1, 0);
	item = (char *)calloc(strlen(str) + 1, 1);  // To hold 1 of the 8 elements
	if (strsplit(str, ",", 0, item) != 8)
	{
		// VALUES is missing or not properly defined (8 elements needed),
		// so fall back to defaults:
		//    empty variable name and unit
		strcpy(h->Vname, "?");
		strcpy(h->Vunit, "?");

		//    the value range is set to full range of the data type
		//    (set by DTextremes above)
		h->Vmin = h->Vlo;
		h->Vmax = h->Vhi;



		//    No encoding: scale = 1.0, offset = 0.0
		h->Vint = 0.0;
		h->Vslo = 1.0;
	} else {
		// VALUES has 8 elements as expected, so parse them
		strsplit(str, ",", 1, item);
		if(strlen(item) > (sizeof(h->Vname)-1)) {
			item[sizeof(h->Vname)-1] = 0;
		}

		// element 1: variable name
		strcpy(h->Vname, item);
		if(strlen(h->Vname)==0) {
			strcpy(h->Vname, "?");
		}
		// element 2: variable unit
		strsplit(str, ",", 2, item);
		if(strlen(item) > (sizeof(h->Vunit)-1)) {
			item[sizeof(h->Vunit)-1] = 0;
		}
		strcpy(h->Vunit, item);
		if(strlen(h->Vunit)==0) { strcpy(h->Vunit, "?"); }
		// elements 3&4: significant (non-flag) digital value range
		strsplit(str, ",", 3, item);
		f = atof(item);
		f1=min(f, h->Vlo);
		f2=max(f, h->Vhi);
		if(num_diff(f1, h->Vlo, 0.000001) || num_diff(f2, h->Vhi, 0.000001)) {
			sprintf (ERRmess, "VALUE Vlo=%f beyond range [%f ... %f] in HDR %s", f, h-
			>Vlo, h->Vhi, hdr);
			ERROR(fun, ERRmess, 1);
		}
		h->Vlo = f;
		strsplit(str, ",", 4, item);
		f = atof(item);
		f1=min(f, h->Vlo);
		f2=max(f, h->Vhi);
		if(num_diff(f1, h->Vlo, 0.000001) || num_diff(f2, h->Vhi, 0.000001)) {
			sprintf (ERRmess, "VALUE Vhi=%f beyond range [%f ... %f] in HDR %s", f, h-
			>Vlo, h->Vhi, hdr);
			ERROR(fun, ERRmess, 1);
		}
		h->Vhi = f;
		// elements 5&6: value range actually used (or fall-back to significant range)
		strsplit(str, ",", 5, item);  h->Vmin = atof(item);
		strsplit(str, ",", 6, item);  h->Vmax = atof(item);
		if ( (h->Vlo > h->Vmin) || (h->Vmin > h->Vmax) || (h->Vmax > h->Vhi) ) {
			printf("\n     WARNING: Bad sequence of VALUEs in %s (Vlo=%g, Vmin=%g,
			Vmax=%g, Vhi=%g)  \n\n", hdr, h->Vlo, h->Vmin, h->Vmax, h->Vhi);
		}
```

```
		// element 7&8: intercept and slope
		strsplit(str, ",", 7, item);  h->Vint = atof(item);
		strsplit(str, ",", 8, item);  h->Vslo = atof(item);
	}
	free(item);

	//	Date (optional)
	h->date = atol(strHDR("date", str, fp, &multi, 1, 0)); // YYYYMMDD, default: 0
	//	Days (optional), e.g. 10 for dekad composite, 30 for month, 365 for year
	h->days = atoi(strHDR("days", str, fp, &multi, 1, 0));
	if(h->date) {
		date_test(h->date, 1, &yyyy, &yy, &mm, m, &dd, &dd_st, &ttm, &tty, &dpm, &doy);
	}

	//	Flags (optional), flag name = value pairs
	strHDR("flags"  , str, fp, &multi, 1, 0); if(strlen(str) > (sizeof(h->flags  )-1)) {
		str[sizeof(h->flags  )-1] = 0;
	}
	strcpy(h->flags  , str);
	strHDR("sensor" , str, fp, &multi, 1, 0);

	if(strlen(str) > (sizeof(h->sensor )-1)) {
		str[sizeof(h->sensor )-1] = 0;
	}

	//	Sensor (optional), e.g. MSG
	strcpy(h->sensor , str);
	//	Comment (optional)
	strHDR("comment", str, fp, &multi, 1, 0);
	if(strlen(str) > (sizeof(h->comment)-1)) {
		str[sizeof(h->comment)-1] = 0;
	}
	strcpy(h->comment, str);
	//	Program (optional)
	strHDR("program", str, fp, &multi, 1, 0);
	if(strlen(str) > (sizeof(h->program)-1)) {
		str[sizeof(h->program)-1] = 0;
	}
	strcpy(h->program, str);

	// Close HDR-file
	fclose(fp);

	// Check file size of image data file against header information
	if(tis)
	{
		__int64 byt_exp, byt_tru;
		struct _stati64   filestat;
		if(_stati64(img, &filestat)) {
			sprintf(ERRmess, "Cannot open IMG %s", img);
			ERROR(fun, ERRmess, 1);
		}
		byt_tru = filestat.st_size;
		byt_exp = h->offset + (h->pixels * h->bands) * h->bpp;
		if (byt_tru < byt_exp)    {
			sprintf(ERRmess, "IMG %s smaller (%I64d) than expected (%I64d bytes)",
			byt_tru, byt_exp, img);
			ERROR(fun, ERRmess, 1);
		}
	}
}
```

```
/*************************************************************************************
strHDR: Search and retrieve the value (dest) of an item (src) in an ENVI header-file.
       Values can be either scalar (multi = 0), typically on a single line
              Header file syntax: src = dest
       Or multi-element (multi=1), possibly multi-line
              Header file syntax: src = {v1,v2,v3,…}
Parameters:
    Fptr : pointer to the open header file
    src  : keyword to search for, case-insensitive
    dest : retrieved value (string)
    rew  : first seek to start of file (0/1)
    fatal: 0 for optional keyword (returns empty string if not found), 1 for mandatory (error if
        missing)
    multi: indicates if the retrieved value is single-line (0) or multi-line (1)
*************************************************************************************/
char *strHDR(char *src, char *dest, FILE *fptr, short *multi, short rew, short fatal) {
       char           record[HDRlmax+1], ss[501], *p;
       short          i, l;

       if(rew) { rewind(fptr); }
       strcpy(ss, src); strtrim(ss, 2); // ss=copy of source but trimmed
       l = strlen(ss);
       // scan file
       while (!feof(fptr))
       {
              record[0]=0;
              fgets(record, HDRlmax, fptr);
              strtrim(record, 2);

              if(strnicmp(record, ss, l)==0)
              {
                     // line with keyword src found
                     p = strstr(record, "=");
                            if(p==NULL) { continue; }
                     strmid(p+1, 1, HDRlmax, dest);
                     strtrim(dest, 2);
                     if(strnicmp(dest, "{", 1)) {
                            // found: scalar parameter
                            *multi = 0;
                            return dest;
                     }
                     // look for list of values, between {}, maybe multi-line
                     *multi = 1;
                     for(i=0;i<strlen(dest);i++) { dest[i] = dest[i+1]; }
                     strtrim(dest, 0);
                     l = strlen(dest);
                     while(!feof(fptr) && dest[strlen(dest)-1] != '}')
                     {
                            record[0]=0; fgets(record, HDRlmax, fptr);
                            strtrim(record, 2);
                            l = l + 1 + strlen(record);
                            if (l>HDRlmax) { sprintf(ERRmess, "ITEM '%s' longer than allowed %d
                            bytes", src, HDRlmax) ; ERROR("LIBfunc strHDR", ERRmess, 1); }
                            strcat(dest, " "); strcat(dest, record);
                            (*multi)++;
                     }
                     if (dest[strlen(dest)-1] != '}')
                     {
                            if(fatal) {
                                   sprintf(ERRmess, "ITEM '%s' should end with '}'", src);
                                   ERROR("LIBfunc strHDR", ERRmess, 1);
                            } else { *multi = 0; break; }
                     }
                     dest[strlen(dest)-1] = 0;
                     strtrim(dest, 2);
                     return dest;
              }
       }
       if (fatal) {
              sprintf(ERRmess, "HDR-item '%s' not found" , src);
              ERROR("LIBfunc strHDR", ERRmess, 1);
       }
       strcpy(dest, "");
       return dest;
}
```

```
/*************************************************************************************
map_find: parses map info spatial information into MAP INFO structure
Parameters:
	*s: string containing the map info string in ENVI header file
	*gr: resulting structure
	samples, lines: values retrieved from samples/lines in header file (grid size)
*************************************************************************************/
void map_find(char *s, struct MAPINFO *gr, long samples, long lines) {
	char	str[HDRlmax+1];
	short	i, N0, N;
	double	a, b;

	strcpy(gr->name, "");
	if(strlen(s)<1) {return;}
	N = strsplit(s, ",", 0, str);
	if(N < 7) {
		sprintf (ERRmess, "Incomplete MAP INFO: %s", s);
		ERROR("LIBfunc MAP_FIND", ERRmess, 1);
	}

	strsplit(s, ",", 1, str);
	if(strlen(str) > (sizeof(gr->name)-1)) {
		str[sizeof(gr->name)-1] = 0;
	}
	strcpy(gr->name, str);
	strsplit(s, ",", 2, str); gr->mc = atof(str);
	strsplit(s, ",", 3, str); gr->mr = atof(str);
	strsplit(s, ",", 4, str); gr->mx = atof(str);
	strsplit(s, ",", 5, str); gr->my = atof(str);
	strsplit(s, ",", 6, str); gr->dx = atof(str);
	strsplit(s, ",", 7, str); gr->dy = atof(str);

	N0 = 7;
	gr->zone = 0;
	strcpy(gr->ns, "");
	if (!stricmp(gr->name, "UTM"))
	{
		N0 = 9;
		strsplit(s, ",", 8, str); gr->zone = atoi(str);
		strsplit(s, ",", 9, str); strcpy(gr->ns, str);
		if(gr->zone<1 || !strlen(gr->ns)) {
			sprintf (ERRmess, "Incomplete MAP INFO for UTM: zone=%d, NS=%s", gr->zone, gr-
			>ns);
			ERROR("LIBfunc MAP_FIND", ERRmess, 1); }
		}
		strcpy(gr->datum, "");
		strcpy(gr->units, "");
		for(i=0;i<(N-N0);i++)
		{
			strsplit(s, ",", N0+i+1, str);
			if(strnicmp(str, "units", 5)==0) {
				strsplit(str, "=", 2, gr->units);
			} else {
				strcpy(gr->datum, str);
			}
		}
		b =  1./gr->dx;
		a = gr->mc - b * gr->mx;
		gr->xmin = (CRmin - a)/b;
		gr->xmax = (CRmin + samples - a)/b;

		b = -1./gr->dy;
		a = gr->mr - b * gr->my;
		gr->ymax = (CRmin - a)/b;
		gr->ymin = (CRmin + lines   - a)/b;
	}
}
```

```
/***************************************************************************************
Extract from main routine
***************************************************************************************/
void main( int argc, char **argv ) {
        char            fileIn[_MAX_PATH];
        short           BPPi;
        long            line, column;
        struct ENVIHDR      headerIn,    hm,    ho;
        FILE                *filePointerIn;
        unsigned char       *bufferIn;

        //Read header file for 1 layer; accepting any datatype, byte offset or endian-ness
        envi_hdr_read(fileIn, &headerIn, 1, 0, 1, 4, 1, 1);

        //Open binary image file
        if((filePointerIn = fopen(fileIn,"rb"))==NULL) {
                sprintf(ERRmess, "Opening input image   %s", fileIn);
                ERROR("PRE-PROCESSING", ERRmess, 1);
        }
        //Skip the byte offset indicated in the header
        fseek(filePointerIn, headerIn.offset, SEEK_SET);

        // read the image file line-by-line
        for (line =0; line < headerIn.lines; line ++)
        {
                fread(bufferIn, BPPi, headerIn.samples, filePointerIn);
                if(headerIn.flip) {flip_byte(bufferIn, headerIn.samples, BPPi); // byte-swap
                // process each pixel (column) in current line
                for(column=0; column < headerIn.samples; column++)
                {
                … // some processing
                }
        }
}
```

In Fortran90 programming language

```
program img2ascci

! The main goal of this sample program is to clarify how to access .img
! files data with Fortran, using the directives present in the
! respective .hdr file.
!
! Compiling (gfortran):
!
!     gfortran -o img2ascci img2ascci.f90
!
! Usage (Linux command line):
!
!     img2ascci <filename> <lines> <samples> [<bands>] [<header offset>]
!
! Input arguments:
!
! <filename> - The path to the .img file
! <lines> - The number of rows in the image
! <samples> - The number of columns in the image
! <bands> - The number of spectral bands in the image. (Optional if
!     no <header offset> argument is given. Default value is 1).
! <header offset> - The number of bytes of data in the image file to skip in
!     order to reach the start of the image data. (Optional. The default
!     value is 0 bytes).

!---------------------------------------------------------------
!   Auxiliary variables:
!---------------------------------------------------------------
    implicit none

    character(255) :: img_filename  ! input filename
    integer        :: nrows, ncols, nbands, skipbytes  ! lines, samples, bands and header offset
    integer , parameter :: data_kind = 4  ! number of bytes (data type)
    integer (kind=data_kind) , dimension(:,:,:), allocatable :: img_data  ! data handler
    integer        :: nargs, f_unit, io_stat, lin, col, band, band_offset, row_offset
    character(16)  :: str_buffer
    integer , parameter :: nshow = 5
```

```
!-------------------------------------------------------------
!   Get Input arguments from command line:
!   (The given input arguments (2,3,4 and 5) should be present
!    in the .hdr respective file of the given .img filename)
!-------------------------------------------------------------

    nargs = iargc()

    call getarg(1,img_filename)

    call getarg(2,str_buffer)
    read(str_buffer,*) nrows

    call getarg(3,str_buffer)
    read(str_buffer,*) ncols

    if (nargs > 3) then
        call getarg(4,str_buffer)
        read(str_buffer,*) nbands
    else
        nbands=1
    end if

    if (nargs > 4) then
        call getarg(5,str_buffer)
        read(str_buffer,*) skipbytes
    else
        skipbytes=0
    end if

    write(*,*)new_line('a'),"Input arguments:", new_line('a'), &
    &"filename = ", trim(img_filename), new_line('a'), &
    &"lines     =", nrows, new_line('a'), &
    &"samples  =", ncols, new_line('a'), &
    &"bands     =", nbands, new_line('a'), &
    &"header offset =", skipbytes, new_line('a'), new_line('a'), &
    &"The following arguments are immutable for this program:", new_line('a'), &
    &"data type  = 3 (Long: 32-bit signed integer)", new_line('a'), &
    &"interleave = bsq", new_line('a'), &
    &"byte order = 0", new_line('a')

!-----------------------------------------------------------
!   Read data from the .img file:
!-----------------------------------------------------------

    allocate(img_data(0:ncols-1,0:nrows-1,0:nbands-1))

    if (skipbytes == 0) then

        open (newunit=f_unit, file=trim(img_filename), status='old', action='read', &
        & recl=data_kind*nrows*ncols, access='direct', iostat=io_stat)

        do band=0, nbands-1
            band_offset = band*ncols*nrows
            read(f_unit, rec=1+band_offset, iostat=io_stat)img_data(:,:,band)
        end do

    else

        open (newunit=f_unit, file=trim(img_filename), status='old', action='read', &
        & recl=data_kind, access='direct', iostat=io_stat)

        do band=0, nbands-1
            band_offset = band*ncols*nrows
            do lin=0, nrows-1
                row_offset = lin*ncols
                do col=0, ncols-1
                    read(f_unit,     rec=1+skipbytes+band_offset+row_offset+col,      iostat=io_stat)
img_data(col,lin,band)
                end do
            end do
        end do

    endif
```

```
!-----------------------------------------------------------
!   Print corner pixels (Output):
!-----------------------------------------------------------

    do band=0, nbands-1
        write(*,*)"Band:",band,new_line('a')
        do lin=0, nshow-1
            write(*,*)img_data(0:nshow-1,lin,band),"     ...",img_data(ncols-5:ncols-1,lin,band)
        end do
        write(*,*)"        ...",new_line('a'),"         ...",new_line('a'),"         ..."
        do lin=nrows-nshow, nrows-1
            write(*,*)img_data(0:nshow-1,lin,band),"     ...",img_data(ncols-5:ncols-1,lin,band)
        end do
    end do

    deallocate(img_data)

end Program img2ascci
```