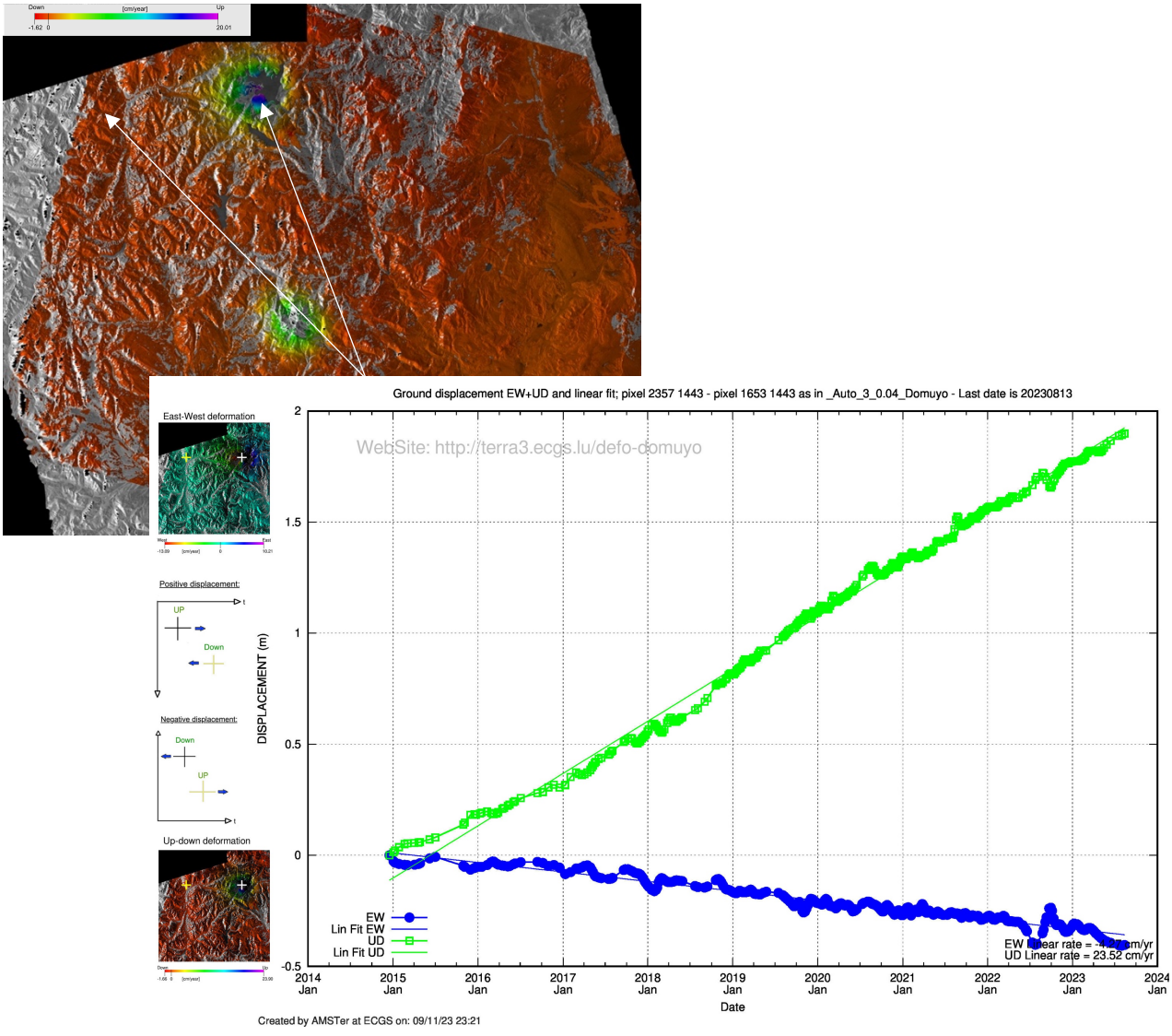
AMSTer:

SAR & InSAR Automated Mass processing Software for Multidimensional Time series

Nicolas d'Oreye[1,2], Dominique Derauw[3,4], Sergey Samsonov[5],
Delphine Smittarello[1], Maxime Jaspard[1], Gilles Celli[1]

[1] *European Centre for Geodynamics and Seismology (ECGS), 19 rue Josy Welter, L-7256 Walferdange, Luxembourg*
[2] *National Museum of Natural History (NMNH), 19 rue Josy Welter, L-7256 Walferdange, Luxembourg*
[3] *Centre Spatial de Liège (CSL), Avenue du Pré Aily, B-4031 Angleur, Belgium*
[4] *SAREOS, 1 Rue des Violettes, 4557 Fraiture, Belgium*
[5] *Canada Centre for Mapping and Earth Observation, Natural Resources Canada (NRCAN), 560 Rochester Street, Ottawa, ON K1A 0E4, Canada*



AMSTer:
to crunch the SAR and InSAR mass processing

https://github.com/AMSTerUsers/AMSTer_Distribution

Illustration on the front page is an example of double difference of vertical (green) and horizontal (East-West; blue) ground deformation observed with Sentinel-1 images in the Domuyo (Argentina) – Laguna del Maule (Chile) volcanic region between pixels pointed at by arrows on the vertical velocity map. The time series was fully automatically computed using 192 images acquired in ascending orbit and 263 images acquired in ascending orbit spanning October 2014 – Jan 2022. More than 4500 interferograms where computed and 3600 were used for MSBAS inversion considering a coherence threshold.

For more information, see

- Derauw D., d'Oreye N., Jaspard M., Caselli A. and Samsonov S. (2020)
  Ongoing automated Ground Deformation monitoring of Domuyo – Laguna del Maule area (Argentina) using Sentinel-1 MSBAS time series: Methodology description and first observations for the period 2015 – 2020.
  J. South Am. Earth Sc., Vol. 104, 102850. https://doi.org/10.1016/j.jsames.2020.102850
  Open Access here: https://www.sciencedirect.com/science/article/pii/S089598112030393X?via%3Dihub

- d'Oreye N., Derauw, D., S. Samsonov, M. Jaspard, and D. Smittarello (2021), MasTer: a full automatic multi-satellite InSAR mass processing tool for rapid incremental 2D ground deformation time series. Proc. IEEE IGARSS21, Brussels, July 2021.

See also:

- Smittarello, D., d'Oreye, N., Jaspard, M., Derauw, D. & Samsonov, S. Pair Selection Optimization for InSAR Time Series Processing. J Geophys Res Solid Earth 127, (2022).
  Open Access here: https://agupubs.onlinelibrary.wiley.com/doi/10.1029/2021JB022825?af=R

The every-day updated version of these results can be freely accessed at the following web page:
http://terra3.ecgs.lu/defo-domuyo/index.php

Table of contents

0. Introduction

0.1) What is AMSTer ?

AMSTer stands for Automated SAR & InSAR Mass processing Software for Multidimensional Time series.

AMSTer software is a tool dedicated to the processing of Synthetic Aperture Radar (SAR) images. It is mostly designed to allow optimized **mass processing of SAR interferometry** (InSAR) and the production of **2/3D ground deformation time series**. However, AMSTer can also be used for the production of **Digital Elevation Models** (DEM), coherence maps and amplitude images time series e.g. for **land cover or geomorphological changes, flood mapping etc**...

AMSTer software is made of three components:

1. an InSAR command line processor (the **AMSTerEngine**),
2. the **MSBAS** processor (for the computation of 2/3D time series of ground deformation)
3. and a bunch of **shell scripts** automatizing all tasks, from data downloading to updated displacement maps and time series and possible automatic display on a dedicated webpage (**AMSTer Toolbox**).

AMSTer was initially named MasTer. Some publications refer to that former name.

The **AMSTerEngine** (formerly named MasTerEngine) is a command line InSAR processor written in C derived from the Centre Spatial de Liege (CSL, Belgium) InSAR Suite (CIS) [Derauw, 1999]. It is able to process nearly all the SAR sensor currently available (ERS1 & 2, EnviSAT, ALOS, ALOS2, RadarSAT 1& 2, CosmoSkyMed, TerraSAR-X, TanDEM-X (incl. bistatic and pursuit mode), Sentinel1 A & B, Kompsat5, PAZ, SAOCOM, ICEYE...).

Contact: dderauw@sareos.be

The Multidimensional Small BAseline Subset (**MSBAS**) time series software [Samsonov and d'Oreye, 2012, 2017; Samsonov et al. 2020] allows inverting simultaneously several interferometric SAR time series acquired along different acquisition modes, including different sensors, SAR wavelengths, incidence angles, polarization etc., to solve for horizontal and vertical component of the ground displacement. MSBAS is maintained at NRCan/RNCan, Canada.
Contact: sergey.samsonov@canada.ca

The **AMSTer toolbox** explained here (see specificities hereafter) is mostly based on bash scripts and Python and is maintained at ECGS, Luxembourg [Derauw et al. 2020; d'Oreye et al. 2021].
Contact: nicolas.doreye@ecgs.lu

Kind reminder: The software is licensed under CC BY-NC-SA 4.0 (Attribution-NonCommercial-ShareAlike 4.0 International) and freely available for noncommercial use from GitHub[1]. To help us carrying on with the development of the tool, we kindly ask you to cite the **AMSTer** in publications that contains results produced by the software thanks to at least the most recent references (see **References** at the end of the present manual).

You can also drop us a line on the discussion group of the GitHub[1] repository to let us know your usage. It is encouraging for us.

[1] https://github.com/AMSTerUsers/AMSTer_Distribution

0.2) License

"AMSTer: Automated SAR & InSAR Mass processing Software for Multidimensional Time series" © 2023 by Nicolas d'Oreye, Dominique Derauw, Sergey Samsonov, Delphine Smittarello, Maxime Jaspard and Gilles Celli is licensed under CC BY-NC-SA 4.0 (Attribution-NonCommercial-ShareAlike 4.0 International).
http://creativecommons.org/licenses/by-nc-sa/4.0/

You are free to:

Share — copy and redistribute the material in any medium or format.

Adapt — remix, transform, and build upon the material. The licensor cannot revoke these freedoms as long as you follow the license terms.

Under the following terms:

Attribution - You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.

NonCommercial - You may not use the material for commercial purposes.

ShareAlike - If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.

No additional restrictions - You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

Notices: You do not have to comply with the license for elements of the material in the public domain or where your use is permitted by an applicable exception or limitation. No warranties are given. The license may not give you all of the permissions necessary for your intended use. For example, other rights such as publicity, privacy, or moral rights may limit how you use the material.

Moreover, MSBAS is licensed to the https://open.canada.ca/en/open-government-licence-canada

Because AMSTer software is licensed free of charge, there is no warranty for the program.

0.3) AMSTer specificities

- Multiplatform:
  AMSTer is fully Mac OSX and Linux (at least Ubuntu 20.04 and 22.04) compliant

- Automatization:
  AMSTer performs all its tasks thanks to a series of scripts launched by few command lines which are easily integrated in cron jobs to become a fully automated tool. The whole processing chain makes use of single files containing all the parameters required for the mass processing of each satellite mode. Examples of cron jobs for making a fully automatic processing & monitoring tool are provided.

- Optimization:
  The processing chain is designed to split the work in several tasks, some of which can be run in parallel. These tasks are optimized e.g. by sharing intermediate results that are common between some steps. Masking of decorrelated regions allows faster unwrapping. AMSTer also benefits from an optional unwrapping accelerator (DetPhun, [Derauw and d'Oreye, in prep.]). When updated Sentinel1 orbits are made available, AMSTer automatically downloads only the orbits (not the image) and recompute automatically each step (and only those) impacted with that update up to the production of the ground deformation time series.

- Self-evaluation:
  It is self-evaluating and able to automatically tune some parameters and recompute some steps that would be unsatisfactory.

- Incremental:
  Computation of time series is incremental, that is, when a new image is available, AMSTer downloads it automatically, then computes the new interferometric pairs formed using this new image along with all the images already acquired that satisfy the pre-defined spatial and temporal baselines criteria. The new differential deformation maps are then added to the existing database to be inverted by MSBAS. To compute the 2D or 3D linear rates and the cumulative deformation maps and time series, the MSBAS inversion itself only takes few tens of minutes to hours depending on the size of the area and the number of pairs.

- Security:
  The processing architecture is designed in order to minimize the loss of data in case of hardware failure by storing the data and the intermediate and final results in a directories structure that is easily shared between several dedicated hard disks or servers.

- Flexibility:
  AMSTer contains several tools for testing and fine-tuning the processing, for selecting the most appropriate MSBAS inversion parameters, or for cleaning or reprocessing only some steps after a possible crash, for preparing, extracting results, making time series and double difference time series plots, creating maps and kmz for display with Google Earth etc…. If an image presents some problem, it can be quarantined from the automated processing chain.

- User oriented tools:
  Specific scripts exist to select the appropriate InSAR pairs based on temporal and spatial thresholds or Delaunay triangulation etc. Further selection can be performed based on the mean coherence computed over a user-defined footprint (kml). There are also tools to test the installation, to create masks based on mean coherence or to generate sequence of amplitude gif animation (to monitor the ground cover changes or the geomorphologic changes over time)

etc… If you need a tool, there are high chances that you are not the first one to need it and it must exist… If not, let us known.

- Interactive web page:
  Optional scripts allow displaying the results in a dedicated web page as linear rate of deformation maps (wrapped either on SAR amplitude images, Google Maps or Google Earth), amplitude maps, differential cumulated deformation maps, coherence maps, RGB amplitude-coherence maps, baselines plots as well as time series of ground displacement double difference between pairs of pre-selected pixels.
  An interactive tool also allows extracting time series for any other pair of points. The results are then sent by email to the user within a minute [Derauw et al., 2020]. The module to create the web page was developed by M. Jaspard (ECGS) and is described in a separate manual named Web_tool_V*x.x*.pdf (where *x.x* is the number of the latest version of that manual).

The methodology and succession of steps are described below in the main text. Architecture of main scripts are described in annexes.

0.4) Conventions (hopefully) used in the present document:

- Paths are in green ("…/" at the beginning of a path means "whatever your path starts with")
- Parameters are in *italic green*
- External commands or files are in *italic blue*
- AMSTerEngine commands and scripts are in ***bold italic***
- Some warnings or important remarks are in red
- YYYY MM DD stands obviously for year, month and day
- Yellow highlight is coming soon (hopefully) or to be revised after main changes in AMSTer toolbox
- Parameters with square brackets (i.e. [..]) show in command lines are optional
- Names of AMSTer Toolbox scripts usually start with a capital letter. Names starting with a small letter are usually an AMSTer Engine command or an external command.

Some names start by one or more "_" or "zz_". This is only to get them at the top of the list when searching a directory…

In addition, lines of text displayed at the terminal (or logged in log files) starting with a double slash (i.e. " // ") are output by AMSTer Toolbox scripts. Lines starting without these double slashes are usually output by AMSTer Engine or external commands. It can also be errors or warnings, though this must be obvious from the message itself.

MT stands for Main Toolbox (see for instance the directory SCRIPTS_MT or the file FUCNTIONS_FOR_MT.sh).

Among the InSAR community, images used to perform an interferogram were commonly named *Master* (*MAS*) and *Slave* (*SLV*). Given the inappropriate connotations of this terminology, the InSAR scientific community recommended using a different nomenclature in the early 2020s (see for instance the Open Statement on Terminology in InSAR by COMET centre). In the present manual, the images used to compute an interferogram are called *Primary* (*PRM*) and *Secondary* (*SCD*). When all the images are coregistered on a single reference image, this image, formerly commonly named Super Master, is named here the *Global Primary* image (*GLOBPRM*). Despite our efforts, some traces of the old terminology used for ~30 years are still present e.g. in the naming of some scripts, files and directories (e.g. using Super Master, sometimes dubbed SM) or in the naming of variables within the scripts. Because AMSTer is aiming at computing automatic and incremental time series, changing these remaining names would require to recompute or rename millions (literally) of files, directories and parameters within files in the complex architecture currently running in several scientific institutions or they would be rendered inoperative.

0.5) Installation:

All the resources are available from https://github.com/AMSTerUsers/AMSTer_Distribution.

Installation can be performed using the script ***AMSTer_Install.sh***. That script can also be used to update AMSTer and can be run several times. It is stored in SAR/AMSTer/SCRIPTS_MT/zz_Utilities_MT.

The script ***AMSTer_Install.sh*** will guide you through the whole process of installation, through a set of questions and actions. Please read carefully all the messages and questions. It will download the required libraries, apt or ports, gnu tools, ancillary software etc. It will also update your *.bashrc* and *.bash_profile* with the required (mandatory) state variable and update the $PATH.

In a nutshell, it will

- Recommend to delete from your *.bashrc* former information that would result from an old installation, that is old enough for not having been performed by the present installer script.
- After confirming the former step, it will ask you to confirm your $HOME directory.
- It will then ask you if you want a [f]ull installation or an [u]pdate of only the main components (AMSTer Engine, msbas or Toolbox scripts).
- It will then offer you to install / check or skip (if you are sure that it is already installed) mandatory elements (*apts or ports, libraries, gmt and gdal, clang14, gnu utilities, Java, Fiji (imageJ), snaphu, gnuplot, python3*)
- Follow the instructions if it asks you to download external software and copy them where it asks you (/SAR/EXEC/).
- In addition, it will offer you to install / check or skip some useful tools (and often highly recommend) such as *GitKraken, QGIS, GIMP, gnu fortran, cpxfiddle*)
- It will then update the PATH and state variables in your *bashrc*.
  Note about the PATH state variable: you need to have a line in your *.bashrc* like:
  `export PATH=$PATH`
  It will look for it but may find another `export PATH=(something_you_need)` in your *.bashrc* that comes from another installation on your computer. In that case, it will ask you if you are satisfied with that other line. If you say no, it will offer you to enter another variable. Enter `export PATH=$PATH`. The existing one can be deleted or kept after having entered your own new variable. Follow the questions and everything should be fine. At the worst, you can have several exports of `export PATH=$PATH` if you mess up. It is not harmful and you can always remove them manually after by editing your *bashrc* after the installation.
- It will then offer you add an external state variable named EXTERNAL_DEMS_DIR. This can however conflict with automatic usage of AMSTer Toolbox. It is then recommended to answer no. It will never be harmful to not install it as AMSTer will always take care of the external DEM for you through the configuration file you will need anyway.
- It will then ask you to assign state variables to some mandatory disks (or directories) where it will expect to find and/or store several products, intermediate results or final results, that is at least $PATH_DataSAR, $PATH_1650, $PATH_3600, $PATH_3601, $PATH_3602. For each of these variables, enter a path to a disk or a directory. More disks (or directories) are suggested ($SynoData or $HOMEDATA) though they are not mandatory and can be ignored. You can do it or add more later manually in your *bashrc* if needed.
  Note that if the disk (or directory) that you entered as state variable is not reachable (or does not exist yet), it will offer you to enter again the name (in case it would result from a typo), or define it anyway and it will be your responsibility to mount the disk later. (If it is a directory, it will offer you to create it for you).

On Linux, mounting points of external disks are supposed to be in /mnt/ while on Mac, they are in /Volumes/.

- Then it offers to download the geoid data that are mandatory for building DEMs
- Then it offers you to download S1 orbits (providing that the AMSTer Engine is already installed). This can either be done from the first orbit (beware, it takes a lot of time !), or from a given date.
- If your *.bashrc* and/or *.bash_profile* have been modified, a backup is created and renamed by appending the date and time of the modification. In that case it will also offer you to reboot the computer for making these changes active.
- **Note** : in case of problem, ensure that you have the rights to create/change the *.bashrc* and *.bash_profile*
- **Note**: it is mandatory that the path to where some of the tools will be stored (/opt/local/bin) appears first in your *.bashrc*. If AMSTer does not act as expected, ensure that the $PATH with /opt/local/bin is declared first.

Note that the installation can also be performed manually (see documents *Install_AMSTer_Mac_Vx.x.pdf* or *Install_AMSTer_Linux_Vx.x.pdf* where *x.x* is the number of the latest version of that manual).

When installed, all the AMSTer Toolbox scripts are stored in a directory that must be named /SAR/AMSTer/SCRIPTS_MT, and which contains a.o. a subdirectory named zz_Utilities_MT.

Also, several scripts are tools that were used at ECGS for our own needs. They are stored in a sub directory named SCRIPTS_MT/zz_Utilities_MT_Ndo left in the package as they may be of some use for other users, providing (most probably) that the user performs necessary adjustments.

After the installation, you can check it using the script ***Check_Installation.sh*** that is in zz_Utilities_MT. It will try to check mains components (AMSTerEngine, MSBAS and AMSTer shell scripts compilation and access, as well as third party software, libraries, state variables etc…).
Watch the displayed messages. Note that, given the diversity of computer configurations, some displayed messages may be inaccurate (eg. complaining about wrong version and offering instead… the same version). Just ignore these warnings.

The script ***___TEST_ALL_TYPE_OF_PROCESS.sh*** that is in zz_Utilities_MT_Ndo may also help you to run several processings using data from several sensors in order to check that the tools run smoothly. It can also be useful to run this again after major updates. Of course, it was designed for our own needs at ECGS but with a bit of work, you can change the several hard-coded lines in order to tune it to your own needs.

0.6) Current status:

- Because AMSTer is in constant evolution and involves several types of satellite data, it may happen that some old functionalities (or some rarely used functionalities) do not survive the most recent updates. If you observe some problem with some scripts (especially for non S1 data for instance), feel free to let us know asap (ndo@ecgs.lu).
- Geocoding of (too) large images leads to crash while allocating too big memory blocs. This requires in depth modification of the geocoding function. If you do not have enough RAM, consider either processing smaller images, or increase the multilooking.
- Lat/Long geocoding is not implemented. Results are only given in UTM although AMSTerEngine allows lat/long results. Nevertheless, transforming UTM in Lat/Long is straight forward e.g. with gdal (see e.g. the script ***UTM2LL.sh***) or any GIS software.
  Note that geocoding of amplitude files is however possible after computation of all the amplitudes using ***ALL2GIF.sh*** (see paragraph about ***ALL2GIF.sh***).
- Mass re-processing from unwrapping or geocoding was only tested for S1 in WS mode.
- The function used to make raster figures uses *cpxfiddle*. This could be improved e.g. by using *Python* or *convert* (from *ImageMagick*). Currently there is a function in ***FUNCTIONS_FOR_MT.sh*** named ***MakeFigFiji*** that uses *ImageJ* to create a jpg figure instead of raster files. However, this function is NOT used in the scripts because 1) it is much slower than the functions that uses *cpxfiddle*, and 2) because several scripts expect raster files instead of jpg files. Nevertheless, if the user wants to avoid using *cpxfiddle*, and use *ImageJ* instead, he/she can adapt and replace the several ***MakeFig*** functions and then adapt the scripts that would search for raster files.
- Radarsat-1 can be read and processed, though only in slant range. Hence, DInSAR can be performed in 3 passes mode but results can't be geocoded because external DEM can't be taken into account. This is related to the fact that the orbits of RS1 are provided in a different reference frame.

Note about the ***MakeFig*** functions:
In "***FUNCTIONS_FOR_MT.sh***" several functions (named resp. ***MakeFig, MakeFigR, MakeFigR2, MakeFigR3, MakeFigNoNorm, MakeFigRflip, MakeFigNoNormFlip, MakeFigflip***) create one-line scripts that are stored in the processing directories and that use *cpxfiddle* to create raster files. They can be edited to change the way to display the data and re-run from the same directory to re-create the figure.

Note about Sentinel 1 orbits (from Feb 2021): obsolete (since March 2022) use of ASF orbits
Since Feb 2021, ESA changed the way the rapid orbits are dealt with. The orbits delivered within the original images are now not accurate enough to provide with satisfying coregistration. One need Restituted orbits, provided since February 2021 through 3h long segments. Downloading these Precises (POE) and restituted (RES) orbits can be performed from Alaska SAR Facility (ASF).
It requires:

- AMSTerEngine at least from 20210519 (with function ***updateS1Orbits***) to cope with new ESA orbits.
- Procedure to download orbits from ASF as explained here:
  https://wiki.earthdata.nasa.gov/display/EL/How+To+Access+Data+With+cURL+And+Wget
  that is (see web page for details):
  - Make an account to urs.earthdata.nasa.gov and get a *login* and *pwd*
  - While logged in,
    - click on "My Profile", then menu "Applications > Authorized Apps".
    - At the bottom, click on button "APPROVE MORE APPLICATIONS"

  - In the “Search” bar, type “Sentinel”
  - Click on “AUTHORIZE” button next to each application named with Sentinel 1.

- In a terminal on your computer type:

```
> cd ~
> touch .netrc
> echo "machine urs.earthdata.nasa.gov login login password pwd " > .netrc
> chmod 0600 .netrc
```

  Then

```
> cd ~
> touch .urs_cookies
```

- Ensure that you have *S1_ORBITS_DIR* as state variable in *.bashrc* for path to directory where orbits are stored (./AUX_POEORB and ./AUX_RESORB). Old versions of AMSTer had *S1_PRECISES_ORBITS_DIR* but it is not used anymore.
- Note that since March 2021 images may be distributed with original orbits (i.e. not restituted nor precise) which are of very poor quality. Because it is useless for interferometry, S1DataReader is now used with option -p to prevent reading images that only have these orbits. When these orbits will be automatically updated at the occasion of another reading (with restituted or precise orbits), the images will be read normally. This is hence a transparent process for the user.
- Later, ESA changed the way they make the S1 orbits available (see below). The ASF procedure here above is not needed anymore but can still be used as a backup.

Note about Sentinel 1 orbits (from March 2022):
Since March 16 2022, AMSTer takes again the orbits from ESA server using default user and password (i.e. *gnssguest/gnssguest*). This is transparent to the user as long as he/she has a AMSTerEngine more recent than March 2022.
Note however that AMSTerEngine is still able to use ASF orbits using -ASK option at call of ***updateS1Orbits*** function. Nevertheless, AMSTer toolbox (and especially ***Read_All_Img.sh***) is tuned to use the easiest and fastest solution, that is ESA orbits.

Also, a start date can be used to verify the orbit data base (see ***updateS1Orbits*** help). In particular, ***Read_All_Img.sh*** now performs update of local S1 orbit data base only since last available precise orbit date. To force update of older orbits, either add option *from=YYYYMMDD* either

- in ***Read_All_Img.sh*** script where it calls the function ***updateS1Orbits***, or
- performed manually using command ***updateS1Orbits*** (see ***updateS1Orbits*** *-help*), or
- when using ***Read_All_Img.sh*** script with *ForceAllYears* option.

0.7) RECOMMENDATIONS:

Unsorted advices:

- Even if you do not have 4 hard disks where you will store the data, the processing results and the final results, it will spare you a lot of time and troubles to keep the same architecture by defining 4 paths in your PATH state variable to identify 4 places (or disks) where the scripts expect to find some specific directories. The names of these 4 places (or disks) used by AMSTer are based after servers' name at ECGS: 1650, 3600, 3601 and 3602. It might be the easiest solution for you to keep this naming as it is used nearly everywhere. Also, even if you do not work with 4 separate disks, it is required to have these 4 state variables (see below "Expected architecture of disks"): $PATH_1650, $PATH_3600, $PATH_3601 and $PATH_3602. These state variables are created at installation if it is performed with the installer script (see above).

- Avoid naming directories with dash or blank characters. This is always a good practice anyway...

- **Do not use hard drives formatted in FAT format because it does not support symbolic links, which are widely used by AMSTer**

- Keep the same directories structure as what is proposed here (including for instance "/NoCrop") for storing the data and results. This is mandatory…

- In case of crash or problem, start by reading the header of the script(s). It may contain useful information about dependencies, hard coded information etc. After a crash, do not forget to check/clean directories where results are supposed to be stored.

- When raster figures are not processed as expected, before concluding that the processing was wrong, check the original file e.g. with *cpxfiddle*, *Fiji* or *QGis*…. If file is ok, then either
  - the name of the processing directory contains some string that conflicts with one of the functions (such as a star), or
  - as it was observed on some Linux computer, scripts launched through cron jobs can't be run with the user's profile defined in the *bashrc*. State variables must be copied in the crontab.
  - If images look like noise when opened with *Fiji*, check that you have correctly ticked the endianness box in the *Fiji* dialog box asking for image path, size etc…

  In the first case, try the reprocessing with another processing path. In the second case, just relaunch manually the script generating the figure.

- If you stitch your DEM tiles with the appropriate AMSTerEngine tool (i.e. *agregateSRTMTiles*), it will create a *.txt* file that will be used by the ***SlantRangeDEM*** function to extract the necessary values. Be careful because with this technique, the DEM is in mathematical order and referred to the ellipsoid.
  If you create a DEM with a GIS tool in Envi format, it must contain a *.hdr* file instead of a *.txt* file. Be aware that in such a case the DEM will be inverted compared to mathematical logic and may not be referred to the ellipsoid. In order for the AMSTerEngine to cope properly with that difference, ensure that the hdr file contains the proper "description" fields with all the appropriate information about reference frame, datum etc… !! If there is a *.hdr* and *.txt* in the directory, AMSTerEngine will consider only the *.txt*. Beware of possible conflict.

- Ensure that your DEM is referred to ellipsoid and not the geoid !

- As for the DEM, if a mask is provided with .txt description file, AMSTerEngine will expect it to be in mathematical order. If provided with *.hdr* it will expect it to be in GIS order.

- If you intend to do mass processing using S1, ensure that the kml you will use to crop the interferometric products overlaps the same bursts that the kml used for the bulk reading.
  The reason of the care required here is just a matter of disk space and processing time: if the kml file used for cropping the interferogram has a footprint that covers less bursts than what has been read (and hence assembled in the primary image (*PRM*) directory as …/SAR_CSL/*SAT/TRK*/NoCrop/*PRM*.csl/Data/SLCData.*pol*), AMSTerEngine will assemble again the required bursts in a new SLCData.*pol* but saved this time in your *PRM_SCD*/i12 directory (where *SCD* stands for the secondary image name). Hence it will make a new copy of the primary image in **EACH** processing pair, which would be a huge waste of time and disk space. There is of course no problem if you set no kml as a crop region for interferometric computation.

- You can run the script zz_Utilities_MT/***Check_Installation.sh*** which will test some of the required components and configuration. This might be useful at first setup, but also after some upgrades of the OS or whatever you upgrade on your computer(s)…

0.8) Hard coded lines in scripts:

Although it was attempted to avoid it as much as possible, some specific information (e.g. related to the user's computer infrastructure) are sometimes hard-coded in some scripts. We can observe the following type of scripts using hard coded lines:

1. Those related e.g. to the computer infrastructure or some parameters related to the satellite and/or track and/or target. These parameters needed by these scripts are as much as possible stored in a single file named ***__HardCodedLines.sh*** (see below);
2. Those where some parameter(s) or plot style(s) are defined and which are usually OK, thought the user may want/need to change the default value (see below);
3. Scripts to be launched from a terminal without profile definition, hence needing to source the .bashrc (see below);
4. Those that are really specific to the target(s), e.g. containing names, coordinates, positions, dates etc… This is typically the case for the cron scripts (see below);
5. And some specific cases that need short explanations (see below).

When hard coded lines are required, it is mentioned in the header of the script. In addition, the hard coded lines are usually located at the beginning of the script in a dedicated section that looks like:

```
# vvv ----- Hard coded lines to check --- vvv
# Explanation about parameter to define
    Parameter=…
# ^^^ ----- Hard coded lines to check --- ^^^
```

Let's briefly discuss these 5 types of scripts using hard coded lines.

0.7.1 Hard coded lines in *__HardCodedLines.sh*:

Whenever possible, the hard coded lines are gathered in a dedicated file named ***__HardCodedLines.sh***, which is stored in /SCRIPTS_MT. That file is sourced by the scripts requiring these parameters definitions. Parameters are called in the script as functions.

These functions are arranged in sections entitled by the name of the script(s) calling them. Comments with some explanations about the parameter(s) are provided for each function.

Review carefully that file and tune it to your own needs. Ensure to get a backup that will not be overwritten when you update the scripts of your AMSTer toolbox e.g. from the GitHub[2] repository.

The scripts using the ***__HardCodedLines.sh*** file are the following:

- ***__SplitCoreg.sh***
- ***__SplitSession.sh***
- ***ALL2GIF.sh***
- ***FUNCTIONS_FOR_MT.sh***
- ***Geocode_from_ALL2GIF.sh***
- ***Plot_All_EW_UP_ts_inDir.sh***
- ***Plot_All_LOS_ts_inDir.sh***
- ***Plot_Diff_TS.sh***
- ***plot_lcurve.sh***
- ***PlotBaselineGeocMSBAS.sh***
- ***PlotBaselineGeocMSBASmodeTXT.sh***
- ***PlotBaselineGeocRaster.sh***
- ***plotBaselines.sh***
- ***PlotTS_all_comp.sh***
- ***RenamePath_Volumes_MNTtoVOL.sh***
- ***RenamePath_Volumes_VARtoMNT.sh***
- ***RenamePath_Volumes_VARtoVol.sh***
- ***RenamePath_Volumes.sh***
- ***RenamePathAfterMove_in_SAR_MASSPROC.sh***
- ***RenamePathAfterMove_in_SAR_SM_AMPLITUDES.sh***
- ***RenamePathInPlace_Volumes_VOLtoMNT.sh***
- ***SinglePairNoUnwrap.sh***
- ***TimeSeriesInfo_HP.sh***
- ***UpdateAMSTerEngine.sh***

[2] https://github.com/AMSTerUsers/AMSTer_Distribution

0.7.2 Hard coded lines in scripts that should be OK:

Some other scripts have hard coded lines for defining some parameter(s) or plot style(s), which are usually OK. However, the user may want/need to change these default value.

See e.g.:

- ***_Check_S1_SizeAndCoord.sh*** (Tolerance in position of the corners of the Area of Interest)
- ***_Remove_All_Files_With_PairDate_Bp.sh*** (Path to base directory where to operate)
- ***AmpAmpAmp.sh*** (Plot style in Fiji commands)
- ***AmpAmpCoh.sh*** (Plot style in Fiji commands)
- ***AmpDefo_map.sh*** (Plot style in Fiji commands)
- ***CronLaunch2Term.sh*** (path to /SCRIPTS_MT/)
- ***Fiji_Amp_Defo_Coh.sh*** (Plot style in Fiji commands)
- ***Fiji_DEM_Defo_Coh.sh*** (Plot style in Fiji commands)
- ***Fiji_Open_Envi2tif.sh*** (Plot style in Fiji commands)
- ***Plot_All_EW_UP_ts_inDir.sh*** (Plot style)
- ***Plot_All_LOS_ts_inDir.sh*** (Plot style)
- ***Plot_Diff_TS.sh*** (Plot style)
- ***plot_lcurve.sh*** (Plot style)
- ***PlotBaselineGeocMSBAS.sh*** (Plot style)
- ***PlotBaselineGeocMSBASmodeTXT.sh*** (Plot style)
- ***PlotBaselineGeocRaster.sh*** (Plot style)
- ***plotBaselines.sh*** (Plot style)
- ***PlotTS_all_comp.sh*** (Plot style)
- ***Read_All_Img_Rebuild.sh*** (Expected polarization)
- ***ReGeocode_fromList.sh*** (Which products to re-geocode & recompute rasters)
- ***ReGeocode_ManuallyUnwrapped_SinglePair.sh*** (Which products to re-geocode & sat def)
- ***ReGeocode_SinglePair.sh*** (Which products to re-geocode)
- ***RerunFromGeoc.sh*** (Which products to re-geocode)
- ***SinglePairNoUnwrap.sh*** (Max. Sigma considered for successful Coreg & Crop fct)
- ***TSmap2moviegif.sh*** (Plot style in Fiji commands)

See also the gnu templates for plots in /SCRIPTS_MT/TemplatesForPlots

Sometimes, hard coded parameters are not listed at the beginning of the script. However, this happens when changing them is not expected, though it remains possible. For instance:

- ***Envi2ColorKmz.sh:*** may want to change path to the colour table to get a better one
- ***Filter_Interpolate_DefoMaps.sh:*** extension of the files to filter and width of the median filter
- ***Filter_Interpolate_SingleMap.sh:*** width of the median filter
- ***_Remove_All_Files_With_PairDate_Bp.sh:*** *Path to dir where pairs are stored*
- ***Check_S1_SLCImageInfo.sh:*** *Preferred polarization for S1 images*

0.7.3 Hard coded lines specific to targets:

Some scripts have hard coded lines that are specific to the target of interest. Each user will probably define its own set of script such as those for the cron jobs.

At ECGS we have for instance the following ones, which can be an inspiration to build your own processing chain (see also section 7):

- ***Domuyo_S1_Step1_Read_SMCoreg_Pairs.sh*** (specific for processing Domuyo region)
- ***Domuyo_S1_Step2_MassProc.sh*** (specific for processing Domuyo region)
- ***Domuyo_S1_Step3_MSBAS.sh*** (specific for processing Domuyo region)
- ***Lazy_prepa_msbas_all_sets_andPlotBaselines_VVP.sh*** (batch of commands for a specific region)
- ***Nyigo_Nyam_Crater_S1_Read_AMPLI.sh*** (specific for processing amplitude in DRC craters)
- ***Shadows_S1_Nyam_Desc.sh*** (specific for processing amplitude in DRC craters)

Etc…

Also, some small tools are sometimes designed for specific manual tasks.

- ***CompareTimeSeries.sh*** (Path to time series and where to store the results)
- ***Del_All_dir_from_list.sh*** (String to search)
- ***Del_All_files_in_Dirs_Named_With.sh*** (String to search and path to dir to clean)
- ***r4togif.sh*** (Date tag & crop size)
-

Etc…

0.7.4 Hard coded lines for script launched from a terminal without profile definition:

Some scripts must be launched from cron jobs or from a terminal open from a command line in another terminal. In these cases, there is no profile definition yet. Hence the script MUST start by sourcing the .bashrc to get the state variables and the AMSTer environment variables.

Ensure that a *.bashrc* is present in your $HOME directory, which is done automatically if you installed AMSTer with the installation script. Hence, in principle, there is nothing to change.

The following scripts start by sourcing the .bashrc:

- ***ALL2GIF.sh***
- ***AllProd2GIF.sh***
- ***AmpAmpAmp.sh***
- ***AmpAmpCoh.sh***
- ***AmpDefo_map.sh***
- ***build_header_msbas_criteria_From_nvi_name_WithoutAcqTime.sh***
- ***build_header_msbas_criteria.sh***
- ***Check_Installation.sh***
- ***FUNCTIONS_FOR_MT.sh***
- ***MultiLaunch_Ampli_Coh.sh***
- ***MultiLaunch_ForMask.sh***
- ***MultiLaunch.sh***
- ***plot_Multi_BaselinePlot.sh***
- ***Read_All_Img_Rebuild.sh***
- ***Read_All_Img.sh***
- ***Remove_BrokenLinks_and_Clean_txt_file.sh***
- ***SinglePair.sh***
- ***TimeSeriesInfo_HP.sh***
- ***TS_AddLegend_EW_UD.sh***
- ***TS_AddLegend_LOS.sh***

and all the cron jobs of course.

0.7.5 Some specific cases:

The script ***00_RasterPixelCoord.py*** contains the hard coded path to where the deformation maps are stored and where the ***PlotTS.sh*** script is located. Unlike in nearly all the other scripts, this cannot be set as a parameter here because that script is used from the python console in QGIS (see e.g. section 6.4, which cannot source the .bashrc etc.

Also, ***00_RasterPixelCoord.py*** uses ***PlotTS.sh***. In general, when used from the terminal, ***PlotTS.sh*** would get the paths to AMSTerEngine, SCRIPTS_MT and the gnu utilities from the state variables defined in the .bashrc. However, when used from ***00_RasterPixelCoord.py***, that is from the python console in QGIS, these paths cannot not be interpreted from the state variable. For that reason, they are set as hard coded parameters in these scripts.

The script ***CronLaunch2Term.sh*** aims at launching a script in a new terminal from a cron job. It requires the path to the SCRIPTS_MT because it can't be read from the .bashrc.

The script ___***Tune_your_AMSTer.sh*** aims at changing hard coded lines in script. Each change, for each script, must be hard coded (See section 9.15).

The script ___***TEST_ALL_TYPE_OF_PROCESS.sh*** aims at running several processings using data from several sensors in order to check that the AMSTer runs smoothly, e.g. after updates. Of course, it was designed for our own needs at ECGS (sensors and images available at ECGS). Corresponding hard-coded lines must be adapted depending on your data.

The scripts ***_Check_Cron_MassProcessed.sh***, ***_Check_Cron_Ampli.sh, l3.sh*** and ***psn*** aim at testing the status of the last processes run from the cron jobs at ECGS. It requires an in-depth revision to be adapted to another automated set of processes.

The scripts ***ChgeAll_LaunchParamFiles.sh*** aims at changing a given parameter in all the ***LaunchMTparam.txt.*** while ***Chge_Several_Criteria_LaunchParamFiles.sh*** runs several occurrences of ***ChgeAll_LaunchParamFiles.sh***. This must be tuned to your own needs (See Section 9.13).

And of course, although this is not a script, all your ***LaunchMTparam.txt*** files must be tuned to your targets.

0.9) Python scripts:

AMSTer toolbox contains some *python* script. They were written for *python3* and are called in scripts with #!/opt/local/bin/*python*. Hence you must ensure that your *python3* can be launched as *python* from that directory. This may require that you create an alias for that from your *python3* installation to /opt/local/bin/*python*. This will allow you to avoid modifying your python scripts again and again after each synchronization with the GitHub repository when you will update your AMSTer Toolbox. If AMSTer was installed by the installer script, this should be done by default for you. Here are some python scripts used in AMSTer Toolbox :

- *00_RasterPixelCoord.py*
- *checkOnlyNaN.py*
- ***FLIPproducts.py.sh*** and ***FLOPproducts.py.sh***
- *MeanCoh.py* and *MeanAmpl.py*
- *Norm.py*
- *NaN2zero.py* and *zero2NaN.py*
- *byte2float.py* and float2byte.py
- *Mask_Builder.py*
- *FiltMedian.py*
- *flip_raster.py*

Note: *python2* might not be compatible because it had several differences with *python3*, e.g. (see web for more info):

	Python V2	Python V3
print	print “your text”	print (“your text”)
xrange	xrange	range

0.10) Dependencies:

AMSTer needs at least the following:

- AMSTerEngine (ensure to get the latest version and in any case at least **V20231018**).

- MSBAS at least V2: cfr Sergey's web page, in "Research Interest": http://www.insar.ca

- *Fiji* (*ImageJ*)

- ***sed***, ***awk***, ***grep***, ***seq, find, date, uniq, stat, readlink, xargs…***: these commands are usually provided by default on most of the Linux and Mac computers. However, some version (e.g. on Mac computers) are not fully compatible with gnu version and syntax may slightly vary. For the sake of portability, scripts are developed for gnu version of these functions.
  They are expected:
  - to be stored in a specific directory ${PATHGNU}/ (see description of the state variables below).
  - to be named with a "g" (e.g. ***gsed*** instead of ***sed***), at least for ***sed***, ***awk***, ***grep***, ***date, stat.***

  These functions are called in the scripts with their path, e.g. ${PATHGNU}/**gsed** (see description of the state variables below).
  As a consequence, it is MANDATORY to have these functions installed (or copied or linked) in a directory that will be defined as the ${PATHGNU} state variable in your *.bashrc* (avoid using *.bash_profile* because all scripts that needs your state variable call *.bashrc*; if needed, link them).
  Just in case one of these functions would be called without its g-name in a script, it is recommended to have them in ${PATHGNU} with both names, that is with and without heading g (that is **sed** and g**sed**, **awk** and **gawk**, **grep** and **ggrep**, **date** and **gdate**, **stat** and **gstat**).
  This can be simply done with a link.
  If you installed AMSTer with the installer, this should have been done for you.

- **convert** (from *ImageMagick*)

- **bc**

- *gmt*, **gdal**

- *snaphu*.

- *cpxfiddle*: see https://github.com/TUDelftGeodesy/Doris/tree/master/sar_tools.

- *Python3*

- *gnuplot*

- *osascript* (Mac) or *x-terminal-emulator* (Linux)

- *XnView* or *GIMP* (or equivalent, to open sun raster files)

- If you process ENVISAT and/or S1 data, you will of course require their orbits and they will have to be stored in the appropriate directory (see state variables below).

0.11) Environmental (state) variables:

To operate, your *.bashrc* (if you use *.bash_profile* link it to *.bashrc* because all scripts that need you state variables etc call *.bashrc*) must be updated with the following state variables. If you installed AMSTer with the installer, this should have been done for you.

Beware of the spelling!:

- **$PATH_SCRIPTS:**
  Define the path to the folder containing the AMSTer *scripts* (named SCRIPTS_MT), i.e. add the following line to your profile:
  export PATH_SCRIPTS=/${HOME}/SAR/AMSTer
  Note: the scripts MUST be stored in a directory named SCRIPTS_MT. Hence, they will be located in: /${HOME}/SAR/AMSTer/SCRIPTS_MT

- **$PATHGNU:**
  Define the path to the gnu version of executable *gsed*, *gawk*, *ggrep*, *gnuplot*, *gseq, find* and *gdate*, i.e. add a line like this to your profile:
  export PATHGNU=*/opt/local/bin*
  Note: ensure that /opt/local/bin is in your path before the /usr/bin to ensure that it would in priority take into account your GNU commands in case of doubt.

- **$PATHFIJI:**
  Define the path to the executable *Fiji* (which is *ImageJ*), i.e. add a line like this to your profile (for Mac):
  export PATHFIJI=*/Applications/Fiji.app/Contents/MacOS*

- **$PATHCONV:**
  Define the path to the executable *convert* (which is a function of *ImageMagick*), i.e. add a line like this to your profile:
  export PATHCONV=*/opt/local/bin*

- **$PATHTOCPXFIDDLE:**
  Define the path to the executable *cpxfiddle*, i.e. add a line like this to your profile:
  export PATHTOCPXFIDDLE =*/usr/local/bin*

- **$PATH_1650, $PATH_3600, $PATH_3601 and $PATH_3602:**
  Define the path to 4 disks where the scripts are supposed to find some directories. Although naming 1650, 3600, 3601 and 3602 is associated to servers at ECGS, it might be the easiest solution for you to keep this naming as it is used nearly everywhere. Also, even if you do not work with 4 separate disks, it is required to have these 4 variables (see below “Expected architecture of disks”). E.g (for Mac):
  export PATH_1650=*/Volumes/hp-1650-Data_Share1*
  export PATH_3600=*/Volumes/hp-D3600-Data_Share1*
  export PATH_3601=*/Volumes/hp-D3601-Data_RAID6*
  export PATH_3602=*/Volumes/hp-D3602-Data_RAID5*

- **$PATH_DataSAR:**
  Define the path to a disk where mandatory data will be stored, e.g in a directory named **SAR_AUX_FILES** that will contain:
  - **DEM**: directory that contains the digital elevation model(s)
  - **EGM**: directory that contains the geoid information
  - **MASKS**: directory that contains the mask(s) if required
  - **ORBITS**: directory that contains the ENVISAT and/or Sentinel1 orbits

- **$ENVISAT_PRECISES_ORBITS_DIR:**
  Define the path to the folder containing the Envisat orbits, i.e. add the following line to your profile:
  ENVISAT_PRECISES_ORBITS_DIR=${PATH_DataSAR}/SAR_AUX_FILES/ORBITS/ENV_ORB
- **$S1_ORBITS_DIR:**
  Define the path to the folders containing the Restituted (i.e. fast) and Precise Sentinel 1orbits, i.e. add the following line to your profile:
  export S1_ORBITS_DIR=${PATH_DataSAR}/SAR_AUX_FILES/ORBITS/S1_ORB
  Note that the function ***updateS1Orbits*** will automatically create in that directory the following 2 directories to store respectively the Precise and Restituted orbits: /AUX_POEORB and /AUX_RESORB

- **$EARTH_GRAVITATIONAL_MODELS_DIR:**
  Define the path to the folders containing a directory named EGM. This must contain the geoid file WW15MGH.DAC in a sub directory EGM96. The geoid will be mandatory to correct the DEM heights from the geoid heights.
  For this, Earth Gravity Model EGM96 WW15MGH.DAC must be downloaded from site below and stored in that $EARTH_GRAVITATIONAL_MODELS_DIR/EGM96 (link checked 05 10 2022):
  https://web.archive.org/web/20130314064801/http://earth-info.nga.mil/GandG/wgs84/gravitymod/egm96/binary/WW15MGH.DAC
  export EARTH_GRAVITATIONAL_MODELS_DIR=${PATH_DataSAR}/SAR_AUX_FILES/EGM

- **$PATH**
  Add to your $PATH the path to the folders containing respectively the *AMSTerEngine* and *msbas* software and to folder containing AMSTer Toolbox scripts (SCRIPTS_MT), utilities scripts (zz_Utilities_MT), EXEC (used during installation by the installer script)… i.e. add lines like this to your profile:

```
PATH=$PATH:/Users/nicolas/SAR/EXEC
PATH=$PATH:/Users/nicolas/SAR/AMSTer/MSBAS
PATH=$PATH:/Users/nicolas/SAR/AMSTer/AMSTerEngine
PATH=$PATH:/Users/nicolas/SAR/AMSTer/SCRIPTS_MT/zz_Utilities_MT_Ndo
PATH=$PATH:/Users/nicolas/SAR/AMSTer/SCRIPTS_MT/zz_Utilities_MT
PATH=$PATH:/Users/nicolas/SAR/AMSTer/SCRIPTS_MT/_cron_scripts
PATH=$PATH:/Users/nicolas/SAR/AMSTer/SCRIPTS_MT/AMSTerOrganizer
PATH=$PATH:/Users/nicolas/SAR/AMSTer/SCRIPTS_MT/optimtoolbox
PATH=$PATH:/Users/nicolas/SAR/AMSTer/SCRIPTS_MT
```

Hence you should have something in your *.bashrc* such as :

For a Mac installation:

```
[…]
PATH="$PATH:/opt/local/bin:/usr/local/bin:/usr/bin:/bin:/usr/sbin:/sbin"

# AMSTer PATHS
##################
PATH=$PATH:/Users/YourAccount/SAR/EXEC
PATH=$PATH:/Users/YourAccount/SAR/AMSTer/MSBAS
PATH=$PATH:/Users/YourAccount/SAR/AMSTer/AMSTerEngine
PATH=$PATH:/Users/YourAccount/SAR/AMSTer/SCRIPTS_MT
PATH=$PATH:/YourAccount/doris/SAR/AMSTer/SCRIPTS_MT/AMSTerOrganizer
PATH=$PATH:/Users/YourAccount/SAR/AMSTer/SCRIPTS_MT/zz_Utilities_MT_Ndo
PATH=$PATH:/Users/YourAccount/SAR/AMSTer/SCRIPTS_MT/zz_Utilities_MT
PATH=$PATH:/Users/YourAccount/SAR/AMSTer/SCRIPTS_MT/_cron_scripts
PATH=$PATH:/Users/YourAccount/SAR/AMSTer/SCRIPTS_MT/optimtoolbox


# AMSTer VARIABLES
##################
export PATH_HOMEDATA=/Users/YourAccount/
export PATH_3602=/Volumes/hp-D3602-Data_RAID5/
export PATH_3601=/Volumes/hp-D3601-Data_RAID6/
export PATH_3600=/Volumes/hp-D3600-Data_Share1/
export PATH_1650=/Volumes/hp-1650-Data_Share1/
export PATH_DataSAR=/Volumes/DataSAR/

export EARTH_GRAVITATIONAL_MODELS_DIR=${PATH_DataSAR}/SAR_AUX_FILES/EGM
export ENVISAT_PRECISES_ORBITS_DIR=${PATH_DataSAR}/SAR_AUX_FILES/ORBITS/ENV_ORB
export S1_ORBITS_DIR=${PATH_DataSAR}/SAR_AUX_FILES/ORBITS/S1_ORB

export PATH_SCRIPTS=/Users/YourAccount/SAR/AMSTer
export PATHCONV=/opt/local/bin
export PATHFIJI=/Applications/Fiji.app/Contents/MacOS/
export PATHGNU=/opt/local/bin
export PATHTOCPXFIDDLE=/Users/YourAccount/SAR/EXEC/
export PATH=$PATH
```

Attention: PATH_SCRIPTS should **not** be PATH_SCRIPTS=/${HOME}/SAR/AMSTer/SCRIPTS_MT

For a Linux installation:

```
[…]

# If not running interactively, don't do anything
#case $- in
#    *i*) ;;
#      *) return;;
#esac

[…]

PATH="/usr/local/bin/:/opt/local/bin:/usr/bin:/bin:/usr/sbin:/sbin"

# AMSTer PATHS
###################
PATH=$PATH:/usr/local/tigervnc/bin/
PATH=$PATH:/home/YourAccount/SAR/EXEC
PATH=$PATH:/home/YourAccount/SAR/AMSTer/MSBAS
PATH=$PATH:/home/YourAccount/SAR/AMSTer/AMSTerEngine
PATH=$PATH:/home/YourAccount/SAR/AMSTer/SCRIPTS_MT
PATH=$PATH:/home/YourAccount/SAR/AMSTer/SCRIPTS_MT/AMSTerOrganizer
PATH=$PATH:/home/YourAccount/SAR/AMSTer/SCRIPTS_MT/zz_Utilities_MT_Ndo
PATH=$PATH:/home/YourAccount/SAR/AMSTer/SCRIPTS_MT/zz_Utilities_MT
PATH=$PATH:/home/YourAccount/SAR/AMSTer/SCRIPTS_MT/_cron_scripts
PATH=$PATH:/home/YourAccount/SAR/AMSTer/SCRIPTS_MT/optimtoolbox

# AMSTer VARIABLES
###################
export OPENBLAS_NUM_THREADS=1
export JAVA_HOME="/usr/lib/jvm/java-11-openjdk-amd64"
export PATH_3602=/mnt/3602/
export PATH_3601=/mnt/3601/
export PATH_3600=/mnt/3600/
export PATH_1650=/mnt/1650/
export PATH_DataSAR=/mnt/syno_sar/

export EARTH_GRAVITATIONAL_MODELS_DIR=${PATH_DataSAR}/SAR_AUX_FILES/EGM
export ENVISAT_PRECISES_ORBITS_DIR=${PATH_DataSAR}/SAR_AUX_FILES/ORBITS/ENV_ORB
export S1_ORBITS_DIR=${PATH_DataSAR}/SAR_AUX_FILES/ORBITS/S1_ORB

export PATH_SCRIPTS=/home/YourAccount/SAR/AMSTer
export PATHCONV=/usr/bin
export PATHFIJI=/home/YourAccount/SAR/EXEC/Fiji.app/
export PATHGNU=/usr/bin
export PATHTOCPXFIDDLE=/home/nicolas/SAR/EXEC/
export PATH=$PATH
```

Attention: PATH_SCRIPTS should **not** be PATH_SCRIPTS=/${HOME}/SAR/AMSTer/SCRIPTS_MT

See also installation notes:

Install_AMSTer_Mac_V*x.x*.pdf, or

Install_AMSTer_Linux_V*x.x*.pdf

where *x.x* is the number of the latest version of that manual.

0.12) The external DEM:

To perform differential interferometry, AMSTerEngine requires an external DEM operates (unless you need to create a DEM based on interferometric pairs; see usage of ***SinglePair.sh*** in TOPO mode).

That DEM must have the following specificities:
- As most -if not all- the InSAR processors, AMSTerEngine requires a DEM with reference to ellipsoidal heights, not to the geoid. If this is not the case, altitudes being wrongly interpreted, small offsets will be visible at geocoding.
- The DEM is expected to be in the mathematical order (i.e. not the GIS order)

AMSTerEngine format will be composed of a binary matrix and a header file named with a .txt extension.

The DEM can be created from srtm tiles using ***agregateSRTMTile*** function from AMSTerEngine. Since May 2022, the function ***agregateSRTMTile*** will create a DEM directly in the appropriate format by stitching SRTM tiles and correct the heights from the difference between ellipsoidal and geoidal heights.

Older version did not correct the srtm from the geoidal height. However, ***agregateSRTMTile*** can also be used to apply a posteriori that geoidal height correction to DEM in AMSTerEngine format by launching the following command:

agregateSRTMTiles $PATH_DataSAR/SAR_AUX_FILES/DEM/*DEM_REGION/dem_name_to_correct*

It will then modify the DEM accordingly and change the .txt file by adding a line confirming that the DEM is related to the ellipsoid.

Note: If you run several times that command, it will not change the reference of the DEM several times (as long as the .txt file contains the line saying that the correction has already been applied).

Note: to apply the geoid correction, an Earth Gravity Model (e.g. EGM96 geoid) must be stored in $PATH_DataSAR/SAR_AUX_FILES/DEMS (see below).
The EGM96 file named WW15MGH.DAC can be downloaded from (link checked 05 10 2022):
https://web.archive.org/web/20130314064801/http://earth-info.nga.mil/GandG/wgs84/gravitymod/egm96/binary/WW15MGH.DAC

To create a DEM from srtm tiles:

- Log to http://earthexplorer.usgs.gov
  (Login and pwd required. Account can easily be created).
- Download the srtm1 tiles in BIL format.
- Move the files in a dedicated dir (e.g. $PATH_DataSAR/SAR_AUX_FILES/DEM/*DEM_REGION*).
- Decompress the files and delete the zip files. You stay with several subdirs where .bil, .blw, .hdr and .prj files are stored for each tile.
- Ensure you have in your .bashrc a state variable named:
  EARTH_GRAVITATIONAL_MODELS_DIR=$PATH_DataSAR/SAR_AUX_FILES
  and that it contains a directory named **/EGM96** that contains the Geoid file named WW15MGH.DAC (downloaded from web).

- Launch the command :
  agregateSRTMTiles $PATH_DataSAR/SAR_AUX_FILES/DEM/*DEM_REGION*

- After a while you will find two files named by the same way as the directory with and without .txt extension. The one without extension is the DEM and the one with the .txt extension is the one that contains the details about the DEM and which is used for for filling the slantRange.txt file during the AMSTer processing.

- If you move the DEM, it is advised to change the path indicated in the *DEM*.txt file accordingly to the new place where the DEM is stored.

To create a DEM from another source:

If you have an external DEM created from any other source than srtm, you can convert it in the appropriate format using the script ***DEM_Envi_hdr2AMSTer_txt.sh***. That script allows to convert tif or ENVI DEM in format expected by AMSTerEngine, that is in mathematical order, and referred to the ellipsoid (see below).

For instance, if you need to transform a Copernicus DEM into AMSTer format DEM, you will need to:
- Download the tif tiles
- Merge them with QGIS for instance
  (Processing toolbox -> GDAL/OGR -> Miscellaneous -> Merge as Float32, BIL or tif)
- run the following command:
  DEM_Envi_hdr2AMSTer_txt.sh *YOUR_PATH_TO/Your_COPERNICUS_DEM.tif*

And because, following the Copernicus documentation[3],
  - the horizontal reference datum is WGS84-G1150; EPSG 4326
  - the vertical reference datum is the Earth Gravitational Model 2008 (EGM2008; EPSG 3855), that is EGM2008 geoid undulation values with respect to WGS84,

when the script will ask you if "*your DEM referred to the Ellipsoid (E), Geoid (G) or you do not know (Q)"*, you will have to answer G (for Geoid). It will then refer your Copernicus DEM in AMSTer format to the Ellipsoid as expected by AMSTer.

Attention: there are two types of ENVI formats, both made of a binary matrix and a text header with a .hdr extension, but with different coordinate reference:
- the ESRI ArcGis: the coordinate of a pixel is the coordinate of its centre (as geotif format or as for AMSTerEngine format)
- the ENVI HARRIS: the coordinate of a pixel is the coordinate of its top left corner.

One can distinguish them by their hdr file (see below).

Although these two ENVI formats are very similar, that small difference in the way they referred the coordinate of a pixel may introduce geocoging error if not properly taken into account. The script ***DEM_Envi_hdr2AMSTer_txt.sh*** (which uses a python script *flip_raster.py* by Warmerdam & Perry) can manage geotif and both types of ENVI format and will take care of that difference of reference. Just beware that if you create artificially your ENVI format, you must ensure that you create the header in the form that corresponds to your reference.

[3] https://spacedata.copernicus.eu/documents/20126/0/GEO1988-CopernicusDEM-SPE-002_ProductHandbook_I1.00.pdf

Example of ESRI ArcGis header file:
(see https://desktop.arcgis.com/en/arcmap/10.3/manage-data/raster-and-images/bil-bip-and-bsq-raster-files.htm):
For instance, it uses *nrows* and *ncols* for image size.

```
Sample BIL header file
Lines that don't begin with a keyword are treated as comments.
nrows 1024  Comments can be placed here as well.
ncols 1024
nbands 3
nbits 8
layout bil
skipbytes 128
```

Example of ENVI HARRIS header file
(see https://www.l3harrisgeospatial.com/docs/enviheaderfiles.html):
For instance, it uses *lines* and *samples* for image size.

```
description = {
  Registration Result. Method1st degree Polynomial w/ nearest neighbor [Wed Dec 20
23:59:19 1995] }
samples = 709
lines   = 946
bands   = 7
header offset = 0
file type = ENVI Standard
data type = 1
interleave = bsq
sensor type = Landsat TM
byte order = 0
map info = {UTM, 1, 1, 295380.000, 4763640.000, 30.000000, 30.000000, 13, North}
z plot range = {0.00, 255.00}
z plot titles = {Wavelength, Reflectance}
pixel size = {30.000000, 30.000000}
default stretch = 5.0% linear
band names = {
 Warp (Band 1:rs_tm.img), Warp (Band 2:rs_tm.img), Warp (Band 3:rs_tm.img), Warp
(Band 4:rs_tm.img), Warp (Band 5:rs_tm.img), Warp (Band 6:rs_tm.img), Warp (Band
7:rs_tm.img)}
wavelength = {
  0.485000,  0.560000,  0.660000,  0.830000,  1.650000, 11.400000,  2.215000}
fwhm = {
 0.070000, 0.080000, 0.060000, 0.140000, 0.200000, 2.100000, 0.270000}
```

0.13) The masks:

AMSTerEngine can mask interferograms at the unwrapping step. No weight is given to masked pixels during the unwrapping process. If the unwrapping is performed with *snaphu*, the deformation maps are then multiplied by the mask in order to only display the trusted pixels.

For that, AMSTerEngine < V20230928 expects a mask with the following characteristics:

- the mask must be ENVI Harris (not ESRI) format (!)
- the name of the mask must have no extension compared to its header, e.g.
  name_of_your_mask.what_ever_your_want
  name_of_your_mask.what_ever_your_want.hdr
- the zone must be GREATER than image,
- the mask must be in Lat Long (not UTM !)
- the mask must be in Bytes, filled with 1 (to keep) or 0 (to mask)
- the mask can NOT contain NaN !!

Note : The mask is used during the unwrapping process in conjunction with a parameter named COHCLNTHRESH (defined in *LaunchMTparam.txt*), which tells *snaphu* to give no weigh to pixels where the coherence is below that value (see also the paragraph 3.4 about ***MultiLaunch_ForMask.sh***). This can influence the masking process as follow: pixels with mask = 1 will always be kept. However, pixels with mask = 0 will normally be masked UNLESS their coherence is above that threshold. Hence, during a mass processing, you can recover some good quality pixels for some pairs while they are usually of bad coherence and hence masked for all the other ones.

Having said so, if your mask is aiming at filtering out pixels where you NEVER want any information (e.g. on water bodies), then you must set the COHCLNTHRESH = 1, so that it will never use pixels where mask = 0.

Unfortunately, if you mask combines regions where you never want any information (e.g. on water bodies) and regions where coherence is usually bad except at some occasions (e.g. for vegetated areas, created by computing a mean coherence and keeping only the pixels with a value above a given threshold), by setting the COHCLNTHRESH = 1, you will never recover the information when coherence is good enough. And if you set the COHCLNTHRESH to a value between 0 and 1, you will recover some pixels located where you never want to (e.g. the water bodies).

To solve this issue, a new masking method was introduced since AMSTerEngine V20230928. From that version the masks are inverted and multi-layers, that is:

- 0 is to keep (no mask applied)
- 1 is always masked (e.g. water bodies area)
- 2 is masked during unwrapping except if coherence is above a given threshold (see parameter COHCLNTHRESH in *LaunchMTparam.txt*)

You can find several scripts to create and/or transform files to such a kind of mask characteristics (see annexes).

Example to create mask for AMSTerEngine before V20230928:
Rough procedure to create a mask based on water bodies shape file.

1. Download a shape file
2. Process a ***SinglePair.sh*** with a Forced geocoding (no need to unwrap) on an area larger than the images you will need to mask

3. In QGIS import the shape file and a copy (because original file will be changed) of the coherence computed here above
4. RASTERIZE from GDAL toolbox:
    - Input layer = shape
    - Input raster = coherence
    - Fixed value to burn = 2
5. Raster Calculator:
    - If (coherence < 2 , 0 ,1)
    - Save as “mask”
    - Output format = ENVI.hdr (not ESRI !)
    - As Lat Long
      (if needed, use the script ***UTM2LL.sh***, which uses the gdal command gdalwarp -of ENVI -t_srs EPSG:4326 *UTMfileName LLfileName*)
6. Replace NaN with zeros in mask (e.g. using ***NaN2zero.py***)
7. Transform floats to bytes (e.g. using ***float2byte.py***)
8. Copy the mask.hdr as mask.zeroBytes.hdr
9. Edit mask.zeroBytes.hdr and change “data type = 4” with “data type = 1”

Example to create a multi-level mask for AMSTerEngine from V20230928:

1. Invert your old water bodies mask using *Change_Val.py* (see script for usage) in order to change pixels with value = 1 as value = 2, then do it again to change the 0 in 1 and again to finally change the 2 in 0. Your new water bodies mask is now made of 0 (for pixels to keep) and 1 (for pixels to always mask). This can also be done in one step using ***InvertWaterMask.sh*** (see script).

2. Invert you mean coherence mask using *Change_Val.py* to change the 0 in 2, then the 1 in 0. Your new mean coherence mask is now made of 0 (pixels to keep) and 2 (pixels to mask if coherence is below COHCLNTHRESH). This can also be done in one step using ***InvertCohMask.sh*** (see script).

3. Combine (sum) the two masks in QGIS (because they probably do not have the same size) or with Combine_mask_Water_Coh.py (if they have the same size). The combined mask has now values 0, 1, 2 or 3.

4. Transform the combined mask (which is in float) into bytes using e.g. float2byte.py

5. Change the pixels with values 3 in 1 using *Change_Val.py*. Your final product is now as expected : 0 means always keep, 1 means never keep, and 2 means pixels not to keep unless the coherence is above COHCLNTHRESH.

6. Save your new mask e.g. as WaterBodies_CohAboveThresh_012 in ENVI format (_012 is to remind that it is made of these 3 values).

0.14) Notes for Linux:

- See the document Install_AMSTer_Linux_V*x.x*.pdf (where *x.x* is the number of the latest version of that manual).

- Because in Linux *sed* is *gsed* anyway, copy or better, link *sed* as *gsed* in the same directory as where *gawk* is.

- If you want to use some of the scripts with a cron, Linux may require to define the PATH state variables in a system-wide environment.
  - Create a file in /etc/profile.d/, such as : /etc/profile.d/*yourname*_sar.sh
  - Add your state variables in that file, e.g:

```
export PATH_1650="/mnt/1650"
export PATH_3600="/mnt/3600"
export PATH_3601="/mnt/3601"
export PATH_3602="/mnt/3602"

export PATH_SynoData="/mnt/syno_data"
export PATH_SCRIPTS="/home/nicolas/SAR"
export PATHGNU=/bin
export PATHFIJI=$HOME/SAR/EXEC/Fiji.app/
export PATHCONV=/bin

export EARTH_GRAVITATIONAL_MODELS_DIR=$PATH_DataSAR/SAR_AUX_FILES
export S1_ORBITS_DIR="/mnt/3600/ORBITS/S1_ORB/AUX_POEORB"
export ENVISAT_PRECISES_ORBITS_DIR="/mnt/3600/ORBITS/ENV_ORB/vor_gdr_d"

alias gsed='sed'

PATH=$PATH:/home/nicolas/SAR:/home/nicolas/SAR/AMSTer/AMSTerEngine:/home/nicolas/SAR/AMSTer/SCRIPTS_MT
PATH=$PATH:/home/nicolas/SAR/AMSTer/MSBAS/

etc…
```

  - In your crontab, launch your scripts as follow (at 12h00 in example):

```
00 12 * * * cd /full_path_to_the_dir_where_your_scripts_are/; bash -l -c './your_script.sh'
```

Another option (preferred) is to start your cron tab by defining the path, eg:

```
# m h  dom mon dow   command
SHELL=/bin/bash
BASH_ENV=~/.bashrc

PATH_1650=/mnt/1650
PATH_3600=/mnt/3600
PATH_3601=/mnt/3601
PATH_3602=/mnt/3602
PATH_SynoData=/mnt/syno_data
PATH_DataSAR="/mnt/syno_sar"
PATH_HOMEDATA="/mnt/dell3raid5"

PATH_SCRIPTS=/home/nicolas/SAR/AMSTer

#ENVISAT_PRECISES_ORBITS_DIR=/mnt/3600/ORBITS/ENV_ORB/vor_gdr_d
#S1_PRECISES_ORBITS_DIR=/mnt/3600/ORBITS/S1_ORB/AUX_POEORB
S1_ORBITS_DIR=${PATH_DataSAR}/SAR_AUX_FILES/ORBITS/S1_ORB
ENVISAT_PRECISES_ORBITS_DIR=/mnt/syno_sar/SAR_AUX_FILES/ORBITS/ENV_ORB
EARTH_GRAVITATIONAL_MODELS_DIR=$PATH_DataSAR/SAR_AUX_FILES/EGM


PATHGNU=/usr/bin
PATHCONV=/usr/bin
PATHFIJI=$HOME/SAR/EXEC/Fiji.app/
PATHTOCPXFIDDLE=/${HOME}/SAR/EXEC/

# # # # # Domuyo
15 00 * * * /home/nicolas/SAR/AMSTer/SCRIPTS_MT/_cron_scripts/Domuyo_S1_Step1_Read_SMCoreg_Pairs.sh  > /dev/null 2>&1
45 02 * * * /home/nicolas/SAR/AMSTer/SCRIPTS_MT/_cron_scripts/Domuyo_S1_Step2_MassProc.sh > /dev/null 2>&1
02 15 * * * /home/nicolas/SAR/AMSTer/SCRIPTS_MT/_cron_scripts/Domuyo_S1_Step3_MSBAS.sh  > /dev/null 2>&1
```

0.15) Expected architecture of disks:

For safety reasons, it is advised to store raw data and data in CSL format (i.e. from AMSTerEngine) on different hard drives. For the same reason, it is advised to store mass processing results and msbas times series on different hard drives.

AMSTer Toolbox expects the directory structure as below **(in red = mandatory**). Most of the directories where the intermediate and final results are stored will be created by the scripts. For those which must be created by the user, it is required to follow the following architecture and directories naming (again only the green must be adapted to your satellite(s), orbital mode(s) and region(s)) :

Disk 1 (i.e. PATH_1650):

PATH_1650/kml : where to store kml for cropping, or for coherence check
PATH_1650/ Param_files : where to store your LaunchMTparam.tx filet by */SAT/TRK*….
PATH_1650/SAR_CSL/*SAT/TRK*/NoCrop : where data in csl format are stored (incl. crops)
PATH_1650/SAR_SM/AMPLITUDES/*SAT/TRK/Region* : where amplitudes gif will be computed
PATH_1650/SAR_SM/MSBAS/*Region* : where compatible pairs for msbas are computed
PATH_1650/SAR_SM/RESAMPLED/*SAT/TRK* : where data resampled on Global Primary are saved

Disk 2 (i.e. PATH_3600):

PATH_3600/SAR_DATA/*SAT/…* : where raw data are stored (safer to keep them on a different hard drive than 1650)

Disk 3 (i.e. PATH_3601):

PATH_3601/SAR_MASSPROCESS/*SAT/TRK/Crop* : where directories with each pair computation details and results are stored

PATH_3601/SAR_MASSPROCESS/*SAT/TRK/Crop*/Geocoded : where geocoded maps are stored in sub dir by type (Amp, Coh, Defo etc…)
PATH_3601/SAR_MASSPROCESS/*SAT/TRK/Crop*/GeocodedRasters : where rasters of geocoded maps (for quick look) are stored

Disk 4 (i.e. PATH_3602):

PATH_3602/MSBAS_RESULTS/*Region* : where msbas series are computed

Maybe in one of these disks, or somewhere else where enough storage room is available:

PATH_DataSAR/SAR_AUX_FILES/DEM : where DEMs are stored
PATH_DataSAR/SAR_AUX_FILES/EGM : where geoid is stored
PATH_DataSAR/SAR_AUX_FILES/MASKS/*SAT/TRK* : where coherence masks are computed :
PATH_DataSAR/SAR_AUX_FILES/ORBITS/*SAT* : where orbits are stored

You can of course decide to change the location of these mandatory directories. Ensure consistency with paths defined in the *LaunchMTparam.txt.*

Remember: **Do not use hard drives formatted in FAT format because it does not support symbolic links, which are widely used by AMSTer**

0.16) Organizing the work:

When properly installed, AMSTer should be easy to operate. The largest “difficulty” might be to remember where are stored the data, the auxiliary data, the configuration files, the scripts, the intermediate results etc..

To assist the user to figure out where all the files and results are, a small interface was developed using **pyqt6**. Note that pyqt6 requires recent operating system (at least Mac OSX 12.6 Monterey or Ubuntu 22.04).

It can be opened by launching at the Terminal (which can be closed after it was launched) the script ***AMSTerOrganiser.sh*** (located in SCRIPTS_MT/AMSTerOrganizer/).

It will open a window organized in 5 sections as follow (see figure Figure 1):

1. Three (or more) rows of buttons are displayed at the top (separated by light grey line). Their name and function (i.e. the directory to which they refer) are defined in a config file names SCRIPTS_MT/AMSTerOrganizer/config.txt.

   A good practice would be to define in the first row the buttons corresponding to the logical progression of data progressing:
   - where the RAW data are stored,
   - where the data transformed in CSL format are stored,
   - where the data coregistered and RESAMPLED on a Global Primary are stored,
   - where the lists of compatible pairs are computed,
   - where the AMPLI images (coregistered on a Global Primary image, in radar geometry) are stored,
   - where the results of the MASSPROCESS are stored,
   - where the results of the MSBAS are stored.

   If more than one target are considered (e.g. several regions, satellites or tracks) and files of each steps must be stored on different hard drives, one can have another layer of these buttons (see buttons RAW_DATA_Other_Zones Figure 1)

   The second row of buttons could help locating:
   - the mains SCRIPTS,
   - the scripts used to launch CRON jobs,
   - the directory where the LaunchMTparam.txt files are stored,
   - the directory where the DEMs are stored,
   - the directory where the MASKS are stored,
   - the directory where the KML (e.g. for reading the data) are stored,
   - the directory where the EVENTS (e.g. for plotting on the time series) are stored,
   - the directory where the list of POINTS_TS are stored (for automatic plot at during MSBAS inversion).

   The third row of buttons could be the disks where processing’s are operated, that is a list of available disks or directories.

2. Below, a main window displays the content of the directory selected by a button above. One can navigate in that directory and its subdirectories (displayed in columns).
3. Further below, a line labelled “**Go To Dir Where To Run**” followed by two buttons labelled **GOTO** and **HOME**
4. Further below, a line labelled “**Select Command/Param To Run**” followed by a button labelled **ADD**

5. Finally, a window like a Terminal, with a "**>**" displayed as a prompt, and two buttons labelled **EXEC** and **CLEAR**.

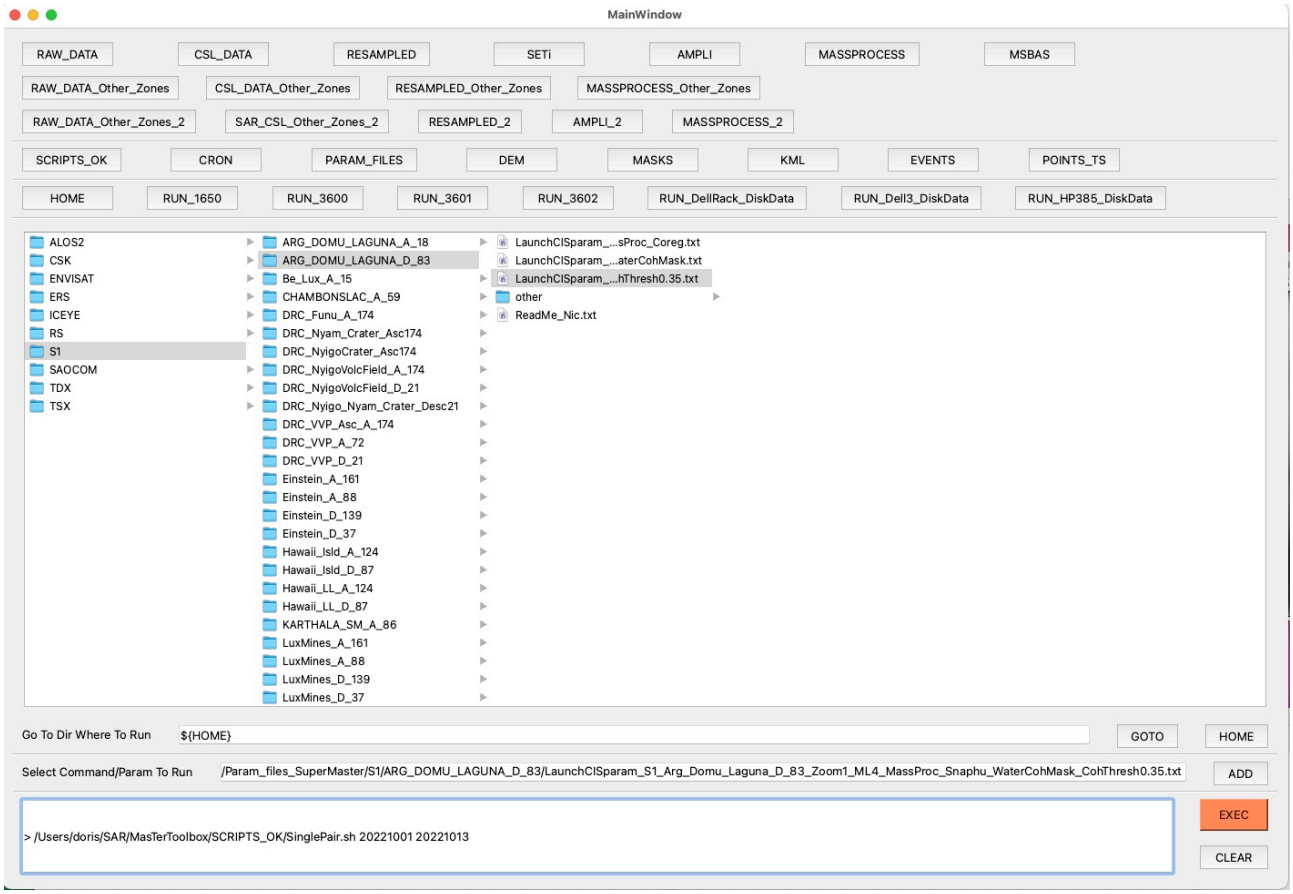


Figure 1: example of AMSTer Organizer window

The goal is to allow the user to find where the files are and to launch commands from the prompt in last section (at the bottom).

Suppose the user wants to launch the processing of a pair of Sentinel 1 data that are already read (i.e. transformed in CSL format).
To do so, the user must

- click on the button SCRIPTS_MT (first button of second layer of buttons) and locate for instance the script ***SinglePair.sh***. The path and name of that script will appear in the section 4 "**Select Command/Param To Run**". By clicking on the button ADD, that path and name will appear in the Terminal in section 5.
- The user can then type in that section 5 after the name of the script, the date of the Primary and Secondary image he/she wants to compute. The user can check the date of these images by navigating for instance in the subdirs obtained by clicking on the 2nd button of first layer of buttons (CSL_DATA)
- Then the user must provide the script with the third expected parameter, which is the file containing all the parameters for the processing. That file can be located by pressing the button PARAM_FILES (3rd button of second layer of buttons). Again, by clicking ADD, that path and name will appear in the Terminal in section 5.
- The user can add an optional remark as 4th parameter by entering a string in the Terminal window.

- If the user press EXEC, the command will be executed in a Terminal window that will pop up and from the path that is visible in the 3rd section. In the example, that is ${HOME}. If the user wants to run the command from another directory, he/she must simply navigate in the directories by pressing e.g. the buttons from the 3rd layer of buttons. When the location is found, the user must press the button GOTO. If the user press HOME, the script will be launched obviously back in $HOME.

  So far, the ***AMSTerOrganizer.sh*** does not allow to drag and drop files, delete files from the main windows, launch bash commands outside of a script etc..

  The script requires **PyQT6** and the following python scripts: *main_window.py*, *start_app.py* and *Ui_main_window_man.py* (located in …/AMSTer/SCRIPTS_MT/AMSTerOrganizer).

0.17) Contributions:

DD wrote the AMSTerEngine software.
SS wrote the MSBAS software.
NdO wrote most of the AMSTer Toolbox scripts.
DS wrote the pair selection optimization module and the recursive unwrapping module.
MJ wrote the web interface and the AMSTer Organizer module.
GC helped to debug, improve, compile several codes and hardware installation

The development of AMSTer commenced in the early 2010s and leveraged the capabilities of various pre-existing tools, including the CSL InSAR suite, which was fully remastered to be the MasTerEngine, renamed later as AMSTerEngine.

Over time, the AMSTer underwent incremental enhancements, both independently and within the context of numerous projects, notably (unsorted) RESIST, MUZUBI, GEORISCA, SMMIP, TIGRES, ECTIC, MODUS, VERSUS, advInSAR, Vi-X... These projects were primarily funded by the Belgian Scientific Policy (BelSPo) and the Luxembourgish Fond National de la Recherche (FNR).

0.18) Acknowledgements:

The authors wish to address special thanks to:

- Benoît Smets, research scientist at the Royal Museum for Central Africa, who contributed as an efficient beta tester and provided the first version of the Linux installation manual.
- Antoine Dille, PhD at the Royal Museum for Central Africa and the Vrije Univ. Brussels, who contributed as an efficient beta tester and suggested several useful features such as the Python script to extract time series from QGIS.
- Ludivine Libert, who developed the initial tools required for preparing the baseline plots during her thesis at the Centre Spatial de Liège (Belgium).

0.19) General remarks:

- We can't repeat enough the importance of directory structure (and naming).
- The use of multiple disks is recommended both for safety (e.g. keep raw and CSL data on separate disks) and for speed (e.g. split processing on several disks). See directory structure).
- Sometimes TRACK is written TRK and SATDIR is written SAT in path in the present manual for the sake of brevity.
- Classical processing(s) will involve several main steps (Figure 2):
    - Downloading the images and reading them in csl format
    - Process interferograms from single pairs of images either for testing and selecting the best parameters before a mass processing, or for producing DEM, or simply because you may need only one interferogram…
    - Selecting a Global Primary image (also named Super Master and dubbed SM in file or directory naming) and computing the list of all the pairs that would fit a spatial and temporal baseline threshold.
    - The coregistration of all the images to the Global Primary
    - The mass processing of all the interferograms
    - The production of the time series
    - The selection of the best parameters for the msbas processing
    - The production of figures and graphs either for display on a web page or for processing in GIS software.

Figure 2: Work flow and corresponding directories infrastructure.

1. Download the SAR data

AMSTer is able to process automatically data that would be made available automatically and incrementally as the time passes. This is for instance possible for Sentinel 1 data that are put online as they are recorded. But of course, you can process any type of image and archives.

If you want to set up your own automated procedure, you will need a script to get the data automatically. An example is provided for the Sentinel 1 data but you can develop your own scripts for other satellite.

If you are not interested by the incremental and continuous automatic processing, you can simply manually store your data.

In any case, if you manually deal with the archiving of satellite images in their native format or if you set up new scripts for that, it is important that the raw images are stored in directories with the same naming and/or structure. This is especially important for the S1 data that must be stored in a S1-DATA-*TARGET*-SLC.UNZIP directory where *TARGET* is the name of the region under concern, at least if you want to operate an automatic processing.

Of course, store only the same sort of data (region/footprint, track, mode, frequency, incidence…) in a given directory and name it in a smart way.

The following steps provide with some help to sort and store the data, depending on the satellites. Attention, the naming of the satellite directory is important. It is advised to keep the same naming as provided here below (e.g. use CSK instead of CosmoSkyMed).

1.1. CSK:

- download manually the images from ASI in …/SAR_DATA/CSK/Original/*info* (e.g. .../SAR_DATA/CSK/OriginalZip/Download_Jan2018).
  The parameter *info* is chosen to help identifying the data and remembering where they come from or when they were downloaded. This is helpful because ASI usually does not provide meaningful naming in the data structure.
- change dir in …/SAR_DATA/CSK/Original/*info*
- run "***ReadDateCSK.sh***" in directory without any parameters (the script is in zz_Utilities_MT/). For each image, it will read the date from the xml file located in the directory of the image and rename it by that date. The original name of the directory is kept as a file.txt within the image directory.

  (For old images, as they came zipped, use zz_Utilities_MT/***Prepa_CSK.sh*** that will first unzip the files and store them in a directory named by the date. Attention it will delete the gz files.).
  (For images obtained from SuperSite, use zz_Utilities_MT/***Prepa_CSK_SuperSite.sh***)

1.2. Sentinel 1:

Use a script (see example in zz_Download_Sentinel/) to download them e.g. in
…/SAR_DATA/S1/S1-DATA-*TARGET*-SLC
where *TARGET* depends on your area of interest.

It is recommended that your script unzip and move the images in an appropriate directory named
…/SAR_DATA/S1/S1-DATA-*TARGET*-SLC.UNZIP
(If need, see for instance the script zz_Utilities_MT_Ndo/***Unzip_S1.sh*** aiming at decompressing all the S1 images from a directory provided as parameter and store them in a directory with the same name, though terminated by .UNZIP. Note that the script tests if the files were already unzipped and if their size is as expected; if not, it unzips them again).

Note that at the step of reading the images, to avoid scanning increasing data base of images, the scripts will archive images older than 6 months in
…/SAR_DATA/S1/S1-DATA-*TARGET*-SLC.UNZIP_FORMER/*_YYYY*

Remember that these Sentinel 1 data require Restituted or Precise Orbits. Downloading these Precises (POE) and restituted (RES) orbits is now performed from ESA or Alaska SAR Facility (ASF). It requires:

- AMSTerEngine at least from 20220510 (with function ***updateS1Orbits***)
- Orbits from ESA are available here using login and password *gnssguest*/*gnssguest*:
  https://scihub.copernicus.eu/gnss/#/home
  AMSTerEngine from 20220316 will however download them for your without requiring further procedure.
- If you still want to use ASF orbits, procedure to download them is explained here:
  https://wiki.earthdata.nasa.gov/display/EL/How+To+Access+Data+With+cURL+And+Wget
  that is (see web page for details) :
  - Make an account to urs.earthdata.nasa.gov and get a *login* and *pwd*
  - While logged in,
    - click on “My Profile”, then menu “Applications > Authorized Apps”.
    - At the bottom, click on button “APPROVE MORE APPLICATIONS”
    - In the “Search” bar, type “Sentinel”
    - Click on “AUTHORIZE” button next to each application named with Sentinel 1.
  - In a terminal on your computer type :

    > cd ~
    > touch .netrc
    > echo "machine urs.earthdata.nasa.gov login *login* password *pwd* " > .netrc
    > chmod 0600 .netrc

    Then

    > cd ~
    > touch .urs_cookies
- Ensure that you have *S1_ORBITS_DIR* as state variable in *.bashrc* for path to directory where orbits are stored (./AUX_POEORB and ./AUX_RESORB).

1.3. Radarsat 2:

Download images manually and store them in
…/SAR_DATA/RADARSAT/*MODE*

1.4. TSX-TDX:

These images are acquired in several modes and footprints. Store them appropriately in directories named after their characteristics, e.g.:
…/SAR_DATA/TSX/*MODE*
or
…/SAR_DATA/TDX/*MODE*
You may want to use the script ***Prepa_TSX.sh*** to change the name of the raw image directories and check their acquisition modes and/or footprints.

1.5. ENVISAT and ERS:

Download images manually and store them in
…/SAR_DATA/ENVISAT/*MODE*

Note: data downloaded on ESA web site after termination of the ENVISAT mission might be stored in a different manner compared to when data were delivered after each acquisition during the operational phase of the mission. Each image must be stored in a subdirectory named by its orbit number. If it is not the case, run ***MoveBulkEnvisat_InSubDirs.sh*** (in zz_Utilities_MT_Ndo) prior reading the data.

1.6. KOMSAT:

Works mostly like CSK

1.7. PAZ:

Works mostly like TSX

1.8. RS1:

Radarsat-1 can be read and processed, though only in slant range. Hence, DInSAR can be performed in 3 passes mode but results can't be geocoded because external DEM can't be taken into account. This is related to the fact that the orbits of RS1 are provided in a different reference frame.

1.9. SAOCOM:

SAOCOM data made available by CONAE might mix several production dates for the same image. Each might be due to a re-focusing or a focusing along a new frame (different start/stop time). Because the file naming does not tell the user which image is what, one can download everything and the reader will later take care of this.

2. Reading all available images for each mode: *Read_All_Img.sh*

This step is aiming at reading all the SAR images available in a given directory for a given satellite in a given mode or track and store them in CSL format where they can be read by AMSTerEngine and used for further processing.

The AMSTerEngine image reader ingests unzipped images. For safety reason, it is recommended to save original (raw) data in native format and data read in *.csl* format on separate disks.

All images must be SLC (level 1). AMSTer does not work with level 0 data.

The process is done by launching the script ***Read_All_Img.sh*** with at least 3 parameters:

- the path to the folder where unzipped *raw* data are stored
- the path to the folder where *CSL* data will be stored
- the *satellite* (beware of the naming that must fit the dir naming, that is also the naming expected in the script)

More parameters might be necessary for some specific cases (there is no specific order in providing these options):

- for S1 and SAOCOM data: the path to a *kml* file delimiting the footprint of the area of interest. It will be used by the bulk reader to select all and only the S1 bursts that overlap that area of interest, or will be used by the bulk reader to crop the SAOCOM data during the reading.
- ForceAllYears: for S1 data, it will force the reader to read not only the last 6 months data that are stored in /SAR_DATA/S1-DATA_*YourPlace*-SLC.UNZIP, but also those stored in /SAR_DATA/S1-DATA_*YourPlace*-SLC.UNZIP_FORMER/*_yyyy* (where *yyyy* stands for each year of data already read and older than 6 months). Note that when ForceAllYears is used, it will also update all the S1 orbits (and not only those since the last update).
- -n: for S1 data, it will skip updating the S1 orbits (precise S1 orbits are made available about 3 weeks after the image acquisition).
- *RESAMPLED_DIR*: for S1 data, the path to the directory that contains the data already resampled (/SAR_SM/RESAMPLED/). The script will check over there which resampled data were computed with old orbits (or with FAST24 images) and move them to /SAR_SM/RESAMPLED/S1_CLN. Hence, at the next run of coregistrating the data, AMSTer will recompute the resampling with the most accurate orbits.

  Notes:

  - The path MUST end with the /RESAMPLED/ string (as expected anyway in the usual architecture)
  - If no path is provided, it will NOT update the results, obviously
  - Do not forget to flush that S1_CLN directory on a regular basis to avoid overloading your hard disks.
- *MASSPROCESS_DIR*: for S1 data, the path to the directory that contains the InSAR pairs already computed (/SAR_MASSPROCESS/). The script will check over there which pairs were computed with old orbits (or FAST24 images) and move them to /SAR_MASSPROCESS /S1_CLN. Hence, at the next run of mass processing the InSAR pairs, AMSTer will recompute them with the most accurate orbits.

  Notes:

- The path MUST end with the /SAR_MASSPROCESS/ string (as expected anyway in the usual architecture)
- If no path is provided, it will NOT update the results, obviously
- Do not forget to flush that S1_CLN directory on a regular basis to avoid overloading your hard disks.

- POL: for S1 or SAOCOM data (mandatory), the preferred polarization (*VV*, *HH*, *VH*, *VV* or *ALLPOL*) in order to read only that polarity and spare room on disk. For S1, if a VV (or HH) polarization is requested but does not exist, it will try read HH (or VV) to avoid gaps in time series. Cross polarization for interferometry (e.g. VV-HH) may however be of a lesser quality.
  To force reading all the polarizations, use the parameter ALLPOL. Reading all the polarizations is however NOT RECOMMANDED for data used for mass processing for time series as it will take a lot of time and resources!

The script makes use of ***FUNCTIONS_FOR_MT.sh, Check_All_S1_ImgReadSize.sh***., ***List_All_S1_ImgSize.sh*** and ***List_S1_Frames_Swaths_Bursts.sh***.

In order to keep track of the version of AMSTerEngine used to read the data, the script will store a small text file named *Read_w_AMSTerEngine_V.txt* in …/SAR_CSL/*SAT*/*REGION*/NoCrop/*date*.CSL/. It contains a single line indicating the date of the last version of the software present in the /$HOME/SAR/AMSTer/AMSTerEngine/_Sources_AE/Older. It is supposed to be the last one that was compiles and hence the one currently used.

NOTES:
In principle, if the script is launched several times while some images may be added in the *raw* directory, it will only process the new data and ignore the images already read. At least, this should be the case

- For the data read using the bulk reader, that is Sentinel-1, SAOCOM, ICEYE and TDX
- For all the other satellites, which are read one by one, as long as the date of the images are included as *yyyymmdd* somewhere in the name of the raw image directory (or, for ALOS2, at least as *yymmdd* in the name).
  When raw images do not contain the date in their name, (e.g. for ALOS, TSX or CSK), it is strongly advised to run a preparatory script that will rename the raw image directories by their dates (and maybe check if / sort all the images that would be acquired in different modes or on different targets). See for instance ***Prepa_CSK.sh*** or ***Prepa_TSX.sh***. Depending on your needs, you may want/need to adapt these scripts.

Note that reading with bulk reader or one by one is defined in the script and is not an option. Bulk reader has the advantage of taking care of image that would be already read etc. but has the inconvenient to stop when it encounters a bad image. Reading one by one requires to check the data that would be already read. If it is not performed correctly, it will ask the user if one wants to overwrite the existing data each time the scripts encounter an image already processed with the same acquisition date.

Note: you can always check the characteristics of all the images in your directory …/SAR_CSL/*SAT*/*REGION* by running the script ***_Check_CSL_Corners_Trk_etc.sh***, which will log the main characteristics of all the images in a file named List_Img_Characteristics.txt. That file logs the following characteristics read in the /Info/SLCImageInfo.txt of each image:

Date:	The acquisition date of the image
Time:	The acquisition time
Mode:	The Orbit geometry (Asc or Desc)
LookDir:	Right or Left
LookAngl[deg]:	The incidence angle
Xsize[pix]:	The size of image in range direction (in pixels)
Ysize[pix]:	The size of image in azimuth direction (in pixels)
Xsampl[m]:	The pixel size in range direction (in m)
Ysampl[m]:	The pixel size in azimuth direction (in m)
Center/Ext-Lon:	The longitude of the center (or the min-max UTM Y value) of the scene
Center/Ext-Lat :	The latitude of the center (or the min-max UTM X value) of the scene

2.1. Sentinel 1 (Wide Swath or Strip Map):

Sentinel-1 Strip Map (SM) or Wide Swath (IW) data are usually downloaded as .zip files. After unzip, each image is stored in a directory named e.g.
S1A_IW_SLC__1SDV_20220801T232721_20220801T232748_044365_054B6D_01E8.SAFE
AMSTer will read Sentinel-1 images from that expected format.

For Sentinel-1 data, the script ***Read_All_Img.sh*** will select all and only the swath(s) and burst(s) of interest to cover the desired footprint (these can be from separate images if the area of interest is larger than a regular S1 frame or if the area of interest is located across two S1 frames). The footprint is defined by providing a *KML* file as the 4th parameter (e.g. produced in Google Earth). The script will also sort the S1 images (Ascending or Descending).

The optional 5th parameter is a text string. If that string is *ForceAllYears*, the script will force the reading of all the images available. If that parameter is left empty (or contains another string than *ForceAllYears*, the script will read only the images that are less than 6 months old (or that were not read yet). See below for explanations about the archiving of raw images after reading.

Note that using the parameter *ForceAllYears* will also force the script to update the whole orbit database instead of updating it only since last available precise orbit date.

The optional 6th parameter is a text string "-n" (without quotes). This option allows skipping the orbits update process.

After having selected the required swaths and bursts and sorted the images, it will assemble the bursts into a single image in each …/SAR_CSL/S1/*REGION*/NoCrop/*IMAGE*.slc directory (where *IMAGE* is the date of the image), then check that the size in bytes of that file is consistent with the expected number of lines and columns. If not, it will re-read (re-stitch actually) the image. This is sometimes needed for unknown reason. Do not remove the separated bursts after stitching as it will need them for coregistration when the image will be used as Secondary image.

Because all Ascending and Descending raw Sentinel-1 images that are stored in …/SAR_DATA/S1/S1_*info*_UNZIP will be read at the same time by the AMSTerEngine reader, the script will sort and store the CSL images in directories named after their geometry (ascending or descending) and their orbit number. However, because the mass reading compares the folder with raw data (…/SAR_DATA/S1/S1_*info*_UNZIP) with the target folder where the CSL data are (…/SAR_CSL/S1/*REGION*/NoCrop), one cannot move the CSL data in sub-directories (these images would be read again at the next execution of the reader). For this reason, the script keeps all the images (whatever their geometry or orbit) as links in the target directory (…/SAR_CSL/S1/*REGION*/NoCrop) while CSL images (to which links point at) are stored in other directories (Figure 3) such as
…/SAR_CSL/S1/*REGION_GEOM_ORB*/NoCrop
where
- *GEOM* is *A* or *D* (for ascending or descending orbit)
- *ORB* is the orbit number

Images in CSL format are named as:
S1*A*(or*B*)_*ORB_DATE_A*(or*D*).csl where
- *A* or *B* depends on the Sentinel 1A or 1B
- *A* or *D* depends on ascending or descending orbit
- *ORB* is the orbit number

Note that the stitched image file
…/SAR_CSL/S1/*REGION_GEOM_ORB*/NoCrop/*IMAGE*.slc/Data/SLCData.*POL* (where *POL* is the polarization and which is usually *VV*; see the preferred polarization chosen in

Read_All_Img.sh) is used during the (mass)processing at coregistration step. If it is missing, it will be recomputed at ***S1Coregistration***, but it will not be overwritten if present and of expected size.

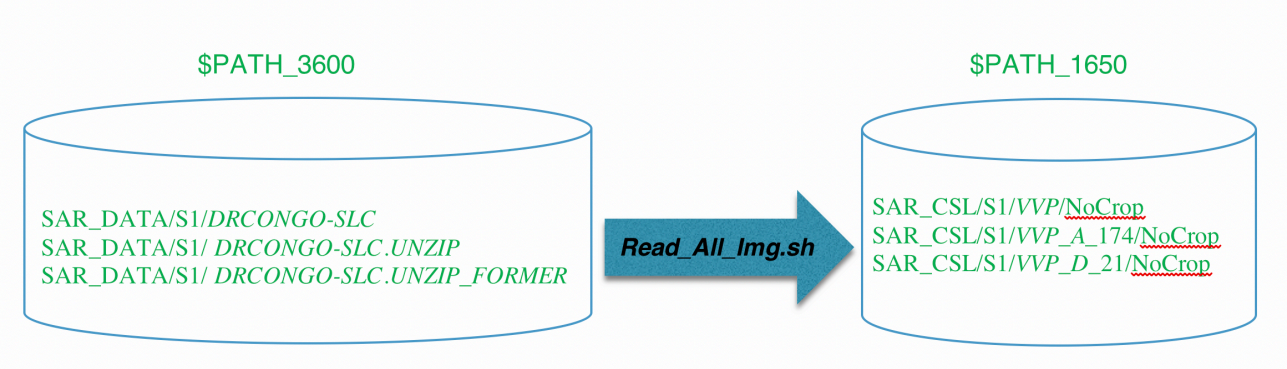


Figure 3: Example of input and output directories associated with the usage of Read_All_Img.sh in the case of S1 data acquired for DR Congo and reading for the Virunga Volcanic Province (VVP)

Attention, if a *kml* is provided at mass processing for cropping, and if that *kml* does not overlaps the same bursts as those assembled at reading (Figure 4), it will stitch again the bursts in the i12 directory during the InSAR processing. This would take time and disk space! To spare time and space, better adjust your *kml*'s properly from beginning, i.e. at reading step.

This was planned for mass processing using S1. The reason is a matter of disk space and processing time: if the *kml* used to crop the interferogram has a footprint that cover less bursts than what has been read (and hence assembled as …/SAR_CSL/*SAT/TRK*/NoCrop/*IMAGE*.csl/Data/SLCData.*pol*), AMSTerEngine will assemble again the required bursts in a new SLCData.*pol* but saved this time in your *PRM_SCD*/i12 directory. Hence it will make a new copy of the primary image in **EACH** processing pair, which would be a huge waste. There is of course no problem if you set no *kml* as a crop region for interferometric computation.

Example:

Black: S1 bursts footprints as selected with KML used for reading (blue)
Blue: KML used for reading.
Green: Example of good KML's used for cropping interferograms.
Red: Examples of bad KML's used for cropping interferograms.

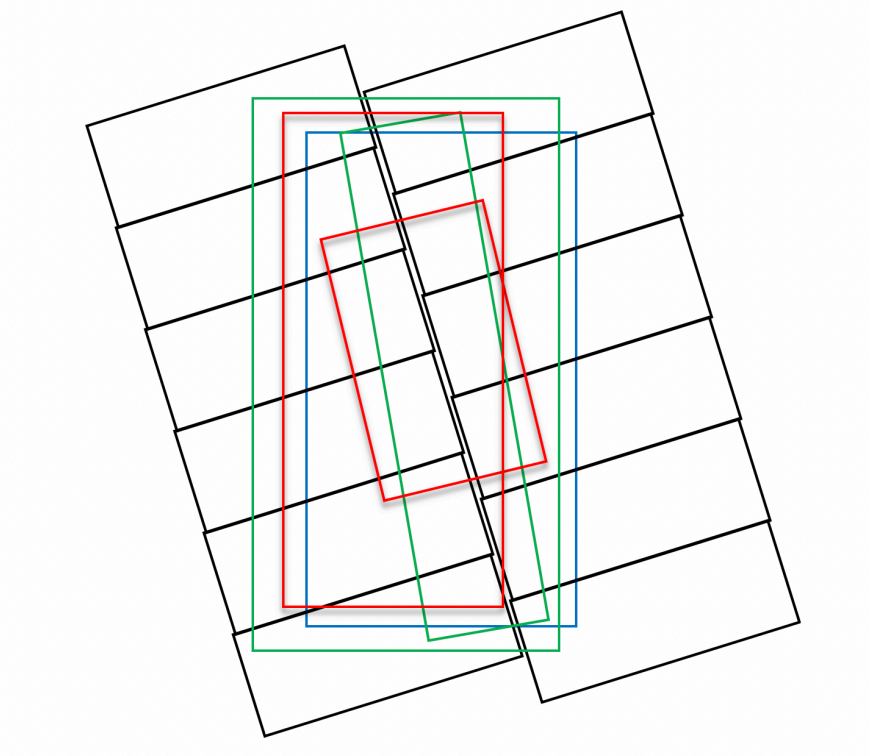


Figure 4: Example of S1 bursts footpint (black), kml footprint used for reading (blue) and for cropping at processing (Greens are good, Reds would require wasted time and disk space).

Notes:

- Remember that the script ***Read_All_Img.sh*** will also restrict the reading to a chosen polarization. If you want all the polarizations, use the option ALLPOL in ***Read_All_Img.sh*** at each use of ***S1DataReader.*** Same applies for SAOCOM.
- If precise orbits are not present in the S1_ORBITS_DIR folder yet, the script will attempt to download them if available (i.e. for image at least ~31 days old). If no precise orbits are available, restitute orbits will be used.
- To gain time, the script attempts to update the S1 orbits only since last available precise orbit date (i.e. it supposes that all the former orbits were already correctly updated). If for any reason you need to update older orbits, you can either change the script to manually hard code the date from when to begin the update (***Read_All_Img.sh*** where it calls the function ***updateS1Orbits***), or launch manually the function ***updateS1Orbits*** with appropriate *from=YYYYMMDD* parameter, (see ***updateS1Orbits*** *-help*) before launching ***Read_All_Img.sh***, or more simply, you can launch ***Read_All_Img.sh*** with the *ForceAllYears* parameter. In that case, it will indeed update the whole S1 orbit database since the beginning of S1.
- Note that the path to the folder with the sentinel orbits must be defined in the */$HOME/.bashrc* as a state variable S1_ORBITS_DIR.
- Note: Do **not** name two S1 REGION with the same beginning of name followed by an underscore (e.g. do NOT use *LUX* and *LUX_TEST*; you can, however, use for instance *LUX* and *LUXTEST*) because it would result in some problems at sorting the orbit modes after reading *LUX* images.

Attention: when an updated precise orbit is found (or when a FAST24 image is updated with a normal one), the script will update the *SLCImageInfo.txt* with the new orbit in
…/SAR_CSL/S1/REGION/NoCrop/S1*AorB*_track_date_*AorD*.csl/Data/Frame*i*.csl/Info
where *i* of *Frame* stands for each of the frame used in your footprint. The former *SLCImageInfo.txt* will be renamed *SLCImageInfo.txt.old*

Because this update impacts the results, the script will search for all the processes already performed with the outdated orbits and move them to specific directories. In that case, the products impacted by that updated will be automatically re-processed at the next mass processing runs. Specifically, the script will move the RESAMPLED images and the MASS_PROCESSED pairs involving the updated images resp. in:

- …/SAR_SM/RESAMPLED/S1_CLN/CLEANED_FAST24 (both Asc and Desc in same dir) or
  …/SAR_SM/RESAMPLED/S1_CLN/CLEANED_ORB (both Asc and Desc in same dir)
- …/SAR_MASSPROCESS/S1_CLN/CLEANED_FAST24 (both Asc and Desc in same dir) or
  …/SAR_MASSPROCESS/S1_CLN/CLEANED_ORB (both Asc and Desc in same dir)

Of course, one must run again the scripts ***SuperMasterCoreg.sh*** and ***SuperMaster_MassProc.sh*** to get the products involving the updated images back into the database. Note that updating the orbits does not requires to recompute the slant Range DEM nor the Slant Range Mask (unless your DEM is precise at few cm.. which is unlikely).

The amplitude products computed with ***ALL2GIF.sh*** (see below) are not updated! This however shouldn't be a problem.

If you do not want to have the images removed from your RESAMPLED and/or MASSPROCESS databases, simply do not provide corresponding paths during ***Read_All_Img.sh*** execution. If you want to remove instead of move these files, or if you want to update also the amplitude products computed with ***ALL2GIF.sh*** (I don't see why you would like that form amplitudes though), see and change script ***Read_All_Img.sh***.

EXECUTION:

For a Sentinel 1 processing, run "***Read_All_Img.sh***" with the following 4 or more parameters:

…/SAR_DATA/S1/S1_*info*_UNZIP	(that is where unzipped raw data are)
…/SAR_CSL/S1/*REGION*/NoCrop	(that is the target folder where CSL images must be stored)
S1	(that is the satellite name; must be consistent with dir naming)
…/SAR_CSL/S1/*REGION*/*region*.kml	(a *kml* file spanning the area of interest)
ForceAllYears	(an option to force reading of images older than roughly 6 months)
-n	(an option to skip updating the orbits)
…/SAR_SM/RESAMPLED/	(a path to the resampled data to recompute them later with updated orbits)
…/SAR_MASSPROCESS/	(a path to the InSAR pairs to recompute them later with updated orbits)
POL	(preferred polarization, i.e. *VV*, *HH*, *VH*, *VV* or *ALLPOL*)

The name *info* is chosen to help identifying the data (e.g. name of a country).
The name *REGION* describes the area of interest; chose it carefully as it will also be part of the TRK parameter. It differs from the *info* because from the same set of S1 image, you may want to process InSAR over different regions (e.g. name of a volcano).
The *region*.kml is a polygon e.g. created with Google Earth contouring the footprint of the area of interest.
Do not forget /NoCrop at the end of dir path where data in csl format must be stored.

To avoid time consuming re-check of former S1 images during the next execution of ***Read_All_Img.sh***, the script will move images older than 6 months in *YYYY* directories in …/SAR_DATA/S1/S1_*info*_UNZIP_FORMER/_*YYYY*.

Next time ***Read_All_Img.sh*** is executed, it will hence only check and read the last 6 months of data. If you want to check and read (again) images older than 6 months, run the script with the 5th parameter *ForceAllYears*. If you want to re-read only some of the FORMER images, you can put them back in …/SAR_DATA/S1/S1_*info*_UNZIP.

When run again, the script will ensure that every link in …/SAR_CSL/S1/*REGION*/NoCrop points toward an image in the corresponding directory and that every image in the …/SAR_CSL/S1/*REGION_GEOM_ORB*/NoCrop directories have a link in …/SAR_CSL/S1/*REGION*/NoCrop. If not, it will read again what is missing and/or clean up the directories.

Note: to read only S1 images from ONLY a given year, you can try to launch ***Read_All_Img.sh*** with the first parameter as …/SAR_DATA/S1/S1_*info*_UNZIP_FORMER/_*YYYY* without *ForceAllYears*.

Note: after reading S1 images, it will make a file listing the size of all the images, saved in …/SAR_CSL/S1/*REGION*/NoCrop/List_Master_Sizes.txt. It might be a good idea to check it and check that all of your images have the same kind of size. If one image is too small, it will most likely be due to missing bursts. That area, if missing in one pair used in your mass processing for msbas, will be lost for the whole msbas series! Reminder: a S1 burst is about 1.200-1.500 lines by 22.000-25.000 columns.

It will also list the bursts in …/SAR_CSL/S1/*REGION*/NoCrop/FramesSwathBurst_*DATE_xx*_bursts.txt for each primary image (where *xx* is the number of bursts). It allows a quick look to identify wrong images.

If some of your images are incomplete (i.e. if not all the bursts are available for some images) or corrupted for any reason, you can move your images from …/SAR_CSL/S1/*REGION*/NoCrop into …/SAR_CSL/S1/*REGION*/Quarantained
Images set in Quarantained will be ignored when re-running the ***Read_All_Img.sh***. Deleting rather than moving these images in Quarantained would result in re-reading them every time you run a ***Read_All_Img.sh,*** which is not what you want unless you have corrected the problem leading to a corrupted image or incomplete image.

Scripts ***_Check_S1_SizeAndCoord.sh*** and ***_Check_ALL_S1_SizeAndCoord_InDir.sh*** can also be used to check if image(s) are as expected, that is 1) the image has the expected number of bursts and 2) corner's coordinates are within a range of 0.05 degree (hard coded tolerance) of expected coordinates from kml corners, or are within the provided min/max coordinates of an Area of interest. (For more info see chapter 9.1).

Example:

Read_All_Img.sh …/SAR_DATA/S1/S1-DATA-DRCONGO-SLC.UNZIP …/SAR_CSL/S1/DRC_Funu/NoCrop S1 …/Share1/SAR_CSL/S1/DRC_Funu/Funu.kml ForceAllYears …/SAR_SM/RESAMPLED/ …/SAR_MASSPROCESS/ VV

2.2. CSK:

CSK data are usually provided in directories named with codes that are not explanatory such as *525071-400979* or *EL20161213_1412143_4408467.6.2* for instance. ***Read_All_Img.sh*** will however expect these CSK images to be renamed by the date to avoid possible confusion. CSK images must hence be prepared with preliminary processing (***ReadDateCSK.sh*** and ***PrepaCSK.sh***; see § 1.).

Then, because ASI's CSK data naming doesn't allow to assess which image is in Asc and which is Desc orbit, one must proceed with a two steps procedure:

- read all the images in a temporary directory named MassRead with ***Read_All_Img.sh***
- then move the corresponding Asc and Desc images toward appropriate respective directories with an additional script ***ReadModeCSK.sh***, while keeping links in the original directory. This is mandatory to allow incremental reading without having to read again images.

Note: if links from …/SAR_CSL/CSK/*REGION_Mode*/NoCrop to …/SAR_CSL/CSK/*REGION* need to be rebuilt, you can use the script ***Rebuild_Ins_CSK.sh***.

EXECUTION:

a) run "***Read_All_Img.sh***" with the following 3 parameters:

… /SAR_DATA/CSK/OriginalZip/*info*	(that is where **zipped** data are stored)
… /SAR_CSL/CSK/MassRead	(that is a temporary folder where Asc and Desc CSK images in csl format will be stored before sorting)
CSK	(satellite name; must be consistent with dir naming)

Example:
Read_All_Img.sh …/SAR_DATA/CSK/OriginalZip/Dec2017 /SAR_CSL/CSK/UnZipNoRename_AndMassRead CSK

b) go to the temporary directory …/SAR_CSL/CSK/MassRead

c) run "***ReadModeCSK.sh***" with the following parameter:

REGION	(that is the region name used to identify the data)

Attention, …/SAR_CSL/CSK/*REGION*_Asc/NoCrop and
…/SAR_CSL/CSK/*REGION*_Desc/NoCrop must exist.

Example:
ReadModeCSK.sh *Bukavu*

2.3. TDX (Bistatic or Pursuit mode):

Usually, TDX data are provided as .zip files. After unzip, each image is stored in a directory named e.g. *dims_op_oc_dfd2_369957642_11*, which is fine like that for the script ***Read_All_Img.sh*** if you are sure to have only one mode and footprint.

However, that naming will be of little help if you have several modes or footprints (hence incompatible for InSAR) and maybe several images acquired on the same days.
In that case, the script ***Read_All_Img.sh*** can't unfortunately sort automatically the images by footprint because coordinates vary. Hence the user must sort them manually or ensure that all the images in …/SAR_DATA/TDX/*REGION* have the same footprint.

If your data contains several footprints, it is advised to sort the raw data prior to read the data. This can be eased by using the script ***SORT_TDX.sh***. That script will:

- copy in specific directories created on purpose (_Check_common_images and _Check_single_images), a quick look of all the images that are stored in all the subdirectories from the current directory (i.e. TSX or TDX amplitude images)
- list the path and date of each image in each of these directories in a file named *List_tif.txt*.

See script for more information about usage.
By looking at these quick looks and files list, you can move the images in appropriate directories such as …/SAR_DATA/TDX/*REGION*.

To further assist the user for sorting the footprints, the script ***Read_All_Img.sh*** can also create a text file …/SAR_CSL/ TDX/*REGION*/Center_Of_Scene.txt with the list of the Latitude and Longitude of each scene coordinates. This allows to quickly check if images have similar footprint.

Note that if the images are incorrectly sorted by footprint or mode, there are some chances that two images would be acquired at the same date. Nevertheless, the script ***Read_All_Img.sh*** can cope with that and avoid missing or overwriting images. Images read will be stored in directories …/SAR_CSL/ TDX/*REGION_BWZ_BIS_MODE_ORB*_TX(orRX)_*ii*/NoCrop, where *ii* is an integer index.
In any case, each of the image directory must contain somewhere two subdirectories which names contain the date coded as *yyyymmdd*T*hhmmss,* eg.:
TDX1_SAR__SSC_BRX2_SM_S_SRA_20120905T035616_20120905T035621
TSX1_SAR__SSC_BTX1_SM_S_SRA_20120905T035616_20120905T035621

"***Read_All_Img.sh***" works with the same 3 parameters, the third being TDX:

…/SAR_DATA/TDX/*TRK_MODE*	(that is where unzipped raw data are)
…/SAR_CSL/TDX/*REGION*/NoCrop	(that is the target folder where CSL images must be stored)
TDX	(that is the satellite name; must be consistent with dir naming)

Remember that TSX and TDX are two satellites operating in tandem and TDX can acquire images in two modes: as emitting (TX) or reception (RX). If you need the tandem specificity (i.e. 2 images at the nearly same time) to perform DEM computation for instance, then you need to read both type of image which are provided in the same directory. Note however that these two images can be labelled TDX and/or TSX. No matter the label, it is indeed the TDX option you should use here as third parameter while using the script ***Read_All_Img.sh***.

Read_All_Img.sh will read all the images that are stored in the sub directories of the …/SAR_DATA/TDX/*REGION* and read them and store then in CSL format in the …/SAR_CSL/ TDX/*REGION*/NoCrop. Because TDX tandem data contains TX (transmit) and RX (receive) images,

the script will then move these two data in directories named
…/SAR_CSL/ TDX/*REGION_BWZ_BIS_MODE_ORB*_TX(orRX)/NoCrop where

BWZ	is the bandwidth [in Mhz]
BIS	is the type of TDX acquisition: BS for Bistatic, PM for Pursuit Mode
MODE	is A or D for Ascending or Descending
ORB	is the orbit number
TX or RX	stands for the Transmit or Receive image,

and keep links to these data in …/SAR_CSL/ TDX/*REGION*. The directory is named with useful information related to the images acquisition mode.

Notes:

- When using TDX for InSAR, the primary image must always be TX. If it needs a Secondary image of the same date (eg for making a DEM from bistatic data), the Secondary will be obviously RX, that is it will be actually stored in another directory. However, the processing scripts always expect the primary and secondary images to be stored in the same …/SAR_CSL directory. To allow the processing, the InSAR scripts will hence link the data from RX dir to TX dir. But to avoid clash in naming with the same date, the first digit of the secondary image's date (i.e. the millennia) will be replaced by 9 (e.g. 9019 instead of 2019). The script ***Read_All_Img.sh*** is able to cope with data too.

2.4. TSX:

Usually, TSX data are provided as .zip files. After unzip, each image is stored in a directory named e.g. *dims_op_oc_dfd2_371205443_1*, which is fine like that for the script ***Read_All_Img.sh*** if you are sure to have only one mode and footprint.

However, that naming will be of little help if you have several modes or footprints (hence incompatible for InSAR) and maybe several images acquired on the same days.
In that case, the script ***Read_All_Img.sh*** can't unfortunately sort automatically the images by footprint because coordinates vary. Hence the user must sort them manually or ensure that all the images in …/SAR_DATA/TSX/*REGION* have the same footprint.

If your data contains several footprints, it is advised to sort the raw data prior to read the data. This can be eased by using the script ***SORT_TDX.sh***. That script will:

- copy in specific directories created on purpose (_Check_common_images and _Check_single_images), a quick look of all the images that are stored in all the subdirectories from the current directory (i.e. TSX or TDX amplitude images)
- list the path and date of each image in each of these directories in a file named *List_tif.txt*.

See script for more information about usage.
By looking at these quick looks and files list, you can move the images in appropriate directories such as …/SAR_DATA/TSX/*REGION*.

To further assist you in sorting out your images if you have images acquired by several modes ad/or footprints, and possibly facing the risk of having two images acquired on the same date, you can run the script ***Prepa_TSX.sh***, which will rename all your images directories by their acquisition dates. If two or more images are acquired on the same date, the script will rename the data to read by their date and add an index in case of redundancy. The script ***Read_All_Img.sh*** will cope with that index and you will have the possibility to further sort your images.

In any case, each of the image directory must contain somewhere a subdirectory which name contains the date coded as *yyyymmdd*T*hhmmss,* eg.:
TSX1_SAR__EEC_SE___SL_D_SRA_20080503T162043_20080503T162044

"***Read_All_Img.sh***" works with the same 3 parameters, the third being TSX:

…/SAR_DATA/TSX/*TRK_MODE*	(that is where unzipped raw data are)
…/SAR_CSL/TSX/*REGION*/NoCrop	(that is the target folder where CSL images must be stored)
TSX	(that is the satellite name; must be consistent with dir naming)

Remember that TSX and TDX are two satellites operating in tandem. TDX can however acquire simple images as TSX. That is why DLR may provide TSX images labelled TDX, but in any case, images can be read the same way using ***Read_All_Img.sh*** using the third parameter as TSX.

Note: if you have TDX data in tandem mode (i.e. with a pair of TX and RX images) and want to use only the TX data to perform a differential interferogram between two dates, just read the TDX data in TDX mode with ***Read_All_Img.sh*** (see 2.3), and then perform your interferograms using the _TX data as input.

2.5. ENVISAT:

Read_All_Img.sh expects ENVISAT data to be provided in directories named by their orbit number (e.g. *33069*). That directory must contain a file with N1 extension, e.g.
ASA_IMS_1PNIPA20120227_075954_000000163112_00107_52276_0144.N1

However, data downloaded on ESA web site after termination of ENVISAT mission might be stored in a different manner compared to when data were delivered after each acquisition. If each image is not stored in a directory named by its orbit number, run ***MoveBulkEnvisat_InSubDirs.sh*** (in zz_Utilities_MT_Ndo) prior reading the data.
Note that ***Read_All_Img.sh*** expects the acquisition date to be written in the third position of the .N1 file name (underscore-separated string).

The path to the folder with the Envisat DORIS precise orbits must be present in the /$HOME/*.bashrc* as a state variable ENVISAT_PRECISES_ORBITS_DIR.

EXECUTION:
Run "***Read_All_Img.sh***" with the following 3 parameters:

…/SAR_DATA/ENVISAT/*TRK_MODE*	(that is where unzipped raw data are)
…/SAR_CSL/ENVISAT/*REGION*/NoCrop	(that is the target folder where CSL images must be stored)
ENVISAT	(that is the satellite name; must be consistent with dir naming)

2.6. ERS:

Read_All_Img.sh expects ERS data in CEOS format to be provided in directories named by their orbit number (e.g. *22712*). That directory must contain a file which name start with the string LEA_ and where ***Read_All_Img.sh*** will be able to dig out the acquisition date.

EXECUTION:
Run "***Read_All_Img.sh***" with the following 3 parameters:

…/SAR_DATA/ERS/*TRK_MODE*	(that is where unzipped raw data are)
…/SAR_CSL/ERS/*REGION*/NoCrop	(that is the target folder where CSL images must be stored)
ERS	(that is the satellite name; must be consistent with dir naming)

2.7. RADARSAT-2 (Fine, Ultra Fine,…) :

Read_All_Img.sh expects RADARSAT-2 data to be provided in directories where the acquisition date is provided in 6th position (from underscore-separated string) as *yyyymmdd*, whatever the mode of acquisition, e.g. for Fine or Ultra Fine acquisitions respectively:
RS2_OK73287_PK656053_DK586304_F2F_**20101217**_034132_HH_HV_SLC
RS2_OK58499_PK545801_DK482829_U9W2_**20140925**_162413_HH_SLC.

EXECUTION:
Run "***Read_All_Img.sh***" with the following 3 parameters:

…/SAR_DATA/RADARSAT/*TRK_MODE*	(that is where unzipped raw data are)
…/SAR_CSL/RADARSAT/*REGION*/NoCrop	(that is the target folder where CSL images must be stored)
RADARSAT	(that is the satellite name; must be consistent with dir naming)

Note: although it is RADARSAT-2, the third parameter is simply RADARSAT.

2.8. RADARSAT-1:

Read_All_Img.sh expects RADARSAT-1 data in CEOS format to be provided in directories named by a code (e.g. M0272711).

EXECUTION:
Run "***Read_All_Img.sh***" with the following 3 parameters:

…/SAR_DATA/RS1/*TRK_MODE*	(that is where unzipped raw data are)
…/SAR_CSL/RS1/*REGION*/NoCrop	(that is the target folder where CSL images must be stored)
RS1	(that is the satellite name; must be consistent with dir naming)

Notes:

- The third parameter can be RADARSAT1 or more simply RS1.
- Because of Radarsat-1 orbital information format incompatible (yet) with AMSTer Toolbox, it is not possible to take into account an external DEM. As a consequence, DInSAR can be performed in 3 passes mode but results can't be geocoded.
- For the same reason, RS1 images can be read and amplitude images can be computed (e.g. with ***MakeAmpliPlotSingleImg.sh***) though it can't be cropped based on a kml file.

2.9. PAZ:

PAZ data are pretty much like TSX data. They are provided as .zip files. After unzip, each image is stored in a directory with a name that contains the date coded as *yyyymmdd*T*hhmmss*, e.g. *PAZ1_SAR__SSC_____SL_S_SRA_20220614T002135_20220614T002137*, which is fine like that for the script ***Read_All_Img.sh***.

"***Read_All_Img.sh***" works with the same 3 parameters, the third being PAZ:

…/SAR_DATA/PAZ/*TRK_MODE*	(that is where unzipped raw data are)
…/SAR_CSL/PAZ/*REGION*/NoCrop	(that is the target folder where CSL images must be stored)
PAZ	(that is the satellite name; must be consistent with dir naming)

2.10. KOMPSAT:

KOPMPSAT data are pretty much like CSK data. They are provided as .zip files. After unzip, each image is stored in a directory with a name that contains the date coded as *yyyymmddhhmmss*, e.g. *K5_20170717094902_004042_21423_D_ES06_HH_SCS_A_L1A*, which is fine like that for the script ***Read_All_Img.sh***.

"***Read_All_Img.sh***" works with the same 3 parameters, the third being PAZ:

…/SAR_DATA/PAZ/*TRK_MODE*	(that is where unzipped raw data are)
…/SAR_CSL/PAZ/*REGION*/NoCrop	(that is the target folder where CSL images must be stored)
PAZ	(that is the satellite name; must be consistent with dir naming)

2.11. ALOS:

ALOS (i.e. ALOS-1) data are provided as .zip files. After unzip, each image is stored in a directory that contains a file named workreport where ***Read_All_Img.sh*** will be able to dig out the acquisition date.

"***Read_All_Img.sh***" works with the same 3 parameters, the third being ALOS:

…/SAR_DATA/ALOS/*TRK_MODE* (that is where unzipped raw data are)
…/SAR_CSL/ALOS/*REGION*/NoCrop (that is the target folder where CSL images must be stored)
ALOS (that is the satellite name; must be consistent with dir naming)

2.12. ALOS-2 (Wide Swath or Strip Map):

ALOS-2 Strip Map (SM) or Wide Swath (WS) data are provided as .zip files. After unzip, each image is stored in a directory named by a string ending with the date coded as yymmdd which will be interpreted by ***Read_All_Img.sh***.

"***Read_All_Img.sh***" works with the same 3 parameters, the third being ALOS2:

…/SAR_DATA/ALOS2/*TRK_MODE* (that is where unzipped raw data are)
…/SAR_CSL/ALOS2/*REGION*/NoCrop (that is the target folder where CSL images must be stored)
ALOS2 (that is the satellite name; must be consistent with dir naming)

Notes:

- Note the difference in third parameter: do not mix ALOS2 with ALOS used for reading ALOS1 data.
- Beware, ALOS2 images may be affected by strong long wavelength residuals. You may need some tricks such as *Deramp_Spotlight.py* to attempt removing long wavelength artefacts (or attempt some cleaning with Split Band interferometry).

2.13. SAOCOM:

SAOCOM data are provided in directories named e.g. *112878-EOL1ASARSAO1A695905* which contain a zip file and a .xemt file. All the data are read using a bulk reader (transparent to the user) which will sort out which data are already read and read only the new ones. Note however that it will stop reading all the images in the input directory if it encounters a bad image.

Because Ascending and Descending raw images can be stored in …/SAR_DATA/SAOCOM/*REGION*, the script will read them all, then sort them and store the CSL images in directories named after their orbit number and geometry (ascending or descending). However, because the bulk reader compares the folder with raw data (…/SAR_DATA/AOCOM/*REGION*) with the target folder where the CSL data are (…/SAR_CSL/SAOCOM/*REGION*/NoCrop), one cannot move the CSL data in sub-directories (these images would be read again at the next execution of the reader). For this reason, the script keeps all the images (whatever their geometry) as links in the target directory (…/SAR_CSL/SAOCOM/*REGION*/NoCrop), while CSL images (to which links point at) are stored in other directories such as
…/SAR_CSL/SAOCOM/*REGION_ORB_GEOM*/NoCrop
where *ORB* is the orbit number and *GEOM* is A or D (for ascending or descending orbit).

Moreover, because the SAOCOM data made available by CONAE might mix several production dates for the same image. Each might be due to a re-focusing or a focusing along a new frame (different start/stop time). To cope with that, the reader will check if the data contains the area of interest (provided as a mandatory kml file). If it does not, the image is skipped. If several images include the kml, the reader will select the one which has its centre closer to the centre of the kml.

If an image is re-focused and updated, at the next reading, the reader will compare. If the new image is more favourable (better overlap with the kml), it will take it instead of the existing one. If they are similar, it will take the most recent one.

Note that if a path to the Coregistered and/or Mass Processed products is provided to the script, it will move all the products computed with the former image in /SAR_SM/RESAMPLED/SAOCOM_CLN and these will be recomputed with the updated image at the next run of the scripts dedicated to the coregistration on a Global Primary and/or Mass Processing.

Finally, to spare time at reading, the images of more than 6 months are moved in SAR_DATA/SAOCOM-DATA_*YourPlace*-SLC.UNZIP_FORMER/*_yyyy* (where *yyyy* stands for each year of data already read and older than 6 months) after being read.

To force the re-reading of the images of more than 6 months, either add the parameter ForceAllYears or move the images to re-read from /SAR_DATA/SAOCOM-DATA_*YourPlace*-SLC.UNZIP_FORMER/*_yyyy* back in /SAR_DATA/SAOCOM-DATA_*YourPlace*-SLC.UNZIP.

EXECUTION:

For a SAOCOM processing, run "***Read_All_Img.sh***" with the following 4 or more parameters:

…/SAR_DATA/SAOCOM/*REGION*	(that is where unzipped raw data are)
…/SAR_CSL/SAOCOM/*REGION*/NoCrop	(that is the target folder where CSL images must be stored)
SAOCOM	(that is the sat. name; must be consistent with dir naming)
…/SAR_CSL/SAOCOM/REGION/*region*.kml	(a *kml* file spanning the area of interest for optional cropping)
ForceAllYears	(an option to force reading of images older than > 6 months)
…/SAR_SM/RESAMPLED/	(a path to the resampled data to recompute them later with updated orbits)
…/SAR_MASSPROCESS/	(a path to the InSAR pairs to recompute them later with updated orbits)
POL	(preferred polarization, i.e. *VV*, *HH*, *VH*, *VV* or *ALLPOL*)

Note: the kml file is mandatory.

2.14. ICEYE:

ICEYE data are provided as files named with a .h5 extension, e.g. *ICEYE_SLC_SM_5415_20190518T211319.h5*.

All the data are read using a bulk reader (transparent to the user) which will sort out which data are already read and read only the new ones. Note however that it will stop reading all the images in the input directory if it encounters a bad image.

Because ICEYE is a constellation of agile satellites, able to acquire data on several geometries (Asc or Desc, Right or Left looking, with several look angles), the images stored in the raw images directory (i.e …/SAR_DATA/ICEYE/*REGION*) may be of a high variety.

The script will read them all, then sort them and store the CSL images in directories named after their geometry: *REGION_GEOM_MODE_RL_INCID*deg, where *GEOM, MODE, RL* and *INCID* are respectively the orbit geometry (A for Ascending, or D for Descending), the acquisition mode (e.g .Spotlight or Stripmap), the looking direction (RL for Right Looking or LL for Left Looking) and the incidence angle.

However, because the bulk reader compares the folder with raw data (…/SAR_DATA/ICEYE/*REGION*) with the target folder where the CSL data are (…/SAR_CSL/SAOCOM/*ICEYE*/NoCrop), one cannot move the CSL data in sub-directories (these images would be read again at the next execution of the reader). For this reason, the script keeps all the images (whatever their geometry) as links in the target directory (…/SAR_CSL/ICEYE/*REGION*/NoCrop), while CSL images (to which links point at) are stored in other directories such as …/SAR_CSL/ICEYE/*REGION_GEOM_MODE_RL_INCID*deg /NoCrop.

EXECUTION:

For a ICEYE processing, run "***Read_All_Img.sh***" with the following 4 or more parameters:

…/SAR_DATA/ICEYE/*REGION*	(that is where unzipped raw data are)
…/SAR_CSL/ICEYE/*REGION*/NoCrop	(that is the target folder where CSL images must be stored)
ICEYE	(that is the satellite name; must be consistent with dir naming)
ForceAllYears	(an option to force reading of images all the images in …/SAR_DATA/ICEYE/*REGION* instead of only those that are not yet in …/SAR_CSL/ICEYE/*REGION*/NoCrop)

Note: ICEYE has a very large orbital tube, which provides most of the time perpendicular baselines too large for a proper interferometric processing. It may only work if you are very lucky !...

3. Single Pair processing

Before running a mass processing, it is advised to test the required processing parameters (saved in a *LaunchMTparam.txt*) on a given pair. Better do that for each satellite, in each mode, for each type of footprint. For that purpose, or maybe because you only need one pair, you can use the automatic processing script named ***SinglePair.sh***. Its architecture is presented in details in Annexes. ***SinglePair.sh*** may also be used to create (geocoded) DEM if option PROCESSMODE is set to *TOPO* in the *LaunchMTparam.txt* (see below).

SinglePair.sh will read the text file (*LaunchMTparam.txt*, or any other name you would have given it) containing all the parameters for the processing and some options you can tune (such as computing figures or skipping the time-consuming unwrapping process) and lead you (hopefully) up to the final products in a single run. ***SinglePair.sh*** also allows the processing of a given pair for which the primary image would be first coregistered on a Global Primary. However, this requires first to launch ***SuperMasterCoreg.sh*** which is the first step of the mass processing - see chapter 5 and annexes.

Even if you do not want to coregister all your products on a Global Primary, but if each of the images from your pair were already coregistered to a common image, the script will offer you the possibility to gain time by benefitting from these former processing. It will indeed benefit from former amplitude and coarse coregistration computations. The script will speak out the offer and be waiting for you to agree on that trick. He will also show you the list of all available Global Primary coregistrations and wait for you to select one. If you do not want that trick and do not want to have the script waiting for your answer, you can simply run it with the following syntax:

Echo "n" | ***SinglePair.sh***

Note that for TSX, TDX or Envisat, orbits should be good enough to skip the coarse coregistration and hence spare time. If you want to do that, you can set *CCOHWIN* to 0 in the LaunchMTparam.txt.

A version named ***SinglePairNoUnwrap.sh*** stops before unwrapping and then does not allow to geocode the products. But this version also has some adds to be used e.g. when one need accurate coregistred amplitude maps (in radar geometry) e.g. for estimating heights using shadow casted by sub-vertical structures. It will also output amplitude figures in JPEG with the date stamped on the corner. (Note that the style and font size of the date are hard coded; look for *DATECELL* parameter in script and in ***__HardCodedLines.sh***). For the specific need of computing a series of amplitude images (and output a gif animation for instance), see also the script ***ALL2GIF.sh***.

You can also use ***MultiLaunch.sh*** to run a set of ***SinglePairNoUnwrap.sh*** processing using the provided Global Primary as Primary image and all the images present in a provided directory as Secondary image. This could be useful for instance to compute a set of geocoded coherence maps that you would combine (and combine with average geocoded coherence maps computed from the other sat/track modes you plan to use e.g. for a (m)sbas processing). That set of coherence maps can be averaged, weighted, complemented with water bodies maps etc... to create a mask to be used before unwrapping all the interferograms in a full mass processing procedure. This will spare you a lot of computing time! Note that ***MultiLaunch.sh*** is so far tuned for launching ***SinglePairNoUnwrap.sh*** but adaptation would be quick for launching other scripts.

Actually, there are slightly different versions named ***MultiLaunch_ForMask.sh*** and ***MultiLaunch_Ampli_Coh.sh*** which run a set of ***SinglePair.sh*** and which have additional features:

- ***MultiLaunch_ForMask.sh*** will only compute the pairs that satisfy the criteria of max Bperp and max Btemp baseline from the provided list of pairs. It will also create a mask based on a chosen coherence threshold.

- ***MultiLaunch_Ampli_Coh.sh*** will compute amplitude and coherence maps aligned on the same grid (in slant range and geocoded). This is useful e.g. to track land cover changes. It will also allow masking the coherence in slant range etc.

See below for more information.

Note that these scripts, as the other mass processing scripts described in chapter 5 and maybe some utilities, requires the mandatory ***FUNCTIONS_FOR_MT.sh*** file which is the compilation of several functions called by the scripts.

The processing also outsources the processing of the DEM. Hence it requires ***MasterDEM.sh***

In order to keep track of the version of AMSTerEngine used to process the data, the script will store a small text file named *Processing_Pair_w_AMSTerEngine_V.txt* in the directory where the pair was processed. It contains a single line indicating the date of the last version of the software present in the /$HOME/SAR/AMSTer/AMSTerEngine/_Sources_AE/Older. It is supposed to be the last one that was compiles and hence the one currently used.

3.1. Single pair automated processing: *SinglePair.sh*

Launch ***SinglePair.sh*** with the following 3 mandatory parameters and optionally a 4th and/or 5th parameter:

PRM	(date of the Primary image)
SCD	(date of the Secondary image)
LaunchMTparam.txt	(text file with all the parameters and options chosen for the run)
REMARK	(Info used in naming the dir where computation will occur)
SuperMaster	(date of the Global Primary image - optional)

where

- *PRM* and *SCD* are the dates of the images in format YYYYMMDD
- *LaunchMTparam.txt* is a text file (and path) with all the parameters and options chosen for the run. It is a good practice to store parameters files in separate subdirectories sorted by satellite and track for instance. Name them appropriately to easily figure out what is that configuration for.

  Attention: the *LaunchMTparam.txt* paramters file must respect some syntax:
  - Parameters in file must be followed by a # and its variable name must be followed by a coma. Description is optional.
  - As reading this file is made using the first occurrence of a search criteria, do not add text with the variable name followed by a coma elsewhere in the text.
  - Always keep the main paths at the end of the file.
  - See the description of the parameters from ParametersFile.txt in annex 1.
- *REMARK* is usually optional. But it is very convenient and it is advised to use it with a bit of attention. The *REMARK* is only used to be added at the end of name of the folder where the results are computed. It is convenient to use it to remember what are the specificities of the test performed with that pair (e.g. "_Increased_Coh_threshold"), especially if you run several tests. Usually, the significance of names such as Test1, Test2… are only remembered for a short time… For the sake or readability, it is a good practice to start that parameter with an underscore.

Note that this *REMARK* is mandatory if one makes uses of *SuperMaster* option (date of Global Primary) here after as a 5th parameter. It will still be used to name the output directory but, as we will coregsitrate the processed pair on a Global Primary, it is advised to put as *REMARK* something like "*_SMSuperMaster_*" where *SuperMaster* is the date of the Global Primary image in the form of YYYYMMDD. This helps to keep track of the Global Primary used.

- *SuperMaster* is the date of an optional Global Primary image for a common coregistration on that image. Obviously if one uses that option, the 4th parameter (i.e. *REMARK*) is mandatory as explained here above.

Even if you do not work with a Global Primary, you can spare time. If each of the images from your pair were already coregistered to a common image, the script will offer you the possibility to gain time by benefitting from these former processing. The script will speak out the offer and be waiting for you to agree on that trick. If a Global Primary date is provided in the *LaunchMTparam.txt* it will use it. If not, the script will show you the list of all available Global Primary used for former coregistrations and wait for you to select one. If specific images of the pair to process are however not yet registered, the script will perform a simple processing without trying to benefit from the coregistration to a Global Primary.

You can also spare time by skipping the unwrapping if not needed (it will still geocode the results). This requires to set parameter SKIPUW as *SKIPyes* in *LaunchMTparam.txt*.

At the end of this processing, a folder will be created where all the conventional results will be stored (e.g. …/PROCESS/MT/*PRM_SCD_CROP_ZOOM_ML_REMARK*) as in a normal manual AMSTerEngine processing.

To use ***SinglePair.sh*** for computing DEM, ensure that parameter PROCESSMODE is set to *TOPO*. It is highly recommended to set also SNAPHU_MODE to *TOPO*. In that case, interpolation before geocoding is not allowed. Note that detrending is not advised when computing DEM. Masking is allowed but probably only make sense for masking water bodies. Remember that AMSTerEngine takes care and masks the layover regions by default. Remember that it also expects an external DEM to compute the topography from the interferometric pair. Indeed, it will unwrap the « residual » phase, then add the "differential phase to the computed topography. By doing so, it strongly reduces the gradients and ease the unwrapping. External DEM will also allow to geocode the final products, including the obtained new DEM.

Notes:

1) It is possible to perform recursive ***snaphu*** unwrapping (see also ***Launch_RecurUnwr.sh*** called from ***FUNCTIONS_FOR_MT.sh*** and which needs ***recur_unwr.sh***, *add_unwrphase.py*, *CropLastCol_float.py*, *findshift_bytes.py* or *findshift_float.py*, *smoothbyconv_bytes.py* or *smoothbyconv_floats.py*, *init_interf_ref.py* or *init_interf_ref_float.py*, *smoothbyconv_bytes.py* or *smoothbyconv_floats.py*, *subtract_interf_bytes.py* or *subtract_interf_float.py*, *filtercut.py*, *Maskwithseuilcoh.py*, *write_unwr_defo.py* and *CropLastLine_float.py*). This might be useful for high gradient or complex interferogram that can't be properly unwrapped. The method is based on the procedure proposed by J-L Froger and Yo Fukushima. That procedure creates a snaphu unwrapping configuration file named recur_unwr.brief.
   To operate that recursive unwrapping, set variable MULTIUWP to MultiSnaphuYes and select the following variables in your *LaunchMTparam.txt* as in the example below:

```
MultiSnaphuYes          # MULTIUWP, MultiSnaphuYes performs recursive snaphu unwrapping
                              # (need 4 params bellow). MultiUnwrapNo (or any other string)
                              # will perform single snaphu unwrapping
ResidInterfFilt         # WHICHINTERF, which interferogram to unwrap, ResidInterf (residual
                              # interfero) or ResidInterfFilt (residual interfero filtered)
0.9                     # COEFREQ, Coefficient of increase of cut-off frequency
12.5                    # CUTINI, Initial cut-off frequency (e.g. 12.5 for a 400x400 image,
```

```
                        #10 for a 2200x1500 img)
10                  # NITMAX, Max total nr of iterrations
0.0627              # COHMUWPTHRESH, coh threshold (between 0 and 1) below which it replaces
                        # the phase by white noise (corresponding mask will be produced).
                        # If set to 0, do not mask with white noise
```

It is also possible to do that during the mass processing. However, beware that this will be time consuming of course (for a single interferogram and a fortiori on a mass processing) !

2) It is possible to crop the images to a region of interest and hence gain computer time and resources. This is performed by adjusting the CROP parameters in *LaunchMTparam.txt, that is* :

```
CROPyes         # CROP, CROPyes or CROPno, or for S1, path to kml that will be used to define area of interest.
10000           # FIRSTP, Crop limits: first point (row) to use
8000            # FIRSTL, Crop limits: first line to use
24000           # LASTP, Crop limits: last point (row) to use
12000           # LASTL, Crop limits: last line to use
1               # ZOOM, factor during crop
aName           # REGION, Text description of area for dir naming
```

Attention :

- CROP: If option is set to *CROPyes*, images from all satellites (including Sentinel1 in Strip Map mode) but Sentinel1 in Wide Swath mode will be cropped based on the coordinates of the corners provided below (in LINES/PIXELS numbers, or with LAT/LONG coordinates).
  Sentinel1 TOPSAR Wide Swath images, when a ZOOM factor of *1* is chosen, must be cropped by providing a *path_to_kml*. The interferometric products and all subsequent products will be cropped at that footprint. Steps up to the S1coregistration will be performed on the whole set of tracks/swaths/bursts that span the area of interest defined at reading.
  However, if a ZOOM factor other than *1* is wanted, Sentinel1 TOPSAR Wide Swath images must be cropped by providing LINES/PIXELS numbers (a bug in AMSTerEngine prevents so far defining crop zone using LAT/LONG coordinates). (See Annex A.1).

- ZOOM: image oversampling (zoom) is possible for images from all satellites (including Sentinel1 in Strip Map mode) but Sentinel1 in Wide Swath mode by setting the CROP parameter as *CROPyes*, by providing the corners of the CROP region as LINES/PIXELS numbers or LAT/LONG coordinates, and by providing the ZOOM factor. Zooming Sentinel1 TOPSAR Wide Swath images can only be performed by providing LINES/PIXELS numbers (a bug in AMSTerEngine prevents so far defining crop zone using LAT/LONG coordinates). (See Annex A.1).
  Note that ML factor of zoomed processing will then be ML of Zoomed pixels.

3) There is a hard-coded parameter at the beginning of the script that may be useful to check: MAXSIGMARGAZ (set by default to 5) is the maximum value of Sigma (in Range and Azimuth) admitted for successful Fine Coregistration. If Sigma obtained during the processing is larger, it attempts to restart a fine coregistration with larger window size (maximum 3 times) until it ends up with an acceptable value of Sigma (that is below MAXSIGMARGAZ).

4) For TDX in Tandem (i.e. Primary date = Secondary date) or for S1, no coregistration on a Global Primary is possible. Hence it will not offer you that possibility. For all the other cases, it will ask you to answer *Yes* or *No* at the prompt.
If you want to answer this question at the launch of the script rather than answering at the prompt, launch the command preceded by echo followed by the answer then pipe followed by the script. E.g.

echo "*n*" | ***SinglePair.sh*** *prm scd LaunchMTparam.txt…*

5) For TDX in Tandem, when the secondary image is used as RX, its image in SLC format is linked in the TX directory although with a dummy date where millennia is changed in 9000 (e.g. 90190730 instead of 20190730). This is mandatory to preserve the integrity of the processing chain and directory naming conventions.

6) If you processed a pair with ***SinglePair.sh*** and want to add these results to you mass processing directories as if it was computed by a normal ***SuperMaster_MassProcess.sh***, you can do that with zz_Utilities_MT_Ndo/***Move_SinglePair_Results_To_MASSPROCESS.sh*** which will rename and move the geocoded results in the appropriate directories and subdirectories in …/SAR_MASSPROCESS/*SAT*/*REGION*/SM*Crop_Zoom_ML*/Geocoded/…, …/SAR_MASSPROCESS/*SAT*/*REGION*/SM*Crop_Zoom_ML*/GeocodedRasters/… and move the directory with the rest of the processing in …/SAR_MASSPROCESS/*SAT*/*REGION*/SM*Crop_Zoom_ML*/

   Of course, ***SinglePair.sh*** must have been performed with geocoding on the SAME Global Primary as what already exists in the mass processing**.**

See the description of ***SinglePair.sh*** script in annexes.

3.2. Single pair automated processing without unwrapping and geocoding: *SinglePairNoUnwrap.sh*

This is processing is the same as ***SinglePair.sh*** except that it will:

- stop before the unwrapping; It will not do the geocoding neither
- If run with 7 parameters, it means you want to output amplitude images in radar geometry with date label. In such a case the *SuperMaster* parameter will be set to *NOSM*. If you want to run several ***SinglePairNoUnwrap.sh*** for shadows measurements or for producing gif animation for instance, see ***ALL2GIF.sh***.
- if run with 7 parameters, it will tag the amplitude images in radar geometry with the date. Position of that tag in the image is adjusted with the parameters *LabelX* and *LabelY* from the command line. Note that the font style and size of that date tag can be adjusted by changing the *DATECELL* hard coded parameter in the script. Some are already pre-defined depending on the satellite.
- Trim the cropped amplitude image in radar geometry.
  Explanation: during the InSAR processing, images are cropped according to the parameters *FIRSTP, FIRSTL, LASTP, LASTL* that are set in the *LaunchMTparam.txt* file. However, if several pairs of images are processed in the intend of producing a stack of amplitude image of exactly the same dimensions (e.g. for shadows measurements or for producing animation gifs using ***ALL2GIF.sh***), it might be useful to trim a little more the cropped images. This might be necessary e.g when the target region is close to the border of the raw image causing the size of the image to vary from one acquisition to another if the satellite does not start/stop exactly on the same footprint.
  To perform that trimming, the script ***SinglePairNoUnwrap.sh*** uses the parameters *LLRGCO* (i.e. the Lower Left Range Coordinate Offset for final interferometric products generation) and *LLAZCO* (i.e. the Lower Left Azimuth Coordinate Offset for final interferometric products generation) that are set in the *LaunchMTparam.txt* file. The same offsets will be assigned to the upper right corner. The final cropped output amplitude images in radar geometry will hence be (*LASTP-FIRSTP*)-(2x*LLRGCO*) wide and (*LASTL-FIRSTL*)-(2x*LLAZCO*) high (Figure 5).

If all the amplitude images in radar geometry do not have the same size and footprint as expected, adjust your trimming offsets.

If no *LLRGCO* and *LLAZCO* are provided in the *LaunchMTparam.txt* file, it will use 50 pixels for both by default.

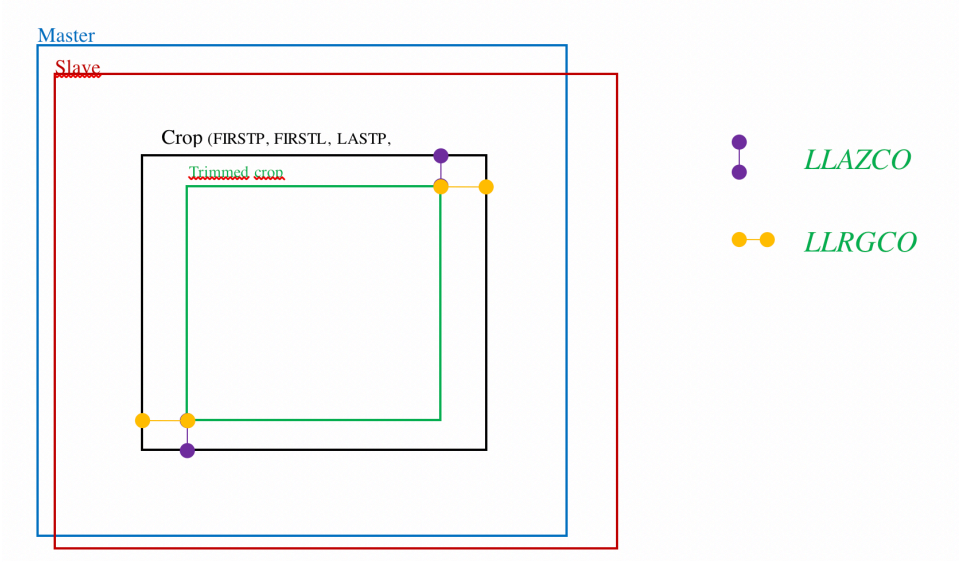


Figure 5: Crop and Trim convention

Launch ***SinglePairNoUnwrap.sh*** with the following 4 - 7 parameters:

PRM	(date of the Primary image)
SCD	(date of the Secondary image)
LaunchMTparam.txt	(text file with all the parameters and options chosen for the run)
REMARK	(Info used in naming the dir where computation will occur)
SuperMaster	(date of the Global Primary image, only if used with 5 parameters. If run with 7 parameters, it will be forced to NOSM)
Label X	(Position (in pixels) of the date label added on amplitude images)
Label Y	(Position (in pixels) of the date label added on amplitude images)

where

- *PRM* and *SCD* are the dates of the images in format YYYYMMDD
- *LaunchMTparam.txt* is a text file (and path) with all the parameters and options chosen for the run. It is a good practice to store parameters files in separate subdirectories sorted by satellite and track for instance.

  Attention: the *LaunchMTparam.txt* parameters file must respect some syntax:
  - Parameters in file must be followed by a # and its variable name must be followed by a coma. Description is optional.
  - As reading this file is made using the first occurrence of a search criteria, do not add text with the variable name followed by a coma elsewhere in the text.

  - Always keep the paths at the end of the file.
  - See the description of the parameters from *ParametersFile.txt* in annex 1.
- *REMARK* is mandatory here, though it is only used to be added at the end of name of the folder where the results are computed. It is convenient to use it to remember what are the specificities of the test performed with that pair (e.g. "_Increased_Coh_threshold"), especially if you run several tests. Usually, the significance of names such as Test1, Test2… are only remembered for a short time… For the sake or readability, it is a good practice to start that parameter with an underscore.
- *SuperMaster* here is either the date of the Global Primary if run with only 5 parameters (similar usage as ***SinglePair.sh*** though without unwrapping), or set to *NOSM* if used with 7 parameters for generating amplitude images with a date tag e.g. for a gif animation (see for instance ***MultiLaunch.sh*** in 3.3 below or ***ALL2GIF.sh*** in 3.5).
- *Label X* and *Label Y* are the X and Y coordinates of the date tag to be overlaid to the amplitude image.

Notes:

- There is a hard-coded parameter at the beginning of the script that may be useful to check: MAXSIGMARGAZ (set by default to 5) is the maximum value of Sigma (in Range and Azimuth) admitted for successful Fine Coregistration. If Sigma obtained during the processing is larger, it attempts to restart a fine coregistration with larger window size (maximum 3 times) until it ends up with an acceptable value of Sigma (that is below MAXSIGMARGAZ).
- The script uses a function to crop the data to a desired zone based on the *cutAndZoom* function from *AMSTerEngine*. However, before June 2022, that function cropped the zone based on the provided coordinates while considering a default zero mean altitude, which could of course result in an offset cropped zone when the area of interest is at high altitude. From June 2022, the *cutAndZoom* function crops the region based on the altitude computed from the provided DEM. The script ***SinglePairNoUnwrap.sh*** can use both type of cropping (of course the most recent one is the best, but older version might still be used for compatibility if a long time series already exists). The selection of new or old cropping method is performed by hard coded *CROPFCT* parameter at the beginning of the script, which can be set as *Crop* (to crop at mean altitude computed from provided DEM) or *CropAtZeroAlt*. Both these functions are defined ***FUNCTIONS_FOR_MT.sh***.

3.3. Batch run of Single pair automated processing: *MultiLaunch.sh*

This script is aiming at launching multiple occurrences of a ***SinglePairNoUnwrap.sh*** processing using a Global Primary image as Primary image, and all other existing images in the input directory as Secondary images in order to obtain a set of amplitude images in radar geometry all cropped and trimmed the same way. It can be run incrementally i.e. it will only compute the new pairs. It can also be used by providing as a 7th parameter to process only a list of Secondary images.

So far it is tuned for using ***SinglePairNoUnwrap.sh*** but it should be straight forward to adapt that to ***SinglePair.sh*** as already done in ***MultiLaunch_ForMask.sh*** (see below 3.4).

Launch ***MultiLaunch.sh*** with the following 4 (or 6 or 7) parameters:

SuperMaster	(date of the Global Primary image)
INPUTDATA	(path to dir where original data are stored in csl format)
OUTPUTDATA	(path to dir where results will be stored)
LaunchMTparam.txt	(text file with all the parameters and options chosen for the run)
Label X	(Position (in pixels) of the date label added on amplitude images)
Label Y	(Position (in pixels) of the date label added on amplitude images)
PairsList.txt	(text file with a list of images to process instead of all possible ones. Images are given as YYYYMMDD or S1 names)

where

- *SuperMaster:* date of the Global Primary image in format YYYYMMDD,
- *INPUTDATA:* Usually something like … /SAR_CSL/*SAT*/*TRK*/NoCrop
- *OUTPUTDATA:* If it is for amplitude stacking purpose e.g. for shadows measurements, it might be something like … /SAR_SM/AMPLITUDES/*SAT*/*TRK*/*REGION*
- *LaunchMTparam.txt* is a text file (and path) with all the parameters and options chosen for the run.
- *Label X* and *Label Y* are the X and Y coordinates of the date tag to be overlaid to the amplitude image
- *PairsList.tx* is a text file with a list of images to process instead of all possible ones. Images are given as YYYYMMDD or S1 names

3.4. Batch run of Single pair automated processing: *MultiLaunch_ForMask.sh*

This script is aiming at launching multiple occurrences of a ***SinglePair.sh*** processing using either:

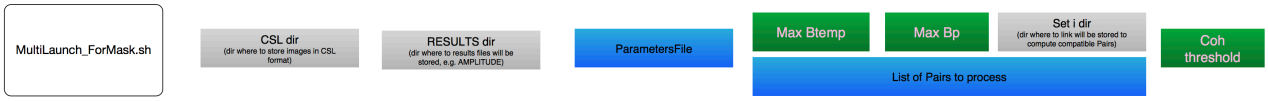
- a Global Primary image as Primary (taken from the ParametersFile.txt) and all other existing images in the input directory as Secondary in order to compute the geocoded coherence maps of every image with the Global Primary, as far as Bp and Bt are smaller than what is input as parameters.
- A list of pairs you can choose manually
  (in the form of …/SAR_SM/MSBAS/*REGION*/set*i*/SM_Approx_baselines.txt or …/SAR_SM/MSBAS/*REGION*/set*i*/table_BpMin_BpMax_BtMin_BtMax.txt, that is 4 columns with Primary and Secondary dates followed by Bp and Bt, with or without header).

It will then create a mask based on a chosen coherence threshold and save it as 8 bits as required by AMSTerEngine. It must however be transformed in LatLong before being used by AMSTerEngine as a mask. Thanks to a line suggested by A. Dille, the script now does it for you but it requires gdal installed on your computer.
Check that the created mask in LatLong (name ending with _LL) has a header with consistent name.
If you use AMSTerEngine version from V20230928, you will need to invert your mask (see below of chapter 0.13).

MultiLaunch_ForMask.sh can be run incrementally i.e. it will only compute the new pairs.

To spare time, it is advised:
- to disable in the *LaunchMTparam.txt* the unwrapping by selecting option *SKIPUW* as *SKIPyes* or even as *Mask*. Option *Mask* geocodes only the coherence and, for check purpose, the amplitudes and residual interferogram.
- to disable in the *LaunchMTparam.txt* the masking by selecting option *APPLYMASK* as *APPLYMASKno*.

Do not forget to set these options back after if you intend to use the same *LaunchMTparam.txt* for mass processing.



Launch ***MultiLaunch_ForMask.sh*** with either 5 or 7 parameters:
If you want to run it with criteria of Bt and Bp and a Global Primary:

INPUTDATA	(path to dir where original data are stored in csl format)
OUTPUTDATA	(path to dir where results will be stored)
LaunchMTparam.txt	(text file with all the parameters and options chosen for the run)
MaxBt	(Maximum temporal baseline to compute pair)
MaxBp	(Maximum spatial baseline to compute pair)
Set_i_dir	(Path to file where baselines are computed)
Coh_threshold	(Coherence threshold)

If you want to run it with a table where you manually selected a list of pairs to process:

INPUTDATA	(path to dir where original data are stored in csl format)
OUTPUTDATA	(path to dir where results will be stored)
LaunchMTparam.txt	(text file with all the parameters and options chosen for the run)
PairsFile.txt	(list of pairs to process, it can have any name)
Coh_threshold	(Coherence threshold)

where
- *INPUTDATA:* Usually something like …/SAR_CSL/*SAT*/*TRK*/NoCrop
- *OUTPUTDATA:* path to where processed pairs will be stored, it might be something like …/SAR_SM/MASKS/*SAT*/*TRK*/*REGION* or …/SAR_SM/AMPLITUDE/*SAT*/*TRK*/*REGION*
- *LaunchMTparam.txt* is a text file (and path) with all the parameters and options chosen for the run. If run with 7 parameters (i.e. with criteria of Bt and Bp), it MUST contains the date of the Global Primary (*SuperMaster*) used to coregister all the Secondary images.
- *MaxBt* and *MaxBp* are the maximum temporal and perpendicular baselines under which the pairs will be computed.
  OR
  A table *PairsFile.txt* with a list of pairs in the form of
  … /SAR_SM/MSBAS/*REGION*/set*i*/SM_Approx_baselines.txt or
  … /SAR_SM/MSBAS/*REGION*/set*i*/table_BpMin_BpMax_BtMin_BtMax.txt).

- *Set_i_dir* is the path **and name of file** containing baselines computed using ***Prepa_MSBAS.sh*** (see chapter 4 below), i.e. a file named like … /SAR_SM/MSBAS/*REGION*/set*i*/SM_Approx_baselines.txt.

  Note: if there is no satisfying pair using the Global Primary (GLOBPRM_SCD or SCD_GLOBPRM) in SM_Approx_baselines.txt, the script will offer to work with table_MinBp_MaxBp_MinBt_MaxBt.txt instead, which contains all pairs satisfying criteria. But that might be too much. It will hence take only into account the first 10 pairs. It is the operator's responsibility to verify that it is significant. If needed, either change your table_MinBp_MaxBp_MinBt_MaxBt.txt or change the script to change the number of pairs to keep.
- *Coh_threshold* is the coherence threshold used to compute the mask. The script will actually compute the mean of all the coherence maps, then apply the threshold and save the mask as coherence_above_*Coh_threshold*.mean and create a gif figure of it. Launching the script for the first time takes time because it must compute the mean of all the coherence maps. However, when this is already computed, relaunching the script with only a different coherence threshold is way much faster.

The script ***MultiLaunch_ForMask.sh*** requires *python*, *numpy* and *OpenCV*. If one prefers to compute the median rather than the mean of coherence maps, see the script *MeanCoh.py*.

Note: With AMSTerEngine version before September 2023, the mask can be combined with coherence when unwrapping is performed with *snaphu*. See parameter *COHCLNTHRESH*. Pixels with coherence above threshold OR where mask = 1 will be unwrapped. If *COHCLNTHRESH* is set to 0, it will only use the mask provided without taking into account the coherence:

Mask	Coh > *COHCLNTHRESH*	unwrapping
1	1	Yes
1	0	Yes
0	1	Yes
0	0	No

Warning: since AMSTerEngine V20230921, masks are inverted and multi-layers, that is:

- 0 is to keep (no mask applied)
- 1 is always masked (e.g. water bodies area)
- 2 is masked during unwrapping except is coherence is above a given threshold (see parameter COHCLNTHRESH in *LaunchMTparam.txt*)

That is:

Mask	Coh > *COHCLNTHRESH*	unwrapping
0	yes	Yes
0	no	Yes
1	yes	No
1	no	No
2	yes	Yes
2	no	No

3.5. Batch run of Single pair automated processing: *MultiLaunch_Ampli_Coh.sh*

This script is aiming at launching multiple occurrences of a ***SinglePair.sh*** processing using either a list of pairs you can choose manually in the form of a list of dates separated by a space, or in the form of …/SAR_SM/MSBAS/*REGION*/set*i*/SM_Approx_baselines.txt or even …/SAR_SM/MSBAS/*REGION*/set*i*/table_BpMin_BpMax_BtMin_BtMax.txt, that is 4 columns with Primary and Secondary dates followed by Bp and Bt, with or without header.

It will compute each pair from the provided list and that was not processed yet in order to :

- create slant range amplitudes and coherence images aligned on the same grid for all the pairs
- geocode Ampli and coh only
- create a fake hdr file for coherence in slant range (to allow reading them with a GIS software)
- mask the coherence in slant range (can be opened with a GIS with the same hdr as unmasked coherence)
- move all the geocoded coherence and amplitude images in dedicated directories:
  - _ALL_COH_GEOC
  - _ALL_COH_SLANTRG
  - _ALL_AMPLI_SLANTRG (or _ALL_AMPLI_SIGMA_SLANTRG for S1 data if Sigma Nought calibration is requested in the *LaunchMTparam.txt*)
  - _ALL_AMPLI_GEOC (or _ALL_AMPLI_SIGMA_GEOC for S1 data if Sigma Nought calibration is requested in the *LaunchMTparam.txt*)

These features are useful for checking land cover changes, or track landslides etc…

To spare time, it is advised to disable in the *LaunchMTparam.txt* the unwrapping by selecting option SKIPUW as *SKIPyes*. Do not crop S1 Wide Swath images with a kml. It would prevent the size of slant range products to remain the same for all pairs.

Launch ***MultiLaunch_Ampli_Coh.sh*** with the 4 parameters:

INPUTDATA	(path to dir where original data are stored in csl format)
OUTPUTDATA	(path to dir where results will be stored)
LaunchMTparam.txt	(text file with all the parameters and options chosen for the run)
PairsFile.txt	(list of pairs to process, it can have any name)

where

- *INPUTDATA:* Usually something like …/SAR_CSL/*SAT*/*TRK*/NoCrop
- *OUTPUTDATA:* Path to where processed pairs will be stored. It will automatically add */SAT/TRK/REGION* to that path
- *LaunchMTparam.txt* A text file (and path) with all the parameters and options chosen for the run.
- *PairsFile.txt:* A table with a list of pairs in the form of *PrimaryDate SecondaryDate* or … /SAR_SM/MSBAS/*REGION*/set*i*/SM_Approx_baselines.txt or … /SAR_SM/MSBAS/*REGION*/set*i*/table_BpMin_BpMax_BtMin_BtMax.txt

Note that it created a file named *_SizeOfCroppedAreaOfInterest.txt* where it stores the limits of the common slant range area (as it does with ***MultiLaunch.sh***).

3.6. Batch run of automated processing for amplitude images: *ALL2GIF.sh*

This script is an additional layer of automation to launch ***MultiLaunch.sh*** (i.e a set of ***SinglePairNoUnwrap.sh***) using a Global Primary image as Primary and all other existing images as Secondary, then other scripts (namely ***Cp_Ampli.sh***, ***CheckAreOfInterest.sh*** and ***jpg2movie_gif.sh***) in order to output an animated gif of amplitude images and figures for shadow tracking for instance. It can be run incrementally i.e. it will only compute the new pairs.

At the end of this processing, amplitude images in radar geometry (flipped or flopped if required) will be stored in Envi format (binary + header txt file) in …/SAR_SM/AMPLITUDES/*SAT*/*TRK*/*REGION*/AMPLI_.

A copy of these files in jpg and an animated gif will be stored in the same directory. Attention, resulting gif file can become huge; opening big ones (e.g. > 2.5 Gb) may crash on some computers.

You can further use *Fiji* to equalise the amplitude gif (and math>log processed) for a better contrast, and/or to crop the gif etc...

In the …/SAR_SM/AMPLITUDES/*SAT*/*TRK*/*REGION*/ directory, there will be also a set of sub-directories containing the intermediate data from computation for each pair.

Launch ***ALL2GIF.sh*** with the following 4 parameters:

SuperMaster	(date of the Global Primary image)
LaunchMTparam.txt	(text file with all the parameters and options chosen for the run)
Label X	(Position (in pixels) of the date label added on amplitude images)
Label Y	(Position (in pixels) of the date label added on amplitude images)

where

- *SuperMaster:* date of the Global Primary image in format YYYYMMDD,
- *LaunchMTparam.txt* is a text file (and path) with all the parameters and options chosen for the run.
- *Label X* and *Label Y* are the X and Y coordinates of the date tag to be overlaid to the amplitude image.

Notes:

- If images "jumps" in the gif file, check _List_Az_Rg_UpperLeft_coord.txt (created by ***CheckAreaOfInterest_InAmplitudesDir.sh*** launched within ***ALL2GIF.sh***) for pairs whose Upper right range and azimuth coordinate are not as expected (that is not the same as in _SizeOfCroppedAreaOfInterest.txt). Either discard these wrong images or try to adjust the parameters LLRGCO and LLAZCO in LaunchMTparam.txt (see ***SinglePairNoUnwrap.sh*** in 3.2.
- ***ALL2GIF.sh*** makes use of ***jpg2movie_gif.sh***
- There is a version that should produce a gif animation with any other type of geocoded products from all pair directories in the current directory. See ***AllProd2GIF.sh*** (in SCRIPTS_MT/zz_Utilities_MT_Ndo)
- During the conversion to jpeg, *convert* uses a ratio to make the image square. Check that its output resolution fits your expectation.
- After completion of ***ALL2GIF.sh*** you can geocode the amplitudes. This is done by running the script ***Geocode_from_ALL2GIF.sh*** in the …/SAR_SM/AMPLITUDES/SAT/TRK/REGION/ directory with a parameter file containing the desired details of the geocoding. This parameter file has a structure similar to the LaunchMTparam.txt files and must contains the following lines (see for instance ___V20200812_LaunchParamReGeocAmpli.txt):

```
# PARAMETERS TO RUN SCRIPT Geocode_from_ALL2GIF.sh.
# PARAMETERS MUST BE FOLLOWED BY A # AND ITS VAR NAME FOLLOWED BY COMA.
#         (DESCRIPTION, THOUGH THIS IS OPTIONAL)
# AS READING THIS FILE IS MADE USING FIRST OCCURENCE OF SEARCH CRITERIA,
#         DO NOT ADD TEXT WTH VARIABLE NAME FOLLOWED BY COMA.
#
# VERSION August 12 2020

# GEOCODING
############
UTM             # PROJ, Chosen projection (UTM or GEOC - both are ok here)
TRI             # RESAMPMETHD, TRI = Triangulation; AV = weighted average; NN = nearest neighbour
LORENTZ         # WEIGHTMETHD, Weighting method : ID = inverse distance; LORENTZ = lorentzian
1.0             # IDSMOOTH,  ID smoothing factor
1.0             # IDWEIGHT, ID weighting exponent
0.1             # FWHM, Lorentzian Full Width at Half Maximum
15              # XPIX, Easting sampling [m] if UTM or Longitude sampling [dd] if LatLong
15              # YPIX, Northing sampling [m] if UTM or Latitude sampling [dd] if LatLong
pathToKmlFile   # AREAOFINT, Forced footprint of geocoded product : Path_to_a_kml_file or pathToKmlFile to ignore forcing
```

Geocoded products will be stored in …/SAR_SM/AMPLITUDES/SAT/TRK/REGION/_GEOCAMPLI

Remark:

A script names ***MakeAmpliPlotSingleImg.sh*** also creates amplitude (module) images, though for only one image at a time. This is convenient 1) if only one image is available (no pair possible), 2) if no coregistration on a Global Primary is required or 3) if speed is important (as it skips all processes other than what is needed to just compute the module image). See script in $PATH_SCRIPTS/SCRIPTS_MT/zz_Utilities_MT.

Note that the script ***MakeAmpliPlotSingleImg.sh*** requires a parameter file as the one named *___V20220719_LaunchParamAmpli.txt* and provided as a template in $PATH_SCRIPTS/SCRIPTS_MT. That file contains information about desired multilooking and zoom factors, output pixel shape (SQUARE or ORIGINAL), crop region (provided either as pixels coordinates in lines and columns or geographic coordinates or kml[4]) and a path to an external DEM. Note that unlike in the case of the interferometric processing, the SQUARE option will compute the ratio between azimuth and range pixel size without taking into account the incidence angle in order to display squared pixels in Slant Range. Taking into account the incidence angle would square the pixel in Ground Range. If needed, uncomment the lines in the script to square the pixels taking into account the incidence angle.
The script is launched by providing it with two parameters: the path to the image (which MUST be in a directory named .../SAR_CSL/*SAT*/*TRK*/NoCrop/*IMAGE*.csl because the name of the satellite, track and date are extracted from that path) and the parameter file *___V20220719_LaunchParamAmpli.txt*.

[4] Attention, cropping using geographic coordinates or kml may be a problem for very small area located e.g. in high relief area as the crop region is defined based on the mean altitude of the whole image in order to define the crop in slant range-azimuth. If altitude of the small cropped region differs significantly from the mean altitude of the scene, the cropped region might be offset significantly. Providing the cutAndZoom parameter file with the path to the DEM may improve. Avoid also locating margins of the crop in lay over region.

4. Preparation for Mass Processing

A series of scripts aims at preparing directories and parameter files that will be used in the mass processing procedure.

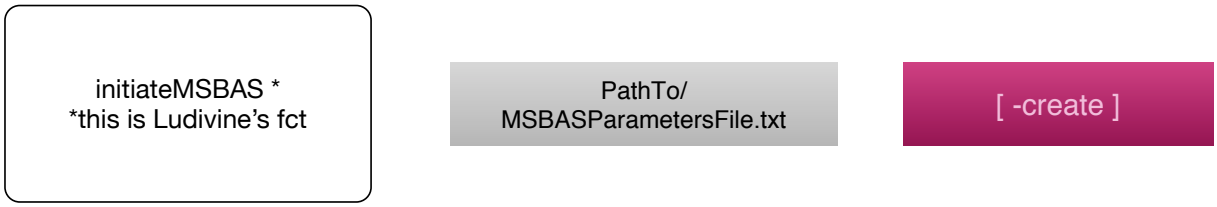
4.1. Prepare the directories used in the mass processing: *initiateMSBAS*

This is a function from AMSTerEngine (not a script) prepared by Ludivine Libert and is aiming at preparing a list of set*i* directories where links to each *SAT/TRK* images in *csl* format will be stored to compute the compatible pairs, as well as creating parameters files that will be needed later (for the MSBAS pair selection etc.).

Example of set for the VVP:

set1 = CSK Asc	set4 = R2 F21N
set2 = CSK Desc	set5 = R2 UF
set3 = R2 F2F	set6 = S1 Asc 174
	Set7 = S1 Desc 21…

Remark: It is advised to keep a track somewhere in a file of the images (sensor, track, region) present in the different *set i.*



To operate this,

- First create an *MSBASParametersFile.txt* in the …/SAR_SM/MSBAS/*REGION/* directory as follows:
  - Launch the command
    initiateMSBAS…/SAR_SM/MSBAS/*REGION*/MSBASParametersFile.txt -create
  - Modify the *MSBASParametersFile.txt* according to your data. e.g.:

```
/* MSBAS Initialization parameter file */
/* ********************************** */
…./SAR_SM/MSBAS/Bukavu/        /* Path to MSBAS directory */
4                              /* Number of sets for MSBAS processing */
40                             /* Total number of CSL images in all sets */
/* The total number of CSL images can be replaced by a maximum number of images */
```

  - The Path to MSBAS directory is the path where your *MSBASParametersFile.txt* is, as well as your different set*i* e.g. .../SAR_SM/MSBAS/*REGION*

- Re-run ***initiateMSBAS*** to create the desired number of set using this newly created and just modified *MSBASParametersFile.txt*:

  initiateMSBAS …/SAR_SM/MSBAS/REGION/MSBASParametersFile.txt

  (Note that the option *-create* is not anymore at the end of the command line).

When done, all the set*i* are in …/SAR_SM/MSBAS/REGION/MSBAS/set*i* directory. This is not necessary. You can move then one level of directory up and delete the empty MSBAS dir. You should end up with a path like …/SAR_SM/MSBAS/REGION/set*i*.

4.2. Create links to the original *csl* images in their respective *set* directory: *Ins_All_Img.sh*

The ***Ins_All_Img.sh*** script is aiming at creating a symbolic link for the *csl* data in the respective set*i* directories where they will be used by the AMSTerEngine automated scripts (e.g. ***Prepa_MSBAS.sh***).

In each set*i*, run ***Ins_All_Img.sh*** with following 3 parameters:

…/SAR_CSL/*SAT*/*TRKDIR*/NoCrop (where the data are stored in *csl* format)
…/SAR_SM/MSBAS/*REGION*/set*i* (where links will be stored in *i*th set)
SAT (name of the satellite; must be compatible with dir naming)

where

- *SAT* is the name of the satellite (must be compatible with dir naming),
- *TRKDIR* is the track of the satellite (must be compatible with dir naming),
- *REGION* is the region
- set*i* is the series of directories names set*1*, set*2*…. set*n* where links to each *SAT/TRK* images in *csl* format will be stored to compute the compatible pairs.

Note that the order of the different set *i* is arbitrary and does not matter but you need to remember which SAT/TRACK is in each set directory.

Make one run like that for each set*i*.

4.3. Compute the compatible pairs and make the baseline plots: *Prepa_MSBAS.sh*

This script is aiming at preparing the MSBAS data sets and computing baselines table and baselines plot. Data must have already been linked in *csl* format in the different directory (see chapter 4.2). One can force the date of the Global Primary if you do not want to compute a new one. This is mandatory when adding image(s) to a large existing data base.

The script is able to create tables and plots using former tools made by Ludivine Libert (using functions ***approximateBaselines, selectInterferometricPairs, globalMaster*** and using for instance the script ***build_bperp_file.sh***), or with more recent tools (after May 2022) by Dominique Derauw (using AMSTerEngine function ***baselinePlot***).

The script ***Prepa_MSBAS.sh*** checks if the AMSTerEngine function ***baselinePlot*** exists, which would mean that the most recent tools by D Derauw are available. Using the former or the most recent tools should be transparent to the user as the scripts will generate all the necessary files. However, the old tool might be slower and may consider the criteria for the baselines search as exclusive, that is if you search for a max Bp of 120m, 120 will not be taken into account.

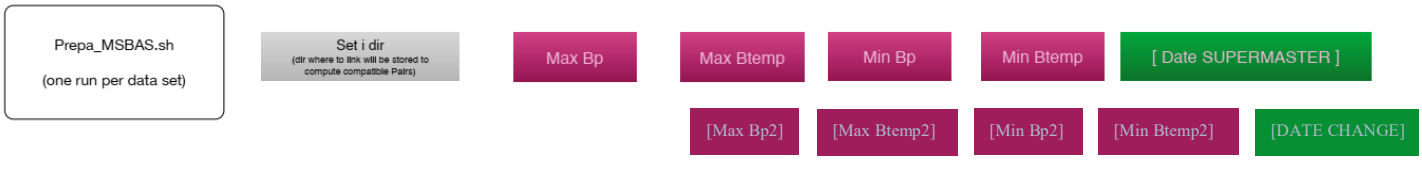
Note that the tools by L. Libert allows selecting a minimum Btemp or Bp, while the tool by D. Derauw considers that the *MinBtemp* and *MinBp* are *zero*. As a consequence, ***Prepa_MSBAS.sh*** may be launched with either 2 or 4 baselines criteria as parameters. The script will automatically check, given the number of baselines criteria, what to do. If you provide it with 4 criteria including *MinBtemp* and *MinBp* different from *zero*, it will launch the old tools. If you provide him with 4 criteria including *MinBtemp* and *MinBp* equal *zero*, it will try to launch the new tool if available, or the old tools if not.

Note that since its version 4.0 (January 2023), the scripts ***Prepa_MSBAS.sh*** is able to cope with two different sets of criteria. This was developed to assist the continuation of long Sentinel-1 time series that were using short Bp and/or Btemp thanks to the narrow orbital tube maintained by ESA. However, since the failure of Sentinel-1B in December 2021, and to spare the Sentinel-1A satellite's fuel, ESA increased the size of the orbital tube. The time series started experiencing gaps in the baseline plots. To overcome the problem, one can launch ***Prepa_MSBAS.sh*** with two sets of baselines criteria and a date from which the second set of criteria must be applied. Hence, one can keep the existing short-baselines baseline plots until e.g. mid 2022, then use larger baseline criteria after. This requires to launch the script with 3 or 5 more parameters (depending if one wants to use the old or the new tools). See below.

The ***Prepa_MSBAS.sh*** has to be run for each set. It will calculate a baseline plot and the optimal Global Primary image for each set, based on the selected baseline parameters.

Note:

- With the new tools to compute the baseline plots, the date of the Global Primary from the dataset satisfying the baseline criteria (*BP* and *BT*) can be found in the baselinePlot_BpMax=*BP*_BTMax=*BT*.png. The Global Primary that is mentioned in the allPairsListing.txt file is the Global Primary from the whole dataset.
- When using the old tools to compute the baseline plots, the date of the Global Primary can be found in setParametersFile.txt.



In each set, run "***Prepa_MSBAS.sh***" with 5 or 6 parameters:

…/SAR_SM/MSBAS/*REGION*/set*i*	(dir containing links to data)
MaxBp	(Max perpendicular baseline criteria for pair selection)
MaxBtemp	(Max temporal baseline criteria for pair selection)
[*MinBp*]	(Min perpendicular baseline criteria for pair selection)
[*MinBtemp*]	(Min temporal baseline criteria for pair selection)
DateOfSuperMaster	(date of Global Primary if it has to be forced)
MaxBp2	(Max perpendicular baseline criteria for pair selection, to be applied after a given date, i.e. *DateChangeCriteria*)
MaxBtemp2	(Max temporal baseline criteria for pair selection, to be applied after a given date, i.e. *DateChangeCriteria*)
[*MinBp2*]	(Min perpendicular baseline criteria for pair selection, to be applied after a given date, i.e. *DateChangeCriteria*)
[*MinBtemp2*]	(Min temporal baseline criteria for pair selection, to be applied after a given date, i.e. *DateChangeCriteria*)
[*DateChangeCriteria*]	(date before which the pair selection is made based on *MaxBp* and *MaxBtemp*, and after which the pair selection is made based on *MaxBp2* and *MaxBtemp2*)

MinBp and *MinBtemp* (and *MinBp2* and *MinBtemp2*) are not mandatory if you operate with the AMSTerEngine tools by D. Derauw after May 2022. They would then be set automatically to zero.

Usually, *MinBp* and *MinBtemp* (and *MinBp2* and *MinBtemp2*) are 0 unless you want to select only a new bunch of pairs with new criteria. For instance, suppose you processed images with Bp comprise between 0-100 meters and you want later to add pairs with Bp comprised between 100 and 150 meters. Running again the script with Bp criteria set to 0-150m would compute all the pairs satisfying that range. Running the script with Bp criteria set to 100-150m would only compute the new pairs, hence you can know exactly how many pairs you add. Note that it would not make a difference at the mass processing step because the mass processing ignores pairs that were already computed.

DateOfSuperMaster is required if one do not want to recompute a new Global Primary image. For instance, when you assess the best Global Primary, then compute the mass processing of all the compatible pairs, then add a new image, you do not want to evaluate a new Global Primary. Indeed, selecting a new Global Primary would result in reprocessing the whole bunch of pairs rather than only the new compatible pairs (i.e. containing only the new image).

During the processing of ***Prepa_MSBAS.sh***, the script will (vocally) ask you if you want to select a new Global Primary. The former Global Primary used can be found in *setParametersFile.txt.*

After that run, a table with all the pairs to be used in the ***SuperMaster_MassProc.sh*** is written in each set (e.g. *Table_0_100_0_100.txt).* It contains something like:

Master	Slave	Bperp	Delay
20180306	20180318	-43	-12
20180306	20180330	-55	-24
20180318	20180330	-12	-12
20180306	20180411	-52	-36
20180318	20180411	-8	-24
20180330	20180411	3	-12
20180306	20180423	-11	-48
20180318	20180423	31	-36
20180330	20180423	44	-24
20180411	20180423	40	-12

Make one run like that for each set*i*. See section 4.4 for doing them all in one run.

Check the baselines plot in: …/SAR_SM/MSBAS/*REGION*/seti/*span_Bmin_Bmax_Tmin_Tmax.jpg* (old tools) or …/SAR_SM/MSBAS/*REGION*/seti/*baselinePlot*_BpMax=*Bmax_BTMax=Tmax.txt.jpg* (tools after May 2022). Note that these recent tools also provide figure with the spatial localization within the orbital tube, that is: *imageSpatialLocalization*_BpMax=*Bmax_BTMax=Tmax.txt.jpg*.

Note: Before running the script, ensure that there are links to *csl* images in the set*i* directory and that *setParametersFile.txt* is present as well! The *setParametersFile.txt* is created using the ***initiateMSBAS*** script (see 4.1).

Note: If, for any reason, you need to add some pairs that do not fit the *MinBp, MaxBp, MinBt* and *MaxBt* baseline criteria (e.g. to ensure phase closure, or have as much usage of an image as Primary than as Secondary image, or want to have a continuous baseline plot), you can create a file named *table_MinBp_MaxBp_MinBt_MaxBt_AdditionalPairs.txt* (with the same values for *MinBp, MaxBp, MinBt* and *MaxBt* as those already used !) that contains the list of additional pairs you would like to process. These pairs must be stored in the file in the form of (without header):
DatePrimary DateSecondary Bp Bt.
Prepa_MSBAS.sh will paste these new pairs to *table_MinBp_MaxBp_MinBt_MaxBt.txt.* The file *table_MinBp_MaxBp_MinBt_MaxBt_AdditionalPairs.txt* will further be copied automatically in SAR_MASSPROCESS/*SAT*/*REGION_TRK*/*SM_ZOOM_ML* directory, where it might be used later.

Note: If one need to add/remove some pairs and plot again the baseline plot:

- add/remove lines in the *Table_Bmin_Bmax_Tmin_Tmax.txt* file
- Launch the script "***plotBaselines.sh***" with 5 parameters:

…/SAR_SM/MSBAS/*REGION*/set*i*	(dir containing links to data and baselines tables)
MinBp	(Min perpendicular baseline criteria for pair selection)
MaxBp	(Max perpendicular baseline criteria for pair selection)
MinBtemp	(Min temporal baseline criteria for pair selection)
MaxBtemp	(Max temporal baseline criteria for pair selection)

Results in plots may differ from one script to another because of X axis units. You may want to force the Y axis limits: see ***plotspan.sh***.

Note: If you run the script on a Linux computer without sound card, it may display an error message as below while attempting to read aloud the question whether or not you want "to search for a new Global Primary image":

```
ALSA lib confmisc.c:767:(parse_card) cannot find card '0'
ALSA lib conf.c:4732:(_snd_config_evaluate) function snd_func_card_driver returned error: No such file or directory
ALSA lib confmisc.c:392:(snd_func_concat) error evaluating strings
ALSA lib conf.c:4732:(_snd_config_evaluate) function snd_func_concat returned error: No such file or directory
ALSA lib confmisc.c:1246:(snd_func_refer) error evaluating name
ALSA lib conf.c:4732:(_snd_config_evaluate) function snd_func_refer returned error: No such file or directory
ALSA lib conf.c:5220:(snd_config_expand) Evaluate error: No such file or directory
ALSA lib pcm.c:2642:(snd_pcm_open_noupdate) Unknown PCM default
error: No such file or directory
ALSA lib confmisc.c:767:(parse_card) cannot find card '0'
ALSA lib conf.c:4732:(_snd_config_evaluate) function snd_func_card_driver returned error: No such file or directory
ALSA lib confmisc.c:392:(snd_func_concat) error evaluating strings
ALSA lib conf.c:4732:(_snd_config_evaluate) function snd_func_concat returned error: No such file or directory
ALSA lib confmisc.c:1246:(snd_func_refer) error evaluating name
ALSA lib conf.c:4732:(_snd_config_evaluate) function snd_func_refer returned error: No such file or directory
ALSA lib conf.c:5220:(snd_config_expand) Evaluate error: No such file or directory
ALSA lib pcm.c:2642:(snd_pcm_open_noupdate) Unknown PCM default
error: No such file or directory
```

These messages can be ignored… Or google them if you want to fix the issue.

Remark:

If a new processing is carried out using the same crop (name), Global Primary image, ML factor and Zoom but with some parameters being different, ensure that you change former directories naming accordingly

- …/SAR_SM/SAR_RESAMPLED/*SAT*/*REGION_TRACK*/SMCrop_SM_DATESM_*CROPNAME_*
  (Used during the mass coregistration)
- …/SAR_MASSPROCESS/*SAT*/*REGION_TRACK*/SMCrop_SM_DATESM_*CROPNAME_*
  (Used during the InSAR mass processing)

Warning: In AMSTerEngine version prior 20210217, the AMSTerEngine tool ***selectInterferometricPairs*** had a bug preventing it to take into account the pairs with Bp=0, although they would be the best ones one can imagine…

Note: if you prefer to have a list of pairs computed on other criteria rather than Max *Bp* and *Bt*, see scripts

- ***Extract_x_Shortest_Connections.sh***, which prepares a similar table table_0_0_MaxShortest_*x*.txt with all the pairs with the *x* shortest connections in the baseline plot. It computes this from the file allPairsListing.txt (created with ***Prepa_MSBAS.sh***) from the current directory.
  It will also create a table allPairsListing_Max*x*_ForPlot.txt (that is all the pairs for being plotted using ***baselinePlot***), and a baseline plot named baselinePlot_table_max_*x*.txt.png.

- ***DelaunayTable.sh***, which uses *DelaunayTable.py* to prepares a table named table_0_0_Delaunay*PARAM*_0.txt (where *PARAM* is Ratio*xxx*MaxBt*xxx*MaxBp*xxx* ; see below) with all the pairs in the baseline plot satisfying a Delaunay triangulation, and a possible filtering of the baselines larger than a maximum temporal and/or spatial baseline provided as parameters. The script computes this selection thanks to the file allPairsListing.txt (created with ***Prepa_MSBAS.sh***) from the current directory. It takes 3 optional parameters (in any order):
  - -Ratio=*float*: where *float* is a number (as float format) representing the ratio between X axis (time; in years) and Y axis (Bp in meters) in the Delaunay plot. For instance, a ratio of *x* will make the 1m baseline orthogonal to 1 day/*x*. Adjusting that ration allows avoiding elongated triangles;
  - -BpMax=*integer*: the Max Bp (as integer format) allowed for the segments of the triangles in y direction;
  - -BtMax=*integer*: the Max Bt (as integer format) allowed for the segments of the triangles in x direction.

  In a first stage, the python script *DelaunayTable.py* creates:
  - a file named table_0_0_DelaunayRatio*xxx*_0.txt
    that contains all the list of pairs for the mass processing (where *xxx* is defined by the parameter -Ratio.),
  - a file named Delaunay_Triangulation_xyRatio*xxx*_PairsforPlot.txt that contains all the list of pairs for plotting with ***baselinePlot*** with the -r option,
  - a figure (for a quick look) Delaunay_Triangulation_Plot_xyRatio*xxx*.png with the baseline plot without any baseline filtering.

  Then the bash script ***DelaunayTable.sh*** performs the optional baseline filtering and creates the following files:
  - a file named table_0_0_DelaunayRatio*xxx*MaxBt*xxx*MaxBp*xxx*_0.txt
    that contains all the list of pairs for the mass processing (where *xxx* is defined by the parameter -Ratio). It is the filtered version of table_0_0_DelaunayRatio*xxx*_0.txt.
  - a file that contains all the list of pairs for plotting with ***baselinePlot*** and named Delaunay_Triangulation_xyRatio*xxx*MaxBt*xxx*MaxBp*xxx*_PairsforPlot.txt. It is the filtered version of Delaunay_Triangulation_xyRatio*xxx*_PairsforPlot.txt.

- a figure named baselinePlot_Delaunay_Triangulation_xyRatio*xxx*MaxBt*xxx*MaxBp*xxx*_PairsforPlot.txt.png with the baseline plot with the baseline filtering. It is the filtered version of Delaunay_Triangulation_Plot_xyRatio*xxx*.png.
- The usual files are also created by ***baselinePlot -r***, e.g:
  imageSpatialLocalization_Delaunay_Triangulation_Ratio*xxx*MaxBt*xxx*MaxBp*xxx*_PairsforPlot.txt.png
  restrictedAcquisitionsRepartition.txt_Delaunay_Triangulation_*xxx*MaxBt*xxx*MaxBp*xxx*_PairsforPlot.txt
  restrictedPairSelection_Delaunay_Triangulation_xyRatio *xxx*MaxBt*xxx*MaxBp*xxx* _PairsforPlot.txt

Note: if one or both MaxBt or MaxBp is/are not used, no corresponding filtering is performed and that/these parameter(s) do(es) not appear in the file naming.

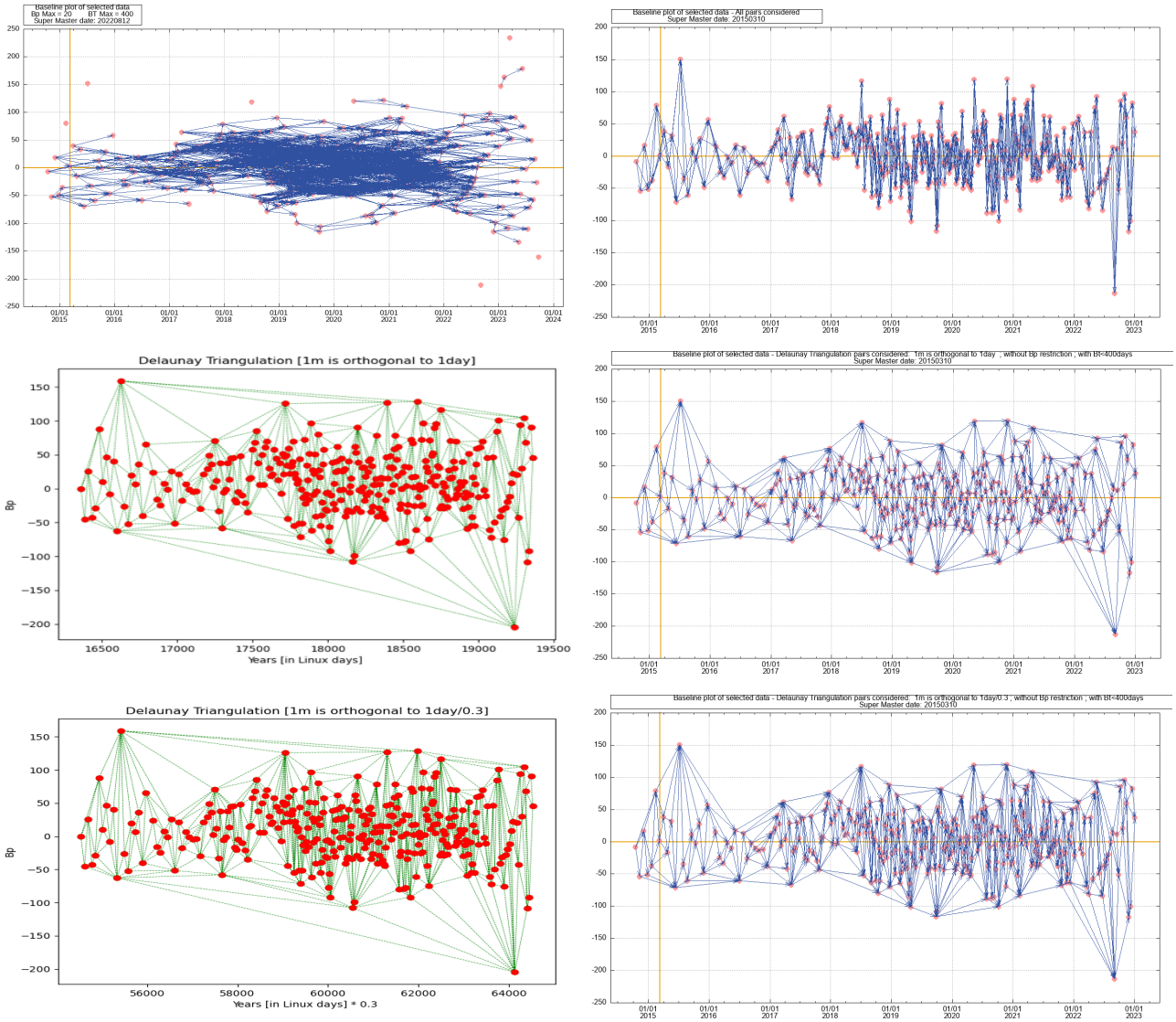


Figure 6: Example of baseline plot for Sentinel 1 images acquired in the Virunga (DRC) computed with several techniques:

*Top Left: All pairs with Bp <20m and Bt < 400 days (obtained with **prepa_MSBAS.sh**).*
*Top Right: Max 3 shortest connection (obtained with **Extract_x_Shortest_Connections.sh**).*

Middle Left: Delaunay Triangulation without Ratio (i.e. 1m is orthogonal to 1 day) and without maximum baseline filtering.
Middle right: Same as above with spatial baseline restricted to Max 400 days (both obtained with ***DelaunaTable.sh***).

Bottom Left: Delaunay Triangulation with Ratio 0.3 (i.e. 1m is orthogonal to 1/.3 day) and without maximum baseline filtering.
*Bottom right: Same as above with spatial baseline restricted to Max 400 days (both obtained with **DelaunaTable.sh**).*

(Note: The top left baseline plot covers a slightly longer time span)

4.4. Process more than one *Prepa_MSBAS.sh* for several set*i* at a time:

You can prepare a script to make several ***Prepa_MSBAS.sh***. See for instance
Lazy_prepa_msbas_all_sets_andPlotBaselines_Funu.sh
or
Lazy_prepa_msbas_all_sets_andPlotBaselines_VVP.sh

In these scripts, there is also a suggestion for plotting baseline plots with several sets on the same plot using ***plot_Multi_span.sh.*** See its syntax therein if you are interested by that option.

5. Mass Processing

Before performing a mass processing, it is advised to carefully check the *LaunchMTparam.txt* that will be used, then run some tests with ***SinglePair.sh***.

If some errors are encountered during a first run of mass processing, it is advised to carefully check the …/SAR_MASSPROCESS directory and clean pair directories and files in /Geocoded and /GeocodedRasters directories.

5.1. Coregistration of images pairs to the Global Primary image: S*uperMasterCoreg.sh*

This script is aiming at coregistering and resampling all the images form a give sat/mode in the geometry of a Global Primary image using AMSTerEngine.

For each set of sat/mode, launch "***SuperMasterCoreg.sh***" with following 1 or 2 parameters:

LaunchMTparam.txt (text file with all the parameters and options chosen for the run)
FORCE (optional; for S1 only, to force recomputing DEM for each image)

where

- *LaunchMTparam.txt* is a text file (and path) with all the parameters and options chosen for the run.
- *FORCE* is an option (for S1 use only) to force the script to compute DEM (and mask if required) in slant range for each S1 image.

- ***SuperMasterCoreg.sh*** will create a directory for each set where it will store the coregistration and resampling of each image to the Global Primary. Its naming is as follow:
  …/SAR_SM/SAR_RESAMPLED/*SAT*/*REGION_TRACK*/SMCrop_SM_*DATESM_CROPNAME_*
  where *DATESM* is the date of the Global Primary image.
  For Sentinel 1 images, orbits and geometries are good enough to bypass the coregistration step of each image to the Global Primary image. However, each image will need its DEM (and mask, if any) computed in slant range. This will be performed at the present step. If no *FORCE* parameter is used, it will only compute the slant range projection of the DEM (and mask) for the new images.

- Note that for specific use at ECGS, because computation is not performed on the same hard disk as the place where final RESAMPLED results are stored, the scripts will update each path in OUTPUTDATA/i12/TextFiles for each processed pair. This is done by calling the script ***RenamePathAfterMove.sh*** at the end of ***SuperMasterCoreg.sh*** script.

- Note that for specific use at ECGS, because computation is performed with a combination of Linux and Mac computers, which have their external disks mounted with different syntax, the scripts will update each path in OUTPUTDATA/i12/TextFiles for each processed pair, with its form as a state variable (e.g. change /Volumes/hp-1650-Data_Share1 with /$PATH_1650).

This is done by calling the script ***RenamePath_Volumes.sh*** at the end of ***SuperMasterCoreg.sh*** script. If you do not need this, just comment that call at the end of the ***SuperMasterCoreg.sh*** script.

- In order to keep track of the version of AMSTerEngine used to coregister the data, the script will store a small text file named *Coreg_w_AMSTerEngine_V.txt* in the directory where the pair was coregistered. It contains a single line indicating the date of the last version of the software present in the /$HOME/SAR/AMSTer/AMSTerEngine/_Sources_AE/Older. It is supposed to be the last one that was compiles and hence the one currently used.

5.2. Parallel run of Coregistration: __*SplitCoreg.sh*

You can speed up the coregistration processing by launching several ***SuperMasterCoreg.sh*** sessions at the same time using __***SplitCoreg.sh***. The script must be provided with:

- A list of images to coregister and to be split (must be in the form of image names if S1 (e.g. S1*x_Trk_DATE_AD*.csl) or *dates* if not S1, in 1 col)
- The *LaunchMTparameters.txt* file (with its path) to be used
- The number of parallel sessions to process the coregistration
- A 4th parameter (named *FORCES1DEM* in the script) may be provided for S1 images: if set to FORCE, it will recompute the DEM for each S1 wide swath image

Note that images already coregisterd will be ignored (unless FORCE is set for S1 of course).

The user must select manually the hard disks where he wants to process each split session. To assist the user, the script will list all the drives (that are hard coded in __***HardCodedLines.sh***) and display the corresponding available disk space. To select a hard disk, the user must select the number provided next to the hard disk proposed by the script. When all the sessions are attributed to a hard drive, the user must confirm that each split session can be run in a separate Terminal. The script then opens as many additional Terminal windows as the required split session. It keeps the main Terminal windows open and wait there the confirmation to clean the processing directories. Do not clean before all the split sessions are finished!

Note that __***SplitCoreg.sh*** only works if you are running it on your local computer, or if you are logged on another computer with a graphical interface such as *Microsoft Remote Desktop*. It can't work if you are logged on a server by ssh !!!
If you want to operate it from a ssh session, you must use ssh -X session instead of ssh and ensure that you have:

- on the client side, in the ~/.ssh/config file:

```
Host *
    ForwardAgent yes
    ForwardX11 yes
    ForwardX11Trusted yes
```

If the client is a Mac, you must install *XQuartz*, which contains *Xterm*, because *Terminal.app* is not X11 compatible.

- on the server side, in the /etc/ssh/sshd_config file:

```
X11Forwarding yes
X11DisplayOffset 10
X11UseLocalhost no
```

You should also have *xauth* installed on the server side (most probably existing by default).

Note for the Linux users: in order to automatically open one or more new Terminal windows from a Terminal and run command(s) in these newly open Terminal(s), *x-terminal-emulator* needs to know in which DISPLAY it must open the new Terminal windows. This is normally automatically determined by the script using the "*who*" command from *coreutils*. If the scripts can't figure out which is your DISPLAY, it will ask you to enter it manually. For this it will show you all the DISPLAY that are currently used on your computer. Enter the DISPLAY value in the form e.g. *:0.0* or *:10.0*…

5.3. InSAR mass processing of the image pairs: *SuperMaster_MassProc.sh*

This script is aiming at processing all compatible pairs form a given sat/mode in the geometry of a given Global Primary image as selected by ***Prepa_MSBAS.sh***.
All Secondary images must have been initially resampled at the Global Primary grid using ***SuperMasterCoreg.sh*** (see chapter 5.1).

For each set, launch "***SuperMaster_MassProc.sh***" with the following 2 (or 3) parameters:

PairsFile.txt	(path to file with the list of compatible pairs)
LaunchMTparam.txt	(text file with all the parameters and options chosen for the run)
-f or -list=*filename*	(optional: resp. to force processing list of pairs based on files in Geocoded/DefoInterpolx2Detrend or based on provided list)

where

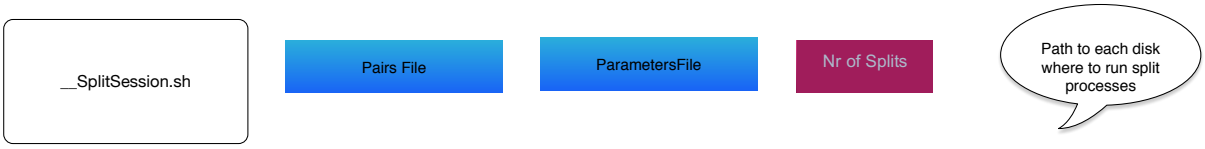
- *PairsFile.txt* is a text file (and path) with the list of all compatible pairs. This file was created using ***Prepa_MSBAS.sh*** and is stored in …/SAR_SM/MSBAS/*REGION*/set*i*. It is named *table_MinBp_MaxBp_MinBt_MaxBt.txt*. The script then compares that list with the list of directories of pairs already computed in …/SAR_MASSPROCESS/*SAT*/*REGION_TRACK*/SMCrop_SM_*DATESM_CROPNAME* and computes the missing pairs.
- *LaunchMTparam.txt* is a text file (and path) with all the parameters and options chosen for the run.
- The third parameter is optional. It can either be -f or -list=*filename*.
  - In the first case, it forces the script to build the list of existing pairs based on the files that are already present in …/SAR_MASSPROCESS/*SAT*/*REGION_TRACK*/SMCrop_SM_*DATESM_CROPNAME*/Geocoded/DefoInterpolx2Detrend
    instead of based on the list of existing pair directories in …/SAR_MASSPROCESS/*SAT*/*REGION_TRACK*/SMCrop_SM_*DATESM_CROPNAME*
  - If -list=*filename*, it forces to compute only pairs in provided in the *filename* list. Pairs in *filename* MUST be in the form of list of *PRM_SCD* dates (i.e. dates of Primary and Secondary images separated by an underscore).

Note that the processing is performed on a hard drive and/or a directory that differs from the final storage place. Directories of processed pairs are kept during the processing in …/PROCESS/MT/SAT/ (path defined in your *LaunchMTparam.txt*) and copied to the final storage place …/SAR_MASSPROCESS/*SAT*/*REGION_TRACK*/SMCrop_SM_*DATESM_CROPNAME* as soon as a pair is finished. The geocoded files of the processed pairs are moved directly in their final location as things progress, that is in …/SAR_MASSPROCESS/*SAT*/*REGION_TRACK*/SMCrop_SM_*DATESM_CROPNAME*/Geocoded
(Similarly, their raster versions are moved at the same place, though in /GeocodedRatsers dir).
All the pair directories are removed from the PROCESS directory only when everything is finished. As a consequence, if a problem occurs before the end of the process, the directories for each pair already processed remains available in …/PROCESS/MT/SAT/… for trouble shooting. In such a case, before relaunching ***SuperMaster_MassProc.sh***, these directories could be check and, if not OK, removed manually from SAR_MASSPROCESS. However, because geocoded products and their raster were moved as soon as they were computed, check that these geocoded and raster products are OK as well. If not, remove them before relaunching the script ***SuperMaster_MassProc.sh.***

In order to keep track of the version of AMSTerEngine used to mass process the data, the script will store a small text file named *Processing_Pair_w_AMSTerEngine_V.txt* in the directory where each pair was processed. It contains a single line indicating the date of the last version of the software present in the /$HOME/SAR/AMSTer/AMSTerEngine/_Sources_AE/Older. It is supposed to be the last one that was compiles and hence the one currently used.

Note that because the geocoded products are moved right after production (without waiting for the end of the whole ***SuperMaster_MassProc.sh*** process), they could be used in an MSBAS processing even if all the images have not been processed yet (see 6.1.b)).

5.4. Parallel runs of *SuperMaster_MassProc.sh*: *__SplitSession.sh*



The script ***__SplitSession.sh*** (Attention: double underscore at the beginning of the name…) allows splitting the processing of ***SuperMaster_MassProc.sh*** in a series of processing, each running in a separate Terminal window. For the sake of efficiency (speed), it is advised to run the split sessions on separate disks. The available disks must be hardcoded in the script (see ***__HardCodedLines.sh***). Check their availability and if sufficient space.

Note that ***__SplitSession.sh*** only works if you are running it on your local computer, or if you are logged on another computer with a graphical interface such as *Microsoft Remote Desktop*. It can't work if you are logged on a server by ssh !!!
If you want to operate it from a ssh session, you must use ssh -X session instead of ssh and ensure that you have:

- on the client side, in the ~/.ssh/config file:

```
Host *
    ForwardAgent yes
    ForwardX11 yes
    ForwardX11Trusted yes
```

If the client is a Mac, you must install *XQuartz*, which contains *Xterm*, because *Terminal.app* is not X11 compatible.

- on the server side, in the /etc/ssh/sshd_config file:

```
X11Forwarding yes
X11DisplayOffset 10
X11UseLocalhost no
```

You should also have *xauth* installed on the server side (most probably existing by default).

__SplitSession.sh needs the following parameters:

PairsFile.txt	(path to file with the list of compatible pairs)
LaunchMTparam.txt	(text file with all the parameters and options chosen for the run)
NumberOfSplit	(Number of set of data to be run independently from the set).

For the sake of efficiency, pairs using the same PRIMARY will not be split into different sessions. For that reason, each set (or batch) of pairs to be run in parallel may have a different number of pairs to process. That number is always at the maximum the total number of pairs divided by the number of sessions (*NumberOfSplit*). Hence often the last batch contains more pairs to process that the others. If that last set of pairs to process contains more than 10% of pairs than the maximum expected, the script offers to divide that last batch in two and hence add a parallel processing to the *NumberOfSplit* that were chosen. This is useful for having all the parallel processings ending at about the same time.

The user must then select manually the hard disks where he wants to process each split session. To assist the user, the script will list all the drives (that are hard coded in ***__HardCodedLines.sh***) and display the corresponding available disk space. To select a hard disk, the user must select the number provided next to the hard disk proposed by the script. When all the sessions are attributed to a hard drive, the user must confirm that each split session can be run in a separate Terminal. The script then opens as many additional Terminal windows as the required split

session. It keeps the main Terminal windows open and wait there the confirmation to clean the processing directories. Do not clean before all the split sessions are finished!

If ***SuperMaster_MassProc.sh*** processing crashes:

- Check which pair folder(s) do(es) not contain the complete set of expected final products in PROCESS directories (usually the pair that caused the process to stop; see Terminal) and check the results in SAR_MASSPROCESS/…. Kill what was wrong or incomplete and remove the copy of these wrong directories that would already be copied in the corresponding SAR_MASSPROCESS folder.
- If the cause of crash is external (e.g. power failure, connection to disk lost or disk full), relaunch the processing. No need to change the list of pairs to process because it will only compute the ones that are not yet in SAR_MASSPROCESS.
- If the cause of crash is due to a faulty image, try to identify and solve the cause of failure (e.g. corrupted raw image due to problem at download), then relaunch the processing. If the cause can't be fixed, edit the list of pairs and remove it from the list (if it is related to a corrupted image, you may need to remove all pairs involving it and/or move that image in …/SAR_CSL/S1/*REGION*/Quarantained), then relaunch the processing.

Note that this script needs the script ***LaunchTerminal.sh*** in order to be able to launch a new Terminal from a Terminal.

Note for the Linux users: in order to automatically open one or more new Terminal windows from a Terminal and run command(s) in these newly open Terminal(s), *x-terminal-emulator* needs to know in which DISPLAY it must open the new Terminal windows. This is normally automatically determined by the script using the "*who*" command from *coreutils*. If the scripts can't figure out which is your DISPLAY, it will ask you to enter it manually. For this it will show you all the DISPLAY that are currently used on your computer. Enter the DISPLAY value in the form e.g. *:0.0* or *:10.0*…

5.5. Combined Mass Processing (Mass Coregistration + InSAR processing): *SuperMaster.sh*

It is possible to launch both coregistration and mass processing with a single script: In each set, launch ***SuperMaster.sh*** with the following 2 parameters:

PairsFile.txt (path to file with the list of compatible pairs)
LaunchMTparam.txt (text file with all the parameters and options chosen for the run)

It will first compute the coregistration then the mass processing. As usual, these runs are incremental i.e. it will only compute the new pairs and ignore the pairs that are already processed.

5.6. Check results: *Verify_MassProcess_Results.sh*

Use ***Verify_MassProcess_Results.sh*** to check if all pairs are processed and each product is created correctly. Missing products will be listed in files. It will also offer to clean incoherent occurrence of files and pair directories. It might then be necessary to re-run a mass process to recompute the missing pairs/products. This is of course useless if the reason of the problem was not identified and sorted.

Verify_MassProcess_Results.sh will read the list of pairs supposed to be processed, then
- check if data are stored in each …/SAR_MASSPROCESS/…/Geocoded/*Modes*, (where *Modes* are respectively Ampli, Coh, Defo, DefoInterpol, DefoInterpolDetrend, DefoInterpolx2Detrend, InterfFilt, InterfResid, UnwrapPhase)
- check if each directory of processed pair is stored in corresponding SAR_MASSPROCESS directory.

It will output a series of text files with the status of each of these checks. It will then check if Geocoded products exist but not the corresponding Pair directory.

Launch ***Verify_MassProcess_Results.sh*** with the following 2 parameters:
PairsFile.txt (path to file with the list of compatible pairs)
MASSPROCESSINGPATH (path to where geocoded products are stored)

where
- *PairsFile.txt:* is a text file (and path) with the list of compatible pairs created using ***Prepa_MSBAS.sh***. This file is stored in …/SAR_SM/MSBAS/*REGION*/set*i* and named *table_MinBp_MaxBp_MinBt_MaxBt.txt*
- *MASSPROCESSINGPATH* is the path to where all the geocoded products are stored (eg. …/SAR_MASSPROCESS/SAT/TRACK/CROP_SM_DATE_ZOOM_ML/Geocoded).

It might be advised to run ***Verify_MassProcess_Results.sh*** twice in case some corrections are performed on the first run.

It is also a good practice to have a quick look at the GeocodedRasters/*Modes*, (where *Modes* are respectively Ampli, Coh, Defo, DefoInterpol, DefoInterpolDetrend, DefoInterpolx2Detrend, InterfFilt, InterfResid, UnwrapPhase) to ensure that each image that you intend to use is ok.

Note: There is similar script named ***Verify_Geocoded_versus_Rasters.sh*** which ensure that the results from the mass processing are successfully stored the same way as /Geocoded and /GeocodedRasters.

6. MSBAS Processing

(M)SBAS is a software developed by Sergey Samsonov (https://doi.org/10.4095/313749) that can be downloaded at http://insar.ca. It aims at performing multi-dimensional time series of ground deformation in EW and UD components [Samsonov and d'Oreye, 2012; 2017[5]]. If only one look angle is provided (or if several sets are acquired in similar looking geometry), the software will perform a conventional SBAS processing [Berardino et al., 2002[6]] and results will be given in radar Line Of Sight (LOS).

6.1. Preparation

When the mass processing is finished, one can prepare the msbas processing by selecting the pairs to invert and preparing the required files. The selection can be performed using several scripts depending on your needs. Note that the pair selection made for the inversion may differ from the pair selection for mass processing. Indeed, you can prepare the mass processing based on a set of criteria, but perform the msbas inversion based on other criteria. Of course, if the pair selection for the msbas inversion expects more pairs than what was prepared during the mass processing(s), it will not compute these missing pairs before performing the msbas inversion.

Basically, you can perform the pair selection for the msbas inversion using two types of scripts:

- ***build_header_msbas_criteria.sh*** (and its variant ***build_header_msbas_criteria_From_nvi_name_WithoutAcqTime.sh***), or
- ***build_header_msbas_Tables.sh***

The first type of scripts prepares the msbas inversion based on baselines criteria (max *Bp* and *Bt*). It will check among all the processed pairs which ones satisfy the baselines criteria. These criteria must be the same for all the modes to invert.

The second type of script prepares the msbas inversion based on provided tables (one for each mode to invert). These tables list the pairs that where selected based either on the *x* shortest connections (with ***Extract_x_Shortest_Connections.sh***), or on a Delaunay triangulation (with ***DelaunayTable.sh***), or based on baselines criteria (with ***Prepa_MSBAS.sh***). Note that unlike the version based on criteria only (***build_header_msbas_criteria.sh***), the version based on Tables can mix any type of criteria among the modes. For instance, you may perform and msbas inversion with 4 modes where pairs for the mode 1 are selected based on a Delaunay table, the mode 2 with given values of BP and Bt, mode 3 with other values of Bp and/or Bt, and mode 4 with a 3 shortest connection table.

These scripts are explained here after.

[5] Samsonov, S., and N. d'Oreye (2012), Multidimensional time-series analysis of ground deformation from multiple InSAR data sets applied to Virunga Volcanic Province, *Geophysical Journal …*, doi:10.1111/j.1365-246X.2012.05669.x.

Samsonov, S. V., and N. d'Oreye (2017), Multidimensional Small Baseline Subset (MSBAS) for two-dimensional deformation analysis: Case study Mexico City, *Canadian Journal of Remote Sensing*, *82*(6), 1–12, doi:10.1080/07038992.2017.1344926.

[6] Berardino, P., G. Fornaro, R. Lanari, and E. Sansosti (2002), A new algorithm for surface deformation monitoring based on small baseline differential SAR interferograms, *IEEE Transactions on Geoscience and Remote Sensing*, *40*(11), 2375–2383, doi:10.1109/TGRS.2002.803792.

6.1.a) After completion of SuperMaster_MassProc.sh: *build_header_msbas_criteria.sh*

When the mass processing is finished, one can prepare the msbas processing by launching ***build_header_msbas_criteria.sh***. This script must be launched in the directory where msbas will be run (e.g. …/MSBAS_RESULTS/MSBAS_SAT_REGION). It will create all the files needed by msbas. For this it will need to read information from files in …/SAR_MASSPROCESS/SAT/TRACK/CROP_SM_DATE_ZOOM_ML/*PRM_SCD* directories.

Note:

build_header_msbas_criteria.sh cannot operate if *PRM_SCD* directories (where it searches information in specific files) are not present (e.g. if these directories where removes or if ***SuperMaster_MassProc.sh*** is not finished yet). In that case, one can still get approximate information from products file names rather than from parameters in files in …/SAR_MASSPROCESS/…/*PRM_SCD* directories by using ***build_header_msbas_criteria_From_nvi_name_WithoutAcqTime.sh*** instead of ***build_header_msbas_criteria.sh*** (see below).

The script ***build_header_msbas_criteria.sh*** is aiming at:

- Copy only the deformation maps that fulfil criteria (Bperp and Btemp) in the given directories
- Creating the required *list_of_InSAR_files.txt* with path to the deformation files, Bperp, PrmDate, ScdDate
- Creating the *header.txt* required for the msbas processing
- Creating hdr files for results
- Check if files are OK or contains NaN
- It also output files with incidence angle, heading and acquisition time.

In the directory where the results of MSBAS will be stored,
(e.g. /MSBAS_RESULTS/MSBAS_SAT_REGION), launch "***build_header_msbas_criteria.sh***" with the following 4+ parameters:

MODE	(which product do you want to use).
NrOfModes	(nr of modes)
MaxBp	(Maximum perpendicular baseline to select pairs)
MaxBtemp	(Maximum temporal baseline to select pairs)
PathToEachSeti	(path to each mode= directory where the interferometric products are stored)

where

- *MODE:* type of products to use such as Defo, DefoInterpol, DefoInterpolDetrend or DefoInterpolx2Detrend; must be the name of the subdir where data are stored
- *NrOfModes* is the number of set*i* to be processed.
- *MaxBp* and *MaxBtemp* are obvious although they can be more restrictive than the amount of data mass processed. E.g if one had prepared pairs with *MaxBp* and *MaxBtemp* say resp. < 100 meters and 100 days, one can still compute msbas with criteria such as *MaxBp* and *MaxBtemp* say resp. < 50 meters and 50 days.
- *PathToEachSeti:* path to the geocoded deformation files produced for the different pairs, for each mode (eg:

…/SAR_MASSPROCESS/*SAT*/*TRK*/*Resampled_YYYYMMDD*_Crop_*REGION_coord_ZOOM_ML;* where *YYYYMMDD* is the date of the Global Primary, *REGION* is the region corresponding to the *coord*, and *ZOOM* and *ML* are the zoom factor and multilooking used for processing that mode.

Notes:

⇒ To spare time, if you run several msbas, the system remembers which pairs in …/SAR_MASSPROCESS/… do not satisfy your baselines criteria. This allows to avoid checking them again if you perform a new msbas with the same criteria.
For example, if a former ***MSBAS.sh*** was already performed with a max *20*m and *450*days Bp and Bt criteria respectively, a file named *Out_Of_Range_20m_450days.txt* will be created in …/MSBAS_RESULTS/LOCATION/MODE*i* directories. These files contain the list of pairs that were in …/SAR_MASSPROCESS/*SAT*/*TRK*/SM*Crop*_SM_*YYYYMMDD*_Crop_*REGION_coord_ZOOM_ML/MODEi* but that do not satisfy the baselines criteria. In order to avoid testing them again while selecting the pairs with the same baseline criteria, the script ***build_header_msbas_criteria.sh***, when run a second time, will ignore these pairs.

⇒ Similarly, if a previous run of ***MSBAS.sh*** was performed with an additional criterion for pair selection based on the minimum coherence computed over a given area (defined by a provided *kmz* file; see script ***restrict_msbas_to_Coh.sh*** and paragraph 6.1.d), files named *List_All_img_For_Coh_THRESHOLD_Region.kml.txt* and *List_Checked_img_For_Coh_THRESHOLD_Region.kml.txt* and *Checked_For_CohThreshold_To_Be_Ignored_At_Next_Rebuild_msbas_Header.txt* will also be present in the …/MSBAS_RESULTS/LOCATION/MODE*i* directories
(where *THRESHOLD* is the coherence threshold above which pairs are rejected for the msbas processing if the mean coherence computed for the *Region* is exceeded; the region footprint is provided by a kml file. See paragraph 6.1.d).
These files contain the list of pairs that were already tested against the coherence criteria (satisfying it or not). In order to avoid testing them again while selecting the pairs with the same coherence criteria, the script ***build_header_msbas_criteria.sh***, when run a second time, will ignore the unsatisfying pairs.

⇒ Because msbas version after October 2020 has new features, the header.txt file need other options. If you run such a version of msbas, ensure that it is named msbasv4.

⇒ ***build_header_msbas_criteria.sh*** will prepare corresponding info for msbas for each pair that satisfies the Bp and Bt criteria. However, one can force also the script to take some more pairs into account (i.e. that do not satisfy the baselines criteria but that are good enough to bring valuable information for the inversion). In that case, pairs must be listed in a specific file named *table_BpMin_BpMax_Btmin_Btmax_AdditionalPairs.txt* and located in the SAR_MASSPROCESS/SAT/TRK/REGION/ directory.
That file must have the same structure as *table_BpMin_BpMax_Btmin_Btmax.txt* though without header.
It is advised to keep track of that list by keeping a copy maybe in your …/SAR_SM/MSBAS/*REGION*/set*i* directory.
If only one *table_BpMin_BpMax_Btmin_Btmax_AdditionalPairs.txt* file exists, it will use it. If more than one *table_BpMin_BpMax_Btmin_Btmax_AdditionalPairs.txt* file exist, it will use the one with the same baselines criteria as *table_BpMin_BpMax_Btmin_Btmax.txt*.

⇒ Similarly, if one wants to process MSBAS with list of pairs from *table_BpMin_BpMax_Btmin_Btmax.txt* though WITHOUT some pairs, this must be done after the execution of ***build_header_msbas_criteria.sh***.
In that case, add in your /MSBAS/…/*MODEi* (where *MODEi* is e.g. DefoInterpolx2Detrend*i*)

a file *_EXCLUDE_PAIRS_ALTHOUGH_CRITERIA_OK.txt* that contains pairs list as DATE_DATE. Then run the script /zz_Utilities_MT/***Exclude_Pairs_From_Mode.txt.sh*** with only one parameter, that is the path to .../MSBAS/*REGION*/MODE*i* where file *_EXCLUDE_PAIRS_ALTHOUGH_CRITERIA_OK.txt* is located.

It is advised to keep track of that list by keeping a copy maybe in your …/SAR_SM/MSBAS/*REGION*/set*i* directory.

⇒ WARNING: when preparing an inversion using several set*i*, *the script* ***build_header_msbas_criteria.sh*** looks in each …/SAR_SM/MSBAS/*REGION*/set*i* for the table *table_0_BpMax_0_Btmax.txt* based on the *BpMax Btmax* parameters provided to ***build_header_msbas_criteria.sh***. However, it might happen (and it is highly probable e.g. when mixing satellites) that the most appropriate baseline plot for each /set*i* required a different set of *BpMax Btmax* parameters.
Because only one set of *BpMax Btmax* is provided to ***build_header_msbas_criteria.sh***, all the *table_0_BpMax_0_Btmax.txt* must have the same name in each /set*i*. Hence it is required to copy (or better, link) each table that would have a name different from what the script will be searching for (*table_0_BpMax_0_Btmax.txt)* with a fake name.

Example: suppose set1 has a table named *table_0_400_0_50.txt*, set2 has a table named *table_0_400_0_70.txt*, set3 has a table *table_0_50_0_50.txt*, then one need to run ***build_header_msbas_criteria.sh*** with only one set of values for Bt and Pb, e.g. 400m and 70days. If one does not copy tables from set1and set3 as *table_0_400_0_70.txt*, ***build_header_msbas_criteria.sh*** will not prepare msbas data for these sets.
A more elegant solution is to use the script ***build_header_msbas_Tables.sh*** (see section 6.1.c).

⇒ The script needs *checkOnlyNaN.py*

6.1.b) Without all the pairs processed: *build_header_msbas_criteria_From_nvi_name_WithoutAcqTime.sh*

This script does the same as the original ***build_header_msbas_criteria.sh,*** although without reading information in each *PRM_SCD* pair directory in SAR_MASSPROCESS. It will gather the required info from the name of the geocoded files instead. The only missing info will be the acquisition time (*hhmmss*) of each mode. This will have to be added manually in the *header.txt* before running the msbas. The list of incidence angle is also provided with less accuracy but for most of the case, it will be of no influence.
Remark: there might be a bug which delivers the incidence angle with a wrong sign when picked from file names!?! Need to be double checked.

build_header_msbas_criteria_From_nvi_name_WithoutAcqTime.sh is aiming at:
- Copy only the deformation maps that fulfils criteria (Bperp and Btemp) in the given directories
- Creating the required list_of_InSAR_files.txt with path to the deformation files, Bperp, PrmDate, ScdDate
- Creating the *header.txt* required for the msbas processing
- Creating *hdr* files for results
- Check if files are OK or contains of NaN
- It also output files with incidence angle and heading

In the directory where the results of MSBAS will be stored
(e.g. …/MSBAS_RESULTS/MSBAS_SAT_REGION), launch
"***build_header_msbas_criteria_From_nvi_name_WithoutAcqTime.sh***" with the following 4+ parameters:

MODE	(which product do you want to use).
NrOfModes	(nr of modes)
MaxBp	(Maximum perpendicular baseline to select pairs)
MaxBtemp	(Maximum temporal baseline to select pairs)
PathToEachSeti	(path to each mode= directory where the interferometric products are stored)

where
- *MODE:* type of products to use such as Defo, DefoInterpol, DefoInterpolDetrend or DefoInterpolx2Detrend; must be the name of the subdir where data are stored
- *NrOfModes* is the number of set*i* to be processed.
- *MaxBp* and *MaxBtemp* are obvious although they can be more restrictive than the amount of data mass processed. E.g if one had prepared pairs with *MaxBp* and *MaxBtemp* say resp. < 100 meters and 100 days, one can still compute msbas with criteria such as *MaxBp* and *MaxBtemp* say resp. < 50 meters and 50 days.
- *PathToEach*Set*i:* path to the geocoded deformation files produced for the different pairs, for each mode (eg: …/SAR_MASSPROCESS/*SAT/TRK/Resampled_YYYYMMDD*_Crop_*REGION_coord_ZOOM_ML;* where *YYYYMMDD* is the date of the Global Primary, *REGION* is the region corresponding to the *coord*, and *ZOOM* and *ML* are the zoom factor and multilooking used for processing that mode.

This script must also be launched in the directory where msbas will be run. It will create all the files needed by msbas.

Attention, this script requires gnu grep (ggrep) V 3.3 or above!

6.1.c) Based on tables: *build_header_msbas_Tables.sh*

Similarly, to ***build_header_msbas_criteria.sh,*** when the mass processing is finished, one can prepare the msbas processing by launching ***build_header_msbas_Tables.sh***.

This script must be launched in the directory where msbas will be run (e.g. …/MSBAS_RESULTS/MSBAS_SAT_REGION). It will create all the files needed by msbas. For this it will need to read information from files in …/SAR_MASSPROCESS/SAT/TRACK/CROP_SM_DATE_ZOOM_ML/*PRM_SCD* directories.

The script ***build_header_msbas_Tables.sh*** is aiming at:
- Copy only the deformation maps from the pairs provided in the given Tables,
- Creating the required *list_of_InSAR_files.txt* with path to the deformation files, Bperp, PrmDate, ScdDate
- Creating the *header.txt* required for the msbas processing
- Creating hdr files for results
- Check if files are OK or contains NaN
- It also output files with incidence angle, heading and acquisition time.

In the directory where the results of MSBAS will be stored,
(e.g. /MSBAS_RESULTS/MSBAS_SAT_REGION), launch "***build_header_msbas_Tables.sh***" with the following 4+ parameters:

MODE	(which product do you want to use).
NrOfModes	(nr of modes)
TableForEachMode	(path to tables with list of pairs and their baselines, one for each mode)
PathToEachSeti	(path to each mode= directory where the interferometric products are stored)
[--msbasvi]	(optional: version of the msbas for which one want to prepare the processing, e.g. msbasv4. If not provided, it will do it for the most recent version available on the computer)

where:
- *MODE:* type of products to use such as Defo, DefoInterpol, DefoInterpolDetrend or DefoInterpolx2Detrend; must be the name of the subdir where data are stored
- *NrOfModes* is the number of set*i* to be processed.
- *TableForEachMode* are path to 4 columns tables (one for each mode), with the list of pairs and their baselines. These can be either (types of tables can be mixed between the modes):
  - table_0_0_DelaunayRatio*xxx*MaxBt*xxx*MaxBp*xxx*_0.txt table (created with ***DelaunayTable.sh***), and/or
  - table_0_0_MaxShortest_*x*.txt table (created with ***Extract_x_Shortest_Connections.sh***), and/or
  - table_*MinBp_MaxBp_MinBt_MaxBt*.txt table (created with ***Prepa_MSBAS.sh***)

  Note: the tables must be entered in the same order as the path to each mode here after!
  Obviously, the pairs listed in the tables must have been processed beforehand. If the criteria in these tables are less restrictive than those used for the mass processing, the script will not notice and pairs will be missing in the inversion.

- *PathToEachSeti:* path to the geocoded deformation files produced for the different pairs, for each mode (eg: …/SAR_MASSPROCESS/*SAT/TRK/Resampled_YYYYMMDD*_Crop_*REGION_coord_ZOOM_ML;* where *YYYYMMDD* is the date of the Global Primary, *REGION* is the region corresponding to the *coord*, and *ZOOM* and *ML* are the zoom factor and multilooking used for processing that mode.
- [--msbasv*i*]: optionally, one can force the creation of the header file for a given version of msbas. If no --msbasv*i* (either --msbas or --msbasv*i* where *i* is a number above 2) is provided, the script searches for the most recent version of msbas available on your computer.

Notes:

⇒ To spare time, if you run several msbas, the system remembers which pairs were already checked. This allows to avoid checking them again if you perform a new msbas with the same criteria.

⇒ Msbas versions before and after October 2020 (msbasv4) require specific features in the header.txt file. Hence you can force the script to prepare the msbas inversion for a given version by adding a last optional parameter --msbasv*x*. If not such a parameter is provided (as it is usually the case), the script checks the last version of msbas available on your computer and prepares the files accordingly.
Also, the msbasv10 (provided in the AMSTerSoftware_Distribution/MSBAS_sources directory) also needs specific features, which are not implemented yet in AMSTer.

⇒ ***build_header_msbas_Tables.sh*** will prepare corresponding info for msbas for each pair provided in the Tables. However, one can force also the script to take some more pairs into account.
In that case, the additional pairs must simply be listed in a table file named like the original table, though with an extra string "_AdditionalPairs" (beware of typos) before the ".txt" extension, that is e.g. :

- table_0_0_DelaunayRatio*xxx*MaxBt*xxx*MaxBp*xxx*_0_AdditionalPairs.txt, and/or
- table_0_0_MaxShortest_*x*_AdditionalPairs.txt, and/or
- table_*MinBp_MaxBp_MinBt_MaxBt*_AdditionalPairs.txt table

These extra tables must be in the same directory as the main tables. They must also have the same structure (with or without the classical 2 lines of header), that is 4 columns as follow:

PRMDATE SCDDATE BP Bt

⇒ Similarly, one can force the rejection of some pairs although they would be listed in the provided tables. To remove (exclude) some pairs listed in the tables, they must simply be listed in a table file named like the original table, though with an extra string "_ExcludePairs" (beware of typos) before the ".txt" extension, that is e.g. :

- table_0_0_DelaunayRatio*xxx*MaxBt*xxx*MaxBp*xxx*_0_ExcludePairs.txt, and/or
- table_0_0_MaxShortest_*x*_ExcludePairs.txt, and/or
- table_*MinBp_MaxBp_MinBt_MaxBt*_ExcludePairs.txt table

These extra tables must be in the same directory as the main tables. They must also have the same structure (with or without the classical 2 lines of header, that is 4 columns as follow:

PRMDATE SCDDATE BP Bt

⇒ If a previous run of ***MSBAS.sh*** was performed with an additional criterion for pair selection based on the minimum coherence computed over a given area (defined by a provided *kmz* file; see script ***restrict_msbas_to_Coh.sh*** and paragraph 6.1.d), files named
List_All_img_For_Coh_THRESHOLD_Region.kml.txt and
List_Checked_img_For_Coh_THRESHOLD_Region.kml.txt and
Checked_For_CohThreshold_To_Be_Ignored_At_Next_Rebuild_msbas_Header.txt
will be present in the …/MSBAS_RESULTS/LOCATION/MODE*i* directories
(where *THRESHOLD* is the coherence threshold above which pairs are rejected for the msbas processing if the mean coherence computed for the *Region* is exceeded; the region footprint is provided by a kml file. See 6.1.d).

These files contain the list of pairs that were already tested against the coherence criteria (satisfying it or not). In order to avoid testing them again while selecting the pairs with the same coherence criteria, the script ***build_header_msbas_Tables.sh***, when run a second time, will ignore the unsatisfying pairs.

⇒ The script needs *checkOnlyNaN.py*

6.1.d) With additional restriction to coherence threshold: *restrict_msbas_to_Coh.sh*

In addition to restrict the pair selection to the spatial and temporal baseline criteria, further selection among the interferometric pairs can be performed based on the mean coherence computed over an area defined by a *kml* file. This for instance may be of interest for monitoring regions that are affected by strong seasonal variations causing decorrelation from the summer to winter transition while preserving coherence from summer to summer (e.g. free from snow cover).

This is performed by launching ***restrict_msbas_to_Coh.sh***, which aiming at removing from a prepared msbas data set (eg DefoInterpolx2Detrend2) all the images that does not satisfy a minimum average coherence on a selected zone provided by a *kml*. It must be run after ***build_header_msbas_Table.sh***, ***build_header_msbas_criteria.sh*** or ***build_header_msbas_criteria_From_nvi_name_WithoutAcqTime.sh***
and before running ***MSBAS.sh***, in the directory where msbas will be run and which contains all the *MODEi* directories and *Modei.txt*.

In the directory where the results of MSBAS will be stored
(e.g. …/MSBAS_RESULTS/MSBAS_SAT_REGION), launch
"***restrict_msbas_to_Coh.sh***" with the
following 4 parameters:

MODETOCLEANi	(which product do you want to clean for unsatisfying coherence).
COHTHRESHOLD	(coherence threshold)
PathToKml	(path to kml file that defines the zone where to test for mean coherence)
PathToCoh	(path to coherence files: SAR_MASSPROCESS/SAT/TRK/REGION_ML/Geocoded/Coh)

where
- *MODETOCLEANi:* products to clean for unsatisfying coherence, such as Defo*i*, DefoInterpol*i*, DefoInterpolDetrend*i* or DefoInterpolx2Detrend*i*, where *i* is the index of the data set. *MODETOCLEANi* must be the name of the subdir where data are stored.
- *COHTHRESHOLD* is the mean coherence value below which pairs will be discarded.
- *PathToKml* is the path to the kml file that defines the footprint of the area where to compute the mean coherence.
- *PathToCoh:* path to the directory that contains all the geocoded coherence files for the selected SAT/TRK, e.g. SAR_MASSPROCESS/SAT/TRK/REGION_ML/Geocoded/Coh.

Notes:

1. If a previous run of ***restrict_msbas_to_Coh.sh*** was performed, files named
   List_All_img_For_Coh_THRESHOLD_Region.kml.txt and
   List_Checked_img_For_Coh_THRESHOLD_Region.kml.txt and
   Checked_For_CohThreshold_To_Be_Ignored_At_Next_Rebuild_msbas_Header.txt
   are present in …/MSBAS_RESULTS/LOCATION/MODE*i* directories (where *THRESHOLD* is the coherence threshold above which pairs are rejected for the msbas processing if the mean coherence computed for the *Region* is exceeded). These files contain the list of pairs that were already tested against the coherence criteria (satisfying it or not). These files are used to avoid testing again the same pairs while using the same coherence criteria.

2. In some rare occurrence, some pairs are satisfying by accident the coherence threshold. Instead of reducing the threshold (which would result in rejecting most probably more pairs than only the one(s) that are problematic), one can force discarding them by listing these pairs in a file named _EXCLUDE_PAIRS_ALTHOUGH_CRITERIA_OK.txt and located in …/MSBAS_RESULTS/LOCATION/MODE*i*/
   That file only contains the list of pairs in the form *PRM_SCD* (i.e. the date of the Primary and Secondary images separated by an underscore).

When ***Exclude_Pairs_From_Mode.txt.sh*** is run, these pairs will be excluded from the msbas inversion.

3. If one wants to process MSBAS with a list of pairs from *table_BpMin_BpMax_Btmin_Btmax.txt* though WITHOUT some pairs, add in your /MSBAS/*…/MODEi* (where *MODEi* is e.g. DefoInterpolx2Detrend*i*) a file *_EXCLUDE_PAIRS_ALTHOUGH_CRITERIA_OK.txt* that contains pairs list as *DATE_DATE*.
When ***Exclude_Pairs_From_Mode.txt.sh*** is run, these pairs will be excluded from the msbas inversion.

6.1.e) Pair selection optimization based on coherence proxy

A procedure to optimize the pair selection during automated mass processing is explained in the following open access paper:
Smittarello, D., d'Oreye, N., Jaspard, M., Derauw, D., & Samsonov, S. (2022). **Pair selection optimization for InSAR time series processing**. Journal of Geophysical Research: Solid Earth, 127, e2021JB022825. https://doi.org/10.1029/2021JB022825

The proposed algorithm narrows the classical pair selection based on the shortest temporal and spatial baselines thanks to a coherence proxy and balances the use of each image as Primary and Secondary images thanks to graph theory methods. The method can help to drastically reduce the total number of computed differential InSAR interferograms, and hence, the computation time. The optimization also allows to reduce the influence of DEM errors and atmospheric phase screen, which increase the signal-to-noise ratio of the inverted displacement time series.

For this, you need:

- A kml file defining the region of interest where the coherence will be estimated to select the best pairs e.g. in: $PATH_1650/kml/*YourRegion*/*Your_ROI*.kml

- A table of pairs to optimize (created by ***build_header_msbas_criteria.sh***, that is a …/SAR_SM/MSBAS/*YourRegion*/set*i*/table_0_*BpMax*_0_*BtMax*.txt)

- For each mode to optimize, a Coherence table calibration to prepare as follow:
  In …/SAR_MASSPROCESS/*SAT*/*TRK*/SMNoCrop_*SM_Zoom_ML*/Geocoded/Coh
  Launch the script ***Baseline_Coh_Table.sh*** with the kml file as parameter:
  $PATH_1650/kml/*YourRegion*/*Your_ROI*.kml

  It will create a table of coherence named Baseline_Coh_Table_*Your_ROI.kml*.txt
  in …/SAR_MASSPROCESS/*SAT*/*TRK*/SMNoCrop_*SM_Zoom_ML*/Geocoded/Coh

The optimization can be done either based on the known coherence or on a coherence proxy.

OPTION 1: optimization based on the coherence

- In each directory of the mode …/SAR_SM/MSBAS/*YourRegion*/set*i*, optimise the pair selection by running:
  Run_optim_module.sh with the following 4 parameters:
  …/SAR_SM/MSBAS/*YourRegion*/set*i*/ table_0_*BpMax*_0_*BtMax*.txt
  …/SAR_MASSPROCESS/*SAT*/*TRK*/SMNoCrop_*SM_Zoom_ML*/Geocoded/Coh/Baseline_Coh_Table_*Your_ROI.kml*.txt
  NrLinks
  CohThreshold
  Where *NrLinks* (or degree of optimisation) is the maximum number of times each image is used as a Primary or Secondary and *CohThreshold* is a coherence threshold below which pairs will be discarded anyway.

  This will output 3 text files in …/SAR_SM/MSBAS/*YourRegion*/set*i*/
  table_0_*BpMax*_0_*BtMax*_listPR2rm4optim_*NrLinks*_th*CohThreshold*.txt
  table_0_*BpMax*_0_*BtMax*_listPR2rm4optim_*NrLinks*_th*CohThreshold*_optimized.txt
  table_0_*BpMax*_0_*BtMax*_orig.txt

OPTION 2: optimization based on a coherence proxy

- In each directory of the mode …/SAR_SM/MSBAS/*YourRegion*/set*i*, optimise the pair selection by running:

  Run_optim_module.sh with the following 10 parameters:

  …/SAR_SM/MSBAS/*YourRegion*/set*i*/ table_0_*BpMax*_0_*BtMax*.txt
  …/SAR_MASSPROCESS/*SAT*/*TRK*/SMNoCrop_*SM_Zoom_ML*/Geocoded/Coh/Baseline_Coh_Table_Y*our_ROI.kml*.txt
  NrLinks
  DOY_{low}
  α
  β
  γ
  Mxc
  Mnc
  CohThreshold

  Where *NrLinks* (or degree of optimisation) is the maximum number of times each image is used as a Primary or Secondary, DOY_{low} [in days] is a calibration factor to be fixed by the user that represents the epoch of the year when coherence is the lowest, $\alpha \in [1; 5]$ is a calibration factor to be fixed by the user that represents the width of this low coherence period, β and γ account for the temporal and spatial decorrelation rates in the studied area, *Mxc* and *Mnc* are the maximum and minimum expected values for the mean coherence on the ROI when acquisition dates vary, and *CohThreshold* is a coherence threshold below which pairs will be discarded anyway.

  This will output 3 text files in …/SAR_SM/MSBAS/*YourRegion*/set*i*/
  table_0_*BpMax*_0_*BtMax*_listPR2rm4optim_*NrLinks*_th*CohThreshold*.txt
  table_0_*BpMax*_0_*BtMax*_listPR2rm4optim_*NrLinks*_th*CohThreshold*_optimized.txt
  table_0_*BpMax*_0_*BtMax*_orig.txt

Restricting pairs to optimized selection:

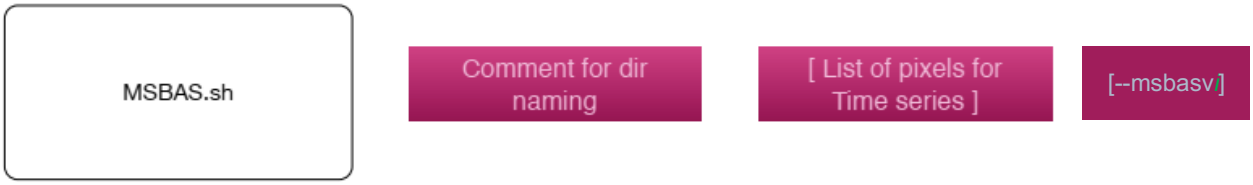
For each mode to optimize, run the script ***Remove_Pairs_From_BaselinePlotOptimisation.sh*** with the following 2 parameters:

DefoInterpolx2Detrend*i*
…/SAR_SM/MSBAS/*YourRegion*/set*i*/table_0_*BpMax*_0_*BtMax*_listPR2rm4optim_*NrLinks*_th*CohThreshold*.txt

It outputs a text file DefoInterpolx2Detrend*i*_Optimized_*TABLE*.txt_*RUNTIME*.txt, which is to be used instead of DefoInterpolx2Detrend*i*.txt for the msbas inversion (see chapter 6).

The script requires the following scripts: *creategraphfromtable_realcoh.py*, *creategraphfromtable.py*, *optimFunctions.py* and ***RemovePairs_fromtableforOptim_V2.sh***
(located in …/AMSTer/SCRIPTS_MT/optimtoolbox).

6.2. MSBAS processing: *MSBAS.sh*



- Modify (if needed), the header file (/MSBAS_RESULTS/MSBAS_SAT_REGION/*header.txt*) according to your parameters. See Samsonov and d'Oreye (2017) for detailed description.

Table 1. MSBAS parameters and supported processing options must have only one uniquely named parameter in header file. Detailed description is provided in text.

Parameter	Type	Value(s)	Description
FORMAT	int	0	4 bytes float, small endian
FORMAT	int	1	4 bytes float, big endian
FILE_SIZE	int(s)	*x*, *y*	For example, 1000, 1000
WINDOW _SIZE	int(s)	*xa*, *xb*, *ya*, *yb*	Default: 0, *x* – 1, 0, *y* – 1
C_FLAG	int	0	No calibration
C_FLAG	int(s)	1, *x*1, *y*1, Δx, Δy	1 reference region
C_FLAG	int(s)	2, *x*1, *y*1, *x*2, *y*2, Δx, Δy	2 reference regions
C_FLAG	int	10	Average set to zero
R_FLAG	int	0	No regularization
R_FLAG	int, float	1, λ	Zero order regularization
R_FLAG	int, float	2, λ	First order regularization
R_FLAG	int, float	3, λ	Second order regularization
T_FLAG	int	0	No topographic correction
T_FLAG	int	1	Topographic correction
I_FLAG	int	0	No interactive mode
I_FLAG	int	1	Interactive mode
I_FLAG	int, s	2, *par.txt*	Process par.txt file
I_FLAG	int	3	Save everything in text file
SET	s, int, int, s	*hhmmss*, θ, φ, *set.txt*	For example, 122435, –189, 34, dsc.txt
#			Comment

Note. Type *s* means *string*. *x* and *y* are width and length of interferograms. x_a, x_b, y_a, and y_b are first and last columns and rows of sub-region to be processed. x_i and y_i are column number and row number of reference region(s), up to 9 regions are supported. Δx and Δy are half-width/length of reference region(s). λ is regularization parameter. *hhmmss* is acquisition time for each set (must be 6 symbols). θ and φ are azimuth and incidence angles, measured in degrees. Lines can be commented with # symbol.

Note that msbas version from October 2020 (must be named msbasv4 or later) have two additional features in the *header.txt* (prepared automatically by scripts with default as in green below):

- V_FLAG=0 to compute displacement time series as before or V_FLAG=1 to compute velocity time series, in this case linear rate is acceleration.
- An additional parameter as first element in defining sets:
  SET=0, ACQTIME, AZIMUTH, INCID, MODEi.txt for InSAR range measurements, or
  SET=1, ACQTIME, AZIMUTH, INCID, MODEi.txt for InSAR azimuth measurements

- Launch "***MSBAS.sh***" with the following 2 optional parameters:
  [*COMMENT*] (optional, string used to add to dir names where results will be stored).
  [*ListOfPixels*] (optional, path and name of file containing a list of pixels)
  [--msbasv*i*] (optional, version of msbas as msbasv*i* where *i* is the version nr. If it does not exist or not provided, it takes the most recent version)

where

- *COMMENT:* string used to be added at end of directories containing the results. This can help to remember e.g. some processing configs.
- *ListOfPixels* is a text file (with path) containing the list of pixels (as COL RAW RADIUS) for which one wants to output time series, eg (*pixlist.txt*):
  125 130 5 (i.e. pixels of coord 125/130 and radius 5)
  160 210 2 … (i.e. pixels of coord 160/210 and radius 2)

- --msbasv*i* is the version of msbas to use (as msbasv*i* where *i* is the version nr, eg. msbasv4). If that version does not exist or if no msbasv*i* is provided, it takes the most recent version.

Note that all parameters are optional, but *COMMENT* is mandatory if one use the pix list output or if one force the version of msbas.

Coordinates of the pixels are provided as X and Y. Radius is the radius [in pixels] around the pixel where the value will be averaged and the stdev will be computed. Script ***msbas_plot_ts.sh*** made by Sergey Samsonov can be used to plot the time series with the error bars. This can be for instance launched for all the time series generated by MSBAS that are in the current directory using the script ***Plot_All_EW_UP_ts_inDir.sh*** as it is done for instance in ***Add_hdr_Files.sh*** in ***MSBAS.sh***.

This will compute the desired processing (hopefully).

If ***MSBAS.sh*** is launched with only one data set (or sets with similar geometries), the scripts will compute msbas deformation times series in LOS direction (that is a SBAS-like processing). Results (cumulative deformation maps at each acquisition date/time + velocity map etc…in ENVI format) and corresponding rasters will be stored in a subdirectory named zz_LOS_*COMMENT*, with an additional directory zz_LOS_TS_*COMMENT* containing the ascii time series of displacements for each pixel that was provided in the *pixlist.txt* (if any) as well as the plot of each time series.

If ***MSBAS.sh*** is launched with several sets with distinct geometries, the scripts will compute msbas deformation times series in EW and UD directions. Results (cumulative deformation maps at each acquisition date/time + velocity map etc…in ENVI format) and corresponding rasters will be stored in subdirectories zz_EW_*COMMENT* and zz_UP_*COMMENT*. If a *pixlist.txt* was provided, an additional zz_UD_EW_TS_*COMMENT* will contain the ascii time series of displacements for each pixel that was provided in the *pixlist.txt* as well as the plot of each time series.

Each product resulting from msbas processing (deformation, velocity maps…) are given in ENVI format and is accompanied with a raster quick look thanks to the use of the script ***Add_hdr_Files.sh*** and the template file *HDR.hdr* called by ***MSBAS.sh***. To spare time and space, another version (***Add_hdr_Files_Less_Ras.sh***) only creates the rasters for the velocity's maps. However, it also creates a script to generate these rasters if needed. It is during the execution of ***Add_hdr_Files.sh*** that time series are plotted (see ***Plot_All_LOS_ts_inDir.sh*** and ***Plot_All_EW_UP_ts_inDir.sh***)

The *header.txt* file that was used to compute the time series will be copied in zz_LOS_TS_*COMMENT* or zz_EW_*COMMENT* in order to keep track of parameters used.

Note: MSBAS will crash if the computer has not enough RAM memory. If you can't increase the RAM, either increase the Multilooking (but this requires performing at the best a new geoprojection of all the deformation pairs) or crop the area of interest (see msbas' header.txt).

6.3. Estimates optimal MSBAS regularization order and lambda factor: *test_lcurve.sh*

This script aims to facilitate the choice of an optimal regularization order (1, 2 or 3) and lambda factor (0→1). The choice of regularization order depends on the signal characteristics and study objectives. In case of the zero-order regularization, the solution is found by the least squares fitting of data and by minimizing the solution norm (i.e., deformation rates between consecutive acquisitions VE and VU). This type of regularization should be used when the mean of the deformation rates is expected to be close to zero (e.g., oscillating motion). See [Samsonov and d'Oreye 2017].

Launch "***test_lcurve.sh***" without parameters in the directory with the MSBAS results. It requires however the presence of a file *Steps_LCurve.txt* (in SCRIPTS_MT) that contains the list of points where one wants to assess the curve such as:

```
0.0001
0.0005
0.0010
0.0025
0.005
0.010
0.015
0.020
0.030
0.040
0.055
0.070
0.090
0.110
0.140
0.170
0.210
0.300
0.400
0.600
0.950
```

It will run that for each of the 3 orders of regularization. It will produce a plot (Figure 7) for each regularization order using ***plot_lcurve.sh***. The most appropriate order of regularization will depend on the deformation encountered (e.g. if the deformation is close to zero over time, there is a relatively steady increase or a gap in the data set). The most appropriate lambda is usually where there is a kink in the curve. An increase in the lambda factor will lead to a smoothing of the time series of deformation.

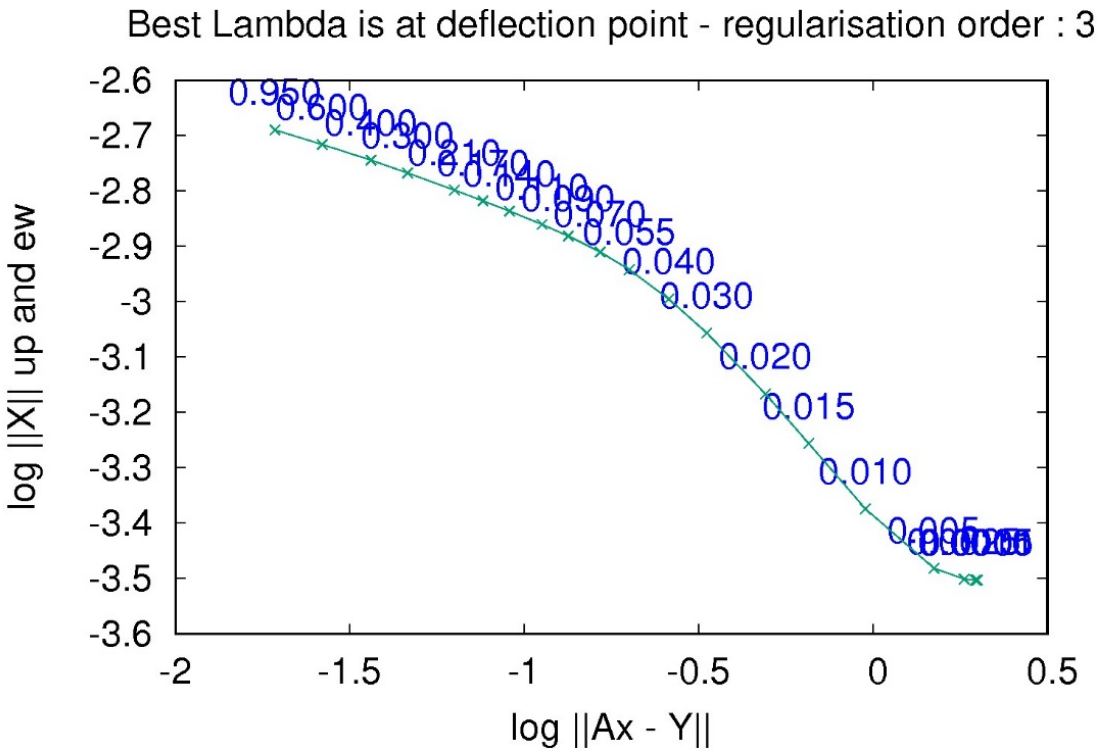


Figure 7: Example of l-curve to assist determination of the regularization. The most appropriate lambda factor is where there is a kink in the curve, that is about 0,02 in the present example.

Results might take a lot of disk space. Delete results that are not to be kept…

This script needs the python script *Norm.py*.

6.4. Plot times series of displacement:

Several scripts allow plotting time series results of msbas. If ***MSBAS.sh*** was run with a *pixlist.txt*, some time series are already provided in zz_LOS_TS_*COMMENT* or zz_UD_EW_TS_*COMMENT*.

However, if no *pixlist.txt* was provided, or if more points are required, or if double differences are wished, more scripts can be used here.

6.4.a) *Plot times series of displacement (or double difference) for a pixel in a given direction: PlotTS.sh*

This script aims to plot a time series of single component displacement for a given pixel. If two pixels are provided, it will compute the time series of both pixels as well as the double difference (pixel 1 – pixel 2).

Several options exist to add on the plot a linear trend (and display the annual rate) or tags with explanation of direction of movement of one pixel with respect to the other in Double Difference.

It is also possible to add several type of events on the plot as lines or rectangles:

- Eruptions (as semi-transparent vertical red rectangles),
- Seismic swarms (as semi-transparent vertical blue rectangles),
- Possible asymmetric acquisition geometry - that is when Asc and Desc acquisitions are not well balanced - (as semi-transparent vertical grey rectangles),
- Satellite coverage by ascending or descending tracks (as semi-transparent horizontal red or blue rectangles respectively),
- Change of polarization scheme (as semi-transparent horizontal grey rectangles),
- Earthquakes (as vertical blue dashed lines), or
- Any other events (as vertical dark blue dashed lines).

Time span of the plot may also be adjusted. If a -start=*YYYYMMDD* and/or -stop=*YYYYMMDD* parameter(s) is/are provided, it takes respectively the first and/or the last date as the limit. If no such a parameters are provided, it displays the time series for the whole duration.

It must be launched in the subdirectory where data are stored i.e. zz_LOS_*COMMENT*, zz_EW_*COMMENT* or zz_UP_*COMMENT*.

Launch ***PlotTS.sh*** with the following parameters:

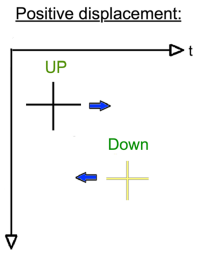
XPixel1	(Position of pixel 1 in X direction, i.e. nr of col in raster map).
YPixel1	(Position of pixel 1 in Y direction, i.e. nr of lines in raster map).
[XPixel2]	(Position of pixel 2 in X direction, i.e. nr of col in raster map).
[YPixel2]	(Position of pixel 2 in Y direction, i.e. nr of lines in raster map).
[-f]	(for optional linear fit to be added on the plot).
[-r]	(for optional linear annual rate displayed on the plot. Needs option -f).
[-t]	(for optional tag with direction of displacement and pixel's location).
[-d]	(for optional cleaning of both TS values text files after completion).
[-D]	(for optional cleaning of both Time Series values text files and double difference text file after completion).

[-g] (for optional cleaning of gnuplot scripts after completion).
[-png] (for optional creation of png versions of some plot).
[-events= PATH] (to add optional events on figures as lines or rectangles, where PATH is e.g. ../EVENTS_TABLES/*NAME*/ and *NAME* is the name of area which is also in events table names – see below).
[-start=YYYYMMDD -stop=YYYYMMDD] (to optionally restricts the plot to corresponding time span).

Notes:

1. ***PlotTS.sh*** requires gnu plot templates *plotTS_template.gnu* or *plotTS_template_fit.gnu* (in SCRIPTS_MT/TemplatesForPlots/)

2. Option *-t* add tags to locate the pixel(s) on the velocity map wrapped on amplitude. It allows add sketches that explains the direction of displacement (see Figure 8 below, although illustrated on figure combining EW and UD on the same plot).
   Attention, this option requires the usage of several other tools and files:
   - ***TS_AddLegend_LOS.sh, TimeSeriesInfo_HP.sh, AmpDefo_map.sh***
   - Python + Numpy + scripts: *CreateColorFrame.py*, *Mask_Builder.py*
   - gnu function *stat* (*gstat*) e.g. from *coreutils*
   - figures in $PATH_SCRIPTS/SCRIPTS_MT/TSCombiFiles/ with the explanations:
     - *TS_Displ_LOS_Neg.png*
     - *TS_Displ_LOS_Pos.png*
     - *TS_Displ_Neg_EW.png*
     - *TS_Displ_Neg_UD.png*
     - *TS_Displ_Neg.png*
     - *TS_Displ_Pos_EW.png*
     - *TS_Displ_Pos_UD.png*
     - *TS_Displ_Pos.png*



   - a figure named *satview.jpg* (see point 3 below and *Web_tool_Vxx.pdf*)
   - a parameters file named *TS_parameters.txt* that contains e.g. the following parameters (to be adjusted by the user):

```
# Data related to HP server path

defo-domuyo                     # WebPage

# Fiji_Amp_Defo_Coh script and ImageCreator.sh
# Value used to build the legend of the deformation maps
1.3                     # IJAmpMin  (Minimum value for brightness to build amplitude image using ImageJ)
2.7                     # IJAmpMax  (Maximum value for brightness to build amplitude image using ImageJ)
40                      # MarkUp    (Legend vertical bar position Up)
90                      # MarkDown (Legend vertical bar position Down)
125                     # LegValPosH            (Vertical Position of the value in the legend)
125                     # LegUnitPosH           (Vertical Position of the unity in the legend)
35                      # LegTxtPosH            (Vertical Position of the text info in the legend)
8                       # LegAdjZero            (Fine adjustment of the horizontal positionnement of Zero)
60                      # LegAdjMin(Fine adjustment of the horizontal positionnement of Min Val)
10                      # LegAdjMax             (Fine adjustment of the horizontal positionnement of Max Val)
70                      # LegAdjLOS             (Fine adjustment of the horizontal positionnement of Max Val for LOS maps)
200                     # LegAdjUnit            (Fine adjustment of the horizontal positionnement of Unity related to the left)

# CreateColorFrame script + Fiji_Amp_Defo_Coh script
100                     # Margin
800                     # LegendWidth
0.6                     # ColorBackgrdLegnd (0 = white --> 1 = grey)
35                      # LegendTxtSize (Size of the text in the legend)
140                     # LegendHeight (Height of the legend in pixels margin include)
40                      # FrameTop (distance between top of colour frame and top of the legend)
60                      # FrameBott (distance between bottom of colour frame and top of the legend)

# ImageCreator.sh and TimeSeriesInfo.sh
0               # Crop_X (Top left X coordinate of the cropped zone)
0               # Crop_Y (Top left Y coordinate of the cropped zone)
3361            # Crop_L (Horizontal size of the cropped zone)
2800            # Crop_H (Vertical size of the cropped zone)

# TimeSeriesInfo.sh
100                     # CrossTresh (Distance between 2 points (Vert or Horiz) which determine the size of the cross)
15                      # CrossBig Size of the cross in defo map as thumbnail if spacement between 2 pts are > treshold)
15                      # CrossSmall Size of the cross in defo map as thumbnail if spacement between 2 pts are < treshold)
```

NOTES:
- Parameters
  - Crop_X (Top left X coordinate of the cropped zone),
  - Crop_Y (Top left Y coordinate of the cropped zone),
  - Crop_L (Horizontal size of the cropped zone)
  - Crop_H (Vertical size of the cropped zone)

  must be adapted to the size of the deformation maps
- if ***PlotTS.sh*** crashes while making the tags using the script, check the script ***AmpDefo_map.sh***, see in that script the command line launching *ImageJ* and check the *--headless* option.

3. When plotting double difference time series in LOS with ***PlotTS.sh*** and option *-t*, the scripts ***TS_AddLegend_LOS.sh*** and ***TimeSeriesInfo_HP.sh*** add a small Google Earth map to locate the pixels with the same crosses as on the small images with the LOS velocity. This feature (set up by Maxime Jaspard) requires however that a referenced Google Earth image is prepared according to the area of interest (named *satview.jpg* in *MSBAS_DIR*/_CombiFiles).
   The procedure to prepare this file is explained in *Web_tool_Vxx.pdf*.
   If that *satview.jpg* image does not exist, it will proceed as usual, with only the velocity map to locate the pixels.
   Note: when plotting time series of LOS only and using option -t, the direction of the orbit (Asc or Desc) **MUST be in the name** of the directory where the msbas results are (i.e. in the *COMMENT* in the directory name zz_LOS_TS_*COMMENT*) to properly tag the direction of displacement.

4. Events can be added on the plot if a path to a directory where events tables are stored is provided. See example in Figure 9 below, although illustrated on a figure combining EW and UD on the same plot.

   Events tables must be named as one or more of the following where *NAME* is the name of the area of interest (must be the name of the directory that contains all the event files, e.g. ../EVENTS_TABLES/*NAME*/):
   - *Eruptions_NAME.txt* (displayed as vertical red rectangles)
   - *EQ_Swarms_NAME.txt* (displayed as vertical blue rectangles)
   - *Asymetric_Acquisition_NAME.txt* (displayed as vertical light grey rectangles)
   - *EQ_NAME.txt* (displayed as vertical blue dashed lines)
   - *Other_events_NAME.txt* (displayed as vertical grey dashed lines)

   Note that these tables MUST be in the form of "*Name StartDate StopDate*" where dates are *YYYYMMDD* and fields are separated by tab.

   Additional marks can be plotted horizontally based on additional tables i.e.:
   - *Sat_Cover_NAME.txt* (displayed as horizontal red and blue rectangles for Asc and Desc respectively)
   - *Polarisation_Change_NAME.txt* (displayed as horizontal grey rectangles)

   Note that these tables MUST be in the form of "*Name Date*" where date is *YYYYMMDD* and fields are separated by tab.
   Note that for the specific case of *Sat_Cover_NAME.txt* the *Name* in the table MUST be *WHATEVER*_Asc or *WHATEVER*_Desc. Asc and Desc string is used by the script to be detected as sat geometry and colour code the rectangles accordingly. If these strings are not detected in second position delimited by underscores, the script will consider that it is not a satellite geometry and the rectangles will be grey, without vertical offsets in the plot.

5. if paths are properly hard coded, it can be used within a python script from QGIS (see *00_RasterPixelCoord.py*) in order to click on the velocity map on a pixel and get its coordinates and plot the time series at the same time. The hard coded paths in ***PlotTS.sh*** are mandatory because python console in QGIS can't read state variables in *.bashrc*. Attention, in python script, the path to the MSBAS_RESULTS directory must be hardcoded as well.

   Procedure to use the *00_RasterPixelCoord.py:*
   - Open QGIS and load a deformation or velocity map. Adjust colour if wanted
   - In QGIS, add a background map and adjust transparency (if wanted)
   - In QGIS: Plugin > Python Console
   - In the column to the right-hand side, click on "open script" icon and load *00_RasterPixelCoord.py*
   - Click on green arrow to start the script
   - Click on the pixel on the map for which you want to see the time series. At the console, you will be prompted the coordinates of the pixel. Its time series will be plotted as usual in the MSBAS_RESULTS directory
   - Note: if you navigate in the map, it will interrupt the python script. You only have to click on the green arrow to start it again.

   Special thanks to Antoine Dille for having provided with the backbone of that python script.

6. With Linux, if an error occurs while generating the eps file using *convert*, change the permissions in /etc/ImageMagick/policy.xml and set:

   </policy domain="coder" rights="none" pattern="PS" />
   as
   </policy domain="coder" rights="readlwrite" pattern="PS" />
   (i.e. with pipe between read and write),
   and
   </policy domain="coder" rights="none" pattern="EPS" />
   as
   </policy domain="coder" rights="readlwrite" pattern="EPS" />

   Also, change as follow:
   </policy domain="resource" name="height" value="16KP" />
   as
   </policy domain="resource" name=" height" value="32KP" />

   and
   </policy domain="resource" name="width" value="16KP" />
   as
   </policy domain="resource" name=" width" value="32KP" />

   and
   </policy domain="resource" name="disk" value="1GiB" />
   as
   </policy domain="resource" name="disk" value="8GiB" />

6.4.b) *Plot times series of displacement for a pixel (or double difference) in all directions in the same plot: PlotTS_all_comp.sh*

This script is only useful for 2D displacements. ***PlotTS_all_comp.sh*** aims to plot in the same figure both EW and UD time series of displacements for a given pixel. If two pixels are provided, it will compute the time series of both pixels as well as the double difference (pixel 1 – pixel 2).

Unlike the former script for single component, ***PlotTS_all_comp.sh*** must be launched in the directory that contains the subdirectories zz_EW_*COMMENT* and zz_UP_*COMMENT*.

Note: it also works for 3D displacements if MSBAS allowing that option is used (i.e. when NS displacement can be deduced from additional information coming from topography [Samsonov et al. 2020]).

Launch ***PlotTS_all_comp.sh*** with the following parameters:

REMARKDIR	(remark that was used at the time of running ***MSBAS.sh*** and that is used to name the UD and EW dirs e.g. zz_UD_*REMARKDIR*).
XPixel1	(Position of pixel 1 in X direction, i.e. nr of col in raster map).
YPixel1	(Position of pixel 1 in Y direction, i.e. nr of lines in raster map).
[XPixel2]	(Position of pixel 2 in X direction, i.e. nr of col in raster map).
[YPixel2]	(Position of pixel 2 in Y direction, i.e. nr of lines in raster map).
[-f]	(for optional linear fit to be added on the plot).
[-r]	(for optional linear annual rate displayed on the plot. Needs option -f).
[-t]	(for optional tag with direction of displacement and pixel's location).
[-d]	(for optional cleaning of both TS values text files after completion).
[-D]	(for optional cleaning of both TS values text files and double difference text file after completion).
[-g]	(for optional cleaning of gnuplot scripts after completion).
[-png]	(for optional creation of png versions of some plot).
[-events= PATH]	(to add optional events on figures as lines or rectangles, where PATH is e.g. ../EVENTS_TABLES/*NAME/* and *NAME* is the name of area which is also in events table names – see below).
[-start=YYYYMMDD -stop=YYYYMMDD]	(to optionally restricts the plot to corresponding time span).

See ***PlotTS.sh*** for explanations about the options above. ***PlotTS_all_comp.sh*** has however an additional option (rather for check, debug or development) that is to add boxes and error-bars-like features that are proportional to the coherence:

[-coh=option]	(to optionally add error bars-like info about coherence) where option is either *avgavg*, *avgmin*, *avgminmax* or *avgavgminmax* to plot data +- Mean and/or Min and/or Max coherence. This is only plotted on single pixel eps plot (not on double difference plot). In a nutshell, the coherence at the location of the pixel is extracted from all the interferometric pairs used to compute the displacement during the msbas inversion. The script then computes the mean coherence (avg), the min and max coherence, and the number of pair files used. Note that the values of the coherence are arbitrary divided by 100 for the sake of scaling in plot. Depending on the option chosen, one can plot boxes above (and below) the data, and/or error bar with min coh value below and max coh above data. Note that because these boxes and bars are

coherence-related information and not error bars, the **larger is the best!**.

Note that because the script must look for all the pairs containing each image and then compute the statistics, this may be very slow at first run. However, it is an incremental process: it only computes that statistic for dates that were not computed yet. As a consequence, next runs will be much faster. Note however that pre-existing coh values will not be estimated again, even if new pairs are added. If this is the case, the line in the data files corresponding to that date in *timeLine_EW_COH_LIN_PIX_REMARK.txt* and *coh_LIN_PIX_REMARK.txt* (in *MSBAS_DIR*/zzz_coh_per_pixels), where *LIN*, *PIX* and *REMARK* are respectively the coordinates of the pixel and the string used to name the directories where the MSBAS results are stored, must be removed.

Color-coded symbols are added along the y=0 axis corresponding to the number of pairs used to compute the coherence statistics. See Figure 10 for illustration.

This script uses ***TS_AddLegend_EW_UD.sh*** to add the legends as for LOS with ***PlotTS.sh***.

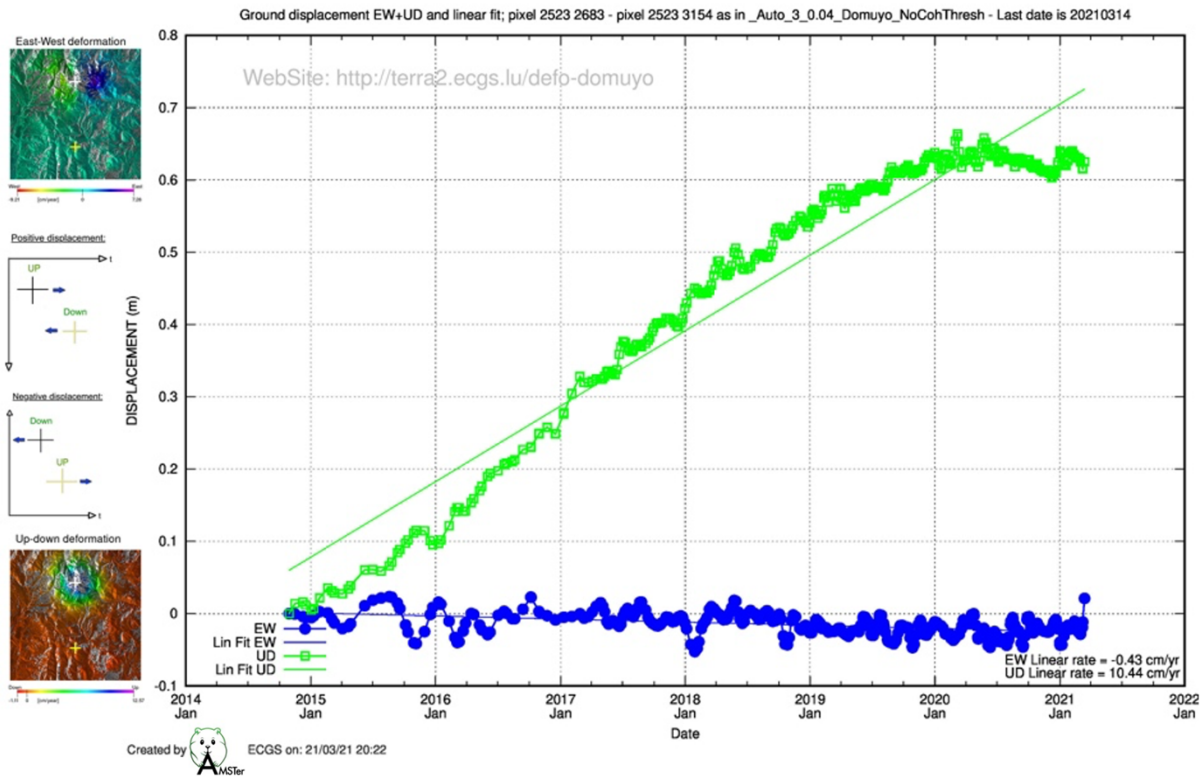


Figure 8: Example of plot (all comp) using options -f -r -t

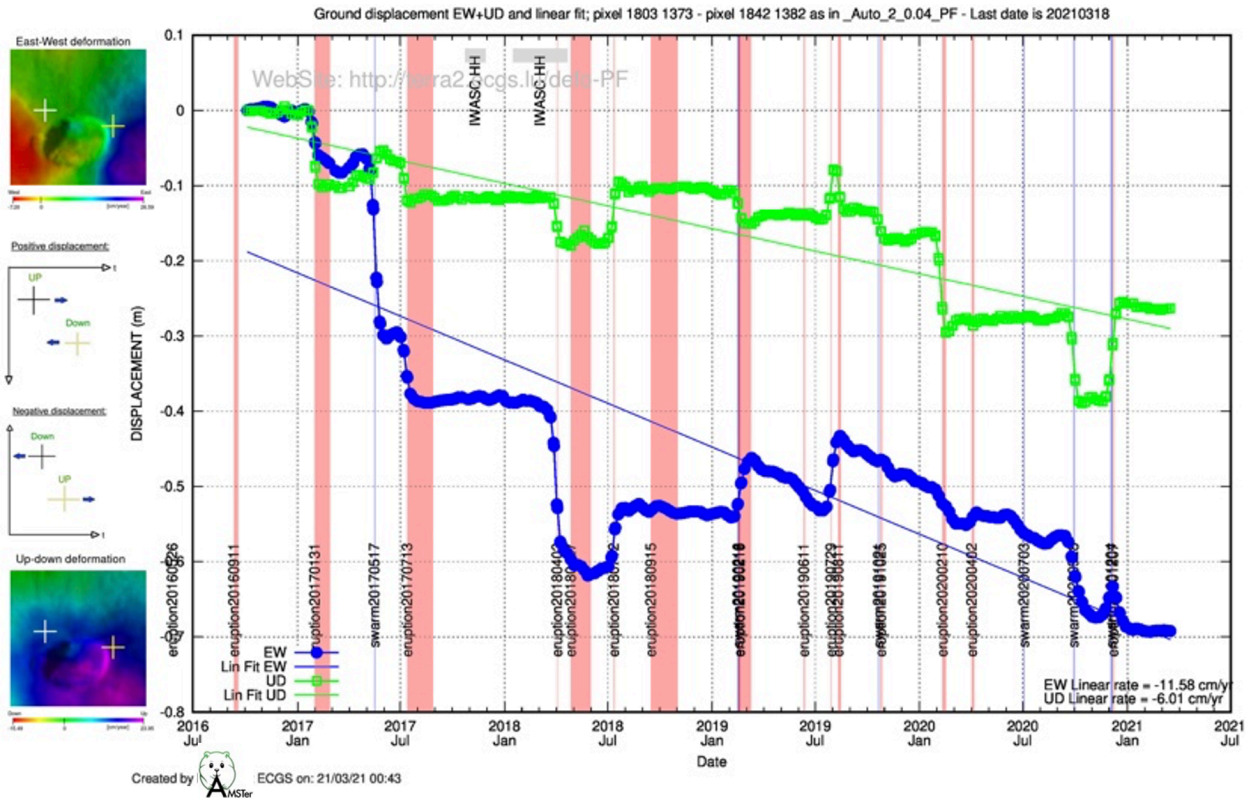


Figure 9: Example of plot (all comp) with option -f -r -t -events=...

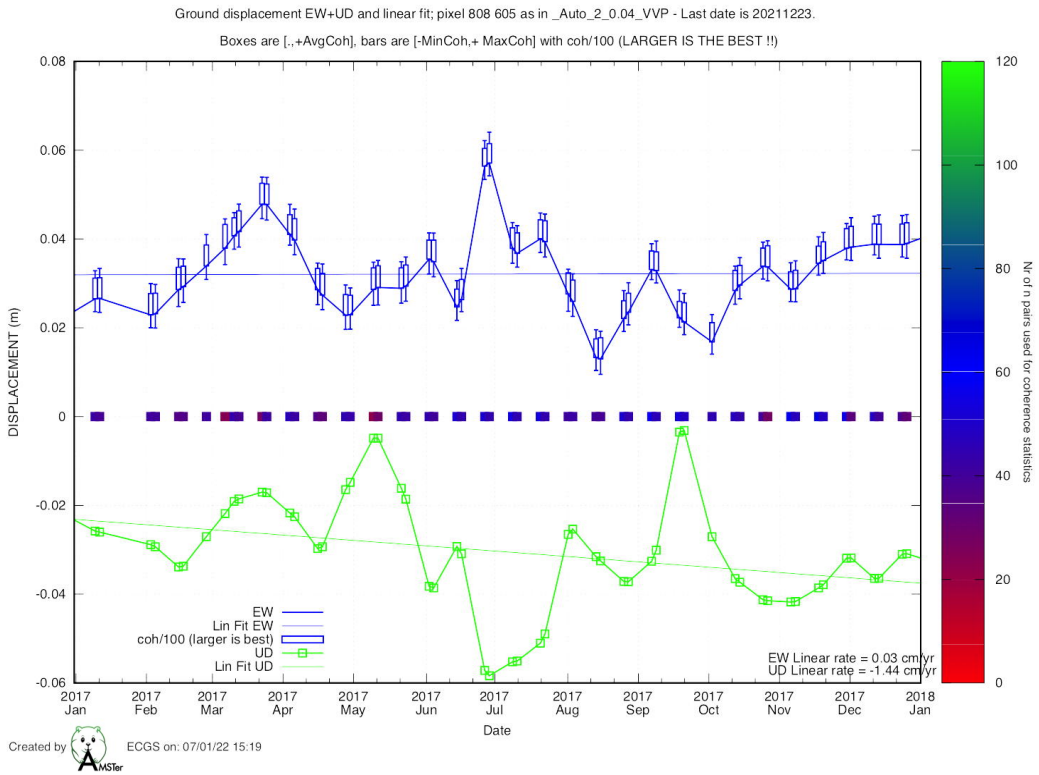


Figure 10: Example of plot (all comp) with option -coh=avgminmax. Bottom of the box is displacement, top of the box is the average coherence (/100), lower and upper bars are min and max coherence (/100) respectively. Color coded symbols is the number of pairs use for the coherence statistics.

6.5. Create customized baselines plot:

6.5.a) *Create baselines plot with the images effectively used for msbas: PlotBaselineGeocMSBAS.sh and PlotBaselineGeocMSBASmodeTXT.sh*

When preparing the msbas at step ***Prepa_MSBAS.sh***, it creates a baseline plot. However, it might be required to add pairs from the default selection of pairs based on solely the Bt and Bp (e.g. when too long gaps are observed in the time series) or remove pairs (e.g. if some pairs can't be computed). This is done by adding/removing files in the directories prepared by ***build_header_msbas_criteria.sh*** (i.e. …/MSBAS_RESULTS/MSBAS_SAT_REGION/DefoMode*i*) and adding/removing the corresponding lines in DefoMode*i.txt* (where *DefoMode* is the first option used in ***build_header_msbas_criteria.sh***).

When pairs are added or removed, one can use ***PlotBaselineGeocMSBAS.sh*** to create new baselines plots based on the list of pairs that were effectively used by msbas i.e. that are in each …/MSBAS_RESULTS/MSBAS_SAT_REGION/DefoMode*i*.

The script will take the characteristics of the images from the image names that are in *DefoMode1*.

Launch ***PlotBaselineGeocMSBAS.sh*** with the following 3 parameters:

*PathTo*Set*i*	(path to dir where pair lists are i.e. …/SAR_SM/SAT/TRK/REGION/set*i*) where *i* is the first mode used in the msbas processing).
DefoMode	(the mode used in msbas as selected at ***build_header_msbas_criteria.sh***).
ith_mode	(*i* th mode of msbas processing for which one wants to create a plot).

The script must be launched from …/MSBAS_RESULTS/MSBAS_SAT_REGION

Probably better, a slightly different version of that script, which is named ***PlotBaselineGeocMSBASmodeTXT.sh***, uses the list of interferograms to draw the baseline plot, i.e. …/MSBAS_RESULTS/MSBAS_SAT_REGION/DefoMode*i*.txt

Launch ***PlotBaselineGeocMSBASmodeTXT.sh*** with the following 2 parameters:

*PathTo*Set*i*	(path to dir where pair lists are i.e. …/SAR_SM/SAT/TRK/REGION/set*i*) where *i* is the first mode used in the msbas processing).
*PathTo*Txt	(path to DefoMode*i*.txt).

6.5.b) *Create baselines plot with the images contained in GeocodedRasters: PlotBaselineGeocRaster.sh*

PlotBaselineGeocRaster.sh allows to create a baselines plot based on the list of rasters contained in …/SAR_MASSPROCESS in subdirectories /GeocodedRasters.

The script will take the characteristics of the images from the image names that are in /GeocodedRasters/Coh.

The script MUST be launched from /GeocodedRasters and one must copy in that directory the *initBaselines.txt* file. Launch ***PlotBaselineGeocRaster.sh*** without parameters.

6.5.c) *Create a combined baselines plot from several data sets: plot_Multi_span.sh*

plot_Multi_span.sh makes a common baselines plot for multiple data sets. The script must be launched from …/MSBAS_RESULTS/MSBAS_SAT_REGION/ and data are expected to be in subdirectories …/MSBAS_RESULTS/MSBAS_SAT_REGION/DefoMode*i*. It supposes that a first plot was already computed with ***Prepa_MSBAS.sh*** for each mode in order to generate the span(1).txt files with the same MaxBp and MaxBt.

Launch ***plot_Multi_span.sh*** with the following 6 parameters:

SETLIST	(file with the list of sets used in the order of DefoMode*i* dir).
MinBp	(minimum Bp)
MaxBp	(maximum Bp)
MinBt	(minimum Bt)
MaxBt	(maximum Bt)
ColorTable	(file with colour table in hex)

For table color, see for instance SCRIPTS_MT/TemplatesForPlots/*ColorTable_AD.txt*, or SCRIPTS_MT/TemplatesForPlots/*ColorTable_ADDA.txt* (where A and D stands for Asc and Desc, and corresponds to the order of DefoMode*i* sets) in.

It is advised to try keeping eg. blueish for Ascending and reddish for Descending orbits.
Blue like colors e.g. : #0000ff, #8a2be2, #6495ed,#00008b, #00ffff, #7fffd4
Red like colors e.g. : #dc143c, #a52a2a, #d2691e, #ff7f50, #ff1493, #ff00ff
see for instance http://cloford.com/resources/colours/500col.htm

6.5.d) *Create a combined baselines plot from 2 data sets with different baselines criteria: plot_Multi_span_multi_Baselines.sh*

plot_Multi_span_multi_Baselines.sh makes a common baselines plot for 2 data sets of different baselines.

The script must be launched from …/MSBAS_RESULTS/MSBAS_SAT_REGION/ and data are expected to be in subdirectories …/MSBAS_RESULTS/MSBAS_SAT_REGION/DefoMode*i*.
It supposes that a first plot was already computed with ***Prepa_MSBAS.sh*** for each mode in order to generate the span(1).txt files with the same MaxBp and MaxBt.

Launch ***plot_Multi_span_multi_Baselines.sh*** with the following 10 parameters:

SETLIST	(file with the list of sets used in the order of DefoMode*i* dir).
MinBp1	(minimum Bp of first data set)
MaxBp1	(maximum Bp of first data set)
MinBt1	(minimum Bt of first data set)
MaxBt1	(maximum Bt of first data set)
ColorTable	(file with colour table in hex)
MinBp2	(minimum Bp of second data set)
MaxBp2	(maximum Bp of second data set)
MinBt2	(minimum Bt of second data set)
MaxBt2	(maximum Bt of second data set)

6.5.e) *Create a combined baselines plot from several data sets with different baselines criteria (using AMSTerEngin version May 2022 or later): plot_Multi_BaselinePlot.sh*

The script ***plot_Multi_BaselinePlot.sh*** makes a common Baseline plot for as much data set as wanted. It is sort of an updated version of ***plot_Multi_span_multi_Baselines.sh*** as it makes also a common imageSpatialLocalization plot for these multiple data sets.

Notes :

- It supposes that a first plot was already computed (see ***Prepa_MSBAS.sh***) for each mode in order to generate the required data and gnuplot files
- It required AMSTerEngine version May 2022 or later as it needs acquisitionsRepartition.txt and gnuplot files prepared during run of ***Prepa_MSBAS.sh***.
- Unlike ***plot_Multi_span_multi_Baselines.sh*** it searches by himself the Bp, Bt and the date of the Global Primary for each dataset
- If more than 5 different datasets are plotted, colours code in the imageSpatialLocalization figure will loop (i.e. set 1 and 6 will be plotted with the same colour). But more than 5 sets would be unreadable anyway... Above 10 sets, see yourself...

It must be launched with only one parameter, that is a text file listing the datasets as e.g. :

$PATH_1650/SAR_SM/MSBAS/*REGION*/set*6*
$PATH_1650/SAR_SM/MSBAS/*REGION*/set*7*

7. Automation with cronjobs:

In SCRIPTS_MT/_cron_scripts/ some scripts are suggested to make a fully automated system using Sentinel 1 data or CSK data.

First, the data must be downloaded. It is the responsibility of the user to set up such a procedure. ESA may provide some hints to set up scripts to automatically download S1 data from their web site for instance. We provide below some examples for S1 data.

From there, 3 main scripts will process the data up to the time series.

A final batch of scripts is used to create and update the web page. See *Web_tool_Vxx.pdf* manual by Maxime Jaspard for more information about that part.

The procedure and the AMSTer software is also illustrated and explained in the following paper (open access), although the software was still named MasTer at the time of its publication:

Derauw D., d'Oreye N., Jaspard M., Caselli A. and Samsonov S.
Ongoing automated Ground Deformation monitoring of Domuyo – Laguna del Maule area (Argentina) using Sentinel-1 MSBAS time series: Methodology description and first observations for the period 2015 – 2020. *J. South Am. Earth Sc., Vol. 104, 102850*.
https://doi.org/10.1016/j.jsames.2020.102850

Open Access here:
https://www.sciencedirect.com/science/article/pii/S089598112030393X?via%3Dihub

You will need to adapt the cron scripts explained below to your needs and prepare crontabs to launch them if required.

Some explanations are illustrated below with the processing of Sentinel 1 data for the Domuyo/Laguna del Maule region (Argentina/Chile) used for the above-mentioned paper. The corresponding scripts are provided in Annex A.6).

Remember:

- Launch your scripts while redirecting the output to /dev/null to avoid unwanted messages filling up you mailbox or logs.
- If you run your scripts on recent Mac OSX, where bash is not the default shell, launch your scheduled scripts as follow

```
hh mm * * * /bin/bash YourScript.sh  > /dev/null 2>&1
```

- If you run your scripts on LINUX, it may not know/read your *.bashrc*. To overcome that, define your environmental variables within you crontab, eg.

```
PATH_1650=/mnt/1650
PATH_3600=/mnt/3600
PATH_3601=/mnt/3601
PATH_3602=/mnt/3602
PATH_DataSAR=/mnt/DataSAR
PATH_SynoData=/mnt/syno_data
PATH_SCRIPTS=/home/nicolas/SAR
ENVISAT_PRECISES_ORBITS_DIR=/mnt/DataSAR/SAR_AUX_FILES/ORBITS/ENV_ORB/vor_gdr_d
S1_ORBITS_DIR=/mnt/DataSAR/SAR_AUX_FILES/ORBITS/S1_ORB
EARTH_GRAVITATIONAL_MODELS_DIR=/mnt/DataSAR/SAR_AUX_FILES
PATHGNU=/usr/bin
PATHCONV=/usr/bin
PATHFIJI=/usr/bin
hh mm * * * /bin/bash YourScript.sh  > /dev/null 2>&1
```

- If cron does not start (e.g. with recent Mac OSX), ensure that you have allowed cron and crontab to have “Full Disk Access” rights. For that:
  - open the System Preferences > Security & Privacy > Full Disk Access.
  - Unlock the dialog widows using the lock and your pwd,
  - click on “+”, cmd + shift + g and enter /usr/bin, then select “crontab”
  - click on “+”, cmd + shift + g and enter /usr/sbin, then select “cron”

- On Linux, ensure your *.bashrc* don’t start with the following lines (as by default in at least Ubuntu 18.04 and later):

```
# ~/.bashrc: executed by bash(1) for non-login shells.
# see /usr/share/doc/bash/examples/startup-files (in the package bash-doc)
# for examples

# If not running interactively, don't do anything
case $- in
   *i*) ;;
     *) return;;
esac
```

  If this is the case, comment them of course.

7.1. Step 0: Download the images Sentinel 1 images (*sentinel1_download_all.sh* and *sentinel1_downloader_ingestiondate.sh*)

The script ***sentinel1_download_all.sh*** is launched on a Mac server 3 times per day (at midnight, 6am and 6pm). It runs several calls of the script ***sentinel1_downloader_ingestiondate.sh***, each one being aimed at a different target we are monitoring at ECGS. In our case, as we are only interested in the Domuyo/Laguna del Maule region (or Domuyo in short), the only lines that are of interest in that script are:

```
## Domuyo
/Users/doris/scripts/sentinel1_downloader_ingestiondate.sh --domuyo18 --slc --startdate=10-DAYS-AGO --enddate=TODAY --skipmd5check --deletezip30days
/Users/doris/scripts/sentinel1_downloader_ingestiondate.sh --domuyo83 --slc --startdate=10-DAYS-AGO --enddate=TODAY --skipmd5check --deletezip30days
```

i.e. it calls the script ***sentinel1_downloader_ingestiondate.sh*** to check and download is any new *slc* data available over the *domuyo* region between today and *10 days ago*. To operate, the script needs (hard coded in the script):

- a login and password to access ESA COPERNICUS’s scihub web site
- the target directory where to store the new data (…/SAR_DATA/S1/S1-DATA-DOMUYO-SLC/)
- the corners of the footprint for the area of interest

After having compared local directory with data provider’s directory and downloaded only the required files, it will unzip the data in …/SAR_DATA/S1/S1-DATA-DOMUYO-SLC.UNZIP and delete zip files older than 30 days.

It will then send an email to dedicated addresses to provide feedback about the status of the operation. The details of the execution and results are stored in a log file.

See scripts for details (Annex A.6.1). Special thanks to Gilles Celli who wrote these scripts.

7.2. Step 1: Read and coregister images on a Global Primary (Super Master) (*Domuyo_S1_Step1_Read_SMCoreg_Pairs.sh*)

The script ***Domuyo_S1_Step1_Read_SMCoreg_Pairs.sh*** is launched once per day by a cronjob e.g. at 3 am.

It will:

- Create a log file $PATH_1650/SAR_CSL/S1/ARG_DOMU_LAGUNA/Last_Run_Cron_Step1.txt which contains the date of execution (start and stop), which can be useful for verification or debugging.
- Read all the images (and stitch the necessary swaths/bursts) with ***Read_All_Img.sh*** (see chapter 0) from $PATH_3600/SAR_DATA/S1/S1-DATA-DOMUYO-SLC.UNZIP and transform then in CSL format in $PATH_1650/SAR_CSL/S1/ARG_DOMU_LAGUNA/NoCrop.
  Because all orbit geometries are read at the same time, it will then sort each image depending on its geometry and store them respectively in $PATH_1650/SAR_CSL/S1/ARG_DOMU_LAGUNA_A_18/NoCrop and $PATH_1650/SAR_CSL/S1/ARG_DOMU_LAGUNA_A_18/NoCrop while keeping links in the original $PATH_1650/SAR_CSL/S1/ARG_DOMU_LAGUNA/NoCrop (to prevent re-downloading the data if they are missing in that directory).
  The script ***Read_All_Img.sh*** will also store data older than 6 months in $PATH_3600/SAR_DATA/S1/S1-DATA-DOMUYO-SLC.UNZIP_FORMER/_YYYY in order to speeding up the processing of reading the data (because it reads all the data in the directory).
  It will then check the orbits (or FAST24 data) and update the new ones (without downloading again the data). If no process is running using the updated products, it will delete all the products created with outdated orbits so that they can be rebuild. If a processing is running using these data, it stores the list of outdated products to be cleaned at next run.
- Check the nr of bursts and coordinates of corners of each image using the script ***_Check_ALL_S1_SizeAndCoord_InDir.sh***. If these values are not as expected, it will move the images in a temporary quarantine directory (__TMP_QUARNATINE) and log the names of these files and the reason why, that is the unexpected values compared to the expected ones.
  Note that the expected values of kml corners can be obtained by running ***_Check_S1_SizeAndCoord.sh*** $PATH_1650/SAR_CSL/*SAT*/*TRK*/NoCrop/*img*.csl *Dummy* where the path is the full path to a good image in SLC format for that region and *Dummy* only tells the script to read and output the values instead of checking them against know values (see 9.1).
  Instead of providing ***_Check_ALL_S1_SizeAndCoord_InDir.sh*** with these kml corners, one can provide the min longitude, max longitude, min latitude and max latitude of the area of interest measured eg. from Google Earth.
  It is the responsibility of the user to check regularly the $PATH_1650/SAR_CSL/*SAT*/*TRK*/NoCrop/__TMP_QUARNATINE directory: images there are either not fully downloaded yet and hence it might be solved the day after, or the image has a problem and must be manually moved to the definitive $PATH_1650/SAR_CSL/*SAT*/*TRK*/quarantained directory.
  Note however that sometimes the S1 tasking may vary a little and the number of bursts required to cover the area of interest may vary if the zone is close to the limit of a burst (Murphy's law… it happens more often that one may think). In such a case, the checking

script is run twice, one with each expected number of bursts and corner's coordinates. This is the case in our example for images acquired in orbit *D_83*.

- Coregister all the new images on the Global Primary image (*20180512* in Asc and *20180222* in Desc orbits) using ***SuperMasterCoreg.sh***. However, because S1 orbits are good enough, coregistration on a Global Primary is not required here. Instead, the script will take that opportunity to compute the DEM (and mask if available) in the slant range geometry of each new image. Results are stored in $PATH_1650/SAR_SM/RESAMPLED/S1/ARG_DOMU_LAGUNA_A_18/SMNoCrop_SM_20180512 and $PATH_1650/SAR_SM/RESAMPLED/S1/ARG_DOMU_LAGUNA_D_83/SMNoCrop_SM_20180222
- Search for new compatible interferometric pairs to be formed with the new data. For this it will first link the new Asc and Desc data respectively in $PATH_1650/SAR_SM/MSBAS/ARGENTINE/set1 and $PATH_1650/SAR_SM/MSBAS/ARGENTINE/set2 thanks to the script ***Ins_All_Img.sh***. Then it will run ***Prepa_MSBAS.sh*** in order to select the new pairs satisfying the spatial and temporal baseline criteria (respectively *20* meters and *450* days) and make the baseline plots.
  Note that the list of
  ⇒ all interferometric combinations (for Asc and Desc geometries resp.),
  ⇒ the list of pairs satisfying the baselines criteria (for Asc and Desc geometries resp.),
  ⇒ the baselines plot (for Asc and Desc geometries resp.), and
  are stored respectively in
  ⇒ $PATH_1650/SAR_SM/MSBAS/ARGENTINE/set1/approximateBaselinesTable.txt and $PATH_1650/SAR_SM/MSBAS/ARGENTINE/set2/approximateBaselinesTable.txt,
  ⇒ $PATH_1650/SAR_SM/MSBAS/ARGENTINE/set1/table_0_20_0_450.txt and $PATH_1650/SAR_SM/MSBAS/ARGENTINE/set2/table_0_20_0_450.txt,
  ⇒ $PATH_1650/SAR_SM/MSBAS/ARGENTINE/set1/span_0_20_0_450.jpg and $PATH_1650/SAR_SM/MSBAS/ARGENTINE/set2/span_0_20_0_450.
- Create a combined baselines plot (Asc+Desc) and store it in $PATH_1650/SAR_SM/MSBAS/ARGENTINE/BaselinePlots_S1_set_1_2/span_2sets_0_20_0_450.jpg

See script for details (Annex A.6.2).

7.3. Step 2: Processing all pairs (*Domuyo_S1_Step2_MassProc.sh*)

The script ***Domuyo_S1_Step2_MassProc.sh*** is launched once per day by a cronjob at 5h30 am.

It will:

- Check that no ***Domuyo_S1_Step1_Read_SMCoreg_Pairs.sh*** is running. If such a process is still running, it will stop and try again the day after.
- Check that no ***SuperMaster_MassProc.sh*** is already running with the same parameters (i.e for Asc and Desc respectively $PATH_1650/Param_files/S1/ARG_DOMU_LAGUNA_A_18/LaunchMTparam_S1_Arg_Domu_Laguna_A_18_Zoom1_ML4_MassProc_MaskCohWater.txt and $PATH_1650/Param_files/S1/ARG_DOMU_LAGUNA_D_83/LaunchMTparam_S1_Arg_Domu_Laguna_D_83_Zoom1_ML4_MassProc_Snaphu_WaterCohMask.txt).
  If such a process is still running, it will stop and try again the day after.
- Launch the script ***SuperMaster_MassProc.sh*** for the Asc and Desc modes.

If it aborts, it logs the info in a corresponding text file.

See script for details (Annex A.6.3).

7.4. Step 3: MSBAS processing and time series computation (*Domuyo_S1_Step3_MSBAS.sh*)

The script ***Domuyo_S1_Step3_MSBAS.sh*** is launched once per day by a cronjob at 5h30 pm.

It will:

- Check that no ***Domuyo_S1_Step3_MSBAS.sh*** is running. If such a process is still running, it will stop and try again the day after.
- Check that no ***SuperMaster_MassProc.sh*** is already running with the same parameters (i.e for Asc and Desc respectively).
- Check that no ***Domuyo_S1_Step2_MassProc.sh*** is running. If such a process is still running, it will stop and try again the day after.

If it stops for any of the previous reasons, it will store the information in
$PATH_3602/MSBAS/_Domuyo_S1_Auto20m_450days/_last_MSBAS_process.txt

- Look for duplicated products in all *Mode* directories based on pair of dates that is in the name. Duplicate products may happen in /Geocoded/*Mode* and /GeocodedRasters/*Mode* (where *Mode* is e.g. DefoInterpolx2Detrend) when a pair is reprocessed with updated orbit with resulting slightly different Bp. It then move them in each *Mode*/___Duplicated_ToKill.
  It DOES NOT PROCESS GEOCODED AMPLITUDE IMAGES because PRM and SCD can be geocoded within the same pair. All this is performed using a script located in zz_Utilities_MT_Ndo that is ***Remove_Duplicate_Pairs_File_All_Modes_But_Ampl.sh***

- Check the date/time at which the last Asc and Desc pairs were computed and compare to the date/time at which the last Asc and Desc pairs were processed during the last run of this ***Domuyo_S1_Step3_MSBAS.sh***. This information was saved at the previous run in
  $PATH_3602/MSBAS/_Domuyo_S1_Auto_20m_450days/_Last_MassProcessed_Pairs_Time.txt

If no new pair were computed, it stops. (If it is the first run, the script will cope with the absence of the file with dates from a previous run).

- Remove possible broken links in MSBAS/.../*Modei* and clean files in corresponding lines in MSBAS/.../*Modei*.txt, i.e. clean files in
  $PATH_3602/MSBAS/_Domuyo_S1_Auto_20m_450days/DefoInterpolx2Detrend*i*
  (where *i* is *1* for Asc and *2* for Desc). Links in /DefoInterpolx2Detrend*i* must points to existing deformation maps in
  $PATH_3601/SAR_MASSPROCESS/S1/ARG_DOMU_LAGUNA_A_18/SMNoCrop_SM_20180512_Zoom1_ML4/Geocoded/DefoInterpolx2Detrend and
  $PATH_3601/SAR_MASSPROCESS/S1/ARG_DOMU_LAGUNA_D_83/SMNoCrop_SM_20180222_Zoom1_ML4/Geocoded/DefoInterpolx2Detrend
  This check is performed with the script ***Remove_BrokenLinks_and_Clean_txt_file.sh***.
- It does it also in case of processing with coherence cleaning threshold in MSBAS/.../*Modei*_Full with the script ***Remove_BrokenLinks_and_Clean_txt_file.sh.***

- If not the first run,
  - Check that each DefoInterpolx2Detrend*i*.txt contains 4 columns and clean it if required (both for full processing and for coherence restricted processing if it exists).
  - Remove lines in MSBAS/ DefoInterpolx2Detrend*i*.txt associated to possible broken links or duplicated lines (i.e. same pairs but name differs because of perpendicular baseline or altitude of ambiguity e.g. after S1 orbit update; in that case it only keeps the most recent file). This is performed using ***_Check_bad_DefoInterpolx2Detrend.sh***
  - Same cleaning in DefoInterpolx2Detrend*i*_Full if previous processing was performed with coherence cleaning threshold

- Prepare the required files for the msbas processing using ***build_header_msbas_criteria.sh*** (see chapter 6.1.a)
- Update the msbas $PATH_3602/MSBAS/_Domuyo_S1_Auto_20m_450days/header.txt file by setting up the appropriate inversion order and lambda factor (*3* and *0.04* in the present case).
- Update the msbas $PATH_3602/MSBAS/_Domuyo_S1_Auto_20m_450days/header.txt file by discarding the calibration. Calibration is indeed not necessary because the deformation maps were detrended (i.e. averaged to zero).

- Check again (if first run or in case some pairs would have been added in the meantime…) that each DefoInterpolx2Detrend*i*.txt contains 4 columns.
- In case of processing with coherence threshold restriction, clean again (if first run, or just in case...) MSBAS/ DefoInterpolx2Detrend*i*.txt from possible lines associated to broken links or duplicated lines using ***_Check_bad_DefoInterpolx2Detrend.sh.***

- If not the first run, Remove again (just in case…) possible broken links and clean files in $PATH_3602/MSBAS/_Domuyo_S1_Auto_20m_450days/DefoInterpolx2Detrend*i* using ***Remove_BrokenLinks_and_Clean_txt_file.sh***

- Run a first msbas processing with ***MSBAS.sh*** using all the available Asc and Desc deformation maps satisfying to the baselines criteria (i.e. without coherence restriction) and output the time series for all the reference points provided in list $PATH_SCRIPTS/SCRIPTS_MT/_cron_scripts/*Cln_Domuyo.txt*.
  These plots are simple time series (no double difference) with error bars (stdev computed in the radius around the pixel as provided in Cln_Domuyo.txt, which contains 4 columns: *Xpix Ypix Xrad Yrad* where *X pix* and *Y pix* are coordinates of the pixel (position in file) and *rad* is radius (in pixel) around the pixel to estimate the stdev. Plots are named by the pixel coordinates.
- Make a baseline plot using ***PlotBaselineGeocMSBASmodeTXT.sh*** for the data set without coherence restriction
- Add description to pdf plots and txt time series of single points with their error bars. Description comes from point file $PATH_SCRIPTS/SCRIPTS_MT/_cron_scripts/Points_TS_Domuyo.txt that contains (after a header "*name x y radiusX raduisY*") the following fields : "*description_as_desired Xcoord Ycoord Xrad Yrad*", where *rad* is the radius (in pixels) around the *X*/*Y* pixel where the stdev is computed. Beware that valid pixels must exist everywhere in that radius (if not, i.e. if *X*/*Y* pixel to plot is located at the border of a zone, set *rad* to zero; no stdev will be computed of course).
- Create plots of double difference between the time series of pixels provided in $PATH_SCRIPTS/SCRIPTS_MT/_cron_scripts/List_DoubleDiff_EW_UD_Domuyo.txt, which contains 5 columns: *Xpix1 Ypix1 Xpix2 Ypix2 string* where *X* and *Y* are coordinates of the two pixels and *string* is a text to describe the area where are located the pixels. Plots are named by the pixel coordinates and *string*. A small inset is added in the plot to help the reader figuring out where the pixels are located and interpreting the sense of deformation observed and displayed as a double difference (see description of ***PlotTS_all_comp.sh*** for explanations).
  Plots are stored in $PATH_3602/MSBAS/_Domuyo_S1_Auto_20m_450days/zz_UD_EW_TS_Auto_3_0.04_Domuyo_NoCohThresh/__Combi/ and text files with time series values are stored in $PATH_3602/MSBAS/_Domuyo_S1_Auto_20m_450days/zz_UD_EW_TS_Auto_3_0.04_Domuyo_NoCohThresh/_Time_series/.

- Apply the coherence restriction: Remove all the interferometric pairs for which the mean coherence computed in an area defined by a provided kml file

($PATH_1650/kml/ARGENTINA/Laguna_Maule.kml in the present case) is below a provided Coherence Threshold (*0.235* in the present case). This selection is performed using the script ***restrict_msbas_to_Coh.sh***

- Offers the possibility to force exclusion of pairs that satisfies the baseline criteria and the threshold although they are obviously decorrelated on the area under concern. Instead of lowering the Coherence Threshold (which would result in removing probably much more pairs that only the faulty one), one can exclude here some pairs that are be provided in a list stored in $PATH_3602/MSBAS/_Domuyo_S1_Auto_20m_450days/DefoInterpolx2Detrend*i*/_EXCLUDE_PAIRS_ALTHOUGH_CRITERIA_OK.txt. This is performed with the script ***Exclude_Pairs_From_Mode.txt.sh***
- Run a second msbas processing using Asc and Desc deformation maps satisfying to the baselines criteria and after coherence threshold restriction.
- Make a baseline plot using ***PlotBaselineGeocMSBASmodeTXT.sh*** for the data set with coherence restriction
- Add description to pdf plots and txt time series of single points with their error bars. Description comes from point file $PATH_SCRIPTS/SCRIPTS_MT/_cron_scripts/Points_TS_Domuyo.txt.
- Create again plots of double difference between the time series of pixels provided in $PATH_SCRIPTS/SCRIPTS_MT/_cron_scripts/List_DoubleDiff_EW_UD_Domuyo.txt. Results are stored in directories with names consistent with the processing.

- Run a third msbas processing using only Asc deformation maps satisfying to the baselines criteria though after removing all the interferometric pairs for which the mean coherence computed in $PATH_1650/kml/ARGENTINA/Laguna_Maule.kml is below the Coherence Threshold of *0.235* and make the corresponding plots.
- Create again plots of double difference in LOS Asc between the time series of pixels provided in $PATH_SCRIPTS/SCRIPTS_MT/_cron_scripts/List_DoubleDiff_EW_UD_Domuyo.txt. (Note that the list of pixels could have been specific to Asc orbit). Results are stored in directories with names consistent with the processing.

- Run a fourth msbas processing using only Desc deformation maps satisfying to the baselines criteria though after removing all the interferometric pairs for which the mean coherence computed in $PATH_1650/kml/ARGENTINA/Laguna_Maule.kml is below the Coherence Threshold of *0.235* and make the corresponding plots.
- Create again plots of double difference in LOS Desc between the time series of pixels provided in $PATH_SCRIPTS/SCRIPTS_MT/_cron_scripts/List_DoubleDiff_EW_UD_Domuyo.txt. (Note that the list of pixels could have been specific to Desc orbit). Results are stored in directories with names consistent with the processing.

Notes:

1) To spare time, the processing keeps track of pairs already tested against the baseline criteria as well as the Coherence Threshold. This information is stored in several files in $PATH_3602/MSBAS/_Domuyo_S1_Auto_20m_450days or DefoInterpolx2Detrend*i*/.
   See e.g. in /$PATH_3602/MSBAS/_Domuyo_S1_Auto_20m_450days/:
   - Checked_For_CohThreshold_To_Be_Ignored_At_Next_Rebuild_msbas_Header.txt

   See e.g. in /$PATH_3602/MSBAS/_Domuyo_S1_Auto_20m_450days/DefoInterpolx2Detrend*i*:
   - Checked_For_CohThreshold_To_Be_Ignored_At_Next_Rebuild_msbas_Header.txt
   - Checked_For_CohThreshold_To_Be_Ignored_At_Next_Rebuild_msbas_Header.txt_Inclunding_Broken_Links.txt
   - List_Checked_img_For_Coh_0.235_Laguna_Maule.kml.txt
   - Out_Of_Range_20m_450days.txt

See e.g. in
/$PATH_3602/MSBAS/_Domuyo_S1_Auto_20m_450days/DefoInterpolx2Detrend*i*_Full:

- Checked_For_CohThreshold_To_Be_Ignored_At_Next_Rebuild_msbas_Header.txt
- Checked_For_CohThreshold_To_Be_Ignored_At_Next_Rebuild_msbas_Header.txt_Inclunding_Broken_Links.txt
- Checked_For_CohThreshold_To_Be_Ignored_At_Next_Rebuild_msbas_Header.txt_Inclunding_Missing_Files.txt
- List_All_img_For_Coh_0.235_Laguna_Maule.kml.txt
- List_Checked_img_For_Coh_0.235_Laguna_Maule.kml.txt
- List_Out_Of_Range_CohThreshold_0.235_in_Laguna_Maule.kml.txt

In case of problem or doubts, it is advised to check these files and if needed remove them or start from scratch (that is erasing the whole directory $PATH_3602/MSBAS/_Domuyo_S1_Auto_20m_450days).
This will however require a much longer processing time at next run.

2) Remember that msbas will crash if the computer has insufficient RAM. In that case you may want to crop the area of interest by editing the header.txt.

3) Status of last MSBAS automatic processing (date and time in human readable format of last processing and info if finished without new pairs to process) is stored in a file $PATH_3602/MSBAS/*_Domuyo_S1_Auto_20m_450days*/_last_MSBAS_process.txt

4) See some logs in
/$PATH_3602/MSBAS/_Domuyo_S1_Auto_20m_450days/_MSBAS_log.txt
More details about the processing is stored e.g. in:
$PATH_3602/MSBAS/*_Domuyo_S1_Auto_20m_450days*/zz_LOS_TS_Asc_Auto_3_0.04_Domuyo/_Time_series/MSBAS_LOG.txt
$PATH_3602/MSBAS/*_Domuyo_S1_Auto_20m_450days*/zz_LOS_TS_Desc_Auto_3_0.04_Domuyo/_Time_series/MSBAS_LOG.txt
$PATH_3602/MSBAS/*_Domuyo_S1_Auto_20m_450days*/zz_UD_EW_TS_Asc_Auto_3_0.04_Domuyo/_Time_series/MSBAS_LOG.txt
$PATH_3602/MSBAS/*_Domuyo_S1_Auto_20m_450days*/zz_UD_EW_TS_Asc_Auto_3_0.04_Domuyo_NoCohThresh/_Time_series/MSBAS_LOG.txt

GENERAL NOTE: If these cron scripts are launched with cron tabs, it is usually a good practice to end cron lines with > /dev/null 2>&1 to mute output and avoid filling up internal mailbox. It is also required for those scripts that are launched with cronjobs and that must first check if another occurrence is already running. In that case, the scripts will search for that string (dev/null) to discard cron line from counting occurrence.

7.5. Step 4: Web page

The code for this part was written by Maxime Jaspard (Maxime@ecgs.lu) and is explained in the document entitled *Web_tool_Vx.x.pdf* (where *x.x* is the number of the latest version of that manual).

We briefly describe here the structure of the web page. See also the open access paper: Derauw et al., J. South Am. Earth Sc. (2020).

The web page of the automatic AMSTer processing of the Laguna del Maule – Domuyo region discussed here as an example is openly available at http://terra2.ecgs.lu/defo-domuyo.

Below and introductory header, the web page offers to use a tool to Calculate Time series on demand (see section 7.5.e), displays specific interferograms on demand by selecting the dates (see section 7.5.f), displays specific deformation maps between selected dates (it is not mandatory to combine images acquired in the same geometry because this is based on displacement maps resulting from MSBAS inversion; see section 7.5.g), or a RGB amplitude/coherence combination (see section 7.5.i) (Figure 11).

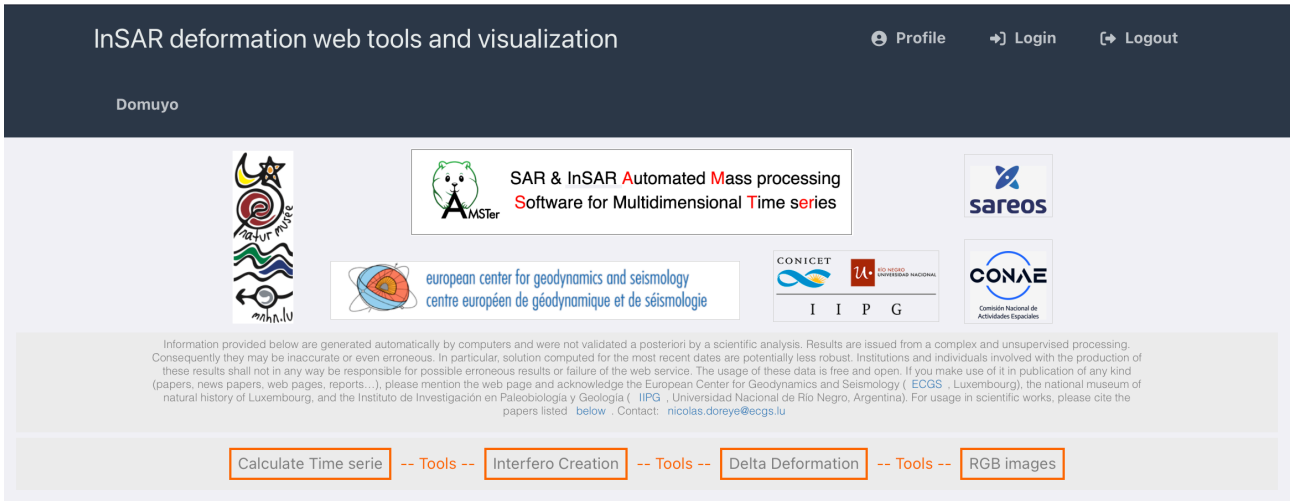


Figure 11: Header of the web page and buttons to switch to Time Series calculation on demand, display specific interferograms by selecting the dates, display specific deformation maps between selected dates (dates must not be from images acquired in the same geometry because this is based on displacement maps resulting from MSBAS inversion), or an RGB amplitude/coherence combination.

Then it offers to jump lower in the page directly to the panel showing either the Amplitude images (see section 7.5.b), the pre-defined time series (see section 7.5.d), the Baseline Plots (see section 7.5.h), or maps of Deformation linear rate Standard Deviation (see section7.5.j), followed with the information about the date of the last synchronization between the web server and the processing server (Figure 12).

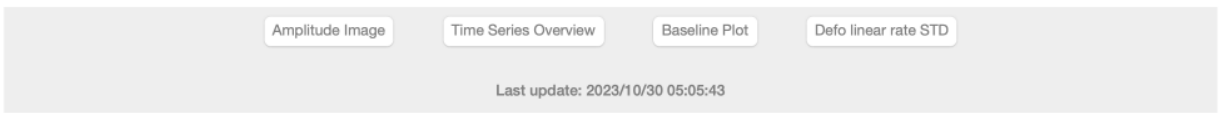


Figure 12: Button to jump directly to panel showing the Amplitude images, the pre-defined time series, the Baseline Plots, or Deformation Linear Rate Standard Deformation maps, followed with the date of the last synchronization between the web server and the processing server.

Further down, the results are displayed in five different main sections, accessible by scrolling down the page or by clicking on the buttons: the velocity maps (see section 7.5.a), the RGB maps

(see section 7.5.i), the amplitude maps (see section 7.5.b), the pixel location of pre-defined double difference time series (see section 7.5.c) and the pre-defined double difference time series (see section 7.5.d).

7.5.a) Velocity maps

The first panel (Figure 13) presents colour figures of linear deformation rates estimates along vertical (UD) and East-West (EW) axes (MSBAS processing) and along each LOS views (SBAS processing).

Clicking on the images allows to zoom in; clicking on the “Open KMZ” blue link allows downloading the kmz colour map to be displayed e.g. with *Google Earth*. Dates of the first and last images used are displayed above the maps.

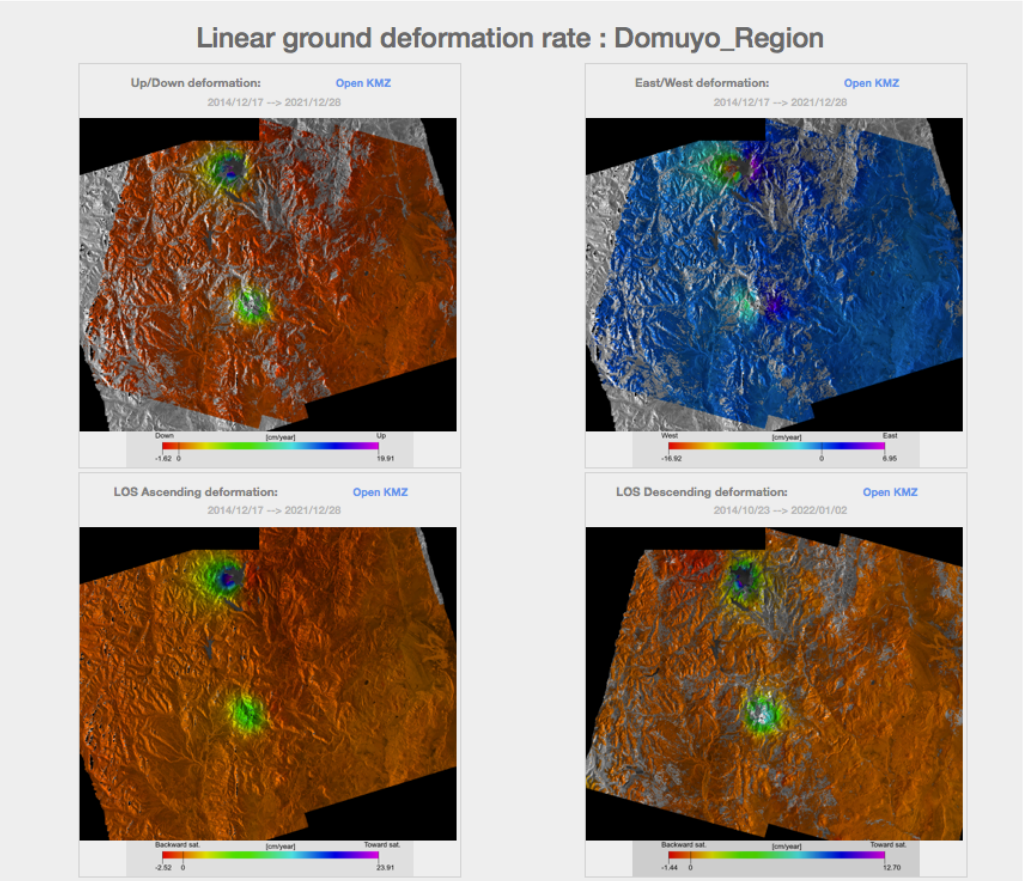


Figure 13: First panel of web page displaying the results of the AMSTer automated processing at Laguna del Maule – Domuyo region. Linear deformation rate maps computed using MSBAS processing along Up-Down (UD) and East-West (EW) directions (Above), and using SBAS processing along Ascending and Descending line of sight directions (below). Maps can be downloaded in kmz format and imported in Google Earth.

7.5.b) Amplitude maps

The section below (Figure 14) shows the last ascending and descending amplitude images of the area of interest and the averages of the last 10 amplitudes images in both modes. Dates of last images are displayed below the images.

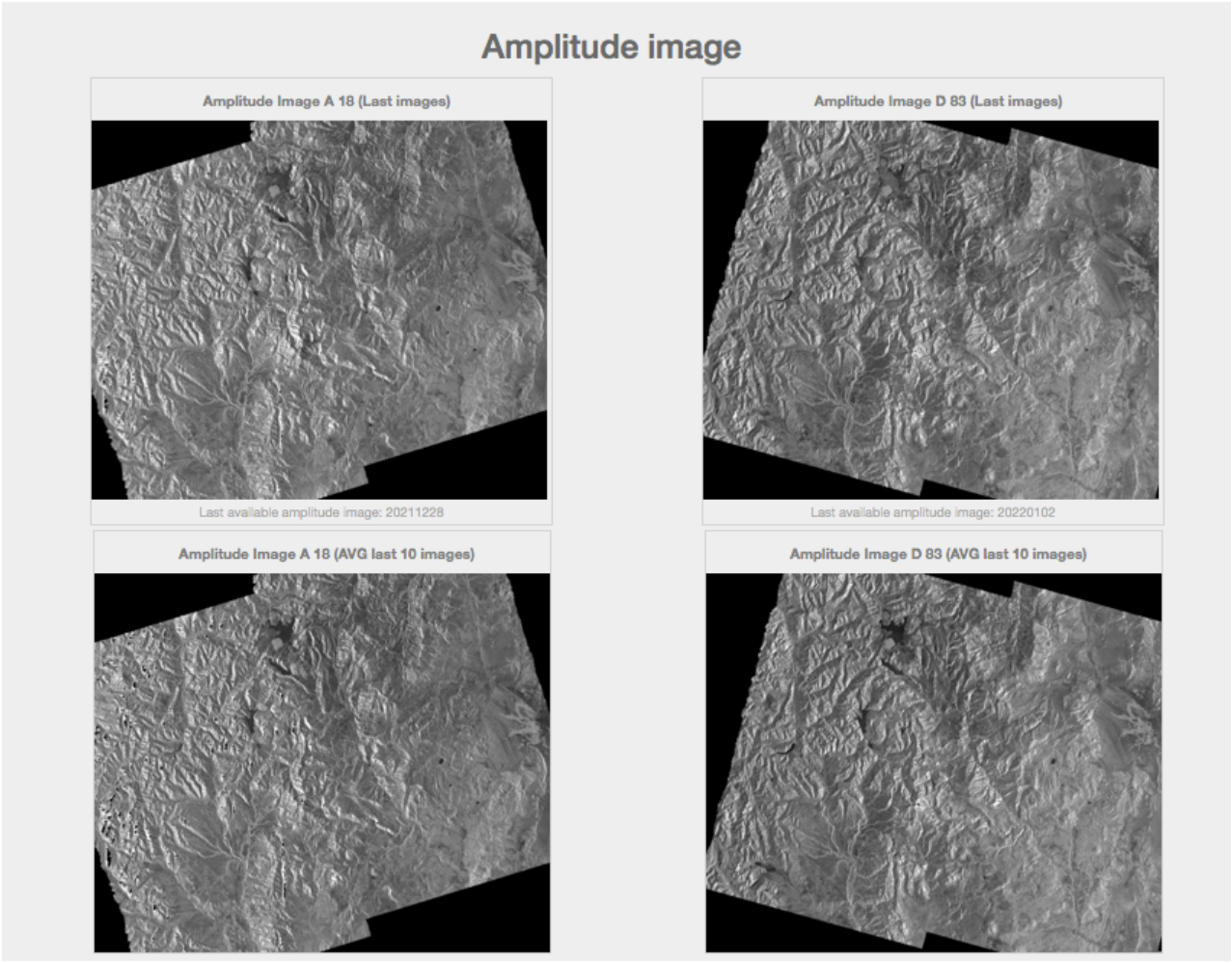


Figure 14: Amplitude maps of last Ascending and Descending images (Upper Left and Right resp.) and Amplitude average of the last 10 images acquired in Ascending and Descending orbits (Lower Left and Right resp.)

7.5.c) Pixel localization maps

The next panel (Figure 15) shows a full-size SAR amplitude image with colour framed zooms indicating the location of some pairs of pre-defined points of interest. That map and the zooms can be displayed as SAR amplitude image or a Google map or a Google Earth view (see buttons at the upper right corner). Clicking on the small maps contoured in colour make the web page to jump directly to panel displaying the differential time series of corresponding pixels (see 7.5.d).

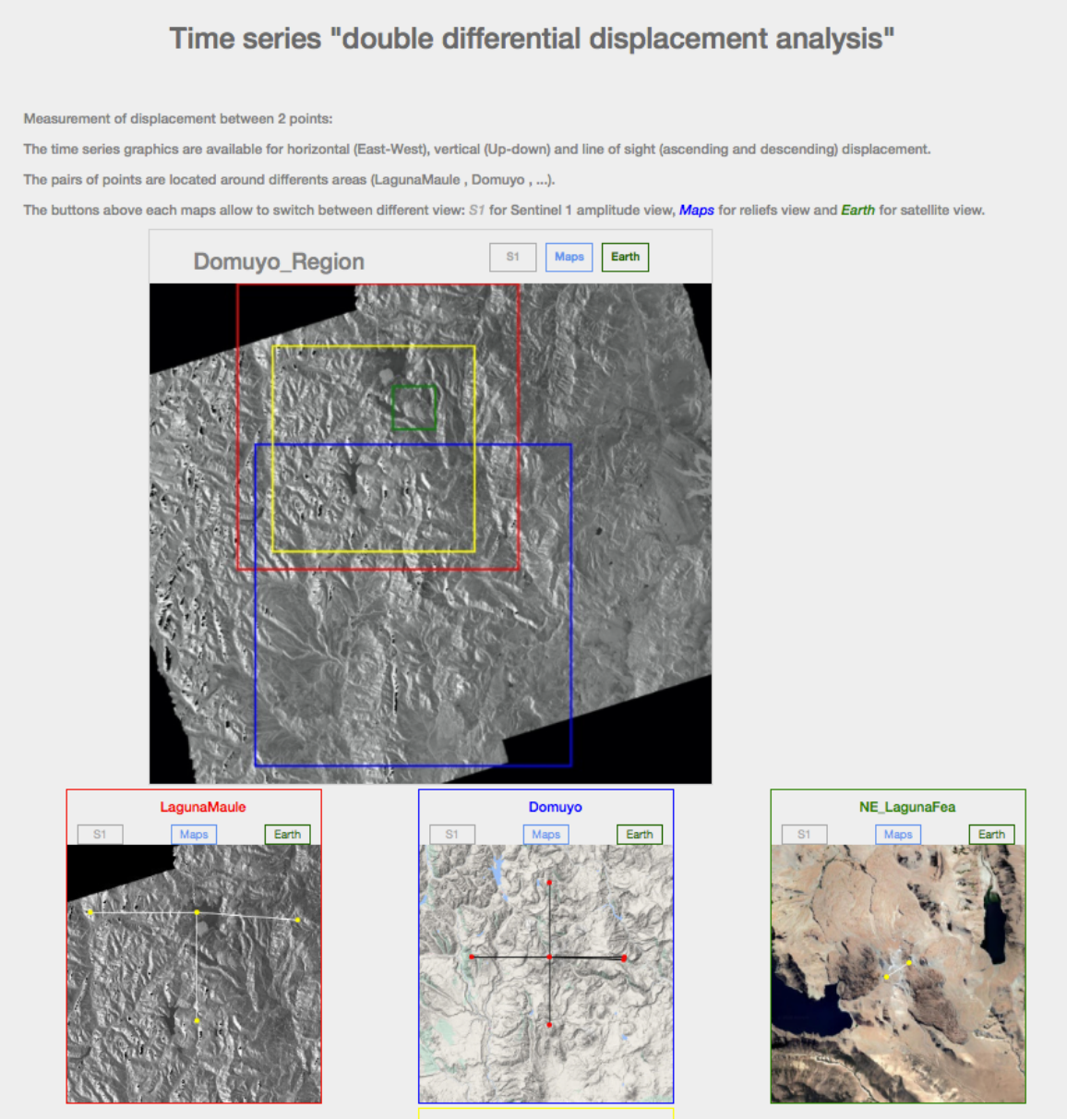


Figure 15: Example of maps displaying the location of pairs of pre-defined points of interest at the Laguna del Maule Volcanic Complex. The user can choose to display the location on a SAR amplitude image (main map and lower left zoom), a Google map view (lower center) or Google Earth (Lower right).

7.5.d) Differential time series of pre-defined pairs of pixels

Below are displayed the corresponding double difference time series of ground displacement at pre-defined pairs of points in UD & EW without coherence threshold, UD & EW with a coherence threshold (if applicable), and in both LOS (Figure 16).

To the left of the panel, pixels are located on an amplitude image (which can be changed to Google Map or Google Earth). For each pair, the point used as the reference is displayed in yellow and the “moving” point is displayed in white. The double difference is computed as the moving minus the reference point (i.e. values at the white point minus the values at the yellow point).

To help the reader to interpret the time series plot and understand the physical direction of the observed relative ground displacement, small explanatory sketches and location of points on the corresponding linear rate deformation maps are also plotted on the side of the time series graphs, best seen on the full-size plot obtained by clicking on the corresponding time series (Figure 17).

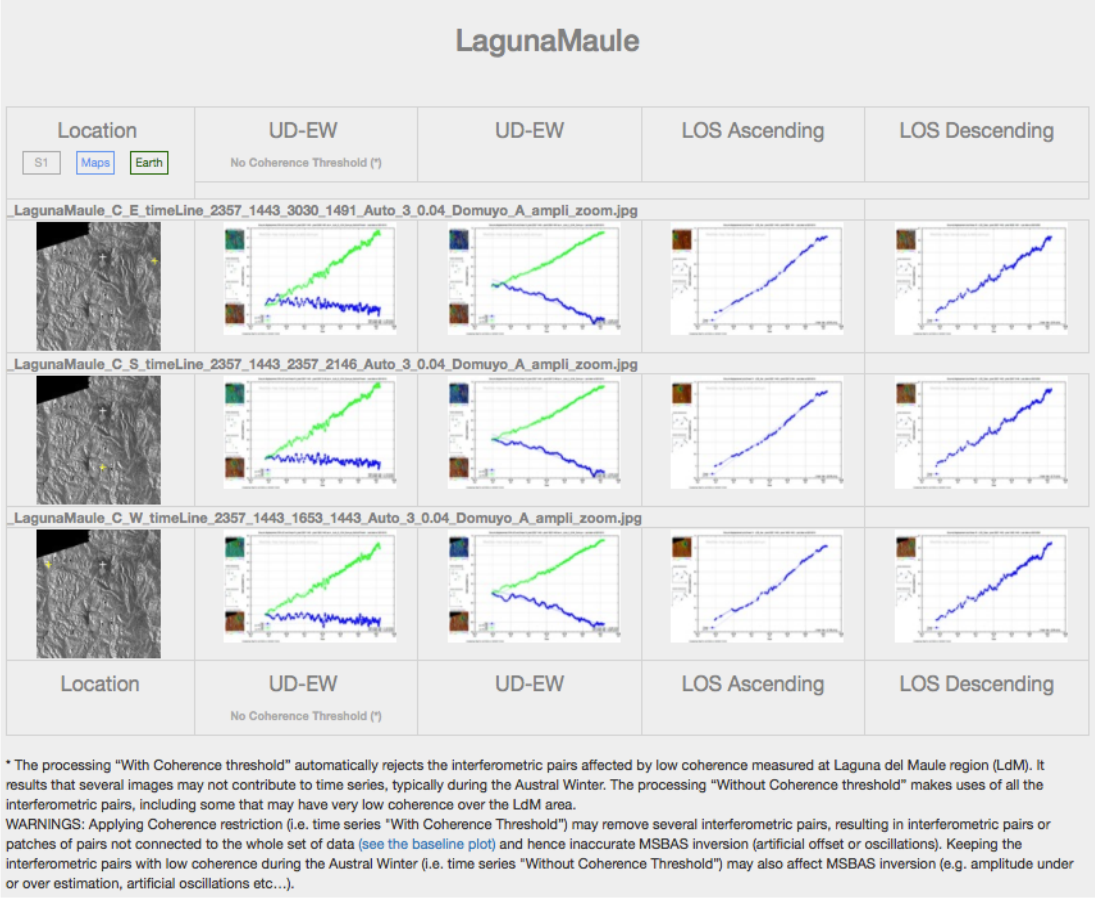


Figure 16: Web page sample showing some automatically generated double difference time series of ground deformation spanning 2015 – 2021 for pairs of points at Laguna del Maule (above). For each pair of points, UD-EW graph shows Up-Down (green) and East-West (blue) displacement and Line of Sight (LOS) displacements for Ascending and Descending orbits. Clicking on these plots allows enlarged view with detailed information about direction of displacement. Note that in the present case the processing automatically rejects pairs of low coherence measured on Laguna del Maule region (see second column of graphs). It results that several images do not contribute to time series (compared to first column), such as those acquired during the Austral Winter.

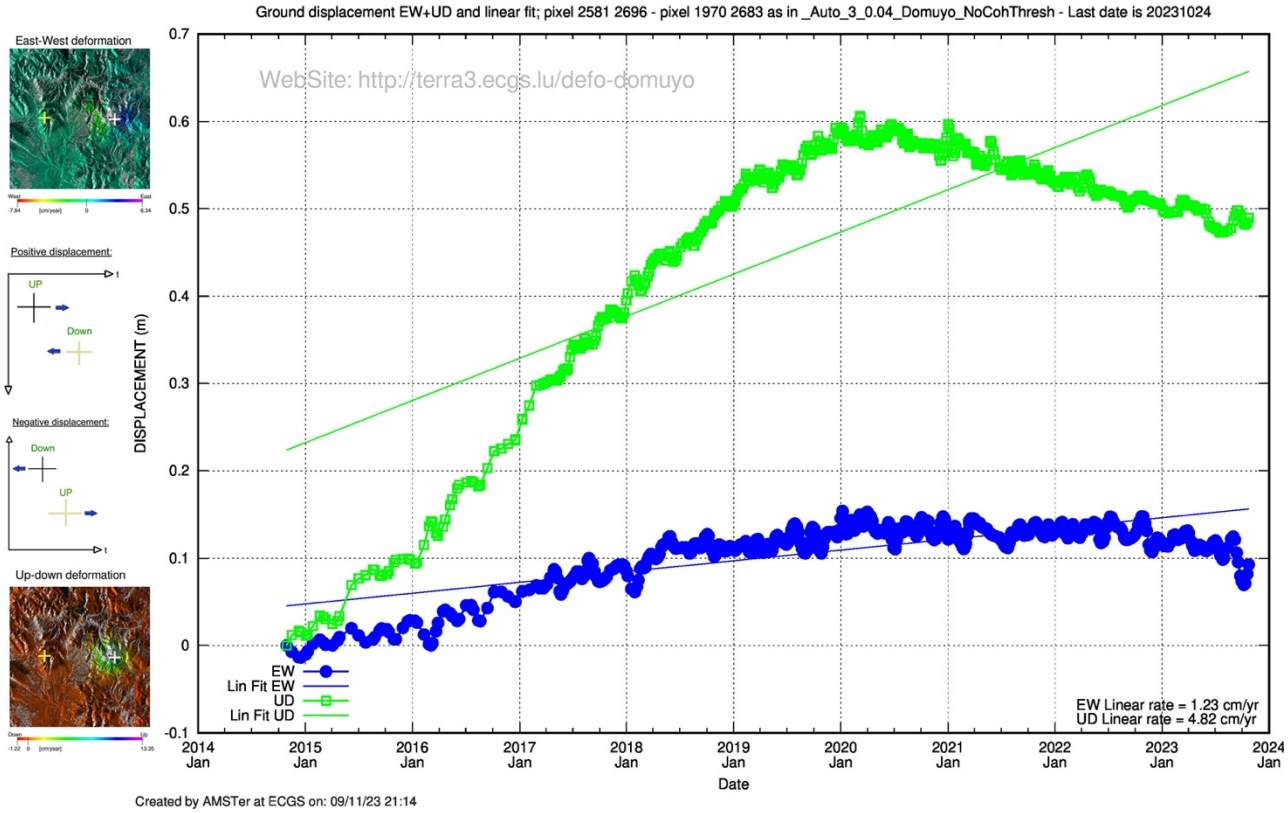


Figure 17: Zoom obtained by clicking on a graph from the web page. In the present case it shows the double difference of pixels located to the West and the summit of Domuyo volcano. See location on the insets to the left. Small sketch also explains the direction of displacement. A linear fit is displayed as a straight line and value of the mean annual rate is provided in the lower right of the picture (in cm/yr).

However, if the user wants to have a look at points that are not in the pre-defined list of points of interest, and hence displayed on the web page, see section 7.5.e).

7.5.e) Differential time series on demand

By clicking on the button “Calculate Time series” at the top of the page opens a new section that allows to select new points on a map and create its own time series plots (Figure 18).

The user can first select which layer to display by clicking on the small layers’ icon in the upper right corner (LOS Asc, LOS Desc, UP-DWN and/or EW). These layers can be displayed on the Google Earth image (terrain) or Google Map (relief). Amplitude Asc and Desc can also be displayed.

Note that the results of all the time series will be provided (LOS Asc, LOS Desc, UP-DWN and EW), whatever the layers were displayed (providing of course that the pixels are coherent for each mode).

One can zoom in and out in the map either by scrolling with the mouse or by clicking on the “+/-” icons in the upper left corner. To move laterally or vertically the map, click and grab the map while moving the mouse.

To select the points for the double difference, 3 options are possible:

1) Move the blue pins that are displayed on the map where you want your pixels. You can see the position of the pixels in number of lines and columns in the light grey text lines below the map. Latitude and Longitude is also displayed there.
2) If you know the position of the pixels in number of lines and columns, these coordinates can be manually entered in the small boxes below the map.
3) Similarly, the position of the pixels in decimal latitude and longitude can be manually entered in the small boxes below the map.

The user can move the points as much as wanted before submitting the time series request. Keeping the mouse inactive on a pin will display “Point A” or “Point B” for the user convenience in order to select the most appropriate order of double difference.

User can also change the time span of the desired time series.

To receive the figures with the selected differential time series, the user must provide a name for the request (to be chosen smartly at the user best convenience in order to conveniently remember what was requested) and must provide its e-mail address.

Notes:

1) To receive the double difference time series by email, the two pixels must be located in a coherent region.
2) It may happen that one of the pixels selected by the user is only coherent in Ascending or Descending orbit. In that case, the user will only receive the results in that coherent component.

Clicking on the orange button “Submit Time Series request” opens a new box summarizing the request while the server prepare the data and send the plots. **Check that box carefully!** If a pixel is located in a decorrelated region both in Asc and Desc, no time series will be computed and the user will receive no mail.

Plots are delivered by email within one or two minutes. If mail does not arrive, please check your spams.

Server will refuse another request while the first one is still processed.

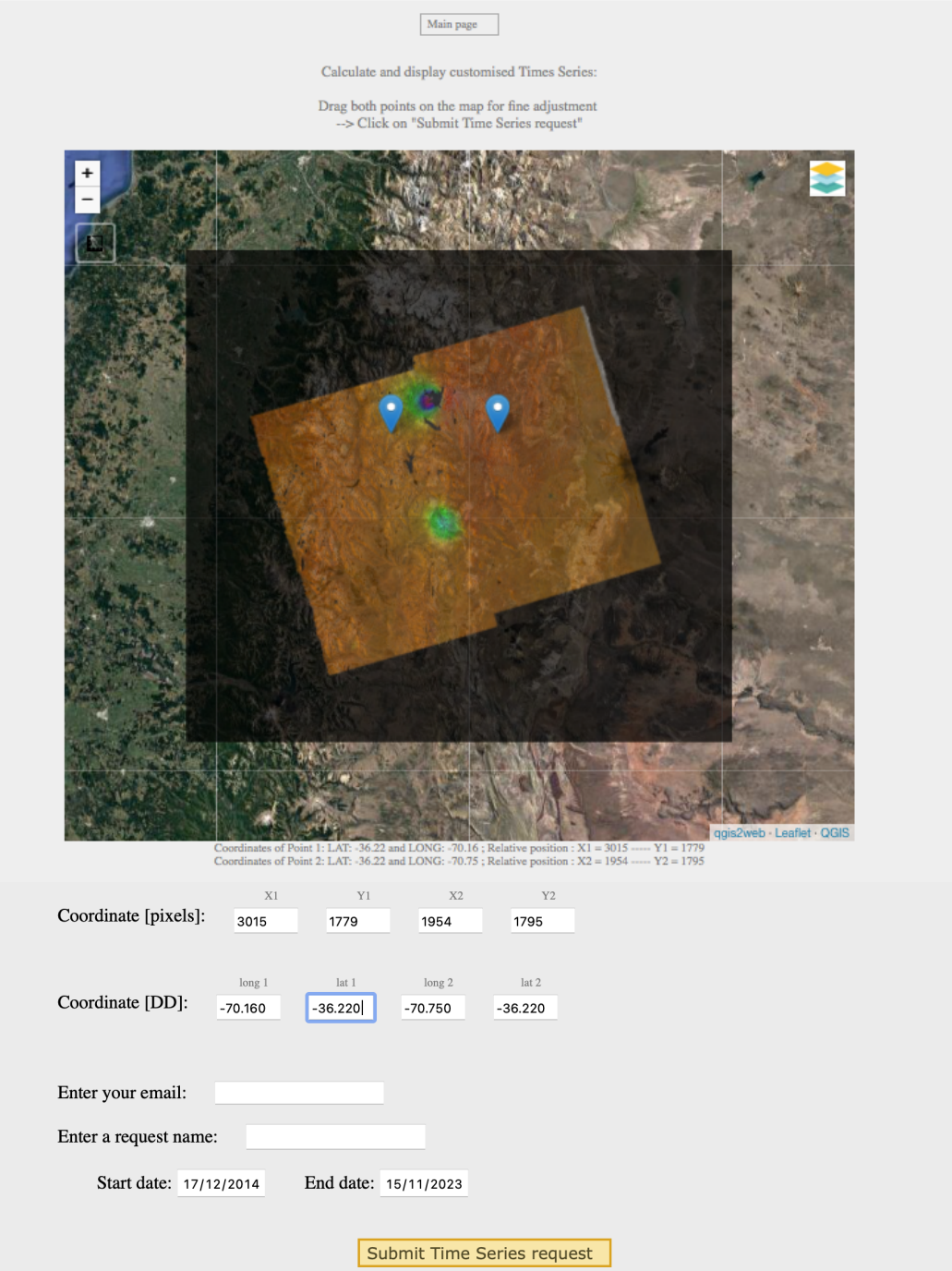


Figure 18: Example of time series request. User must select the 2 pixels (see procedure in main text), select the layers, select the time span, provide a name for the request (optional) and an email address where to receive the plots. Check the box that opens after submitting the request. It summarizes the request and warns the user if request is invalid (and hence no plot will be sent).

Note the small measurement tool in the upper left corner: it allows to click on the map an see the coordinates and distances between the points.

7.5.f) Display specific interferograms and coherence map on demand

By clicking on the button “Interefro Creation” at the top of the page opens a new section that allows to display specific wrapped interferograms in Line of Sight. This can be done by selecting first the Primary or Secondary image using dedicated grey buttons above the map (Figure 19).

The user must select first the satellite, then the orbit, then the dates using the menus. Note that to select the value that is displayed by default on the scrolling menu requires to click on Submit button. Selecting another a value form the scrolling menu directly jumps to the next thing to select (no need to click “Submit”).

When all parameters are selected, a last page summarizing the information is displayed. If satisfied, click “Submit”. If not, get back or cancel.

Displaying the wrapped interferogram may take one or two minutes (Figure 20). Note that coherence map is also available by clicking on the layer icon in the upper right corner.

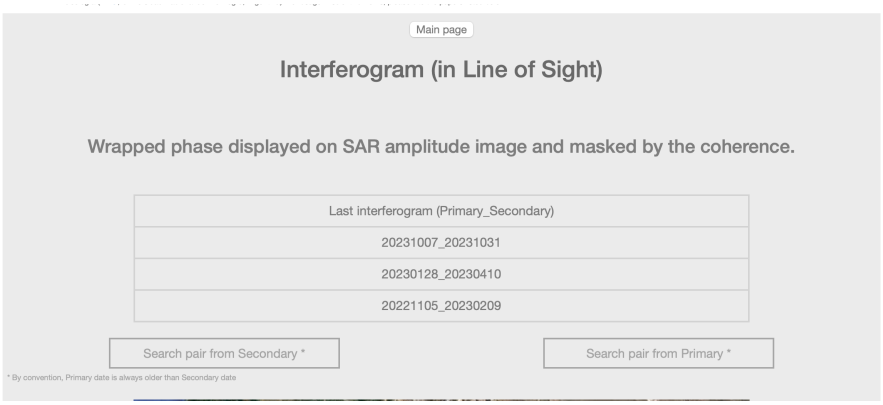


Figure 19: Web page displaying the beginning of the procedure for displaying wrapped interferogram on demand.

Figure 20: Example of wrapped interferogram on demand (left). Coherence map can also be displayed by clicking on the layer icon in the upper right corner (right).

7.5.g) Display specific deformation map on demand

By clicking on the button “Delta Deformation” at the top of the page opens a new section that allows to display specific deformation maps in EW, UD or LOS. This can be done by selecting first the deformation direction (EW, UD, Asc LOS or Desc LOS) using dedicated grey buttons above the map (Figure 21)

The user must first select the two dates. The most recent dates are already suggested by clicking on the layer icon in the upper right corner but any date can be otherwise chosen with the “Choose date” boxes. Note that it is not mandatory to combine images acquired in the same geometry because this is based on displacement maps resulting from MSBAS inversion, hence the use can combine any dates).

When satisfied with choices, click “Calculate differential deformation map (date 1 – date 2)”.

Displaying the deformation map may take one or two minutes (Figure 21)

Note again the small measurement tool in the upper left corner: it allows to click on the map an see the coordinates and distances between the points.

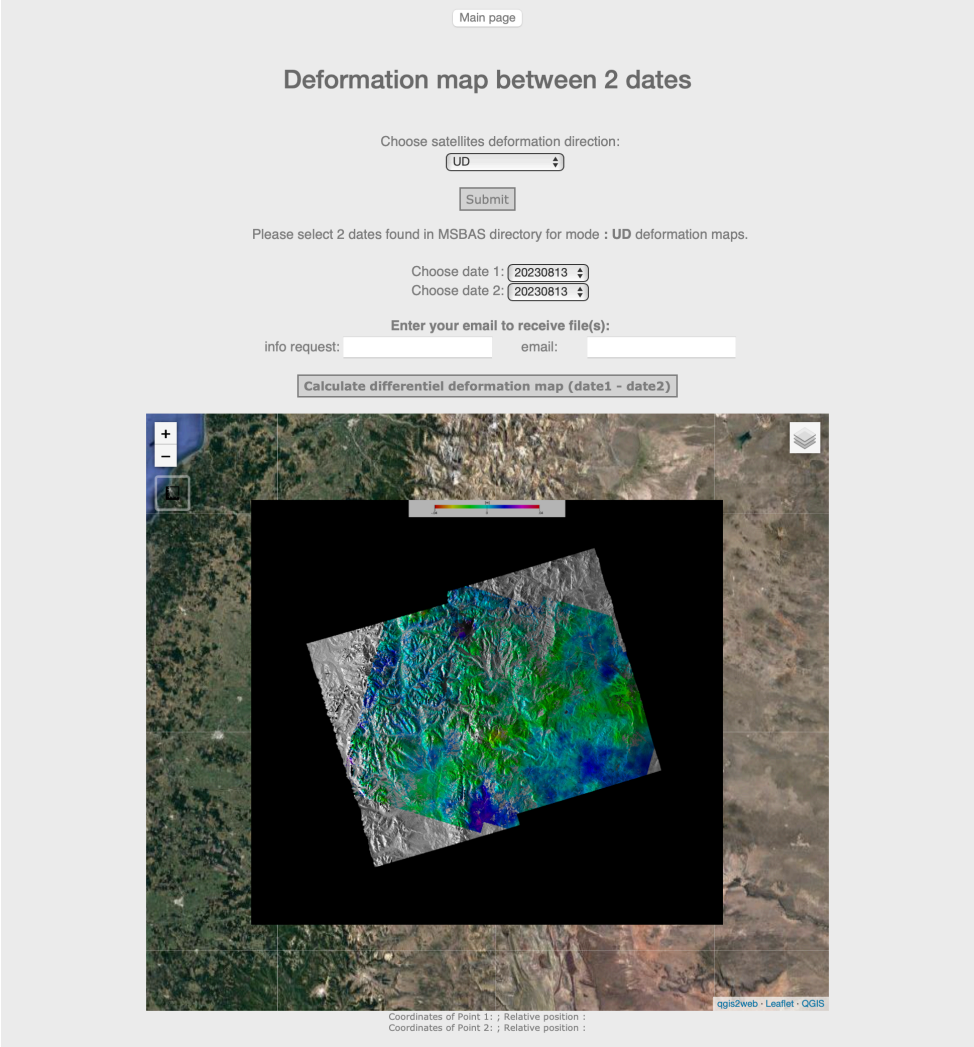


Figure 21: Deformation map on demand.

7.5.h) Display time-baselines plots

By clicking on the button “Baseline Plot” at the top of the page opens a new section that allows to display the baseline plots for each mode (Figure 22): Ascending and Descending orbits, with and without coherence threshold restriction (if applicable).

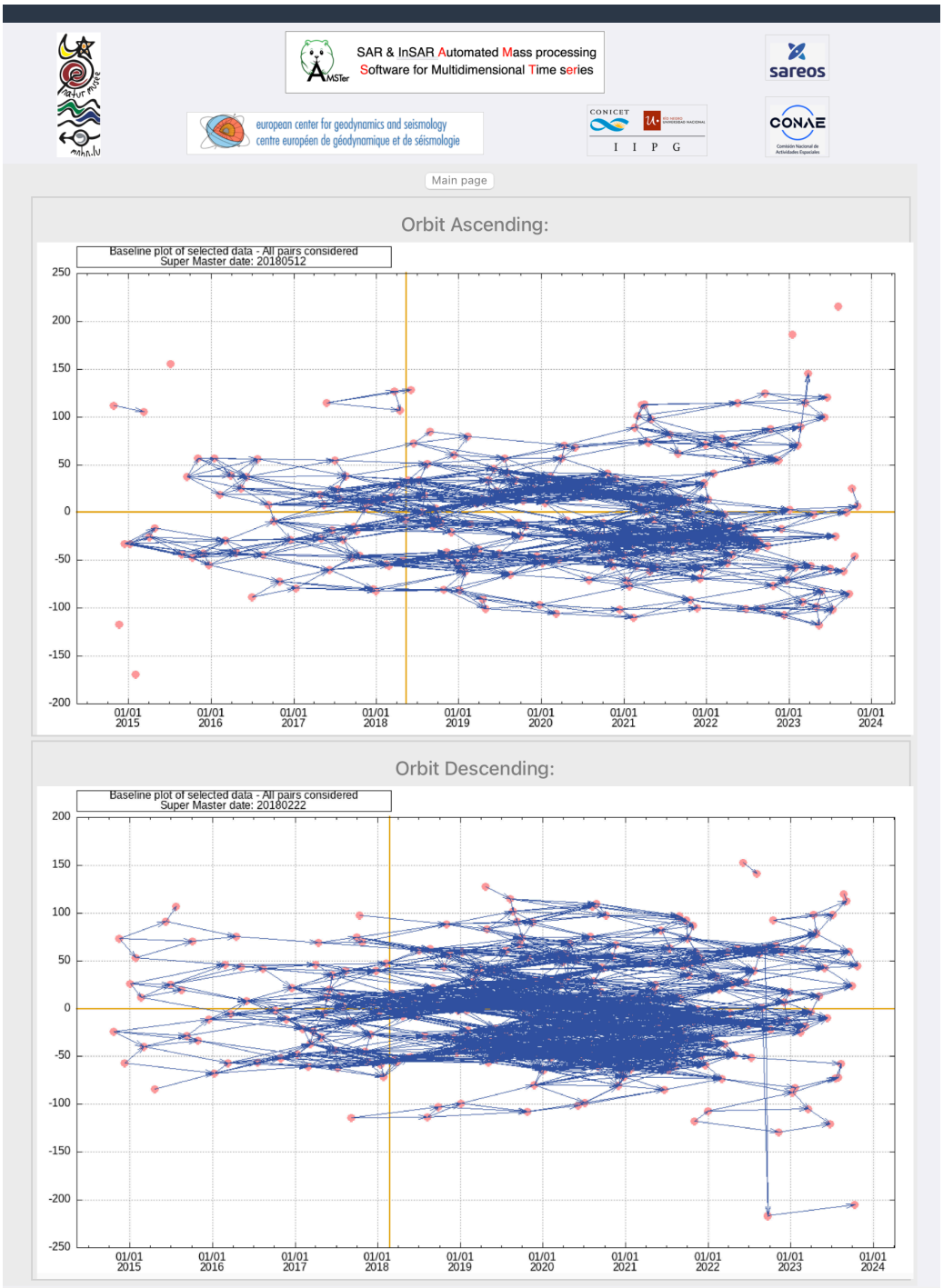


Figure 22: Example of baselines plot for Domuyo volcano region.

7.5.i) Display RGB coded amplitude changes

By clicking on the button “RGB images” at the top of the page opens a new section that allows to create and visualize RGB composition with amplitude of Primary image as Green, amplitude of Secondary image as Red and coherence as Blue.

For details about RGB composition, see *Web_tool_Vxx.pdf* by Maxime Jaspard.

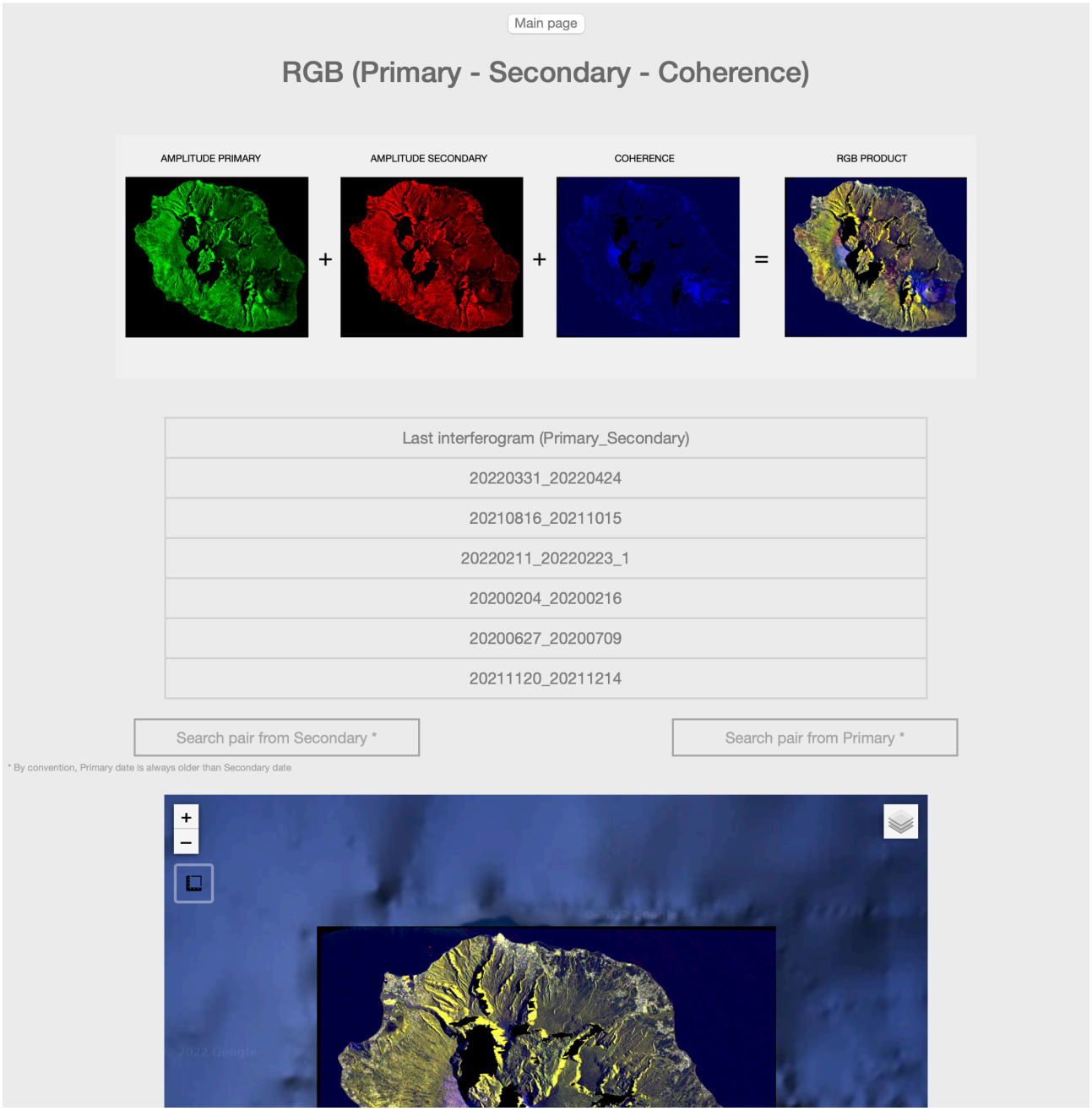


Figure 23: Example of RGB image for Réunion Island.

7.5.j) Display Deformation Linear Rate Standard Deviation

By clicking on the button “Defo linear rate STD” at the top of the page opens a new section that allows to create and visualize maps of Standard Deviation of the Deformation Linear Rate.

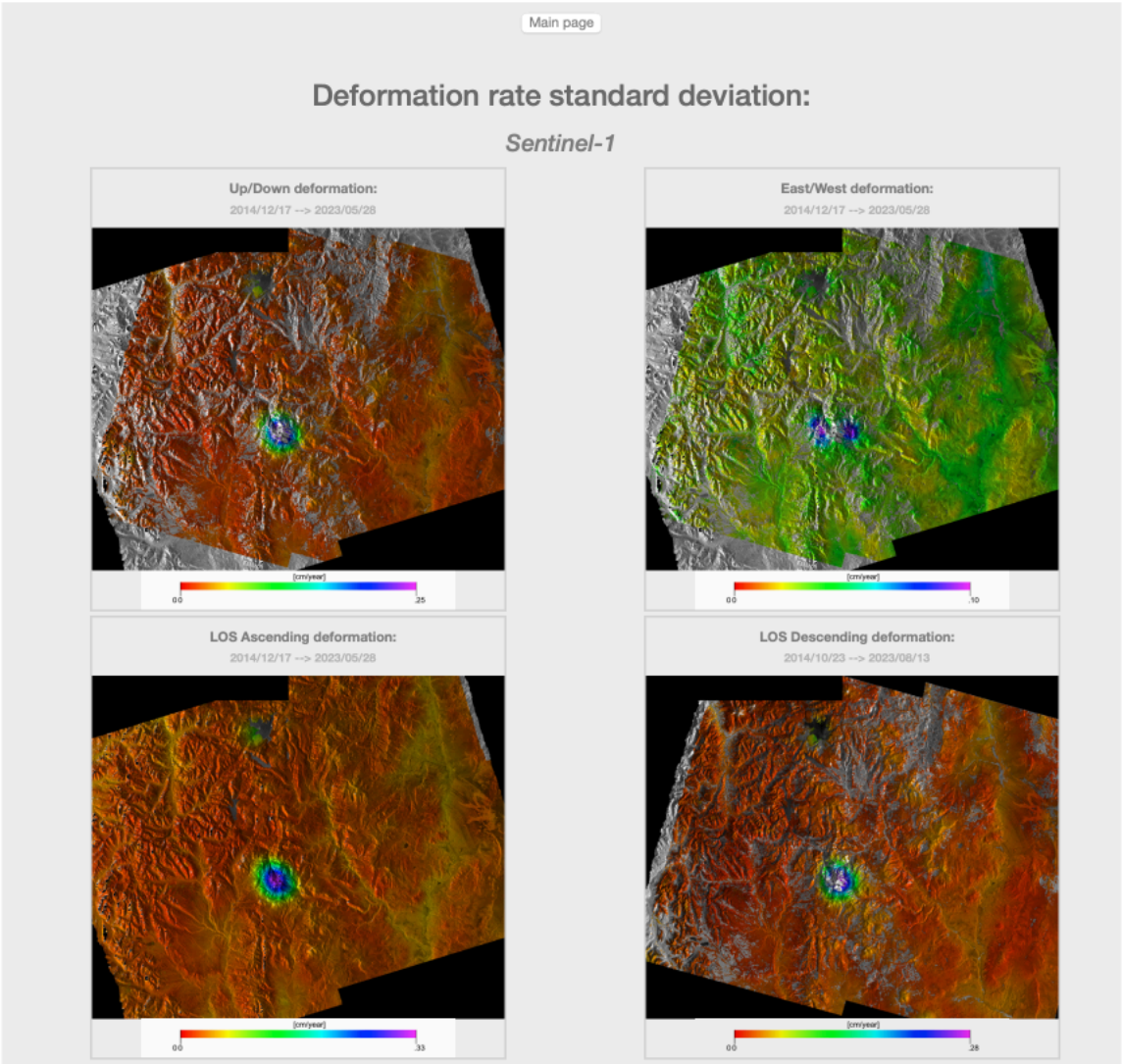


Figure 24: Example of maps of Standard Deviation of the Deformation Linear Rate (Domuyo – Laguna Del Maule region).

8. Useful additional scripts:

The following scripts are also stored in SCRIPTS_MT or zz_Utilities_MT or even zz_Utilities_MT_Ndo. Some haven't been used for a while. In any case, these scripts can be of interest at some point. Use with care. Read carefully at least the header of the scripts. It is strongly advised to check first these scripts before launching them.

8.1) Changing path in Parameters test files:

Because scripts are run on several computers, sometimes operated under different OS, it happens that the external hard drives are mounted differently. As a consequence, the results obtained from the processing performed on these different computers are accompanied by parameters files where path may not be understood by the other computers. The following scripts allow to rename these paths. Note that the details about the renaming is hard coded in ***__HardCodedLines.sh.***

See scripts (in SCRIPTS_MT/zz_Utilities_MT_Ndo) for usage:

- ***RenamePath_Volumes_MNTtoVOL.sh***,
  which transforms paths such as /mnt/1650 in /hp-1650-Data_Share1
- ***RenamePath_Volumes_VARtoMNT.sh***
  which transforms paths such as $PATH_1650 in /mnt/1650
- ***RenamePath_Volumes_VARtoVOL.sh***
  which transforms paths such as $PATH_1650 in /hp-1650-Data_Share1
- ***RenamePathInPlace_Volumes_VOLtoMNT.sh***
  which rename IN PLACE (in various parameters text files) external HD path such as Volumes/hp-1650-Data_Share1 (from OSX) in mnt/1650 (from Linux). This may have an interest when processing data prepared on a computer using another OS

See also (in SCRIPTS_MT):

- ***RenamePath_Volumes.sh***
  which transforms paths such as /mnt/1650 or /hp-1650-Data_Share1 in $PATH_1650
- ***RenamePath_AfterMove.sh***
  which renames path of files in InSARParameters.txt after moving them from where they were computed to where the mass processing wants them to be stored. Need to be run in dir where all /PRM_SCD/i12/TextFiles/InSARParameters.txt were moved, e.g. .../SAR_SM/RESAMPLED/SAT/TRK/CROPDIR/
- ***RenamePathAfterMove_in_SAR_MASSPROC.sh***
  which renames path of files in TextFiles in PAIR DIRS in SAR_MASSPROCESS after having moved them from where they were computed. Need to be run in dir where all …/SAR_MASSPROCESS/.../i12/TextFiles/InSARParameters.txt were moved, e.g. .../SAR_MASSPROCESS/S1/DRC_VVP_A_174/SMNoCrop_SM_20150310_Zoom1_ML8

See also (in SCRIPTS_MT/zz_Utilities_MT):

- ***RenamePathCropSM.sh***
  which rename path of files in InSARParameters.txt that were computed with CSL images stored in dir named by SM instead of normal name (e.g …/SAR_CSL/CSK/Virunga_Desc/SMCrop_SM_20160105_NyigoCrater_-1.510--1.560_29.280-29.330_Zoom1_ML8 instead of …/SAR_CSL/CSK/Virunga_Desc/Crop_NyigoCrater_-1.510--1.560_29.280-29.330_Zoom1_ML8.
  Attention, the input must contain enough info (SAT/TRK) to avoid confusion. Parameters are the beginning of existing dir name and the beginning of wished dir name

- ***UpdateLinkAfterMove.sh***
  Which updates link of S1 resampled files in INPUTDATA/ after moving them from where they were computed to where the mass processing wants them to be stored. Need to be run in dir where all /PRM_SCD/i12/InSARProducts were moved (i.e. OUTPUTDATA), e.g. .../SAR_SM/RESAMPLED/SAT/TRK/CROPDIR/
- ***UpdateLinksInMassProc.sh***
  which updates all path in all type of parameters file (including bestPlaneRemoval.txt) for a given pair. Need to be run in dir PRM_SCD where i12/InSARProducts are stored.

See also (in SCRIPTS_MT/zz_Utilities_MT_Ndo):

- ***RenamePathAfterMove_in_SAR_SM_AMPLITUDES.sh***
  which renames (in parameters text files) the path of files in TextFiles in PAIR DIRS in SAR_SM/AMPLITUDES after having moved them from where they were computed. This may have an interest in case of geocoding in SAR_SM/AMPLITUDES directories.

8.2) Updating links if point toward disks mounted on Mac or Linux:

This script update links that are correct except for the two first directories levels (which are the external mounted disk that we want to change).

Example: suppose that we have links in MSBAS/sat/crop/Defomode/
that points toward
/Volumes/hp-D3601-Data_RAID6/SAR_MASSPROCESS/sat/crop/Geocoded/Defomode
(i.e. computed on Mac), but we need to run it from a computer where $PATH_3601 is /mnt/1650 instead of /Volumes/hp-D3601-Data_RAID6, that in a disk mounted from Linux.

The script will change all the links to that new mount point. It is used for instance in some versions

See script (in SCRIPTS_MT/zz_Utilities_MT_Ndo) for usage:
Update_links_Mac_vs_Linux.sh

8.3) Removing or repairing links:

The script ***Remove_BrokenLinks_and_Clean_txt_file.sh*** will remove broken links from …/MSBAS/*Site*/*MODEi* and clear corresponding text file …/MSBAS/*Site*/*MODEi.txt*

The script ***Remove_BrokenLinks.sh*** only removes broken links in a given directory.

The script ***Rebuild_lns.sh*** search for all files with a name containing a string (parameter 1) in the current directory. For each one, it checks that the file is a link and that it points toward an existing file in a target directory (parameter 2). If it fails but the file exists in that target directory, it will rebuild the link. If not, it will tell the user that the target file does not exist.

The script ***Rebuild_lns_Ampli.sh*** aims at rebuilding links to amplitude files in all pairs directories in current directory (usually SAR_SM/AMPLITUDES/SAT/TRK/REGION/) after the original file were moved to _AMPLI. It also remove possible wrong .ras and .sh links resulting from usage of wrong ***Cp_Ampli.sh*** V2.0.

The script ***Rebuild_Ins_CSK.sh*** recreates the links for CSK images from …/SAR_CSL/CSK/*Region_Mode*/NoCrop to …/SAR_CSL/CSK/*Region*. It takes into account the possible occurrence of multiple images acquired on the same day.

See scripts for usage:
- ***Remove_BrokenLinks_and_Clean_txt_file.sh*** (in SCRIPTS_MT)
- ***Remove_BrokenLinks.sh*** (in SCRIPTS_MT/zz_Utilities_MT)
- ***Rebuild_Ins.sh*** (in SCRIPTS_MT/zz_Utilities_MT)
- ***Rebuild_Ins_Ampli.sh*** (in SCRIPTS_MT/zz_Utilities_MT_Ndo)
- ***Rebuild_Ins_CSK.sh*** (in SCRIPTS_MT/zz_Utilities_MT_Ndo)

8.4) Moving results from SinglePair.sh into SAR_MASSPROCESS or from MassProcess pairs to Geocoded

Use this script if you processed a pair with **SinglePair.sh** and want to add these results to you mass processing directories as if it was computed by a normal ***SuperMaster_MassProcess.sh***, providing that the ***SinglePair.sh*** was performed with geocoding on the SAME Global Primary (SuperMaster) as what already exists.

The script ***Move_SinglePair_Results_To_MASSPROCESS.sh*** renames and move the geocoded results from a ***SinglePair.sh*** processing in the appropriate directories and subdirectories of a mass processing, i.e. in
…/SAR_MASSPROCESS/SAT/REGION/SMCrop_Zoom_ML/Geocoded/…,
…/SAR_MASSPROCESS/SAT/REGION/SMCrop_Zoom_ML/GeocodedRasters/…
and move the directory with the rest of the processing in
…/SAR_MASSPROCESS/SAT/REGION/SMCrop_Zoom_ML/

The script ***Move_MassProcessPair_Results_To_Geocoded.sh*** aims at moving the geocoded results from a ***SuperMaster_MassProc.sh*** and that would be still located in a pair directory, that is SAR_MASSPROCESS/SAT/TRK/SMCrop.../PRM_SCD/i12/GeoProjection to where it was supposed to be, that is
SAR_MASSPROCESS/SAT/TRK/SMCrop.../Geocoded/... directories.
It also copies the corresponding rasters in /GeocodedRasters.

See script for usage:
- ***Move_SinglePair_Results_To_MASSPROCESS.sh***
- ***Move_MassProcessPair_Results_To_Geocoded.sh***
(in SCRIPTS_MT/zz_Utilities_MT_Ndo)

8.5) Reprocessing unwrapping and/or geocoding and/or detrending after MassProcessing:

- For the unwrapping of all the given products from a list of pairs:
  See script for usage: ***ReUnwrap_fromList.sh*** (in SCRIPTS_MT)

  Note: as for the other reprocessing scripts below, it is advised to run first ***RenamePathAfterMove_in_SAR_MASSPROC.sh*** in order to update the path in the parameters file.).

  Pair list must be in the form of
  YYYYMMDD_YYYYMMDD, or
  S1imageDateFormat_ S1imageDateFormat

  After running ***ReUnwrap_fromList.sh*** you can of course run the scripts below to re-process the next steps.

- For the geocoding of all the given products from a list of PAIR DIRS in SAR_MASSPROCESS :
  See script for usage: ***ReGeocode_fromList.sh*** (in SCRIPTS_MT)

- For reprocessing all steps from Detrend (included) till geocoding and renaming:
  See script for usage: ***ReRunFromDetrend.sh***
  (in SCRIPTS_MT/zz_Utilities_MT_Ndo/Need_testing_and_update)

- For reprocessing the interpolating of SAR_MASSPROCESS/Geocoded/DefoInterpolDetrend images and store them in SAR_MASSPROCESS/Geocoded/DefoInterpolx2Detrend:
  See script for usage: ***Interpol_From_Geocoded_DefoInterpolDetrend.sh***
  (in SCRIPTS_MT/zz_Utilities_MT_Ndo)

- For reprocessing the detrending and interpolation of SAR_MASSPROCESS/Geocoded/Defo images and store them in SAR_MASSPROCESS/Geocoded/DefoInterpolDetrend and /DefoInterpolx2Detrend although they will be missing the first interpolation (i.e. before geocoding):
  See script for usage: ***Detrend_From_Geocoded_Defo.sh***
  (in SCRIPTS_MT/zz_Utilities_MT_Ndo)

- For geocoding the snapuZoneMap (obtained with unwrapping the interferogram with the option ZONEMAP set to ZoneMapYes), one must use the script ***GeocSnaphuZoneMap.sh***. It tricks AMSTerEngine by asking to unwrap again the coherence while the coherence file is actually the snaphuZoneMap, which was renamed and transformed into float (using *byte2float.py*).

  There is another version named ***GeocSnaphuZM_and_corr.sh*** which also geocodes the manually corrected unwrapped phase and deformation map (maybe detrended as well). These corrections must be so far performed using Matlab tool such as developed by Delphine Smittarello. Final geocoded products are renamed with long names as all the other geocoded products.

  See scripts for usage (in SCRIPTS_MT/zz_Utilities_MT_Ndo):
  - ***GeocSnaphuZoneMap.sh***
  - ***GeocSnaphuZM_and_corr.sh***

- For geocoding the slantRangemask, one must use the script ***GeocodeMask.sh***. It is supposed to tricks AMSTerEngine by asking to unwrap again the interferogram while it is actually the slantRangemask. Note however that this version is old and MUST be revised (see for instance errors in float to byte conversion), but it must be straight forward. See script in SCRIPTS_MT/zz_Utilities_MT_Ndo.

- For geocoding a series of files (e.g. time series of amplitudes to geocode after a pixel tracking processing), one must use the script ***ReGeocode_AmpliSeries.sh***. The script must be launched from the pair directory where at least one of the original products was computed. It expects all the files to (re-)geocode to be in /InSARProducts and have the same size as the original products. The name of all the files to geocode MUST start by the string REGEOC. It will geocode all and only these files and store the results in /GeoProjection directory as usual and create raster quick looks. It will perform the geocoding based on the criteria provided in the LaunchMTparam.txt file as usual. See script for usage (in SCRIPTS_MT/zz_Utilities_MT).

8.6) Reprocessing unwrapping and/or geocoding and/or detrending after SinglePair.sh:

- For unwrapping again an interferogram computed using the ***SinglePair.sh*** processing, and interpolate and detrend again the results if required, see script ***ReUnwrap_SinglePair.sh***.

  Former unwrapped phase is saved as unwrappedPhase_*LongName*.bak.
  If you have enough room on your disk, it might be wise to duplicate the directory where you will re-process the unwrapping(s).

  From there, you can then run a ***ReGeocode_SinglePair.sh*** if required (see below).

  Because this script will need information from several text files containing the information about the processing, it requires to be run in the same directory and with the same type of computer (Mac or Linux) as the one where ***SinglePair.sh*** was performed.

  Note that this procedure was only tested for S1 so far, and not in STRIPMAP !!!!!

- For geocoding again the products computed using the ***SinglePair.sh*** processing, and interpolate again the deformation maps if required, see script ***ReGeocode_SinglePair.sh***.

  This script must be run in the directory where ***SinglePair.sh*** was run. It takes as parameter the type of interpolation: "BEFORE", "AFTER", "BOTH" or "NONE".

  It contains a hard-coded list of products to geocode. Note that it skips the geocoding of amplitude if it exists. You can however force it again by editing the script (around lines 177-186).
  Note that Incidence angle map is geocoded by default if present in InSARProducts. If you do not want to re-geocode it, it must be renamed temporarily. For this, let the script know by adding an additional *NO* at the hardcoded list of geocoded files.

  Note: only verified for interpolation BOTH and with detrend !
  The script was only tested for S1 so far and not in STRIPMAP !!!!!

- For geocoding the unwrapped phase resulting from a ***SinglePair.sh*** processing that was unwrapped again manually e.g. by iterative procedure (such as proposed e.g. by J.L. Froger and Y. Fukujima), see script ***ReGeocode_ManuallyUnwrapped_SinglePair.sh***. It will also detrend and/or interpolate the deformation map if required. This script must be run in the directory where ***SinglePair.sh*** was run.

  The script expects:
  - in /InSARProducts, the Unwrapped phase (e.g. from MATLAB or python code) named as unwrappedPhase (without polarizations). *InSARParameters.txt* must be updated accordingly.
  - the number of lines and columns of Unwrapped file issued by the MATLAB (or python) code IS ALWAYS even. If these numbers in the original file were not the even, InSARParameters.txt must be updated accordingly.

  Note that before reprocessing the phase-to-height conversion and geocoding, the former results are moved in a directory one level above /InSARProducts and /GeoProjection named /ORIGINAL. The script reads again the LaunchMTparam.txt file to assess how to re-geocode. Size of pixel and/or method of geocoding can hence be changed. It can also geocode it forced on a kml file.

  Note: only verified for interpolation BOTH and with detrend !
  The script was only tested for S1 so far and not in STRIPMAP !!!!!

  See scripts for usage (in SCRIPTS_MT/):
  - ***ReUnwrap_SinglePair.sh***
  - ***ReGeocode_SinglePair.sh***
  - ***ReGeocode_ManuallyUnwrapped_SinglePair.sh***

8.6) Detecting and cleaning duplicate Geocoded and GeocodedRasters products:

Duplicate products may happen in /Geocoded/*Mode* and /GeocodedRasters/*Mode* (where *Mode* is e.g. DefoInterpolx2Detrend) when a pair is reprocessed with updated orbit with resulting slightly different Bp for instance.

Remove_Duplicate_Pairs_File.sh looks for duplicated products in directories based on pair of dates that is in the name. It then offers to interactively delete the oldest version of duplicated files, or delete them without warning if you are brave enough (USE WITH CARE), or safer, to move them in SAR_MASSPROCESS/…/Geocoded/*Mode*/___Duplicated_ToKill/.
The script must be launched in the Mode directory that need to be cleaned, i.e. SAR_MASSPROCESS/…/Geocoded/*Mode*

Remove_Duplicate_Pairs_File_ras.sh looks for duplicated raster products in directories based on pair of dates that is in the name. It then offers to interactively delete the oldest version of duplicated files, or delete them without warning if you are brave enough (USE WITH CARE), or safer, to move them in SAR_MASSPROCESS/…/GeocodedRasters/*Mode*/___Duplicated_ToKill/.
The script must be launched in the Mode directory that need to be cleaned, i.e. SAR_MASSPROCESS/…/ GeocodedRasters/*Mode*

Note that ***Remove_Duplicate_Pairs_File.sh*** and ***Remove_Duplicate_Pairs_File_ras.sh*** do not allow to clean Ampli mode because Primary and Secondary images can be geocoded within the same pair.

Remove_Duplicate_Pairs_File_All_Modes_But_Ampl.sh look for duplicated products in all Mode directories (but Ampli) based on pair of dates that is in the name (i.e. /Geocoded/*Mode* and /GeocodedRasters/*Mode)*. It then moves them in ___Duplicated_ToKill in each corresponding *Mode*.

_Remove_All_Files_With_PairDate_Bp.sh deletes all files (deg, deg.hdr and ras) in MSBAS/*region*/*DefoInterpolDetrendi*(*_Full*) and its texts and in /Geocoded/*Mode* and /GeocodedRasters/*Mode* for given pairs of date and for a given Bp. DO NOT USE IT ON GEOCODED AMPLITUDE IMAGES because PRM and SCD images can be geocoded within the same pair.

See scripts for usage: ***Remove_Duplicate_Pairs_File.sh***
Remove_Duplicate_Pairs_File_ras.sh
Remove_Duplicate_Pairs_File_All_Modes_But_Ampl.sh
_Remove_All_Files_With_PairDate_Bp.sh
(in SCRIPTS_MT/zz_Utilities_MT_Ndo)

Note:
there are also old scripts ***Check_Duplicate_Geocoded.sh*** and***CompareGeocodedDates.sh*** (in SCRIPTS_MT/zz_Utilities_MT_Ndo).
Not sure what the first one is doing (must be verified…) but the second one checks in each Mode (but the Ampli) that files exist in /Geocoded and /GeocodedRasters with the same dates. It may not be with the same Bt, Bp, ha etc… though. Those two are old stuffs and must be taken with care.

See also ***Verify_MassProcess_Results.sh*** and ***Verify_Geocoded_versus_Rasters.sh*** (in SCRIPTS_MT) at chapter 5.6.

8.7) Checking DefoInterpolx2Detrend.txt:

The script ***_Check_bad_DefoInterpolx2Detrend.sh*** look for bad file names (pointing toward inexistent files) in DefoInterpolx2Detrend.txt prepared for MSBAS processing. This may result from crashes in MSBAS preparation or cleaning from update of S1 orbits.

It also checks for duplicated lines in DefoInterpolx2Detrend.txt. Indeed, it may happen that after updating Primary and/or Secondary orbits of S1 data, file name might be the same but col 2 (Bp) might differ while it provides the value with more digits.

See script for usage (in SCRIPTS_MT): ***_Check_bad_DefoInterpolx2Detrend.sh***

8.8) Creating a table with PRM SCD Bp Bt Coh from files in /Geocoded

The script ***Baseline_Coh_Table.sh*** is aiming at making a table PRM SCD Bp Bt Coh from all the files that are in ../SAR_MASSPROCESS/../Geocoded/Coh
named Baseline_Coh_Table_*KMLNAME*.txt and where *KMLNAME* is the name of the kml file defining the footprint where to compute the average coherence (Coh).

See script for usage (in SCRIPTS_MT/zz_Utilities_MT):
Baseline_Coh_Table.sh

8.9) Selecting the pairs such as each image is used a maximum of 3 times as Primary and as a Secondary:

See also ***Stat_Mas_Slv.sh*** to assess how many times an image is taken as a Primary and as a Secondary.

When all pairs are selected to be taken max 3 times as Primary and Secondary, you can run ***Keep_Pairs_From_Extract_Baseline_3.sh*** to limit the pairs in …/MASBAS/…/*Modei*.txt to only those taken max 3 times.

See scripts for usage (in SCRIPTS_MT/zz_Utilities_MT):
- ***Extract_Baselines_3.sh***
- ***Keep_Pairs_From_Extract_Baseline_3.sh***

8.10) Tools to select Bt and Bp:

The python script ***Analyse_BaselineCoh_data.py*** takes as input a table of mean coherence computed on a kml (created with ***Baseline_Coh_Table.sh***) and plot several figures to show how coherence changes with respect Bt, Bp and acquisition dates.

The python script ***Analyse_AllPairslisting_data.py*** takes as input a table AllPairslisting.txt (created with ***Prepa_MSBAS.sh***) as well as a *BpMax* and *Btmax*. It creates several figures showing the characteristics of the selected subgraph based on the given baselines. It also displays at the terminal:
- some information about the images that were not selected and the possible additional pairs with these images;
- some statistics.

See scripts for usage (in SCRIPTS_MT/zz_Utilities_MT):
- ***Analyse_BaselineCoh_data.py***
- ***Analyse_AllPairslisting_data.py***

8.11) Testing phase closure:

The objective is to search for possible interferograms affected by unwrapping error. This is done by checking phase closure consistency between triangles of pairs: for each triplet of interferograms spanning 3 dates (says date1-date2, date2-date3 and date1-date3), the mean unwrapped phase Φ_{ij} (where *i* and *j* are the index of the date) is computed over an area defined by a user-provided kml file.

If $| \Phi_{13} - (\Phi_{12} + \Phi_{23}) |$ is larger than a chosen threshold, the triplet of interferograms are flagged as affected by possible unwrapping errors. Comparing all the flagged triplets allows identifying the faulty interferograms, which must be manually checked and/or removed from the database.

One must first run ***Extract_Triangles.sh*** to list all the triangles from the list of pairs such as $PATH_1650/SAR_SM/MSBAS/*Region*/set*i*/table_*MinBp_MaxBp_MinBt_MaxBt*.txt, which is computed by ***Prepa_MSBAS.sh.*** That list is named *List_Triangels.txt*. The script also lists the pairs that are not used in triangles (*List_No_Triangels.txt*).

One should then run the script ***Check_Closure_All_Triangles.sh*** which is aiming at checking unwrapping error in all triangles from *List_Triangels.txt*. It must be launched with 4 parameters:

- the file with list of triangles (*List_Triangels.txt*)
- the path to deformation files:
  (e.g. .../SAR_MASSPROCESS/*SAT*/*TRK*/SM*Crop*_SM_*DATE_Zoom_ML*/Geocoded/*Mode*)
- the path to a kml of zone where to check the mean phase
- a threshold above which one consider that there is a phase error

It will output 3 files: *_Good_Closure.txt*, *_Wrong_Closure.txt* and *_Pairs_To_Clean_From_WrongClosure_NotIn_GoodClosure.txt*

Wrong pairs in *_Pairs_To_Clean_From_WrongClosure_NotIn_GoodClosure.txt* can be rejected from MSBAS by running ***Remove_Pairs_From_BaselineOptimisation.sh***.

Note that there is another script that checks the phase closure of only a single triangle (***Check_Closure_Triangle.sh***).

See scripts (in SCRIPTS_MT/zz_Utilities_MT) for usage:

- ***Check_Closure_All_Triangles.sh,***
- ***Extract_Triangles.sh*** and
- ***Check_Closure_Triangle.sh***

8.12) Remove lines in DefoInterpolDetrendi.txt or in table_0_Bp_0_Bt.txt from a list of pairs:

The script ***Remove_Pairs_From_BaselinePlotOptimisation.sh*** deletes all lines in MSBAS/region/DefoInterpolDetrend*i*.txt or in table_0_*Bp*_0_*Bt*.txt that contain pairs computed from baseline plot optimization (provided as list of *PRMdate_SCDdate*). It keeps the links in MSBAS/region/DefoInterpolDetrend*i*. It also saves uncleaned DefoInterpolDetrendi.txt as DefoInterpolDetrendi_NotOptimized.txt.

See scripts (in SCRIPTS_MT/zz_Utilities_MT_Ndo) for usage:
Remove_Pairs_From_BaselinePlotOptimisation.sh

8.13) Testing if images have values at given places:

The script ***Check_Bursts_Results.sh*** is aiming at testing if a set of all the images from the current directory (e.g. from MSBAS results, that is list of cumulated deformation maps in the form of *MSBAS_YYYYMMDDThhmmss...*), or list of other type of data such as amplitudes (and hence of other format or naming, thought they must be all of the same size, e.g. in geocoded envi format) have valid values at given places.

The places to be tested are provided as a list of coordinates in *LINES* and *PIXELS* in a file named *BurstMap_REGION_TRK.txt* (e.g. one point per burst if one wants to check if all the bursts are present – ensure that you select your points in coherent zones if you test deformation maps). Note: the file *BurstMap_REGION_TRK.txt* MUST end with one and only one carriage return.

Remember that you can use “Value Tool” plugin in QGIS to see the coordinate in LINES and PIXELS from a geocoded image. Remember also that burst KMLs can be found in each SAR_CSL/SAT/TRK/NoCrop/Img.csl/Info/PerBurstInfo directory.

The script will then run a ***getLineThroughStack*** for each point and output a file with the values at each image:

- If the script is used for MSBAS results, the output files *timeLine${COLX}_${LINY}.txt* will contain "*dates times values*" in columns.
- If the script is used for checking other type of data (such as amplitudes...), the output files *timeLine${COLX}_${LINY}.txt* will contain "*name_of_the_files values*" in columns.

If there is a **NaN** in the time series, or if all the values are null, the script outputs a notification.

The script must be launched in the directory where all envi files are, e.g.:
- SAR_MASSPROCESS/*SAT/REGION_TRK*/SM_ZOOM_ML/Geocoded/Ampli, or
- SAR_MASSPROCESS/*SAT/REGION_TRK*/SM_ZOOM_ML/Geocoded/DefoInterpolx2Detrend, or
- MSBAS/*REGION*/DefoInterpolx2Detrend

Parameters:
- path to *BurstMap_REGION_TRK.txt* : i.e. to an ascii file that may contain a header "Lin(Y) Col(X) Swath,Burst" and contains the list of points stored as lines and columns separated by a space and followed by BurstPosition as Swath,Burst.
  Notes:
  + regions that are overlapped by two bursts can be tested as well and can be noted by their Swath,Burst+Burst numbers
  + if some regions are depicted by the same burst but holding two different names, it can be noted as Swath,Burst-SwathBurst
- Path to the directory with envi files to check
- KEEPFILES if you want to keep the *timeLine${COLX}_${LINY}.txt* files

8.14) Testing if interferograms are empty in given zone:

The script ***Check_Interfero_Not_Empty_In_Zone.sh*** is aiming at removing from a prepared msbas data set (eg DefoInterpolx2Detrend2) all the images that would be empty on a selected zone provided by a kml. It is intended to run after ***build_header_msbas_criteria.sh*** or after ***build_header_msbas_criteria_From_nvi_name_WithoutAcqTime.sh*** and before running ***MSBAS.sh***.

Run this for each mode that need to be cleaned.

Select the kml zone where you know signal must not be zero.

The script must be launched in the directory where msbas will be run, which contains all the Mode*i* and Mode*i*.txt.

Parameters are:
- mode to clean; beware of index at the end of name (eg DefoInterpolx2Detrend2)
- kml of zone where to test the validity
- path to defo (from mode) files:
  SAR_MASSPROCESS/SAT/TRK/REGION_ML/Geocoded/DefoInterpolx2Detrend

A version of the script does the same things although without deleting the files where the zone is empty. It only output at terminal the list of wrong images.

See scripts for usage (in SCRIPTS_MT/zz_Utilities_MT):
- ***Check_Interfero_Not_Empty_In_Zone.sh***
- ***Check_Interfero_Not_Empty_In_Zone_TestWithoutDel.sh***

8.15) Interpolating times series of deformation maps for msbas

This is not a script but a function developed by Ludivine Libert named *interpolateDataSet*. See sources and make your own script if needed. It is not used so far in the mass processing.

8.16) Filtering deformation maps:

Additional filtering may be computed using GMT for instance
(see scripts in SCRIPTS_MT/zz_Utilities_MT):
- ***Filter_Interpolate_DefoMaps.sh***
- ***Filter_Interpolate_SingleMap.sh***

or using python *FiltMedian.py*

8.17) Creating gif from all Geocoded Amplitudes or deformation or r4 files in a directory:

The script ***AmpliGeocRas2GIF.sh*** is aiming at creating a gif from all geocoded amplitude rasters that are in SAR_MASSPROCESS/GeocodedRasters/Ampli. It must be launched from the directory where the amplitude files are stored.

Parameters:
- X and Y coord of date tag position
- font size for date tag
- output images resolution
- output directory where to store results

The script ***TSmap2moviegif.sh*** is aiming at converting binary deformation maps from MSBAS processing to jpg then combine all jpg into a gif movie.

Parameter: - path to directory where MSBAS_????????T??????_*.bin and .hdr are.

The script ***r4togif.sh*** is aiming at converting all r4 (float32; eg from MATLAB post processing) files from the current directory to jpg (cropped to dimensions hard coded in script) and tagged with date (in position hard coded in script), then combine all jpg into a gif movie.

Parameter: - width of r4 files (in pixels)

See scripts for usage:

AmpliGeocRas2GIF.sh	(in SCRIPTS_MT/zz_Utilities_MT)
TSmap2moviegif.sh	(in SCRIPTS_MT/zz_Utilities_MT)
r4togif.sh	(in SCRIPTS_MT/zz_Utilities_MT_Ndo)

8.18) Creating kmz file from Geotiff or Envi files:

The script ***Geotif2kmz.sh*** creates a kmz from a Geotiff file.

The script ***Envi2kmz.sh*** creates a kmz from an Envi format file. Note however that colour table is not set automatically yet; rather, it uses SCRIPTS_MT/TemplatesForPlots/*ColorTableGDAL.txt*. Must be done when time… or see ***Envi2ColorKmz.sh***, which uses SCRIPTS_MT/TemplatesForPlots/*ColorTableKMZ.txt* or SCRIPTS_MT/TemplatesForPlots/*ColorTableKMZ_2.txt*.

See scripts for usage:
- ***Geotif2kmz.sh*** (in SCRIPTS_MT/zz_Utilities_MT)
- ***Envi2kmz.sh*** (in SCRIPTS_MT/zz_Utilities_MT)
- ***Envi2ColorKmz.sh*** (in SCRIPTS_MT)

8.19) Creating black and white kmz figure from amplitude (or other) file in Envi format:

Script to create figures of amplitude (or what you want) in black and white kmz, based on Envi image: ***Envi2BlAndWhKmz.sh***
Pretty useful to see image in Google Earth for instance.

See script for usage (in SCRIPTS_MT): ***Envi2BlAndWhKmz.sh***

8.20) Creating a X/Y scatter plot between DEM and deformation map (or linear rate map) and compute linear regression:

Python script to create a scatter plot between a DEM and a deformation (or a linear rate map) and compute linear regression e.g. to detect possible atmospheric artefacts or topographic residuals: ***RegLin_DEM_Defo.py***

DEM (in meters) and deformation map (in meters) must be of the same size and may contain NaN. The plot reads the deformation in “mm” and the slope of the linear regression between the DEM and the deformation in “mm/m” because input deformation map is multiplied by 1000 in the scripts. Note that if a linear velocity is provided instead of a deformation, you should change in the script the label of the Y axe and the units of the slope in the title.

See script for usage (in SCRIPTS_MT/zz_Utilities_MT): ***RegLin_DEM_Defo.py***

8.21) Creating figures:

Using ImageJ (Fiji), the script ***Fiji_Amp_Defo_Coh.sh*** creates RGB figures of deformation wrapped on amplitude and damped with coherence, based on Envi images and using Fiji.
The script ***Fiji_DEM_Defo_Coh.sh*** creates RGB figures of deformation wrapped on DEM amplitude and damped with coherence, based on Envi images and using Fiji.
All these files are from any pair processing.

Another version exists also as ***AmpDefo_map.sh***. This uses the deformation maps from the MSBAS inversion.

Similarly, ***AmpAmpAmp.sh*** and ***AmpAmpCoh.sh*** allow to create RGB composition using respectively 3 amplitude images and 2 amplitudes and one coherence image. They will highlight changes that occurred during the dates of the amplitude files. These are basic scripts that can be improved. See scripts.

Using GMT, scripts such as ***plotall_gmt5_funuML4.sh*** call ***plot_gmt5.sh*** to create eps figures. These scripts are based on Sergey Samsonov's scripts.

The script ***Rebuild_ras_and_jpg_ampli_figs_in_all_dirs.sh*** aims at rebuilding amplitude ras and jpg figures from envi files in all pairs directories in current directory (usually SAR_SM/AMPLITUDES/SAT/TRK/REGION/). It also adds the date as label in the jpg figure at provided position. See script.

See scripts for usage:
- ***Fiji_Amp_Defo_Coh.sh*** (in SCRIPTS_MT)
- ***Fiji_DEM_Defo_Coh.sh*** (in SCRIPTS_MT)
- ***AmpDefo_map.sh*** (in SCRIPTS_MT)
- ***AmpAmpAmp.sh*** (in SCRIPTS_MT)
- ***AmpAmpCoh.sh*** (in SCRIPTS_MT)
- ***plotall_gmt5_funuML4.sh*** (in SCRIPTS_MT/zz_Utilities_MT)
- ***plot_gmt5.sh*** (in SCRIPTS_MT/zz_Utilities_MT)
- ***Rebuild_ras_and_jpg_ampli_figs_in_all_dirs.sh*** (in SCRIPTS_MT/zz_Utilities_MT_Ndo)

8.22) Transform all sun raster files from directory in png

The script ***Ras2Png.sh*** transforms all the sun raster files (*.ras) from the current directory into png files. It is parallelized and incremental. Size of png file is smaller than ras.

See scripts for usage (in SCRIPTS_MT_Ndo):
- ***Ras2Png.sh***

8.23) Opening Envi image with Fiji and save it as tif format:

Script to open Envi image (i.e. with its hdr) with Fiji and save it in tif format:
Fiji_Open_Envi2tif.sh

See script for usage (in SCRIPTS_MT): ***Fiji_Open_Envi2tif.sh***

8.24) Sorting results from msbasv3 processing when used for 3D, and make some plots:

Script to plot sort the results from a msbasv3 processing and make some plots: ***MSBASV3_results_plots.sh***

It also takes care of possible byte inversion problem if images were processed on a different hardware.

See script for usage (in SCRIPTS_MT): ***MSBASV3_results_plots.sh***

8.25) Plotting time series and times series with GPS data

Sergey Samsonov suggested gnuplot scripts to combine msbas time series and GPS data. See for instance something like ***PlotEW_UP_ts_GPS_BUK.sh*** (in SCRIPTS_MT/zz_Utilities_MT).

8.26) Extracting time series values of several pixels from a stack of ground deformation maps

The script ***Extract_Pixels_TimeSeries_Values.sh*** aims at running a series of ***PlotTS.sh*** to extract the values of time series from a list of pixels. The list, provided as a parameter, is in the form of *LINE COL* nr. The script may run several processes in parallel on a fixed number of CPU's (*MAXCPU*; hard coded in script), or max CPU-1 if less than *MAXCPU* are available.
However, experience showed that it might actually be slower while run in parallel. Keep *MAXCPU* =1 or test on your computer and target first if you have intensive computation to run.

The script also removes the eps figure. If you want to keep it, comment the line in script below.
It will then store all the time series values timeLine*LINE_COL*.txt in a dir SVD.
Ensure that there is no unwanted timeLine*LINE_COL*.txt list already present in the directory.

NOTE: Must be run where all the deformation maps are located, i.e.
/MSBAS/REGION/zz_*LOS*or*UD*or*EW*_...

See script for usage:
Extract_Pixels_TimeSeries_Values.sh (in SCRIPTS_MT/zz_Utilities_MT_Ndo)

8.27) Plotting difference between two time series e.g. computed with different msbas inversion parameters:

If you process the same data set with different tuning of your parameters (either at the ***SuperMaster_MassProc.sh*** or the ***MSBAS.sh***), you may want to look at the difference between the results. The difference between the deformation or velocity maps are easy with *Fiji* or any GIS tool.

Plot_Diff_TS.sh allows to make the difference between the time series of two pixels.

- First, compute the time series of a given pixel in your first MSBAS process directory (using ***PlotTS.sh*** for instance).
- Second, do the same for the same pixel in the second MSBAS process directory.
- Then, preferably in a third directory dedicated to the comparison of the two data processings, launch ***Plot_Diff_TS.sh*** with the path to the two timeseries as parameters to get the plot of the difference.

See also ***CompareTimeSeries.sh*** which calls ***Plot_Diff_TS.sh*** with the coordinates of a pixel to automatically create the plot. Some hard-coded lines are required though for locating the directories.

You can also use this for making a double difference between two pixels from the same processing, but then it is the same as using ***PlotTS.sh*** with two pixels as input…

See scripts for usage:
- ***Plot_Diff_TS.sh*** (in SCRIPTS_MT)
- ***CompareTimeSeries.sh*** (in SCRIPTS_MT/zz_Utilities_MT)

See the gnu templates (in SCRIPTS_MT/TemplatesForPlots/) used by ***Plot_Diff_TS.sh***, ***PlotTS.sh*** or ***PlotTS_all_comp.sh***, that is:
- *plotTS_template_fit.gnu*,
- *plotTS_template_multi_fit.gnu*,
- *plotTS_template_multi.gnu* and
- *plotTS_template.gnu*.

Modify these templates e.g. if you want add tags, change the colour or fonts etc…

You may also want to have a look at the template in SCRIPTS_MT/TemplatesForPlots/:
plotTS_808_605_830_605_multi_Auto_2_0.04_VVP_WithInset.gnu
which contains some features for inserting vertical bars or rectangles.

8.28) Removing all pairs names from a list of images pairs when the Primary or the Secondary is before or after a given date, or if they are between or outside a date interval:

The scripts ***RemovePairsFromFlist_WithImagesBefore.sh*** and ***RemovePairsFromFlist_WithImagesAfter.sh*** aim at removing from a list of interferometric pairs those which have the Primary or the Secondary acquired before/after a given date.

The list of pairs must be like those created using ***Prepa_MSBAS.sh***, that is with a 2 lines header followed by lines containing 4 columns: *yyyymmdd yyyymmdd (-)val (-)val*. It must be provided with its full path.

The script will output a list of cleaned pairs named as follow
*PAIRFILE*_Above*MAXDATE*_NoBaselines_*RNDM*.txt or
*PAIRFILE*_Below*MAXDATE*_NoBaselines_*PROCDATE*.txt
Where *PAIRFILE* is the original file (with path) containing the list of pairs, *MAXDATE* is the threshold date and *PROCDATE* is the date of processing (to avoid possible clash in naming).

The script ***RemovePairsFromModeList_WithImagesAfter.sh*** does the same although the provided list is the text file with the list of images to be inverted by msbas, that is …/MSBAS/*Region/Modei*.txt (e.g. DefoInterpolx2Detrendi.txt) which contains 4 columns (*Modei/PairName.xdeg Bp yyyymmdd yyyymmdd*) without header. ***RemovePairsFromModeList_WithImagesBefore.sh*** obviously do the same for pairs before the given date.

The script ***RemovePairsFromTableList_Between_dates.sh*** aims at removing all pairs of images that contains at least one image between two dates provided as parameters. It will then store the results in a file named ${PAIRFILE}_Without_${MINDATE}_${MAXDATE}.txt that contains all pairs from the input file except those with at least one images in the interval]${MINDATE},${MAXDATE}[. This file is a 4 columns file with header, as the input file.

Similarly, the script ***RemovePairsFromTableList_Outside_dates.sh*** aims at removing all pairs of images that does not contains at least one image between two dates provided as parameters. It will then store the results in a file named
${PAIRFILE}_Between_${MINDATE}_${MAXDATE}.txt

that contains all pairs with at least one images in the interval [${MINDATE},${MAXDATE}]. This file is a 4 columns file with header, as the input file. It also output the results in a file named
_EXCLUDE_PAIRS_ALTHOUGH_CRITERIA_OK.txt
that contains only one column without header in the form of *datePRM_dateSCD.*
Remember that file _EXCLUDE_PAIRS_ALTHOUGH_CRITERIA_OK.txt must be put in the MSBAS/*REGION*/Mode*i* directory, then when ***Exclude_Pairs_From_Mode.txt.sh*** is run, these pairs will be excluded from the msbas inversion.

See scripts for usage (in SCRIPTS_MT):
- ***RemovePairsFromFlist_WithImagesBefore.sh*** and
- ***RemovePairsFromFlist_WithImagesAfter.sh*** or
- ***RemovePairsFromModeList_WithImagesAfter.sh***
- ***RemovePairsFromTableList_Between_dates.sh***
- ***RemovePairsFromTableList_Outside_dates.sh***

8.29) Deleting directories of interferometric pairs:

The following scripts delete directories named by Primary_Secondary dates:
- ***Del_All_dir_from_list.sh***: remove from the current directory (or move them to .../Prblm; see code) all the directories that are listed in a provided file and that contains only the date name of the Primary and Secondary images separated by an underscore (PRM_SCD)
- ***Del_All_dir_Named_With.sh*** remove from the current directory (or move them to …/Prblm; see code) all the directories (detected with "ls") that contains a certain string in their name. That string is provided as a parameter in the form of two elements separated by an underscore, that is *string1_string2* (i.e. usually PRM_SCD)

See scripts for usage (in SCRIPTS_MT/zz_Utilities_MT):
- ***Del_All_dir_from_list.sh***
- ***Del_All_dir_Named_With.sh***

8.30) Deleting files based on their naming:

The following scripts delete files:
- ***Del_All_files_in_Dirs_Named_With.sh*** remove all files containing a given string (parameter 1) in a given directory (parameter 2).
- ***Del_Geocoded_and_GeocodedRasters_from_DateList.sh*** remove in /Geocoded and /GeocodedRasters all the files that contains strings contained in a list provided as parameter. These strings are usually pairs of dates such as PRM_SCD.
- ***Del_Geocoded_and_GeocodedRasters_from_FileName.sh*** remove all files in /Geocoded and /GeocodedRasters if their name contains a provided string as parameter. The criteria to search for the files to delete is usually something such as:
S1_DRC_VVP_A_174-37.0deg_20190717_20191108_Bp-27.3m_HA518.9m_BT114days_Head102.1deg
- ***Del_Geocoded_Files_Wrong_Bt.sh*** remove all files in /Geocoded and /GeocodedRasters that have a Bt larger than a value provided in parameter. This was needed after a bug was detected in AMSTerEngine computation of Bt. The script must be launched in the SAR_MASSPROCESS directory where /Geocoded and /GeocodedRasters are.

See scripts for usage (in SCRIPTS_MT/zz_Utilities_MT):
- ***Del_All_files_in_Dirs_Named_With.sh***

- ***Del_Geocoded_and_GeocodedRasters_from_DateList.sh***
- ***Del_Geocoded_and_GeocodedRasters_from_FileName.sh***
- ***Del_Geocoded_Files_Wrong_Bt.sh***

8.31) FLIP or FLOP products:

These are python tools used for instance in ***SinglePair.sh, SinglePairNoUnwrap.sh, Envi2CISdem.sh, GeocodeMask.sh***,… and aiming at inverting the lines or columns in binary files.

The script ***Envi2CISdem.sh*** uses these two tools to transform an Envi format DEM computed by AMSTerEngine into a pseudo Envi file usable by AMSTerEngine using CIS format. This is however very old and was developed before some changes in the way AMSTerEngine outputs its products. It must hence be tested. Most probably you would prefer ***DEM_Envi_hdr2AMSTer_txt.sh*** to transform an Envi or GEOTIFF format DEM into DEM file usable by AMSTerEngine.

See scripts for usage (in SCRIPTS_MT):
- ***FLIPproducts.py.sh*** (in SCRIPTS_MT)
- ***FLOPproducts.py.sh*** (in SCRIPTS_MT)
- ***FLIPFLOPproducts.py.sh*** (in SCRIPTS_MT/zz_Utilities_MT_Ndo)

and (in SCRIPTS_MT/zz_Utilities_MT):
- ***Envi2CISdem.sh***
- ***DEM_Envi_hdr2AMSTer_txt.sh***

8.32) Transforming NaN to zero and inversely or change value:

Python scripts *NaN2zero.py* or *zero2NaN.py* transform NaN into zero and inversely.

Parameters are file to transform and input file format (byte or float32).
Note that setting NaN in byte file is not appropriate. Hence, if byte file is passed to *zero2NaN.py*, the output file will not be in byte anymore but in float32.

Note that there is a script ***Check_All_Nan.sh*** which checks all the Envi files in the current directory and checks if they contain at least one NaN or if they are full of NaN.
Parameters are:
- ALL or ANY if want to search resp. for files filled with only NaN or files with at least one NaN. For that it will either use *checkNaN.py* or *checkOnlyNaN.py*
- input file format (byte or float32)

Python script *Change_Val.py* replaces a value by another in a file.
Parameters are file to transform, value to search, value to replace with and input file format (byte or float32).

See script (in SCRIPTS_MT/zz_Utilities_MT):
- *NaN2zero.py*
- *zero2NaN.py*
- *Change_Val.py*
- ***Check_All_Nan.sh***

Transforming bytes to float and inversely:

See script (in SCRIPTS_MT/zz_Utilities_MT_Ndo):
- *float2byte.py*
- *byte2float.py*

8.33) Cropping last col or line from binary files:

See script (in SCRIPTS_MT/zz_Utilities_MT):
- *CropLastCol.py* *or* *CropLastCol_float.py*
- *CropLastLine.py* *or* *CropLastLine_float.py*

9. Utilities for repair or development:

There are several scripts stored in SCRIPTS_MT/zz_Utilities_MT or /zz_Utilities_MT_Ndo. They were required at some points for repairing problems I had during faulty processing or during developments, or just small utilities that were useful to me at some point. Some are explained here below though more are present in these directories. If not explained here, it may mean that they haven't been used for a while or were never finished… Use with care; Read carefully at least the header of the scripts. It is strongly advised to check first these scripts before launching them.

9.1) Checking S1 images:

The script ***CheckS1Pol.sh*** aims at checking the polarization for each burst and ensure it is the same everywhere. It must be launched in the directory where all the S1 images are stored in csl format. There is a similar version of the script that also clean those for which the polarisation is not the same as the one provided in parameter: see
Remove_S1Pol_Not_as_in_LaunchParamFile.sh

The script ***Check_S1_Mas_Slv_sizes.sh*** checks the size (in lines and columns) of the Primary and interpolated Secondary SLCs and compare them with the file sizes (in bytes). It must be launched in the pair directory where interferogram was computed.

The script ***Check_S1_SLCImageInfo.sh*** checks that there is a *SLCImageInfo.txt* in each S1.CSL directory and that polarisation is ok. Note that the polarization is hard coded in the script.

The script ***ReCreateLink_S1_Read.sh*** re-creating the link of S1 images in REGION after reading in csl format and stored in REGION_MODE where they will be used by the AMSTerEngine automated scripts. The script will do this from S1 images in Asc and Desc specific directory. This script is based on ***Read_All_Img.sh***, which can explain some useless stuffs here and there...

The script ***Check_S1_Before_MSBAS.sh*** checks that S1 data are OK in SAR_MASSPROCESS/Geocoded (except Ampl) and in MSBAS/region/MODES*i* i.e. from a previous MSBAS processing and for next preparation for a MSBAS.

The script ***Check_S1_RawImageSize.sh*** checks the size of the UNZIP S1 image files; that is the size of directory is at least *6 Gb* (must be integer) and contains *57* files & subdir (these numbers are hard-coded as parameters at the beginning of the script).

When performing a manual data download with another method as the daily automatic download, raw S1 images might be unzipped in a specific directory. The script ***CheckS1unzip.sh*** checks if directories with S1 unzipped images in that unconventional directory would already exist in S1-DATA-*TARGET*-SLC.UNZIP or S1-DATA-*TARGET*-SLC.UNZIP_FORMER/*_YYYY* directories and have the same size. If not, it displays the nr of files and the total size for comparison.

The script ***_Check_S1_SizeAndCoord.sh*** checks the number of bursts and coordinates of corners of S1 images. It compares them to expected values provided as input parameters. These can either be the 4 corners of a kml (obtained e.g. by running ***_Check_S1_SizeAndCoord.sh*** $PATH_1650/SAR_CSL/*SAT*/*TRK*/NoCrop/*img*.csl *Dummy* where the path is the full path to a good image in SLC format for that region), or the min longitude, max longitude, min latitude and max latitude of the area of interest measured eg. from Google Earth.

If the number of bursts is not as expected, or if the corner's coordinates are not in a given range, it echoes an error message with the explanations about the difference. It also attributes to a parameter named *STATUS* a value (OK or FAIL) that can be used in other scripts such as ***_Check_ALL_S1_SizeAndCoord_InDir.sh***, which used to perform the check for all images in a directory (see below).
If you only want to read the nr of bursts and coordinates of an image, provide the script with the PATH to the image and a Dummy parameter.

The script ***_Check_ALL_S1_SizeAndCoord_InDir.sh*** checks the size (in bursts) and coordinates of corners of all S1 images in a $PATH_1650/SAR_CSL/S1/*TARGET*/NoCrop directory. Images that are not compliant are moved in a temporary quarantine directory __TMP_QUARANTINE. Result of each checked image is stored in a text log file depending on the status, that is either

__Unverifiable_Images_*date*.txt
___Wrong_Images_Images_*date*.txt
__Good_Images_Images_*date*.txt

These log files are removed after 15 days. It is the responsibility of the user to check regularly the $PATH_1650/SAR_CSL/S1/ *TARGET* /NoCrop/__TMP_QUARNATINE directory: images there are either not fully downloaded yet and hence it might be solved the day after, or the image has a problem and must be manually moved to the definitive $PATH_1650/SAR_CSL/S1/ *TARGET* /quarantained directory.

The script ***_Check_S1orbits_InSubDirs.sh*** checks the S1 orbits in all the subdirectories from current directory and list in a file named ___List_Orbits_*DATE_TIME_RND*.txt the name of each S1 image and the SLCImageInfo.txt versions that are present in the image directory, that is the .txt and possibly the .ori if the S1 orbit was updated. See script.

The script ***_Check_S1_annotation.sh*** checks that the annotation directory is present in each image directory from the current directory. See script for usage.

The script ***_Which_S1_Is_Missing.sh*** lists the expected S1 images since provided dates (one for a first expected S1A image, and one for a first expected S1B image) until today. It compares these dates to what is existing in directories where raw and read images are expected (based on provided paths, e.g. *$PATH_3600*/SAR_DATA/S1/S1-DATA-*REGION*-SLC.UNZIP and *$PATH_1650*/SAR_CSL/S1/*REGION*/NoCrop respectively). Note that it will also check if RAW files are in …/S1-DATA-*REGION*-SLC.UNZIP_FORMER and if CSL files are in /Quarantained or __TMP_QUARANTINE).
Missing RAW images are listed in _Missing_Raw_S1A/B.txt in the directory where you have run the script and missing CSL images (only if RAW exists) are listed in _Missing_CSL_S1A/B.txt.

See scripts for usage (in SCRIPTS_MT/zz_Utilities_MT):

- ***CheckS1Pol.sh***
- ***Remove_S1Pol_Not_as_in_LaunchParamFile.sh***
- ***Check_S1_Mas_Slv_sizes.sh***
- ***Check_S1_SLCImageInfo.sh***
- ***ReCreateLink_S1_Read.sh*** (in SCRIPTS_MT/zz_Utilities_MT_Ndo)
- ***Check_S1_Before_MSBAS.sh*** (in SCRIPTS_MT/zz_Utilities_MT_Ndo)
- ***Check_S1_RawImageSize.sh***
- ***CheckS1unzip.sh*** (in SCRIPTS_MT/zz_Utilities_MT_Ndo)
- ***_Check_S1_SizeAndCoord.sh*** (in SCRIPTS_MT/zz_Utilities_MT_Ndo)
- ***_Check_ALL_S1_SizeAndCoord_InDir.sh*** (in SCRIPTS_MT/zz_Utilities_MT_Ndo)
- ***_Check_S1orbits_InSubDirs.sh*** (in SCRIPTS_MT/zz_Utilities_MT_Ndo)
- ***_Check_S1_annotation.sh*** (in SCRIPTS_MT/zz_Utilities_MT_Ndo)
- ***_Which_S1_Is_Missing.sh*** (in SCRIPTS_MT/zz_Utilities_MT_Ndo)

9.2) Checking the characteristics of all images.scl from current directory:

The script ***_Check_CSL_Corners_Trk_etc.sh***, is aiming at logging the main characteristics of all the images stored in your …/SAR_CSL/*SAT*/*REGION*/*NoCrop* directory in a file named List_Img_Characteristics.txt.
That file logs the following characteristics read in the /Info/SLCImageInfo.txt of each image:

Date:	The acquisition date of the image
Time:	The acquisition time
Mode:	The Orbit geometry (Asc or Desc)
LookDir:	Right or Left
LookAngl[deg]:	The incidence angle
Xsize[pix]:	The size of image in range direction (in pixels)
Ysize[pix]:	The size of image in azimuth direction (in pixels)
Xsampl[m]:	The pixel size in range direction (in m)
Ysampl[m]:	The pixel size in azimuth direction (in m)
Center/Ext-Lon:	The longitude of the center (or the min-max UTM Y value) of the scene
Center/Ext-Lat :	The latitude of the center (or the min-max UTM X value) of the scene

It must be launched from the directory where the images.scl are stored, that is…/SAR_CSL/*SAT*/*REGION*/*NoCrop*. It does not expect parameter to run.

See script for usage (in SCRIPTS_MT/zz_Utilities_MT_Ndo):
- ***_Check_CSL_Corners_Trk_etc.sh***

9.3) Removing unnecessary data from slave.interpolated.csl in pair directory:

In the former versions of *AMSTerEngine*, the interpolated Secondary (slave) data was stored in full in the interferometric pair directory (i12/InSARProducts/*slave*.interpolated.csl).
However, this took a lot of space and was useless. Now it only keeps the i12/InSARProducts/*slaves*.interpolated.csl/Info directory with the required information if some reprocessing is necessary.

The script ***Del_All_Slave.csl_But_TextFile.sh*** removes all the unnecessary data resulting from usage of an old version of *AMSTerEngine* in each *slave*.interpolated.csl subdirectory from a mass processing directory. It is hence only needed to clean directories to spare room on hard disk. It will also tell you how much disk space you gained after performing that cleaning. Use with care though, as usual when it deals with deleting stuffs…

Note that there is a simpler version of that cleaning, that is ***Clean_Slave.csl.sh***. It only clean /SLCData sub folder, not the /Headers subfolder. It requires *Crop_interf.py*.

See script for usage (in SCRIPTS_MT/zz_Utilities_MT):
- ***Del_All_Slave.csl_But_TextFile.sh***
- ***Clean_Slave.csl.sh***

9.4) Removing files from PAIR directories in SAR_MASSPROCESS:

In order to spare room on your disks, you may want to delete some files from *PAIR* directories located in SAR_MASSPROCESS/*SAT*/*REGION_TRK*/*SM_ZOOM_ML*/.

You can do that using script ***Clean_MassProcessedPairs.sh***. Note that a parameter allows you to choose the degree of cleaning:

- *All* :
  remove all files and subdirectories in each *PAIR* but */i12/TextFiles/InSARParameters.txt* and *masterSCLImageInfo.txt*., which are required by the script ***build_header_msbas_criteria.sh*** when preparing the MSBAS inversion. You can of course delete these text files as well, but then you will need to prepare you msbas inversion using (for more info, see 6.1.a):
  build_header_msbas_criteria_From_nvi_name_WithoutAcqTime.sh..
- *Moderate* :
  remove all but keeps what is needed if you want to be able to re-run a geocoding later on.
- *Light* :
  remove all but keeps what is needed if you want to be able to re-run an unwrapping and a geocoding later on.

Remember that if you need to update your time series, it is recommended to keep all the *PAIR* directories (at least empty) for ***SuperMaster_MassProc.sh*** to identify which pairs were already processed.

See script for usage (in SCRIPTS_MT/zz_Utilities_MT):
- ***Clean_MassProcessedPairs.sh***

9.5) Deleting a large number of files:

The script ***___Kill_All.sh*** delete as quickly as possible the whole content of a directory (provided as parameter to the script). It uses *rsync* for the highest efficiency when dealing with a very large number of files.

Use it with a LOT OF CARE because there is no undo!
The name contains 3 leading underscores. Does it prevent accidental use? Who knows…

See script for usage (in SCRIPTS_MT/zz_Utilities_MT):
- ***___Kill_All.sh***

9.6) Checking or creating links from each *.csl in a source directory to another directory:

The basic script ***MakeHardLink.sh*** creates a hard link (if it does not exist yet) from each subdirectory *.csl located in a source directory (path given as parameter 1) into a target directory (path given as parameter 2).

The script ***Remove_Broken_links.sh*** check in the directory provided as parameter if each *.csl subdirectory is a directory or a link pointing toward a valid target. If no, it deletes the *.csl.

There is also a script named ***Remove_Links_In_Dir.sh*** which removes possible erroneous links in *image*.csl directories. See

See script for usage:
- ***MakeHardLink.sh*** (in SCRIPTS_MT/zz_Utilities_MT)
- ***Remove_Broken_links.sh*** (in SCRIPTS_MT/zz_Utilities_MT)
- ***Remove_Links_In_Dir.sh*** (in SCRIPTS_MT/zz_Utilities_MT_Ndo)

9.5) Linking or copying Geocoded files to msbas directory

The script ***Ins_msbas.sh*** is aiming at creating a symbolic link (or copying ; see code) from Envi geocoded files to where they will be used by MSBAS.

See script for usage (in SCRIPTS_MT/zz_Utilities_MT):
- ***Ins_msbas.sh***

9.6) Swapping columns in files

The scripts ***Swap_Col3and4.sh*** makes use of *awk* to swap the 3rd and the 4th columns of an ascii file.
The version ***Swap_Col3and4_IfNeeded.sh*** does the same but only if the value of the 4th column is larger than the value of the 3rd column.

Search_Mas_After_Slv.sh list the lines of a file that contains 4 columns and where date in col1 is after date in col2.

See script for usage (in SCRIPTS_MT/zz_Utilities_MT):
- ***Swap_Col3and4.sh***
- ***Swap_Col3and4_IfNeeded.sh***
- ***Search_Mas_After_Slv.sh***

9.7) Searching and removing files or directories if size is smaller than a given threshold

The script ***RemoveDEM_SmallerThan.sh*** searches for the externalSlantRangeDEM files in the current directory and all its subdirectories and delete them if their size is smaller than a threshold value provided as parameter.

The script ***Move_All_Dir_SmallerThan.sh*** searches in the current directory for all the directories that are smaller than a certain amount (in Mb) and move them in a provided path.

The script ***Move_All_Files_SmallerThan.sh*** searches in the current directory for all the files named "*deg" (and its hdr) that are not of the given size (in Bytes) and move them in a provided path.

See script for usage (in SCRIPTS_MT/zz_Utilities_MT):
- ***RemoveDEM_SmallerThan.sh***
- ***Move_All_Dir_SmallerThan.sh*** (in SCRIPTS_MT/zz_Utilities_MT_Ndo)
- ***Move_All_Files_SmallerThan.sh*** (in SCRIPTS_MT/zz_Utilities_MT_Ndo)

9.8) Renaming, searching or copying files (or dir) based on string(s) in name

The very basic script ***Rename.sh*** search for all files in the current directory containing a given string in the name and rename these files by replacing that string with something else. (String to search and to replace with are hard-coded).

The script ***CP_All_files_as.sh*** copies all files from current directory (incl. subdirs) that contains a certain string (parameter 1) in their name into a given dir. (parameter 2).

The script ***echo_missing_files_as.sh*** searches in the current directory each subdirectory which name contains a given string, then check in that directory for the presence of a /i12/InSARproducts/*.jpg file. If not, it echoes the name of the problematic pair directory.

The script ***ren.sh*** add a .bin string between the basename and the extension of all the .hdr files in dir.

The script ***Rename_CSK_From_MassReader.sh*** renames the CSK directories obtained from CSK mass reading by only keeping *DATE.csl* instead of *CSKx_DATE_TIME.csl*

The script ***Find_Dir_Name_WIth_Date_Twice.sh*** looks for all files in current directory that are named with duplicated date (e.g. search for pairs that would have been accidentally processed with twice the same date as Primary and Secondary image. The script can be easily adapted to search for files instead of dir and in subdirs... and/or do something with results (see script in SCRIPTS_MT/zz_Utilities_MT_Ndo). It must obviously be launched in dir to search in.

See script for usage (in SCRIPTS_MT/zz_Utilities_MT):
- ***Rename.sh***
- ***CP_All_files_as.sh***
- ***echo_missing_files_as.sh***
- ***ren.sh***
- ***Rename_CSK_From_MassReader.sh***
- ***Find_Dir_Name_WIth_Date_Twice.sh*** (in SCRIPTS_MT/zz_Utilities_MT_Ndo)

9.9) Check value of the Ellipsoid in Geocode Parameters file

The script ***Check_Ellipsoid_in_GeocParam.sh*** checks the size of the ellipsoid in all the i12/TextFiles/geoProjectionParameters.txt from the SAR_MASSPROCESS. If zero, it updates the value and store the pair name in file named *_Updated_GeoProjParamFiles.txt*. The script must be launched in SAR_MASSPROCESS/sat/trk/crop/, i.e. where all pair directories are.

See script for usage (in SCRIPTS_MT/zz_Utilities_MT_Ndo):
- ***Check_Ellipsoid_in_GeocParam.sh***

9.10) Check coordinates of corners from all geocoded products in SAR_MASSPROCESS pair dirs.

The script ***Check_GeocCropCoord_GeocParam.sh*** checks the coordinates of corners from all geocoded products in the i12/TextFiles/*geoProjectionParameters.txt* from the SAR_MASSPROCESS pair directories. Coordinates are logged into *_GeoProjCoord.txt.*.

Size can be compared to expected values (hard coded); if not as expected, the script lists the pairs names in *_GeoProjCoord_WrongPairs.txt*, where it will also log the creation date

See script for usage (in SCRIPTS_MT/zz_Utilities_MT_Ndo):

- ***Check_GeocCropCoord_GeocParam.sh***

9.11) Changing the endianness of a binary file

Using gdal, ***Change_Bin_Order.sh*** changes the byte ordering of all ENVI .bin files in the current directory.

See script for usage (in SCRIPTS_MT/zz_Utilities_MT):

- ***Change_Bin_Order.sh***

9.12) Changing date format

The script ***dat2sec.sh*** transforms the first column of a file (except the first header line) from *YYYYMMDD* into *Linux seconds*.

The script ***date2decimalyr.sh*** transforms the first column of a file from *YYYYMMDD* into *decimal year*.

The script ***decimalyr2date.sh*** transforms the first column of *decimal year* into *YYYYMMDD*.

The script ***sec2date.sh*** transforms the first column of *Linux seconds* into *YYYYMMDD*.

See script for usage (in SCRIPTS_MT/zz_Utilities_MT):

- ***dat2sec.sh***
- ***date2decimalyr.sh***
- ***decimalyr2date.sh***
- ***sec2date.sh***

9.13) Update all LaunchMTparameters.txt files

The script ***ChgeAll_LaunchParamFiles.sh*** is dedicate to change all lines containing a given string with another one from all param files (such as *___V20231026_LaunchMTparam.txt*) in sub dirs. This might be useful if a new feature is added to AMSTer that requires for instance to change all the LaunchMTparameters.txt files.
It was also sometimes used in a script ***Chge_Several_Criteria_LaunchParamFiles.sh*** dedicated to operate multiple occurrences of parameters changes.

The script ***RemoveFromAll_LaunchParamFiles.sh*** removes all lines containing a given string from all parameters files in sub directories.

These scripts haven't been used for a while though. Not sure they still work properly.

See script for usage (in SCRIPTS_MT/zz_Utilities_MT_Ndo):
- ***ChgeAll_LaunchParamFiles.sh***
- ***Chge_Several_Criteria_LaunchParamFiles.sh***
- ***RemoveFromAll_LaunchParamFiles.sh***

9.14) Compile and store files when updating AMSTerEngine

The script ***UpdateAMSTerEngine.sh*** is aiming at updating AMSTerEngine by compiling the sources and sorting them in appropriate place and clean compilation directories. It compiles the main software and the tools. It expects the following parameters:
- path to zipper source of AMSTerEngine to update
- Date of version to update (YYYYMMDD)

See script for usage (in SCRIPTS_MT/zz_Utilities_MT_Ndo):
- ***UpdateAMSTerEngine.sh***

9.15) Update hard coded lines after update of AMSTer toolbox

Script ***__Tune_your_AMSTer.sh*** is aiming at updating all hard coded lines in AMSTer scripts used in your installation, i.e. that would differ from the scripts downloaded or synchronized e.g. from GitHub.
It helps to keep the updated scripts consistent with your own installation. It is hence recommended to run it after each synchronization with the GitHub repository for instance.

The script contains (among others) a function named ***UpdateLineInScript*** dedicate to replace a given line in a given script with a new line.

Hence you must first build your own set of calls of the function ***UpdateLineInScript*** for each hard coded line to change. Everything is then hardcoded in that script and there is no parameter for this script.

Because special characters (such as *, $, \ etc…) are interpreted by the shell scripts, finding and replacing them in the scripts requires a little trick. They are replaced by a chosen string (which one must be sure is not already used in the scripts) beforehand. After the replacement of the line, all these strings are replaced back with the special character. In principle,

__Tune_your_AMSTer.sh is able to cope with most of the special characters. However, if a new special character would mess up in the line to change during update, you may need to:

- Chose a replacement string
- add that special character in the functions ***RemoveSpecChar*** and ***RestoreSpecChar***
- add the replacement string in the test array LISTOFSCTRINGS

ATTENTION:

- It is mandatory that your initial line (the one you search for) and the new line (the one that will be used to replace the initial one) **MUST** be clean of
  - double quotes (single are ok)
  - $
  - @ because we use it as sed separator...

  If some of these are present, they **MUST** be preceded by a backslash.
- Be aware that if your line appears multiple time, it will change it multiple time

- Each script to be updated **MUST** be provided to ***UpdateLineInScript*** with its full path.

It is advised to test the script for instance by creating a file named Test_Syntax.txt and copy the following 3 lines in it:

```
This line will be unchanged, but the following
abc ${a var}, == [why] {not} (brackets) "and quotes" && 'double quotes' || /a/PATH/${here} and 1 ; 2 : 3 , 4 . 5 - 6 -- 7 _ 8 __ "seems 'ok' though" hopefully should be capitalized
```

then run ***__Tune_your_AMSTer.sh*** with the 3 following lines hard coded in the body of the present script, i.e. below the functions definition

```
DEFAULTLINE="abc \${a var}, == [why] {not} (brackets) \"and quotes\" && 'double quotes' || /a/PATH/\${here} and 1 ; 2 : 3 , 4 . 5 - 6 -- 7 _ 8 __ \"seems 'ok' though\" hopefully"
PERSOLINE="ABC \${A Var}, == [Why] {Not} (Brackets) \"And Quotes\" && 'Double Quotes' || /A/path/\${Here} And 1 ; 2 : 3 , 4 . 5 - 6 -- 7 _ 8 __ \"Seems 'OK' Though\" Hopefully"
UpdateLineInScript /PATH_TO_YOUR/Test_Syntax.txt  "${DEFAULTLINE}" "${PERSOLINE}"
```

After successful run of the script, the second line of Test_Syntax.txt must have been updated and the file should now look like:

```
This line will be unchanged, but the following
ABC ${A Var}, == [Why] {Not} (Brackets) "And Quotes" && 'Double Quotes' || /A/path/${Here} And 1 ; 2 : 3 , 4 . 5 - 6 -- 7 _ 8 __ "Seems 'OK' Though" Hopefully will be capitalized
```

NOTE THAT A \ (back slash sign) MUST BE ADDED MANUALLY BEFORE EACH OCCURRENCE OF A $ (dollar sign) OR A " (double quote sign) IN THE DEFINITION OF THE LINES TO SEARCH (DEFAULTLINE) AND REPLACE (PERSOLINE) IN THE PRESENT SCRIPT.
YOU DO NOT HAVE TO ADD THESE \ IN THE ORIGINAL FILE TO CHANGE OF COURSE.

For security reason, the script will back up your original scripts (e.g. Test_Syntax.txt in the case of the test) with an .original.perso extension (e.g. Test_Syntax.txt.original.perso). If you are satisfied with your updated script, you can delete that backup because it will not be overwritten at next run of the script.

See script for usage (in SCRIPTS_MT/zz_Utilities_MT).

9.16) Small maintenance or check tools

Script ***l3.sh*** lists
- the 3 last S1 images read from VVP; Lux and Domuyo in Asc and Desc modes,
- then the 3 last resampled,
- then the last 3 mass processed (from Geocoded, i.e. maybe not finished yet)
- then the last 3 finished directories in MASSPROCESSED
- then the last 3 images included in msbas processing in Asc and Desc LOS and in UD-EW combi (incl. with and w/o Coh threshold when appropriate).

Then it sorts the results in a table that helps to see quickly if the automated processes are up to date. See script for usage (in SCRIPTS_MT/zz_Utilities_MT_Ndo).

Script ***psn*** uses the Linux command *ps -eaf* to make a summary of all the running processes using AMSTer tools. It will provide with valuable information such as on which hard disk some computations are running, how many pairs were already processed (on a total of how many) and when were these processes launched. See script for usage (in SCRIPTS_MT).

The script ***_Check_Cron_MassProcessed.sh*** (in SCRIPTS_MT/zz_Utilities_MT_Ndo) is specific to our cron jobs at ECGS. It lists the last *5* (hard-coded parameter) scheduled S1 and CSK mass processes and checks what was effectively performed. Knowing the revisiting time of S1, it will warn in case of missing image. However, because there is no pre-defined revisiting time for CSK, it only looks for the *5* last existing raw images. The script outputs a table summarizing the status of the processings at the different stages and offers comments to help the user sorting out what could have happened in case of problem.

Similarly, the script ***_Check_Cron_Ampli.sh*** (in SCRIPTS_MT/zz_Utilities_MT_Ndo) is specific to our cron jobs at ECGS. It lists the last *5* (hard-coded parameter) scheduled S1 and CSK AMPLITUDE processes and check what was effectively performed. Knowing the revisiting time of S1, it will warn in case of missing amplitude image. However, because there is no pre-defined revisiting time for CSK, it only looks for the *5* last existing amplitude images. These amplitude images are computed from automatic ***ALL2GIF.sh*** processes used to track geomorphological changes in the crater of some volcanoes.
The script outputs a table summarizing the status of the processings at the different stages and offers comments to help the user sorting out what could have happened in case of problem.

The script ***Get_DateFomFileName_And_CreationDate.sh*** aims at getting the date from the name of all the image files (or directories) in the current directory as well as their creation date. It also displays the difference in days between name and creation, as well as some statistics related to latency with former images etc. It must be launched from directory where all the images are stored. See script in SCRIPTS_MT/zz_Utilities_MT_Ndo.
This script is used to check e.g. the latency of CSK images provided by the Virunga Supersite and outputs a message like:

```
Get_DateFomFileName_And_CreationDate.sh Distro V1.0.2 AMSTer script utilities, Nicolas d'Oreye, (c)2016-2019, Last modified on Mar 15, 2022
Processing launched on Fri Jun 10 10:48:17 CEST 2022

[…]
Image 20220127 was created on 20220203, i.e. nr of days between acquisition and delivered: 7 ;   Nr of days between two last images: 2   ;   Nr of days withtout info: 9
Image 20220210 was created on 20220216, i.e. nr of days between acquisition and delivered: 6 ;   Nr of days between two last images: 14  ;   Nr of days withtout info: 20
Image 20220212 was created on 20220216, i.e. nr of days between acquisition and delivered: 4 ;   Nr of days between two last images: 2   ;   Nr of days withtout info: 6
Image 20220228 was created on 20220305, i.e. nr of days between acquisition and delivered: 5 ;   Nr of days between two last images: 16  ;   Nr of days withtout info: 21
Image 20220305 was created on 20220311, i.e. nr of days between acquisition and delivered: 6 ;   Nr of days between two last images: 5   ;   Nr of days withtout info: 11
Image 20220307 was created on 20220311, i.e. nr of days between acquisition and delivered: 4 ;   Nr of days between two last images: 2   ;   Nr of days withtout info: 6
Image 20220316 was created on 20220415, i.e. nr of days between acquisition and delivered: 29 ;  Nr of days between two last images: 9   ;   Nr of days withtout info: 38
Image 20220321 was created on 20220415, i.e. nr of days between acquisition and delivered: 24 ;  Nr of days between two last images: 5   ;   Nr of days withtout info: 29
Image 20220323 was created on 20220415, i.e. nr of days between acquisition and delivered: 22 ;  Nr of days between two last images: 2   ;   Nr of days withtout info: 24
Image 20220324 was created on 20220415, i.e. nr of days between acquisition and delivered: 21 ;  Nr of days between two last images: 1   ;   Nr of days withtout info: 22
Image 20220330 was created on 20220415, i.e. nr of days between acquisition and delivered: 16 ;  Nr of days between two last images: 5   ;   Nr of days withtout info: 21
Image 20220401 was created on 20220423, i.e. nr of days between acquisition and delivered: 22 ;  Nr of days between two last images: 2   ;   Nr of days withtout info: 24
Image 20220409 was created on 20220429, i.e. nr of days between acquisition and delivered: 20 ;  Nr of days between two last images: 8   ;   Nr of days withtout info: 28
Image 20220422 was created on 20220429, i.e. nr of days between acquisition and delivered: 7 ;   Nr of days between two last images: 13  ;   Nr of days withtout info: 20
Image 20220423 was created on 20220429, i.e. nr of days between acquisition and delivered: 6 ;   Nr of days between two last images: 1   ;   Nr of days withtout info: 7
Image 20220424 was created on 20220429, i.e. nr of days between acquisition and delivered: 5 ;   Nr of days between two last images: 1   ;   Nr of days withtout info: 6
Image 20220425 was created on 20220513, i.e. nr of days between acquisition and delivered: 18 ;  Nr of days between two last images: 1   ;   Nr of days withtout info: 19
Image 20220503 was created on 20220513, i.e. nr of days between acquisition and delivered: 10 ;  Nr of days between two last images: 8   ;   Nr of days withtout info: 18
Image 20220509 was created on 20220526, i.e. nr of days between acquisition and delivered: 17 ;  Nr of days between two last images: 6   ;   Nr of days withtout info: 23
Image 20220510 was created on 20220526, i.e. nr of days between acquisition and delivered: 16 ;  Nr of days between two last images: 1   ;   Nr of days withtout info: 17
Image 20220511 was created on 20220526, i.e. nr of days between acquisition and delivered: 15 ;  Nr of days between two last images: 1   ;   Nr of days withtout info: 16
Image 20220519 was created on 20220526, i.e. nr of days between acquisition and delivered: 7 ;   Nr of days between two last images: 8   ;   Nr of days withtout info: 15
Image 20220524 was created on 20220526, i.e. nr of days between acquisition and delivered: 2 ;   Nr of days between two last images: 5   ;   Nr of days withtout info: 7
Today, 20220610, nr of days withtout info since last image was acquired is:  17

Statistics over last 60 images:
  Average delay between acquisition and delivered: 9.31 days ;   Average delay between two images: 4.20 days  ;   Average nr of days without info: 13.51
```

Because samba sometimes mess up with hidden temporary files that prevent deleting directories, the script ***KillGhost.sh*** aims at attempting circumventing the problem by moving these hidden unclosed files in a directory named ToKill. See script for usage (in SCRIPTS_MT/zz_Utilities_MT_Ndo).

I n some cases, raster files created by a processing run on Linux computer were empty. The script ***Recreate_Emplty_GeocodedRasters.sh*** will recreate all empty raster files from /GeocodedRasters/DefoInterpolx2Detrend.
It must be launched in SAR_MASSPROCESS/region where
/Geocoded and /GeocodedRasters are.

The script ***Which_S1_Is_Missing.sh*** lists the expected S1 images since provided date until today and compare to raw and read images. It performs the check first for S1A then S1B images. Missing RAW images are listed in _Missing_Raw_S1A/B.txt in the current directory. Missing CSL images (when RAW exists) are listed in _Missing_CSL_S1A/B.txt in the current directory. See script for usage (in SCRIPTS_MT/zz_Utilities_MT_Ndo)

The script ***Check_All_DEMs_in_CSL.sh*** checks the name and the size of the *slantRangeDEM* in each image directory (e.g. SAR_CSL/Sat/Region/NoCrop/imgage.csl directories). It also displays the version of AMSTer used to create the *slantRangeDEM.*
See script for usage (in SCRIPTS_MT/zz_Utilities_MT_Ndo)

9.17) Manually unzip and store S1 data

When downloading a new batch of S1 data, it might be useful to use ***Unzip_S1.sh*** to decompress all .zip file in a directory provided as parameter (including in sub dir) and store them in a new directory with the same named, though with tailed with *.UNZIP*. The script also check that each image hasn't been unzipped yet. If so, it checks it size and redo the unzip if size is not as expected.

See script for usage (in SCRIPTS_MT/zz_Utilities_MT_Ndo):

- ***Unzip_S1.sh***

9.18) Mac only features: Add colour bullets to dir names or de-quarantine files

The script ***MacColorFile.sh*** aims at tagging a directory with a colour (This is a Mac only feature!):

0 No color
1 Orange
2 Red
3 Yellow
4 Blue
5 Purple
6 Green
7 Gray

This script is called e.g. by the script ***CheckJpg.sh***, which aims at checking all PAIR sub-directories and tag them with a green or red colour bullet if they do or do not contain a jpg file in /i12/InSARProducts.

See scripts for usage (in SCRIPTS_MT/zz_Utilities_MT_Ndo):

- ***MacColorFile.sh***
- ***CheckJpg.sh***

Because for unknown reason, some scripts are set in quarantine by recent Mac OSX, it happens sometimes that they can't be executed (e.g. sometimes after modifications). To solve this, the script ***MacOSX_DeQuarantine_File.sh*** removes either one file (provided as parameter) or all the files in the current directory (if param : -d) and sub-dirs (if param : -dd) from security quarantine. (This is a Mac only feature!).
See script in SCRIPTS_MT/zz_Utilities_MT_Ndo.

Similarly, ***Unlock_all_in_dir.sh*** unlocks all Mac locked files from a directory and its sub directories.

9.19) Single coordinates transformation:

The scripts ***_UTM2LatLong.py*** and ***_LatLong2 UTM.py*** transform UTM coordinates to LatLong and inversely (single values).

See scripts for usage (in SCRIPTS_MT/zz_Utilities_MT_Ndo):

- ***_UTM2LatLong.py***
- ***_LatLong2 UTM.py***

9.20) Set up script:

The scripts ***Build_Geocoded_and_GeocodedRasters_Dirs.sh*** do nothing else that manually creating Geocoded and GeocodedRasters directories and subdirectories normally created during the mass processing.

10. Troubleshooting:

What could possibly go wrong? Well… here are some ideas.

Also:

- think about reading the manual (e.g. search for keywords or look in the index);
- consult the discussion group in the GitHub repository.

10.1. Can't find …

One of the most classical errors is a file or a directory that can't be found. It can result from a typo in your parameters file or in your command line. It can also be due to missing data or results of course.

You can try to list at the terminal the directory where the missing file or directory is supposed to exist. If it does not exist, remove step by step the sub-directories from the path to see from where there is a failure.
You can also compare in a text editor the path of the missing file or directory as copied from the error message and the path from your file manager: see if there is a typo somewhere.

10.2. Problem with mask:

Check that the mask has the proper name and format.

In particular, it the name of the mask must have no extension compared to its header, e.g.
name_of_your_mask.what_ever_your_want
name_of_your_mask.what_ever_your_want.hdr
and not something like
name_of_your_mask.what_ever_your_want.envi
name_of_your_mask.what_ever_your_want.hdr

(See more about the format in section 0.13)

10.3. /bin/bash: bad interpreter

If you encounter the following message:
*ScriptName**.sh***: /bin/bash: bad interpreter: Operation not permitted
means that the shebang (first line of your script containing the string "#!/bin/bash" is corrupted. Usually erasing and rewriting it will not solve the issue. By experience, the most efficient way to solve it is

- create a new empty file
- copy in it all the script but the first line with the shebang
- add manually that first line #!/bin/bash
- delete the initial corrupted script
- save your new file with the name of the script to be replaced
- you may need to make that new script executable by typing
  chmod +x *ScriptName**.sh***

10.4. Problem with cron

With Linux, if your script launched by a cron does not behave as when launched from the terminal, check:

- that your state variables are all defined in your cron tab as they are in your .bashrc
- that no scripts related to the mass processing are open in a text editor. Indeed, the scripts for cron are usually checking that no other crons are running (same of previous steps), nor manual run of the same step for the same target. However, one of these scripts open with a text editor would be detected by the "***ps***" command. To be sure, close all the scripts open in text editor.
- The several logs that may give you a hint about the reason of the crash…

With Mac, if your script does not launch from cron, check that cron is allowed to have full disk access:

- open the **System Settings**
- Select **Privacy & Security** menu
- Select **Full Disk Access** menu
- Add /usr/bin/crontab and /usr/bin/cron

(this may differ depending on the Mac OSX version though).

10.5. Problem with msbas

In case of problem, start by checking:

- The size of each DefoInterpolx2Detrendi/*deg file. Search for files that would be significantly smaller or bigger than the majority.
- The links are not broken or pointing toward empty images
- The header.txt
- If during the SVD inversion, it outputs a line mentioning "*xxx* Killed" where *xxx* is a number, it means that you ran out of RAM. Decrease the amount of data to invert either:
    - restricting the number of pairs to invert by optimizing the baseline plot (see ***Run_optim_module.sh***) – preferred solution;
    - by cropping the zone to invert (see *WINDOW_SIZE* in header.txt);
    - restricting the number of pairs to invert by selecting pairs with each image taken max 3 times as Primary and Secondary (see ***Extract_Baselines_3.sh***);
    - restricting the number of pairs to invert by selecting a coherence threshold (see ***restrict_msbas_to_Coh.sh***);
    - restricting the number of pairs to invert by checking the phase closure (see ***Extract_Triangles.sh*** and ***Check_Closure_All_Triangles.sh***);

  Or increase you swap memory if possible (see technical note Create_Swap_Linux.pdf for Linux), or add more RAM…

10.6. Problem with jpg time series plots

With Linux, if your jpg figure merging the eps time series plot and the tags (position of the pixels on the velocity maps and direction of deformation) is not as expected, check:

- The /etc/ImageMagick/policy.xml or /etc/ImageMagick-6/policy.xml file, which must contain the following values:
  - `<policy domain="coder" rights="read|write" pattern="PS" />`
  - `<policy domain="coder" rights="read|write" pattern="EPS" />`
  - `<policy domain="resource" name="width" value="32KP"/>`
  - `<policy domain="resource" name="height" value="32KP"/>`
  - `<policy domain="resource" name="disk" value="8GiB"/>`
- The scripts ***AmpDefo_map.sh***, ***TimeSeriesInfo_HP.sh*** etc… use the right font, that is
  - Font="FreeSans"

With Mac or Linux, in case of problem, also check that:

- The necessary files from SCRIPTS_MT/TSCombiFiles were correctly copied in the directory .../MSBAS/*YourTarget*/_CombiFiles
- The .../MSBAS/*YourTarget*/_CombiFiles/TS_parameters.txt file has been correctly updated (see Section 6.4.a)
- The .../MSBAS/*YourTarget*/_CombiFiles/satview.jpg file has been correctly created (see Section 6.4.a)

10.7. Problem with Java

If a script crashes with the following message, re-install java and then Fiji (e.g. using ***AMSTer_Install.sh***)

```
<< The operation couldn't be completed. Unable to locate Java Runtime that supports (null). >>
```

10.8. Problem reading h5 format data (like CSK data)

Check hdf5 library version with the command:

- h5dump –version (Mac)
- pkg-config --modversion hdf5 (Linux)

On Mac, versions before 1.14.2 seems to have problems.

Note that the compiler checks the version before compiling and takes into account the major changes from v 1.12.

10.9. Problem with Libraries

If, after an update, some executables show messages such as

```
Library not loaded: 'opt/local/liblibtiff.5.dylib'
```

try to recompile the sources. It may happen e.g. after the update of some libraries…

Annexes:

A.1) Description of the *LaunchMTparam.txt* file

ATTENTION:

- options are case sensitive!
- Parameter value must be followed by a # and its variable name followed by coma.
- The description that follows the coma is optional though.
- As reading this file is made by searching for the first occurrence of the parameter's name followed by a coma, do not add text with the variable name followed by coma in the description.
- Do not put several lines with different values of the same variable. Scripts will only read the first occurrence.
- Always keep the path parameters at the bottom.

Example (___V20231026_LaunchMTparam.txt):

```
# PARAMETERS TO RUN SCRIPT LAUNCHING AMSTerEngine.
# PARAMETERS MUST BE FOLLOWED BY A # AND ITS VAR NAME FOLLOWED BY COMA.
#         (DESCRIPTION, THOUGH THIS IS OPTIONAL)
# AS READING THIS FILE IS MADE USING FIRST OCCURENCE OF SEARCH CRITERIA,
#         DO NOT ADD TEXT WTH VARIABLE NAME FOLLOWED BY COMA.
# ALWAYS KEEP THE PATH PARAMETERS AT THE BOTTOM
#
# VERSION Oct 07 2022

# AUTOMATIC FIGURES DISPLAY
############################
FIGyes                  # FIG, option to compute or not the quick look using cpxfiddle
POPno                   # POP, option to pop up figs created with cpxfiddle, or POPno

# DATA
######
S1                      # SATDIR, Satellite system (must be the same as dirname structure: RADARSAT, TSX, TDX, CSK, S1, ENVISAT)
LUX_A_15                # TRKDIR, Processing directory and dir where data are stored E.g. SM/Asc160 (must be the same as dirname structure)

# For mass processing only
###########################
20170429                # SUPERMASTER, date of the Global Primary as selected by Prepa_MSBAS.sh in
                        # e.g. /Volumes/hp-1650-Data_Share1/SAR_SM/MSBAS/Bukavu/seti/setParametersFile.txt

# DEM
###########
GreaterRegion           # DEMNAME, name of DEM (in mathematical order). Need txt file in same dir
SIMAMPno                # SIMAMP, compute Simutaled Amplitude during Ext Dem Generation - usually not needed (maybe ERS). SIMAMPno or SIMAMPyes
KEEP                    # RECOMPDEM, recompute DEM or mask in slant range even if already there (FORCE), or check the one that would exist (KEEP).
                                #   DO NOT RUN TWO "FORCE" OR A "FORCE" AND A "KEEP" PROCESS AT THE SAME TIME USING SAME PRIMARY
                                #   It may cause prblm if externalSlantRangeDEM and maybe slantRangeMask are being modified by the first FORCE run.

# CROP
###########
/$PATH_1650/SAR_CSL/S1/LUX/Lux.kml          # CROP, CROPyes or CROPno, or for S1, path to kml that will be used to define area of interest.
10000                   # FIRSTP, Crop limits: first point (row) to use
8000                    # FIRSTL, Crop limits: first line to use
24000                   # LASTP, Crop limits: last point (row) to use
12000                   # LASTL, Crop limits: last line to use
1                       # ZOOM, factor during crop
LUX                     # REGION, Text description of area for dir naming

# AMPLITUDE
###########
4                       # MLAMPLI, Multilooking factor for amplitude images reduction (used for coregistration - 4-6 is appropriate).
                                # If rectangular pixel, it will be multiplied by corresponding ratio.
SQUARE                  # PIXSHAPE, pix shape for product : SQUARE, ORIGINALFORM, SQUAREUNITY or ORIGINALFORMUNITY
SIGMANO                 # CALIBSIGMA, if SIGMAYES it will output sigma nought calibrated amplitude file (for S1 only)

# COARSE COREG
##############
64          # CCOHWIN, Coarse coreg window size (64 by default but may want less for very small crop).
            #   Can be set to 0 to skip coarse coreg when using god orbit sat such as TSX, TDX and Envisat
0.4         # COH, Coarse Coherence threshold coregistration
24          # CCDISTANCHOR, Coarse registration range & az distance between anchor points [pix] (eg 24 for large img, 16 for medium and 2-8 for very small crops)

# FINE COREG
############
7           # FCOHWIN, Fine coreg window size (eg 3 for ERS/ENV or 7 for CSK, TSX and RS; must have win of eg 50 pixels; computed on full resol img)
0.5         # FCOH, Fine Coherence threshold coregistration
24          # FCDISTANCHOR, Fine registration range & az distance between anchor points [pix] (eg 24 for large img, 16 for medium and 2-8 for very small crops)
```

```
# INSAR
########
DEFO        # PROCESSMODE, DEFO to produce DInSAR or TOPO to produce DEM (bytes, 1/0, UTM, Envi Harris format, larger than img) used only in SinglePair.sh)
VV          # INITPOL, For multi pol images; force polarisation at initInSAR for InSAR processing. If it does not exists it will find the first compatible PRM-SCD pol.
50          # LLRGCO, Lower Left Range coord offset for final interferometric products generation. Used in SinglePairNoUnwrap only for Shadow measurements
50          # LLAZCO, Lower Left Azimuth coord offset for final interferometric products generation. Used in SinglePairNoUnwrap only for Shadow measurements

4           # INTERFML, multilook factor for final interferometric products generation (to multiply to the LARGEST
            #   side of the pixel); when used with zoom, it is ML to apply to zommed pixels
1           # FILTFACTOR, filtering factor for interfero (2 might be too strong when used with POWSPECSMOOTFACT filtering)
1           # POWSPECSMOOTFACT, Power spectrum filtering factor (for adaptative filtering) (0 = no filtering; 1 or less is possible though stronger)
2       # COHESTIMFACT, in pixels. Must be similar to INTERFML as far as it is not a ML higher than 5 or 7 non ML for instance. For ML1, if -le 1, will be forced to 2.
        #   If INTERFML is larger than 5 or 7, limit anyway COHESTIMFACT to 5 or 7 (Remember: computations load goes as square of win size)

# MASK
#######
APPLYMASKyes    # APPLYMASK, Apply mask (bytes, LatLong, Envi Harris, larger than img) before unwrapping (APPLYMASKyes or APPLYMASKno);
            # Mask for AMSTer Engine <2 0230928: 1 = keep, 0 = mask. However, at unwrapping, 0-masked pixels are kept if their coh > COHCLNTHRESH
            # Mask for AMSTer Engine>20230928: 0 = keep, 1 = always mask, 2 = mask. However, at unwrapping, 2-masked pixels are kept
            #         if their coh > COHCLNTHRESH
            # If a mask is requested but no Snaphu, one can also mask manually files with ffa (eg ffa residualInterferogram.HH-HH.f x slantRangeMask)
/$PATH_3600/MASKS/LakeKivu_LatLong.envi  # PATHTOMASK, geocoded mask file name and path

# UNWRAPPING
############
SKIPno      # SKIPUW, SKIPno unwraps and geocode all products, SKIPyes skips unwrapping and geocode only available products, Mask geocode
            #   only ampli and coh (for mask geenration)
SNAPHU      # UW_METHOD, Select phase unwrapping method (SNAPHU, CIS, DETPHUN1ONLY, DETPHUN2ONLY, DETPHUN1SNAPHU,
            # DETPHUN2SNAPHU, DETPHUN1CIS, DETPHUN2CIS)

# if snaphu unwrapping:
1.2         # DEFOTHRESHFACTOR, Snaphu : Factor applied to rho0 to get threshold for whether or not phase discontinuity is possible. rho0 is the expected,
            # biased correlation measure if true correlation is 0. Increase if not good.
0.9         # DEFOCONST, Snaphu : Ratio of phase discontinuity probability density to peak probability density expected for discontinuity-possible pixel differences. #
            # Value of 1 means zero cost for discontinuity, 0 means infinite cost. Decrease if prblm.
0.2         # DEFOMAX_CYCLE, Snaphu : Max nr of expected phase cycle discontinuity. For topo where no phase jump is expected, it can be set to zero.
DEFO        # SNAPHUMODE, Snaphu : TOPO, DEFO, SMOOTH, or NOSTATCOSTS.
ZoneMapYes # ZONEMAP, if ZoneMapYes, will create a map with the unwrapped zones named snaphuZoneMap.
            # Each continuously unwrapped zone is numbered (from 1 to...)
0.00001     # ZONEMAPSIZE, Minimum size of unwrapped zone to map (in frazction of total nr of pixels)
300         # ZONEMAPCOST, Cost threshold for connected components (zones). Higher threshold will give smaller connected zones
50          # ZONEMAPTOTAL, Maximum number of mapped zones

MultiSnaphuNo           # MULTIUWP, MultiSnaphuYes performs recursive snaphu unwrapping (need 4 params bellow).
                        #     MultiUnwrapNo (or any other string) will perform single snaphu unwrapping
ResidInterfFilt         # WHICHINTERF, which interferogram to unwrap, ResidInterf (residual interfero) or ResidInterfFilt (residual interfero filtered)
0.9                     # COEFREQ, Coefficient of increase of cut-off frequency
12.5                    # CUTINI, Initial cut-off frequency (e.g. 12.5 for a 400x400 image, 10 for a 2200x1500 img)
10                      # NITMAX, Max total nr of iterrations
0.0627                  # COHMUWPTHRESH, coh threshold (between 0 and 1) below which it replaces the phase by white noise
                        #     (corresponding mask will be produced). If set to 0, do not mask with white noise

# if snaphu or CIS unwrapping:
0.25        # COHCLNTHRESH, Coherence cleaning threshold. Snaphu gives 0 weight at pixels below that threshold.
            # Moreover, if a mask is used, snaphu (or CIS) also unwraps 0-masked pixels (for AMSTerEngine < 20230928)
            #           or 2-masked pixels (for AMSTerEngine > 20230928)
            # if their coherence is above COHCLNTHRESH.

# if CIS unwrapping:
0.1         # FALSERESCOHTHR, False Residue Coherence Threshold: higher is much slower. Use max 0.15 e.g. in crater
3           # CONNEXION_MODE,number of times that connexion search radius is augmented when stable connections are found ; 0 search along all coh zone
3           # BIASCOHESTIM, Biased coherence estimator range & Az window size (do not apply pix ratio)
3           # BIASCOHSPIR, Biased coherence square spiral size (if residual fringes are not unwrapped decrease it; must be odd)

# if DETPHUN unwrapping:
3           # DETITERR, Number of iterration for detPhUn (Integer: 1, 2 or 3 is generaly OK)
0.3         # DETCOHTHRESH, Coherence threshold

BOTH        # INTERPOL, interpolate the unwrapped interfero BEFORE or AFTER geocoding or BOTH.
DETREND     # REMOVEPLANE, if DETREND it will remove a best plane after unwrapping. Anything else will ignore the detrending.

# GEOCODING
###########
UTM         # PROJ, Chosen projection (UTM or GEOC – GEOC OPTION IS NOT READY YET)
Forced      # GEOCMETHD, Resampling Size of Geocoded product: Forced (at FORCEGEOPIXSIZE - mandatory for further MSBAS),
            #       Auto (closest multiple of 10), Closest (closest to ML az sampling), ClosestMassProc (Closest even for a Mass Process),

LetCIS      # RADIUSMETHD, LetCIS (CIS will compute best radius) or forced to a given radius
TRI         # RESAMPMETHD, TRI = Triangulation; AV = weighted average; NN = nearest neighbour
LORENTZ     # WEIGHTMETHD, Weighting method : ID = inverse distance; LORENTZ = lorentzian
1.0         # IDSMOOTH,  ID smoothing factor
1.0         # IDWEIGHT, ID weighting exponent
1.0         # FWHM, Lorentzian Full Width at Half Maximum
1           # ZONEINDEX, Zone index

100                     # FORCEGEOPIXSIZE, Pix size wanted eg as you want for your final MSBAS database
                        # UTMZONE, letter of row and nr of col of the zone where coordinates below are computed (e.g. U32)
225000                  # XMIN, minimum X UTM coord of final geocoded product
426000                  # XMAX, maximum X UTM coord of final geocoded product
5417000                 # YMIN, minimum Y UTM coord of final geocoded product
5593000                 # YMAX, maximum Y coord of final geocoded product
/$PATH_1650/kml/        # GEOCKML, a kml file to define final geocoded product. If not found, it will use the coordinates above
```

```
#########
# PATHS #
#########
/$PATH_3601/PROCESS/MT          # PROROOTPATH, path to dir where data will be processed in sub dir named by the sat name (SATDIR).
/$PATH_1650/SAR_CSL/            # DATAPATH, path to dir where data are stored
/$PATH_DataSAR/SAR_AUX_FILES/DEM/SRTM30/ALL                    # DEMDIR, path to dir where DEM is stored
/$PATH_SCRIPTS/SCRIPTS_MT/FUNCTIONS_FOR_MT.sh                  # FCTFILE, path to file where all functions are stored

# for coregistration mass processing (required if coresitration on a Global Primary i.e. SuperMaster)
/$PATH_1650/SAR_SM/RESAMPLED            # RESAMPDATPATH, path to dir where resampled data will be stored

# for insar mass processing
/$PATH_3601/SAR_MASSPROCESS/            # MASSPROCESSPATH, path to dir where all processed pairs will be stored in sub dir named by the sat/trk name
```

Explanations:

AUTOMATIC FIGURES DISPLAY
############################

- **# FIG,**
  - (*FIGyes* or *FIGno*). Option to compute or not the quick look using *cpxfiddle*. If option is set to *FIGyes*, it will use *cpxfiddle* to create sunrasters figures that allow a quick look in the data. Note that some of these figures can be computed with a degraded resolution or can be displayed with distortion if multilooking in range and azimuth is not appropriate. The script created by AMSTer to generate these figures is usually stored in the directory where the file and its figure are. You can edit it (e.g. to change the multilooking, the resolution or the contrasts) and re-run it manually. Setting the option to *FIGno* will of course ignore the creation of these figures.
- **# POP,**
  - (*POPyes* or *POPno*). Option to pop up figures created with *cpxfiddle*. If option is set to *POPyes*, it will open each figure as soon as it is created. It is NOT advised to do this during a mass processing because it would overload the RAM.

DATA
#######

- **# SATDIR,**
  - Satellite system. It must be the same as the directory naming structure where the data are stored in CSL format (e.g. *CSK, ENVISAT, ERS, RADARSAT, S1, TDX, TSX,*…)
- **# TRKDIR,**
  - Track (or orbit) number/mode. This name must correspond to the subdir where data are stored in CSL format. (e.g. *DRC_Bukavu_A_174*)

For mass processing only
############################

- **# SUPERMASTER,**
  - Date of the Global Primary image (selected by ***Prepa_MSBAS.sh*** and to be found in the …/SAR_SM/MSBAS/*REGION*/seti/*setParametersFile.txt* created)
    → Only needed for mass processing with a Global Primary (SuperMaster). Needed also for ***Prepa_MSBAS.sh*** when this script is used to compute new pairs to add to an existing mass processing and when one does not want to compute a new Global Primary.

DEM
######

- **# DEMNAME,**
  - Name of the Digital Elevation Model in Lat-Long. If computed using GIS software (e.g. Envi), it must be inverted by lines and columns because math and GIS order differs… If DEM is created using *agregateSRTMTiles* from AMSTerEngine, it needs its *.txt* file in

same directory. Do not put a .txt and a .hdr in the same dir.
Do not put the path to DEM here; only the name (see below).

- **# SIMAMP,**
  - (*SIMAMPno* or *SIMAMPyes*). Option to compute Simulated Amplitude during External Dem Generation - usually not needed (maybe to assist ERS coregistration).
- **# RECOMPDEM,**
  - (*KEEP* or *FORCE*). Option to force recomputing the DEM in radar geometry (Slant Range DEM) and the projection of the mask in Slant Range if mask is required. *FORCE* obviously force to recompute the DEM (and mask if required) even if it/they already exist(s). If *KEEP* is selected, it will only compute the DEM and mask if it is not present yet.
  - Note: For obvious reasons, do not run two " *FORCE* " or a " *FORCE* " and a " *KEEP* " processes at the same time using same Primary image.

CROP
#######

- **# CROP,**
  (*CROPyes, CROPno* or *path_to_kml*). Option to crop images to area of interest. If option is set to *CROPyes*, images from all satellites (including Sentinel1 in Strip Map mode) but Sentinel1 in Wide Swath mode will be cropped based on the coordinates of the corners provided below as LINES/PIXELS numbers, or as LAT/LONG coordinates.
  To crop a Sentinel1 TOPSAR Wide Swath images, with a ZOOM factor of *1*, one must provide here a *path_to_kml*. The interferometric products and all subsequent products will be cropped at that footprint. Steps up to the S1coregistration will be performed on the whole set of tracks/swaths/bursts that span the area of interest defined at reading.
  To crop a Sentinel1 TOPSAR Wide Swath images, with a ZOOM factor other than *1*, one must provide the coordinates of the crop zone corners as LINES and PIXELS numbers (a bug in AMSTerEngine prevents so far defining crop zone using LAT/LONG coordinates).
- **# FIRSTP,**
- **# FIRSTL,**
- **# LASTP,**
- **# LASTL,**
  - Coordinates of the desired crop. Must be given as *Lat-Long*.
- **# ZOOM,**
  - Oversampling factor used during reading and cropping the image. This factor can be *any number* (not only integer).
    Combined with a multilooking factor, it can help to reduce noise (eg. ZOOM=2 and ML=3 → final resolution at 1.5)
  - Image oversampling (ZOOM) is possible for images from all satellites (including Sentinel1 in Strip Map mode) but Sentinel1 in Wide Swath mode by setting the CROP parameter as *CROPyes*, by providing the corners of the CROP region as LINES/PIXELS numbers or LAT/LONG coordinates, and by defining the ZOOM factor.
    Zooming Sentinel1 TOPSAR Wide Swath images can only be performed by providing the corners of the CROP region as LINES/PIXELS numbers (a bug in AMSTerEngine prevents so far defining crop zone using LAT/LONG coordinates).
  - ML factor of zoomed processing will then be ML of Zoomed pixels.
  - Keep *1* if no zoom is necessary.
- **# REGION,**
  - *Text* description of the area. Will be used for dir naming.

AMPLITUDE
#############

- **# MLAMPLI,**
  - Multilooking factor (*Integer number*) for amplitude images reduction during the coregistration (usually, a factor of 4 to 6 is appropriate).
    This is not the same as the Multilooking factor used for the computation of the interferometric product (see **INTERFML** below).
    If pixels are rectangular, MLAMPLI will be multiplied by corresponding ratio in Range or Azimuth.
    Convention: multilooking factor provided is always considered to be used to multiply the longest side of the pixel if the pixel is rectangular. Then, if the shape of the final pixel (see PIXSHAPE below) is requested to be square, the scripts will compute the multilooking ratio to use for the shortest side of the pixel taking into account the incidence angle.
    Example: For an Envisat images acquired in mode i7 (i.e. slant range sampling = 7.8m; Azimuth sampling = 3.2m; incidence angle =43°), a ML of 9 means 9 in range and 31 in Az.
    Note that Sentinel1 pixels have larger Azimuth side than Range pixel side.
- **# PIXSHAPE,**
  - (*SQUARE*, *ORIGINALFORM*, *SQUAREUNITY* or *ORIGINALFORMUNITY*). Shape of the pixels for the final products.
    If option is set to *SQUARE*, scripts will select a ratio between range and azimuth pixels size (taking also into account the looking angle) to multilook the pixels in order to have square-like pixels in the final products. If set to *ORIGINALFORM*, the **INTERFML** multilooking factor will be applied both in range and azimuth and the shape of the pixels in the final products will be similar to the original shape. *ORIGINALFORM* is useful for instance when looking at shadows in amplitude images to estimate height of objects. *SQUAREUNITY* and *ORIGINALFORMUNITY* are experimental options to force pixels at ML factor 1. Do not use it.
- **# CALIBSIGMA**
  - For calibrating images (S1 files only): *SIGMANO* or *SIGMAYES*. *SIGMAYES* will output sigma nought calibrated amplitude file.

COARSE COREG
#################

- **# CCOHWIN,**
  - (*Integer number*). Window size (in pixels) for the computation of the Coarse Coregistration (64 by default but may want less for very small crop).
    This parameter can be set to 0 to skip coarse coregistration when using good orbit satellites such as TSX, TDX and Envisat
- **# COH,**
  - (*Real number*). Coherence threshold for the computation of the Coarse coregistration
- **# CCDISTANCHOR,**
  - (*Integer number*). Distance (in pixels) in range and azimuth between anchor points for the computation of the Coarse coregistration (eg 24 for large images, 16 for medium and 2-8 for very small crops)

FINE COREG
#############

- **# FCOHWIN,**
  - (*Integer number*). Window size (in pixels) for the computation of the Fine Coregistration (eg 3 for ERS/ENVISAT, or 7 for CSK, TSX and RS; must have windows of -say- 50 pixels computed on full resolution images).

- **# FCOH,**
  - (*Real number*). Coherence threshold for the computation of the Fine coregistration
- **# FCDISTANCHOR,**
  - (*Integer number*). Distance (in pixels) in range and azimuth between anchor points for the computation of the Fine coregistration (eg 24 for large images, 16 for medium and 2-8 for very small crops)

INSAR
########
- **# PROCESSMODE,**
  - (*DEFO* or *TOPO*). Option to produce defo or topo interferograms (used for ***SinglePair.sh***)
- **# INITPOL,**
  - (*VV, HH, VH* or *HV*). For multi pol images; force selection of only the chosen polarisation at initInSAR for InSAR processing. If that polarization does not exist, it will find the first compatible PRM-SCD polarisation.
- **# LLRGCO,**
  - (*Integer number*). Lower Left Range coordinate offset (in pixels) for final interferometric products generation. It forces to crop the final products with this additional offset from the corner. Used only in ***SinglePairNoUnwrap.sh*** only for Shadow measurements
- **# LLAZCO,**
  - (*Integer number*). Lower Left Azimuth coordinate offset (in pixels) for final interferometric products generation. Used only in ***SinglePairNoUnwrap.sh*** only for Shadow measurements
- **# INTERFML,**
  - (*Integer number*). MultiLooking factor (ML) for interferometric products.
    Convention (see **MLAMPLI)**: select the INTERFML multilooking factor with respect to the largest side of your pixel. For instance, if you want a squared pixel of 50x50m, - with Envisat i2 (supposing that the original ground range pixel size is 2m in azimuth and 10m in range, i.e. a pixel ratio of 1/5), you would need an INTERFML factor of 5 and it will multiply the pixels in range by 5 and in azimuth by a factor 25.
    - in Sentinel (supposing that the original ground range pixel size is 14m in azimuth and 2m in range, i.e. a pixel ratio of 7), you would need a INTERFML factor of 4 and it will multiply the pixels in range by 28 and in azimuth by a factor 4. Of course, if you select a pixel in ORIGINALFORM, it will multiply the pixels in range and in azimuth by the factor INTERFML factor provided.
  - if ML=1, # COHESTIMFACT will be forced to 2
  - When used with zoom, it is ML to apply to zoomed pixels
- **# FILTFACTOR**
  - (*Integer number*; usually *1*). Filtering factor for filtering the interferogram. Factor 2 might be too strong when used with **POWSPECSMOOTFACT** filtering.
- **# POWSPECSMOOTFACT,**
  - (*Integer number*). Power spectrum filtering factor (for adaptive filtering) (*0* = no filtering; *1* or *<1* for filtering, smaller is stronger)
- **# COHESTIMFACT**
  - (*Integer number* in pixels). Number of pixels used to estimate the coherence. Must be similar to **INTERFML** as far as it is not a ML higher than 5 or 7 for instance. If INTERFML is larger than 5 or 7, limit anyway COHESTIMFACT to 5 or 7.

MASK
#######

- **# MASK**
  - (*APPLYMASKyes* or *APPLYMASKno*). The mask provided in Lat-Long will be projected in SlantRange.
  - The mask must be in Envi HARRIS format: a header text file (.hdr) and a binary 8-bit file.
  - If a mask is provided, snaphu (or CIS) also unwraps 0-masked pixels (for AMSTerEngine < 20230928), or 2-masked pixels (for AMSTerEngine > 20230928) if their coherence is above the **COHCLNTHRESH** (see below).
  - If computed manually, do not add extension at mask name. It must be something like *mask* and *mask.hdr,* not *mask.ext* and *mask.hdr*
  - Masks of coherence can be created using the script ***MultiLaunch_ForMask.sh***
  - If a mask requested but you do not use Snaphu, results will not be masked. One can however also mask manually files with *ffa* (eg *ffa residualInterferogram.HH-HH.f x slantRangeMask*)

- **# PATHTOMASK**
  - (*file path*). Path and file name of mask to be used to mask the coherence before unwrapping (when using snaphu).
  - Mask will be projected in slant range and hence can be used to mask the results of the interferometric computation, eg using the AMSTerEngine command *ffa*:
    ffa residualInterferogram.HH-HH.f x slantRangeMask

UNWRAPPING
############

- **# SKIPUW,**
  - (*SKIPyes, SKIPno* or *MASK*).
    SKIPyes will skip the time-consuming unwrapping step but will still geocode all the available and requested products. *SKIPno* will unwrap and geocode all available and requested products. *MASK* option is used to geocode only amplitude and coherence files; this is used for creating masks based on the coherence.
- **# UW_METHOD,**
  - (*SNAPHU, CIS, DETPHUN1ONLY, DETPHUN2ONLY, DETPHUN1SNAPHU, DETPHUN2SNAPHU, DETPHUN1CIS* or *DETPHUN2CIS*). Method used for the unwrapping. Snaphu is probably the most used method in the InSAR community. DETPHUN methods are fast but usually to be considered as preliminary step.

if snaphu unwrapping:

- **# DEFOTHRESHFACTOR,**
  - (*Real number*). Snaphu : Factor applied to rho0 to get threshold for whether or not phase discontinuity is possible. rho0 is the expected, biased correlation measure if true correlation is 0. Increase if results are not good. See snaphu literature for more information.
- **# DEFOCONST,**
  - (*Real number*). Snaphu : Ratio of phase discontinuity probability density to peak probability density expected for discontinuity-possible pixel differences. Value of 1 means zero cost for discontinuity, 0 means infinite cost. Decrease if results are not good. See snaphu literature for more information.
- **# DEFOMAX_CYCLE,**
  - (*Real number*). Snaphu : Maximum number of expected phase cycle discontinuity. For topo where no phase jump is expected, it can be set to zero. See snaphu literature for more information.
- **# SNAPHUMODE,**

- (*TOPO, DEFO, SMOOTH,* or *NOSTATCOSTS*). Snaphu : pre-defined configuration for snaphu depending of the type of interferogram expected. See snaphu literature for more information.

- **# ZONEMAP,**
  - *ZoneMapYes* or *ZoneMapNo*. if ZoneMapYes, it will create a map with the unwrapped zones named snaphuZoneMap. Each continuously unwrapped zone is numbered (from 1 to...)
- **# ZONEMAPSIZE,**
  - *(Real number)* Minimum size of snaphu unwrapped zone to map (in fraction of total nr of pixels)
- **# ZONEMAPCOST,**
  - *(Integer number)* Cost threshold for connected components (zones) obtained with snaphu. Higher threshold will give smaller connected zones
- **# ZONEMAPTOTAL,**
  - *(integer number)* Maximum number of mapped zones unwrapped with snaphu.

- **# MULTIUWP,**
  - MultiSnaphuYes or MultiSnaphuNo. If MultiSnaphuYes, it performs recursive ***snaphu*** unwrapping (need 4 params below). MultiUnwrapNo (or any other string) will perform single snaphu unwrapping
- **# WHICHINTERF,**
  - (*ResidInterf* or *ResidInterfFilt*). Which interferogram to unwrap: residual interferogram or Filtered Residual interferogram
- **# COEFREQ,**
  - (*Real number*). Coefficient of increase of cut-off frequency (e.g 0.9)
- **# CUTINI,**
  - (*Real number*). Initial cut-off frequency (e.g. 12.5 for a 400x400 image, 10 for a 2200x1500 img)
- **# NITMAX,**
  - (*Integer number*). Maximum total number of iterations
- **# COHMUWPTHRESH,**
  - (*Real number*). Coherence threshold (between 0 and 1) below which it replaces the phase by white noise (corresponding mask will be produced). If set to 0, do not mask with white noise

if snaphu or CIS unwrapping:

- **# COHCLNTHRESH,**
  - (*Real number*). Will use the coherence in conjunction with the mask at unwrapping. Pixels with coherence above threshold OR where mask = 1 will be unwrapped. If COHCLNTHRESH is set to 1, it will only use the mask provided without taking into account the coherence:

Mask	Coh > COHCLNTHRESH	unwrapping
1	1	Yes
1	0	Yes
0	1	Yes
0	0	No

Note: Do not set coherence threshold to 1 when using CIS unwrapping method !

Warning: since AMSTerEngine V20230921, masks are inverted and multi-layers, that is:

- 0 is to keep (no mask applied)
- 1 is always masked (e.g. water bodies area)
- 2 is masked during unwrapping except is coherence is above a given threshold (see parameter COHCLNTHRESH in *LaunchMTparam.txt*)

That is:

Mask	Coh > *COHCLNTHRESH*	unwrapping
0	yes	Yes
0	no	Yes
1	yes	No
1	no	No
2	yes	Yes
2	no	No

if CSL unwrapping:

- **# FALSERESCOHTHR,**
  - (*Real number*). False Residue Coherence Threshold: higher is much slower. Use max 0.15 e.g. in crater.
- **# CONNEXION_MODE,**
  - (*Integer number*). Number of times that connection search radius is augmented when stable connections are found ; *0* search along all coherent zone.
- **# BIASCOHESTIM,**
  - (*Integer number*). Biased coherence estimator range and Az window size
- **# BIASCOHSPIR,**
  - (*Integer number*). Biased coherence square spiral size (if residual fringes are not unwrapped, decrease it; must be odd)

if DETPHUN unwrapping:

- **# DETITERR,**
  - (*Integer number*). Number of iterations for detPhUn (*1, 2* or *3* is generally OK).
- **# DETCOHTHRESH,**
  - (*Real number*). Coherence threshold.

- **# INTERPOL,**
  - (*BOTH, BEFORE* or *AFTER*). Interpolate the unwrapped interferogram *BEFORE* or *AFTER* geocoding or *BOTH*. It allows to reduce very small gaps in the data. In principle the function developed by Ludivine Libert could be applied several times repeatedly, but the scripts only apply it once.
- **# REMOVEPLANE,**
  - (*DETREND* or *???*). If option set to *DETREND,* it will remove a best plane after unwrapping. Anything else will ignore the detrending.
  - Note: selecting DETREND is equivalent to the calibration in MSBAS with C_FLAG= 10. In such a case, you can set C_FLAG at zero when processing msbas, which should speed up the msbas processing, but also would allow you to split the data in smaller batches in case of insufficient RAM. You can for instance run 4 msbas with the WINDOW_SIZE set to each quarter of your foot print then assemble the results. Reminder : using C_FLAG = 10 or DETREND means setting the interferogram averages to zero. If you prefer calibrate the interferograms with another method (one or more points or region), then splitting the msbas will not be possible.

GEOCODING
##########

- **# PROJ,**
  - (*UTM* or *LATLONG*). Projection for geocoded products. Although AMSTerEngine is able to handle both UTM and Lat-Long, only UTM is implemented in the scripts. Whatever you set up here, it will be processed as *UTM* anyway… The only exception is the geocoding of amplitude images using ***Geocode_from_ALL2GIF.sh*** after usage of ***ALL2GIF.sh***.

- **# GEOCMETHD,**
  - (*Forced, Auto, Closest* or *ClosestMassProc*).
    Resampling Size of Geocoded pixels:
    - *Forced*: will force pixels at size **FORCEGEOPIXSIZE** (see below) and along a grid at fixed size and position (see **XMIN**, **XMAX**, **YMIN**, **YMAX** below). This is the mandatory selection for MSBAS because it requires the crop of the geocoded interferometric product to be similar for all images acquired by all type of sensors, in all geometries.
      In that case, **FORCEGEOPIXSIZE** (in meters) must be chosen the closest as possible to the size of the multilooked pixel size.
    - *Auto*: Will force the size of the geocoded pixels to be the closest multiple of 10m to the size of the multilooked pixel size.
    - *Closest* : Will force the size of the geocoded pixels to be the closest to the size of the multilooked pixel size.
    - *ClosestMassProc:* Will force the size of the geocoded pixels to be the closest to the size of the multilooked pixel size, even for a Mass Processing. Usually for a mass processing, which aims at performing MSBAS after, it is mandatory to have all the pixels forced at the same size, for all the sensors and geometries. Hence a security would prevent you to make a mass processing (using ***SuperMaster_MassProc.sh***) without *Forced* option. *ClosestMassProc* allowed to override this security.

- **# RADIUSMETHD,**
  - (*LetCIS* or *Integer*).
    LetCIS let AMSTerEngine chose the best (adaptive) interpolation radius (default and preferred option). If you set an *Integer* number, it forces the use of the same radius throughout the whole image. Note however than if chosen too small, it will be fast but will create gaps in slopy terrain. If too large, it will avoid the gaps but may become very slow. Usually, prefer the adaptive automatic solution.

- **# RESAMPMETHD,**
  - (*TRI, AV* or *NN*). Triangulation, weighted average or nearest neighbour.

- **# WEIGHTMETHD,**
  - (*ID* or *LORENTZ*). Weighting method: inverse distance or = Lorentzian

- **# IDSMOOTH,**
  - (*Real*). ID smoothing factor

- **# IDWEIGHT,**
  - (*Real*). ID weighting exponent

- **# FWHM,**
  - (*Real*). Lorentzian Full Width at Half Maximum. Having less than one (eg. 0.5 or 0.1) may provide less blurred geocoded images

- **# ZONEINDEX,**
  - (*Integer*). Zone index

- **#FORCEGEOPIXSIZE,**
  - (*Integer*). Pixel size (in meters) for geocoded products.

- **#UTMZONE,**
  - (*letter* followed by *Integer*). Letter of row and nr of col of the zone where coordinates below are computed (e.g. U32). This is not mandatory unless your area of interest intersect two UTM zones and the middle of the image is not in the same UTM zone depending on the track/mode used. If not set, the software will compute by itself the UTM zone. If provided, when using GEOCMETHD Forced, the provided UTM coordinates will be computed from the provided UTM zone. In most of the case, you will not need that parameter and it can be left empty. Only use it if your Forced option does not provide you with the expected results. Another way to securely Force the geocoding on an area spanning different UTM zones is to force cropping based on a kml file, which does not suffer from a possible UTM zone confusion.

- **# XMIN,**
- **# XMAX,**
- **# YMIN,**
- **# YMAX,**
  - (*Integer*). Minimum/maximum X and Y UTM coordinates of final geocoded product

- **# GEOCKML,**
  - (*path_to_kml_file*). A kml file to define final forced geocoded product. If file is not found, it will use the coordinates above.

#########
PATHS
#########

- **# PROROOTPATH,**
  - (*Path*). Path to directory where data will be processed in sub directories named by the SAT/TRACK/Crop etc… name. If you run several processes at the same time (either the same SAT/TRK/REGION split in several session using __***SplitSession.sh*** or running several different SAT/TRK/REGION), it is advised to process them on separate disks. Read and Write accesses are indeed often the bottleneck for a fast parallel processing.
    (example: *${HOME}/PROCESS/MT* and/or $PATH_3600/PROCESS/MT*)*
- **# DATAPATH,**
  - (*Path*). Path to directory where data are stored in CSL format
    (example: */$PATH_1650/SAR_CSL/)*
- **# DEMDIR,**
  - (*Path*). Path to directory where DEM is stored.
    (example: */$PATH_DataSAR/SAR_AUX_FILES/DEM/SRTM30/ALL)*
- **# FCTFILE,**
  - (*Path to file*). Path to shell script file where all functions are defined.
    (example: */$PATH_SCRIPTS/SCRIPTS_MT/FUNCTIONS_FOR_MT.sh)*

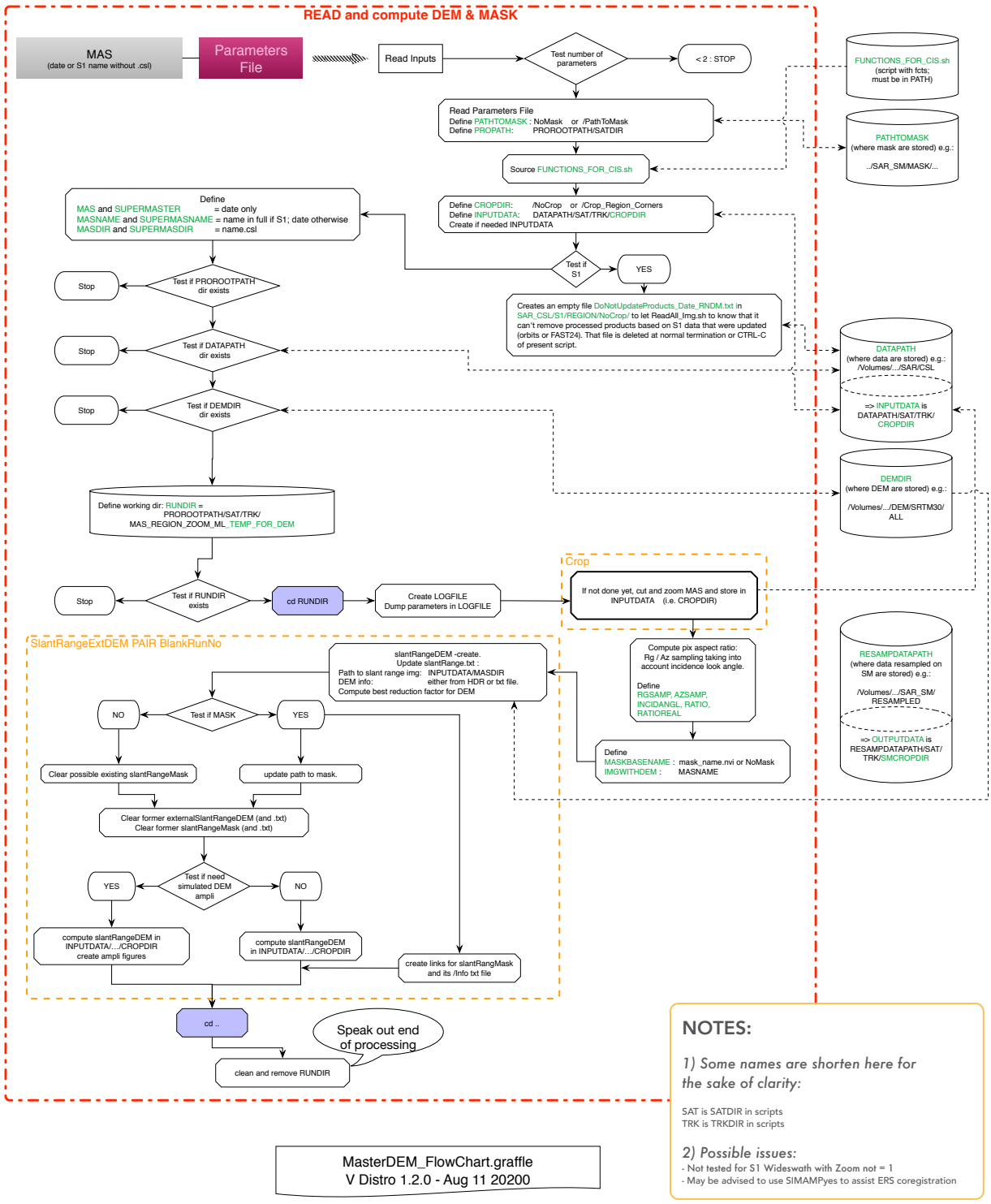
for coregistration mass processing (required if coregistration on a Global Primary i.e. SuperMaster)

- **# RESAMPDATPATH,**
  - (*Path*). path to directory where resampled data will be stored.
    (example: */$PATH_1650/SAR_SM/RESAMPLED)*

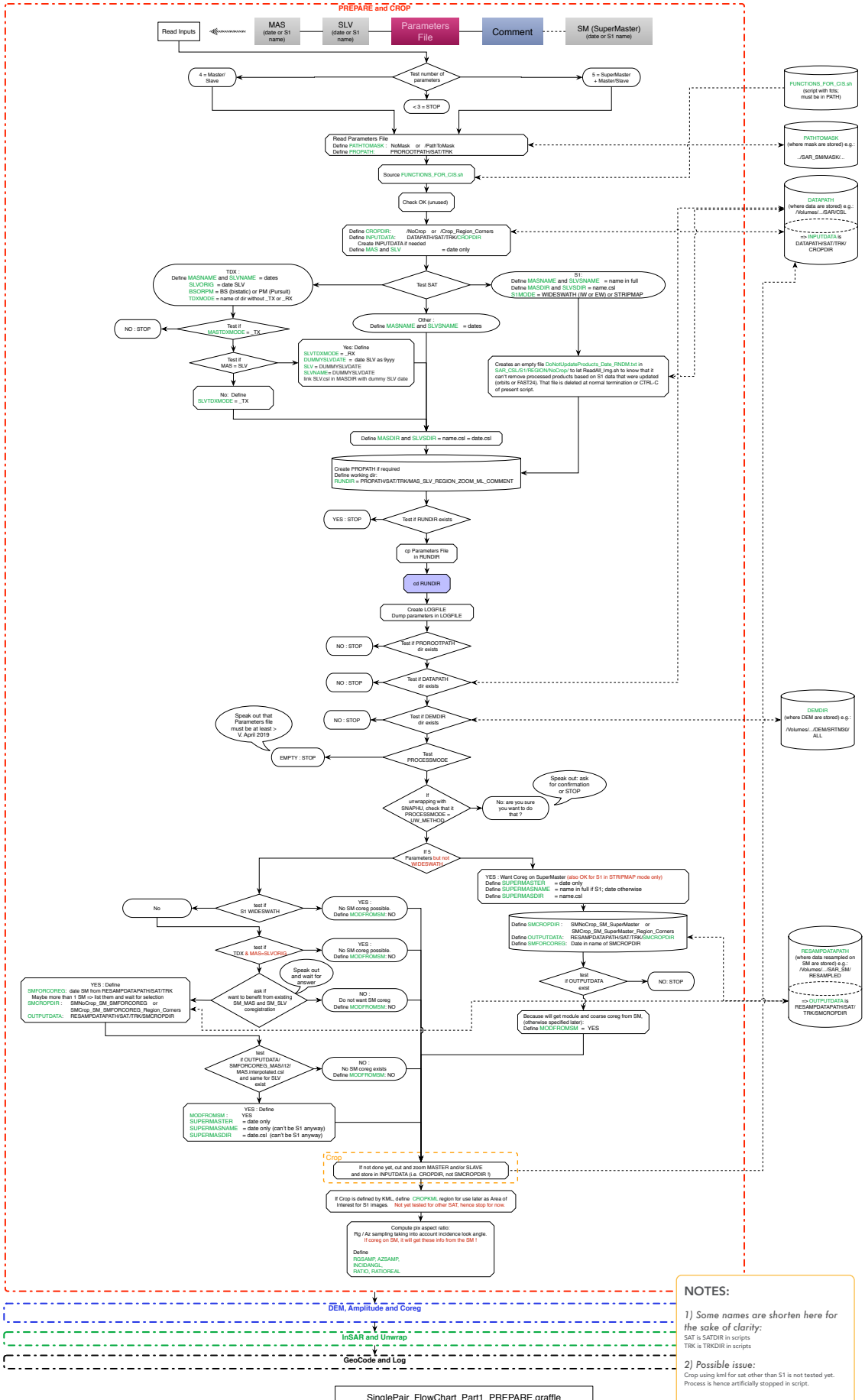
for InSAR mass processing

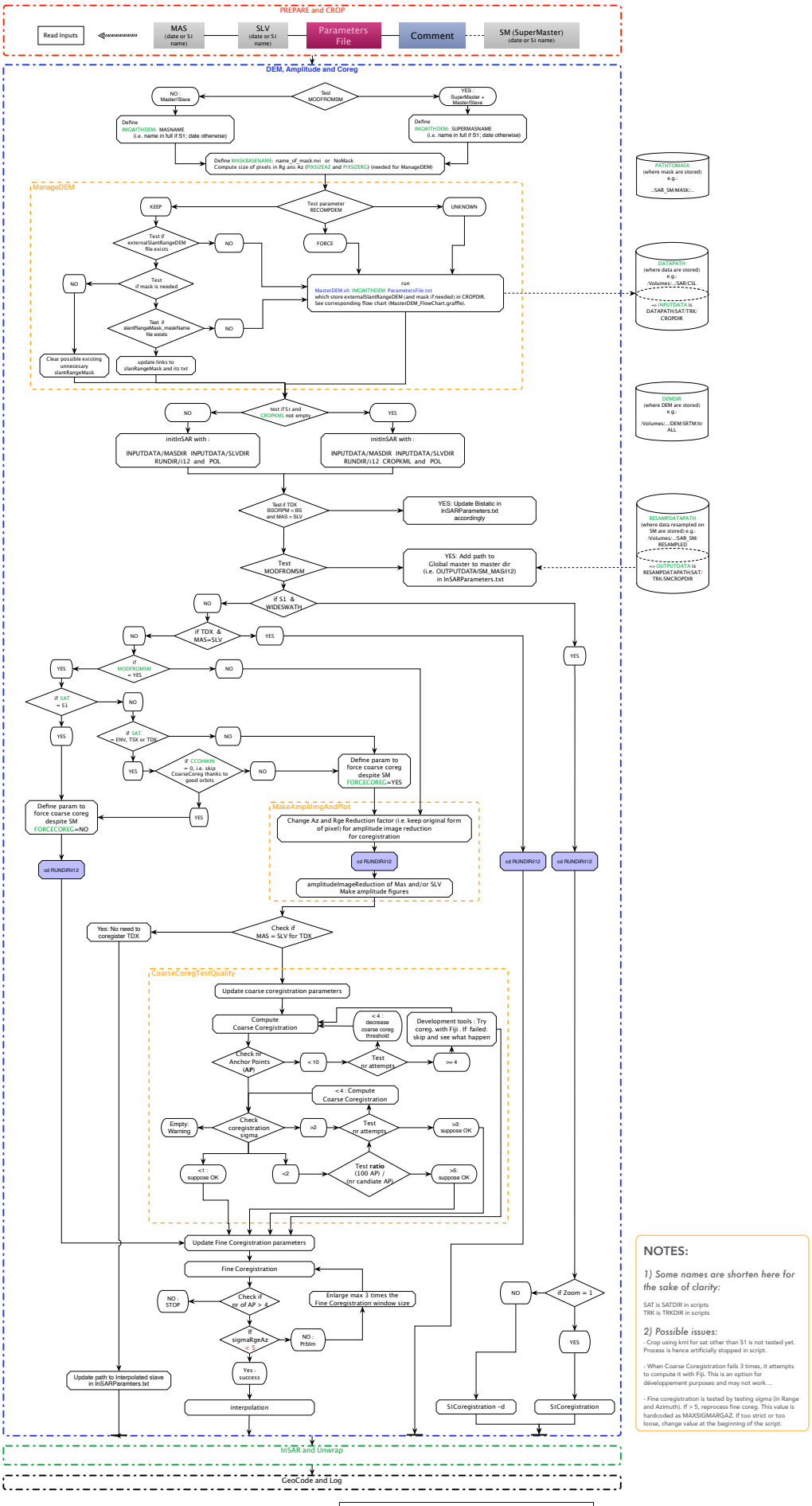
- **# MASSPROCESSPATH,**
  - (*Path*). Path to directory where all processed pairs will be stored in sub directory named by the SAT/TRACK name (SATDIR/TRKDIR).
    (example: */$PATH_3601/SAR_MASSPROCESS/)*

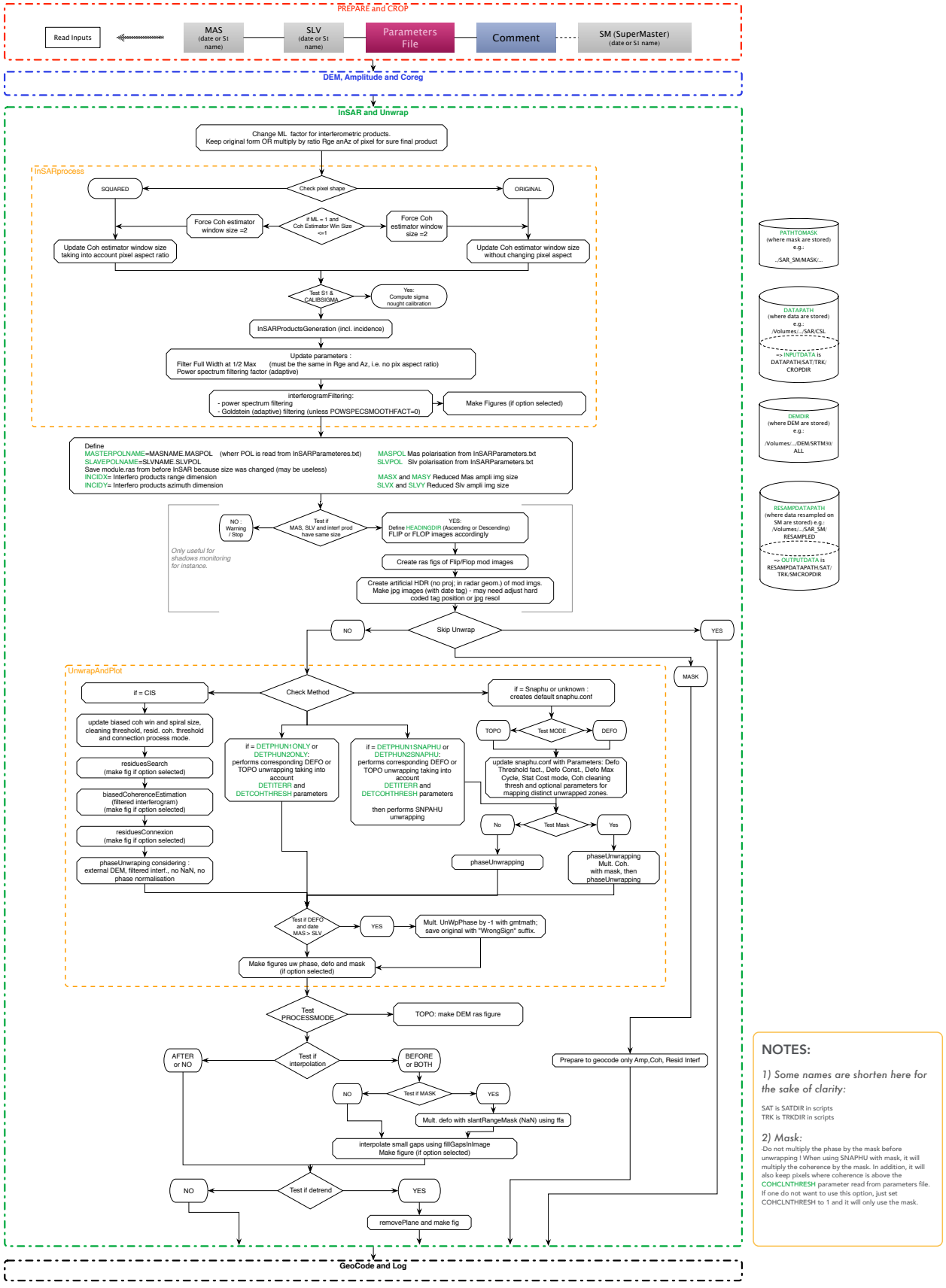
A.2) Flow chart of *MasterDEM.sh*

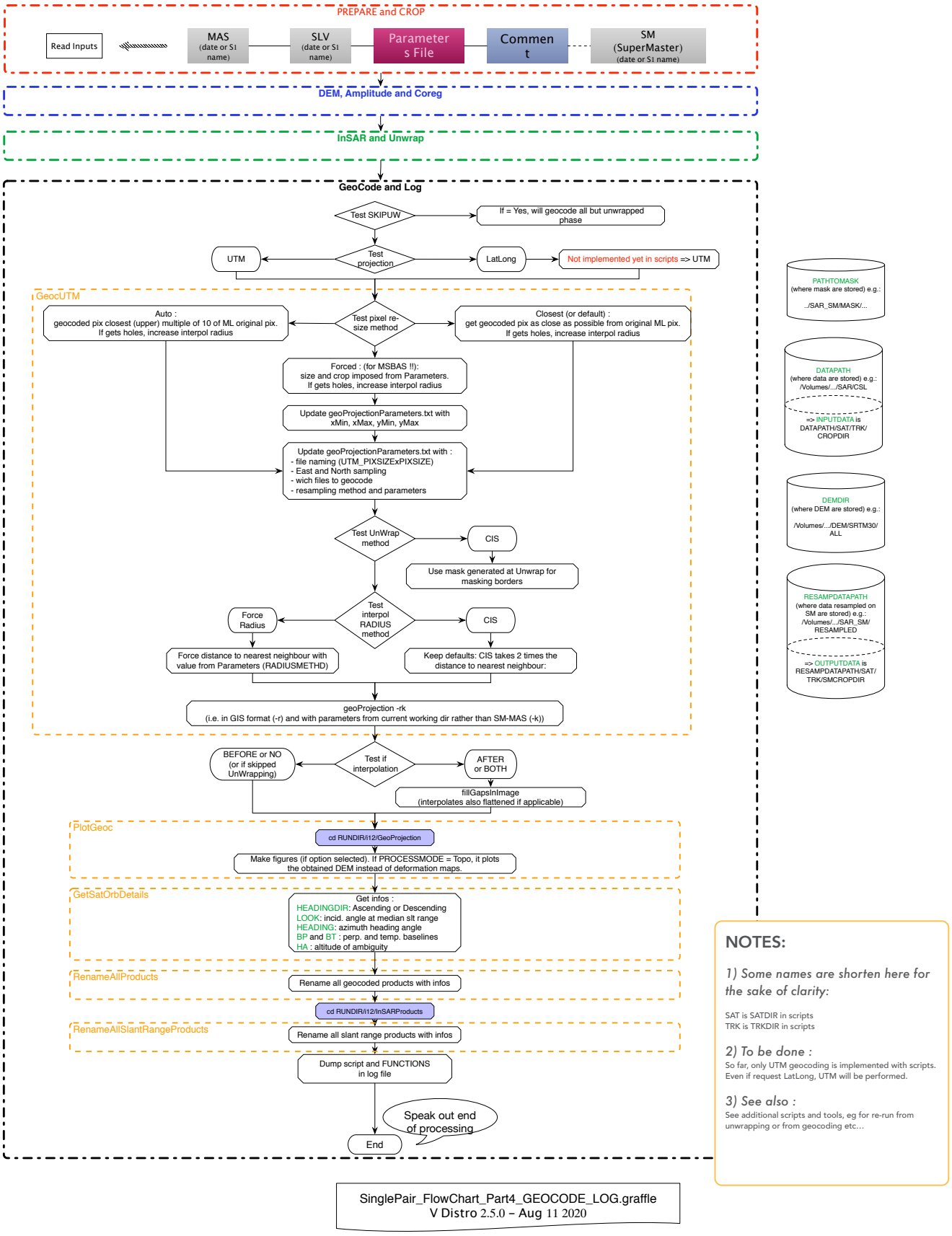


A.3) Flow chart of *SinglePair.sh*

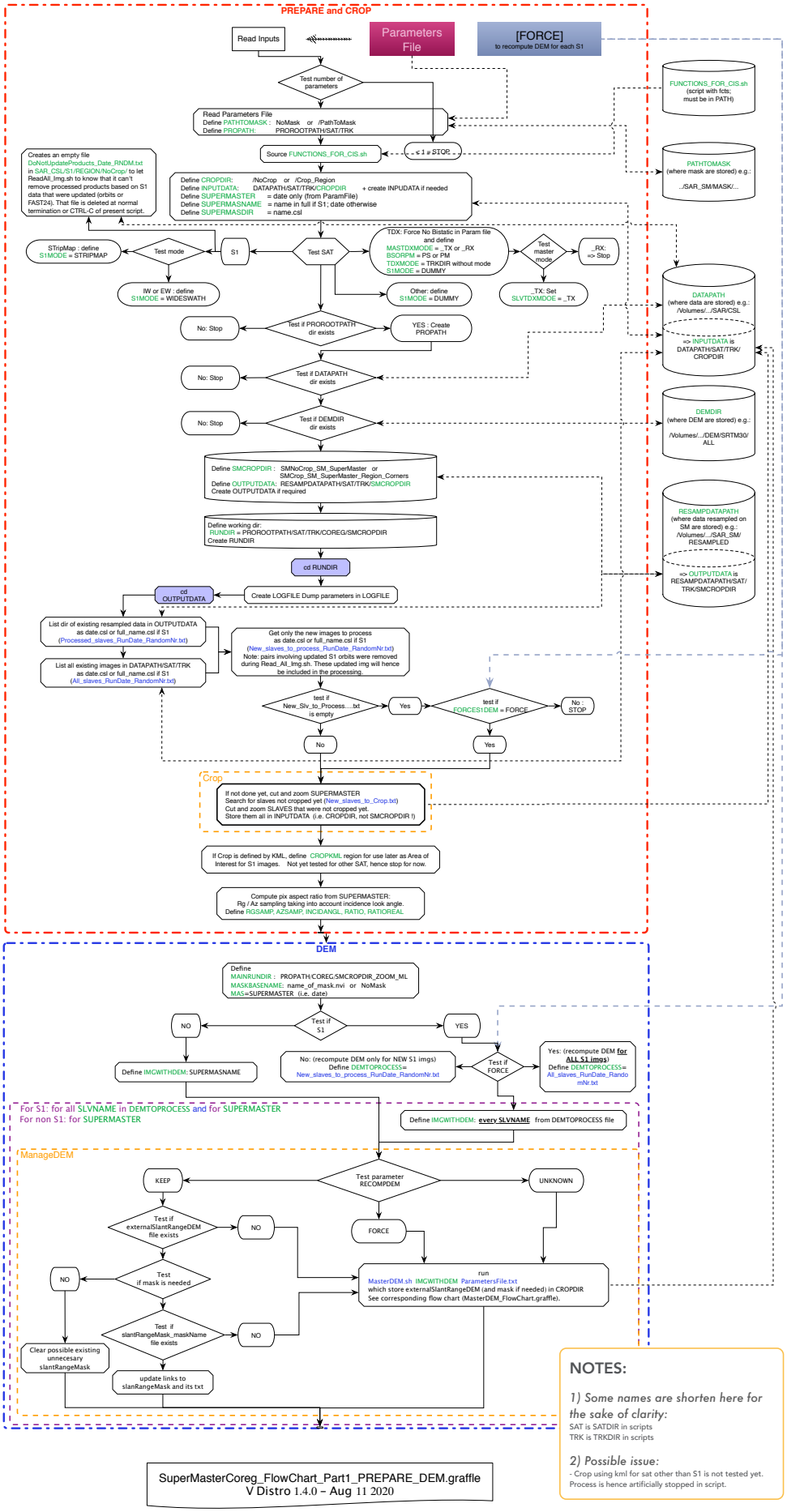


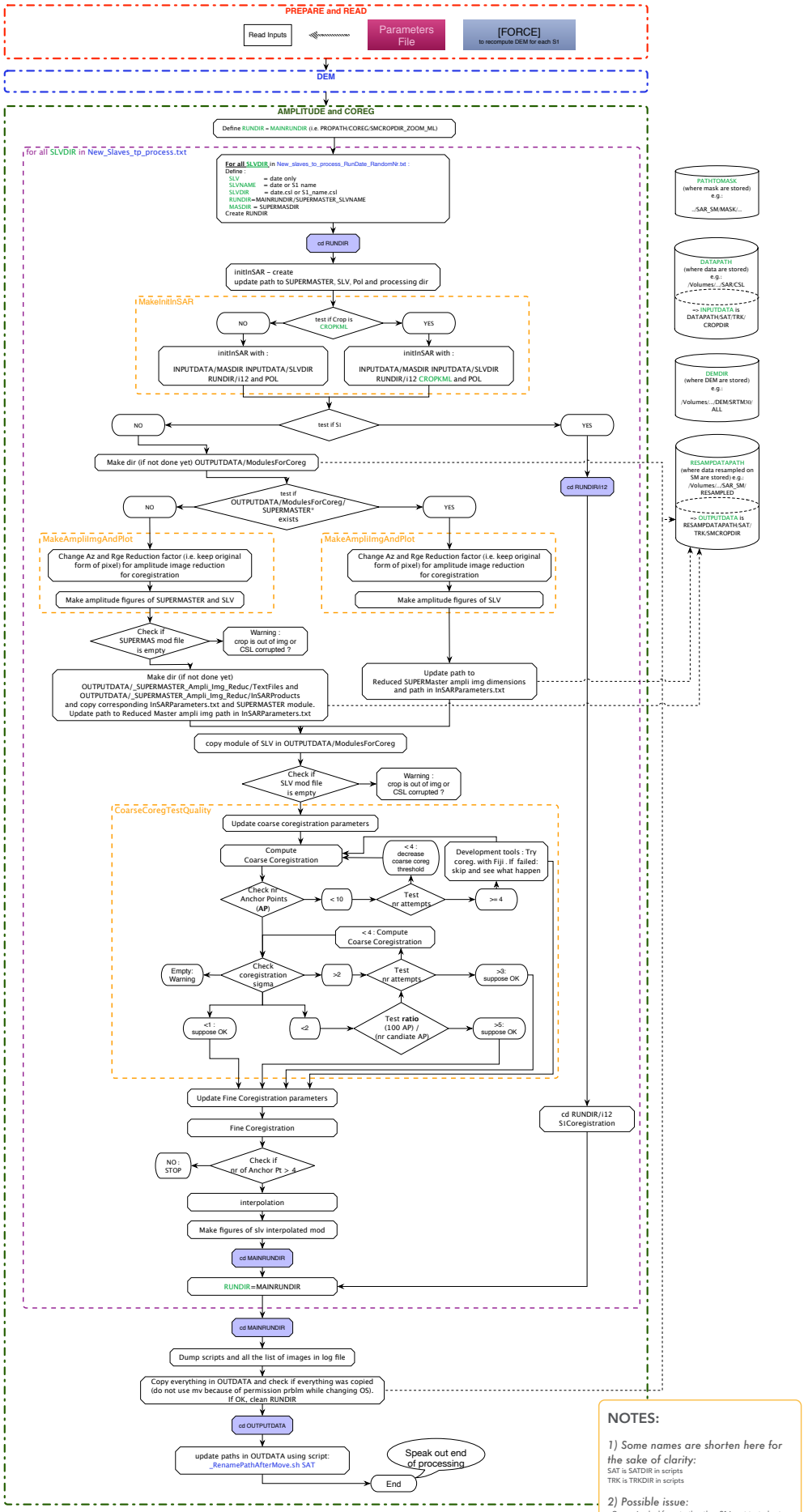

PREPARE and CROP
Read Inputs
MAS (date or S1 name)
SLV (date or S1 name)
Parameters File
Comment
SM (SuperMaster) (date or S1 name)
DEM, Amplitude and Coreg
Test MODFROMSM
NO : Master/Slave
YES : SuperMaster + Master/Slave
Define IMGWITHDEM: MASNAME (i.e. name in full if S1; date otherwise)
Define IMGWITHDEM: SUPERMASNAME (i.e. name in full if S1; date otherwise)
Define MASKBASENAME: name_of_mask.nvi or NoMask
Compute size of pixels in Rg ans Az (PIXSIZEAZ and PIXSIZERG) (needed for ManageDEM)
PATHTOMASK (where mask are stored) e.g. : ...SAR_SM:MASK:...
ManageDEM
Test parameter RECOMPDEM
KEEP
UNKNOWN
FORCE
Test if externalSlantRangeDEM file exists
Test if mask is needed
Test if slantRangeMask_maskName file exists
NO
Clear possible existing unnecessary slantRangeMask
update links to slantRangeMask and its txt
run MasterDEM.sh IMGWITHDEM ParametersFile.txt which store externalSlantRangeDEM (and mask if needed) in CROPDIR. See corresponding flow chart (MasterDEM_FlowChart.graffle).
DATAPATH (where data are stored) e.g.: /Volumes: ../SAR/CSL
=> INPUTDATA is DATAPATH:SAT/TRK/CROPDIR
DEMDIR (where DEM are stored) e.g.: /Volumes:.../DEM:SRTM:N/ALL
test if S1 and CROPKML not empty
initInSAR with : INPUTDATA/MASDIR INPUTDATA/SLVDIR RUNDIR/i12 and POL
initInSAR with : INPUTDATA/MASDIR INPUTDATA/SLVDIR RUNDIR/i12 CROPKML and POL
Test if TDX BISORPM = BIS and MAS = SLV
YES: Update Bistatic in InSARParameters.txt accordingly
RESAMPDATAPATH (where data resampled on SM are stored) e.g.: /Volumes:.../SAR_SM/RESAMPLED
=> OUTPUTDATA is RESAMPDATAPATH:SAT/TRK:SMCROPDIR
Test MODFROMSM
YES: Add path to Global master to master dir (i.e. OUTPUTDATA/SM_MAS/i12) in InSARParameters.txt
if S1 & WIDESWATH
if TDX & MAS=SLV
if MODFROMSM = YES
if SAT = S1
if SAT = ENV, TSX or TDX
if CCOHWIN = 0, i.e. skip CoarseCoreg thanks to good orbits
Define param to force coarse coreg despite SM FORCECOREG=YES
Define param to force coarse coreg despite SM FORCECOREG=NO
cd RUNDIR/i12
MakeAmplitImgAndPlot
Change Az and Rge Reduction factor (i.e. keep original form of pixel) for amplitude image reduction for coregistration
amplitudeImageReduction of Mas and/or SLV Make amplitude figures
Check if MAS = SLV for TDX
Yes: No need to coregister TDX
CoarseCoregTestQuality
Update coarse coregistration parameters
Compute Coarse Coregistration
< 4 : decrease coarse coreg threshold
Development tools : Try coreg. with Fiji . If failed: skip and see what happen
Check nr Anchor Points (AP)
< 10
Test nr attempts
>= 4
< 4 : Compute Coarse Coregistration
Empty: Warning
Check coregistration sigma
>2
Test nr attempts
>3: suppose OK
<1 : suppose OK
<2
Test ratio (100 AP) / (nr candiate AP)
>5: suppose OK
Update Fine Coregistration parameters
Fine Coregistration
NO : STOP
Check if nr of AP > 4
Enlarge max 3 times the Fine Coregistration window size
if sigmaRgeAz < 5
NO : Prblm
Yes : success
Update path to Interpolated slave in InSARParamters.txt
interpolation
if Zoom = 1
SICoregistration -d
SICoregistration
NOTES:
1) Some names are shorten here for the sake of clarity:
SAT is SATDIR in scripts
TRK is TRKDIR in scripts
2) Possible issues:
- Crop using kml for sat other than S1 is not tested yet. Process is hence artificially stopped in script.
- When Coarse Coregistration fails 3 times, it attempts to compute it with Fiji. This is an option for développement purposes and may not work...
- Fine coregistration is tested by testing sigma (in Range and Azimuth). If > 5, reprocess fine coreg. This value is hardcoded as MAXSIGMARGAZ. If too strict or too loose, change value at the beginning of the script.
InSAR and Unwrap
GeoCode and Log
SinglePair_FlowChart_Part2_DEM_COREG.graffle
V Distro 2.5.0 – Aug 11 2020


PREPARE and CROP
Read Inputs
MAS (date or S1 name)
SLV (date or S1 name)
Parameters File
Comment
SM (SuperMaster) (date or S1 name)
DEM, Amplitude and Coreg
InSAR and Unwrap
Change ML factor for interferometric products. Keep original form OR multiply by ratio Rge anAz of pixel for sure final product
InSARprocess
Check pixel shape
SQUARED
ORIGINAL
Force Coh estimator window size =2
if ML = 1 and Coh Estimator Win Size <>1
Force Coh estimator window size =2
Update Coh estimator window size taking into account pixel aspect ratio
Update Coh estimator window size without changing pixel aspect
Test S1 & CALIBSIGMA
Yes: Compute sigma nought calibration
InSARProductsGeneration (incl. incidence)
Update parameters : Filter Full Width at 1/2 Max (must be the same in Rge and Az, i.e. no pix aspect ratio) Power spectrum filtering factor (adaptive)
interferogramFiltering: - power spectrum filtering - Goldstein (adaptive) filtering (unless POWSPECSMOOTHFACT=0)
Make Figures (if option selected)
Define MASTERPOLNAME=MASNAME.MASPOL (wherr POL is read from InSARParameteres.txt) SLAVEPOLNAME=SLVNAME.SLVPOL Save module.ras from before InSAR because size was changed (may be useless) INCIDX= Interfero products range dimension INCIDY= Interfero products azimuth dimension
MASPOL Mas polarisation from InSARParameters.txt
SLVPOL Slv polarisation from InSARParameters.txt
MASX and MASY Reduced Mas ampli img size
SLVX and SLVY Reduced Slv ampli img size
Only useful for shadows monitoring for instance.
NO : Warning / Stop
Test if MAS, SLV and interf prod have same size
YES: Define HEADINGDIR (Ascending or Descending) FLIP or FLOP images accordingly
Create ras figs of Flip/Flop mod images
Create artificial HDR (no proj; in radar geom.) of mod imgs. Make jpg images (with date tag) - may need adjust hard coded tag position or jpg resol
Skip Unwrap
NO
YES
MASK
UnwrapAndPlot
if = CIS
Check Method
if = Snaphu or unknown : creates default snaphu.conf
update biased coh win and spiral size, cleaning threshold, resid. coh. threshold and connection process mode.
residuesSearch (make fig if option selected)
biasedCoherenceEstimation (filtered interferogram) (make fig if option selected)
residuesConnexion (make fig if option selected)
phaseUnwraping considering : external DEM, filtered interf., no NaN, no phase normalisation
if = DETPHUN1ONLY or DETPHUN2ONLY: performs corresponding DEFO or TOPO unwrapping taking into account DETITERR and DETCOHTHRESH parameters
if = DETPHUN1SNAPHU or DETPHUN2SNAPHU: performs corresponding DEFO or TOPO unwrapping taking into account DETITERR and DETCOHTHRESH parameters then performs SNPAHU unwrapping
TOPO
Test MODE
DEFO
update snaphu.conf with Parameters: Defo Threshold fact., Defo Const., Defo Max Cycle, Stat Cost mode, Coh cleaning thresh and optional parameters for mapping distinct unwrapped zones.
No
Test Mask
Yes
phaseUnwrapping
phaseUnwrapping Mult. Coh. with mask, then phaseUnwrapping
Test if DEFO and date MAS > SLV
YES
Mult. UnWpPhase by -1 with gmtmath; save original with "WrongSign" suffix.
Make figures uw phase, defo and mask (if option selected)
Test PROCESSMODE
TOPO: make DEM ras figure
AFTER or NO
Test if interpolation
BEFORE or BOTH
Prepare to geocode only Amp,Coh, Resid Interf
NO
Test if MASK
YES
Mult. defo with slantRangeMask (NaN) using ffa
interpolate small gaps using fillGapsInImage Make figure (if option selected)
NO
Test if detrend
YES
removePlane and make fig
GeoCode and Log
PATHTOMASK (where mask are stored) e.g.: ../SAR_SM/MASK/...
DATAPATH (where data are stored) e.g.: /Volumes/.../SAR/CSL
=> INPUTDATA is DATAPATH/SAT/TRK/CROPDIR
DEMDIR (where DEM are stored) e.g.: /Volumes/.../DEM/SRTM30/ALL
RESAMPDATAPATH (where data resampled on SM are stored) e.g.: /Volumes/.../SAR_SM/RESAMPLED
=> OUTPUTDATA is RESAMPDATAPATH/SAT/TRK/SMCROPDIR
NOTES:
1) Some names are shorten here for the sake of clarity:
SAT is SATDIR in scripts
TRK is TRKDIR in scripts
2) Mask:
-Do not multiply the phase by the mask before unwrapping ! When using SNAPHU with mask, it will multiply the coherence by the mask. In addition, it will also keep pixels where coherence is above the COHCLNTHRESH parameter read from parameters file. If one do not want to use this option, just set COHCLNTHRESH to 1 and it will only use the mask.
SinglePair_FlowChart_Part3_INSAR.graffle
V Distro 2.5.0 – Aug 11 2020

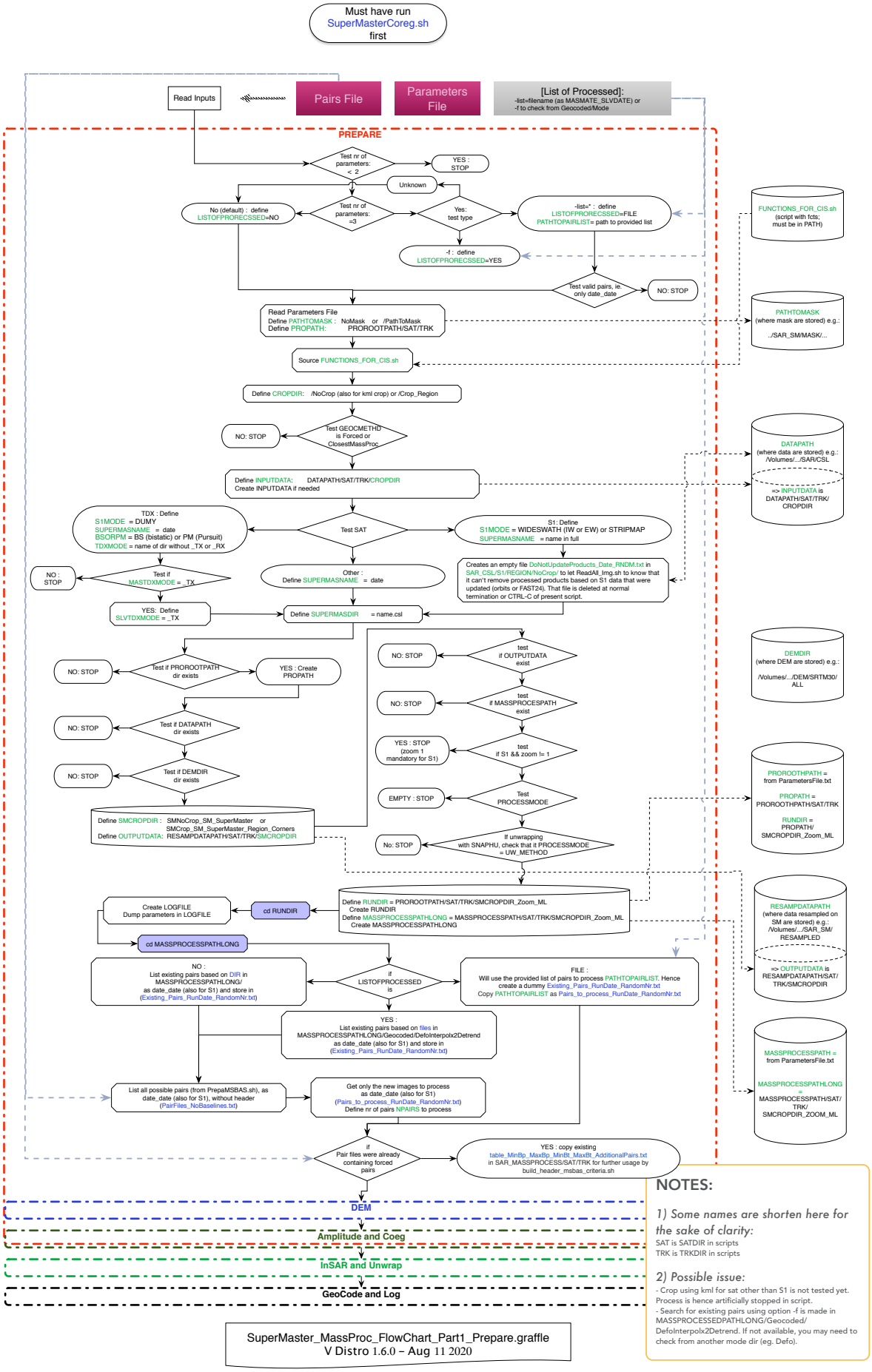

PREPARE and CROP
Read Inputs
MAS (date or S1 name)
SLV (date or S1 name)
Parameters File
Comment
SM (SuperMaster) (date or S1 name)
DEM, Amplitude and Coreg
InSAR and Unwrap
GeoCode and Log
Test SKIPUW
If = Yes, will geocode all but unwrapped phase
Test projection
UTM
LatLong
Not implemented yet in scripts => UTM
GeocUTM
Test pixel re-size method
Auto : geocoded pix closest (upper) multiple of 10 of ML original pix. If gets holes, increase interpol radius
Closest (or default) : get geocoded pix as close as possible from original ML pix. If gets holes, increase interpol radius
Forced : (for MSBAS !!): size and crop imposed from Parameters. If gets holes, increase interpol radius
Update geoProjectionParameters.txt with xMin, xMax, yMin, yMax
Update geoProjectionParameters.txt with : - file naming (UTM_PIXSIZExPIXSIZE) - East and North sampling - wich files to geocode - resampling method and parameters
Test UnWrap method
CIS
Use mask generated at Unwrap for masking borders
Test interpol RADIUS method
Force Radius
CIS
Force distance to nearest neighbour with value from Parameters (RADIUSMETHD)
Keep defaults: CIS takes 2 times the distance to nearest neighbour:
geoProjection -rk (i.e. in GIS format (-r) and with parameters from current working dir rather than SM-MAS (-k))
Test if interpolation
BEFORE or NO (or if skipped UnWrapping)
AFTER or BOTH
fillGapsInImage (interpolates also flattened if applicable)
PlotGeoc
cd RUNDIR/i12/GeoProjection
Make figures (if option selected). If PROCESSMODE = Topo, it plots the obtained DEM instead of deformation maps.
GetSatOrbDetails
Get infos :
HEADINGDIR: Ascending or Descending
LOOK: incid. angle at median slt range
HEADING: azimuth heading angle
BP and BT : perp. and temp. baselines
HA : altitude of ambiguity
RenameAllProducts
Rename all geocoded products with infos
cd RUNDIR/i12/InSARProducts
RenameAllSlantRangeProducts
Rename all slant range products with infos
Dump script and FUNCTIONS in log file
Speak out end of processing
End
PATHTOMASK (where mask are stored) e.g.: ../SAR_SM/MASK/...
DATAPATH (where data are stored) e.g.: /Volumes/.../SAR/CSL
=> INPUTDATA is DATAPATH/SAT/TRK/CROPDIR
DEMDIR (where DEM are stored) e.g.: /Volumes/.../DEM/SRTM30/ALL
RESAMPDATAPATH (where data resampled on SM are stored) e.g.: /Volumes/.../SAR_SM/RESAMPLED
=> OUTPUTDATA is RESAMPDATAPATH/SAT/TRK/SMCROPDIR
NOTES:
1) Some names are shorten here for the sake of clarity:
SAT is SATDIR in scripts
TRK is TRKDIR in scripts
2) To be done :
So far, only UTM geocoding is implemented with scripts. Even if request LatLong, UTM will be performed.
3) See also :
See additional scripts and tools, eg for re-run from unwrapping or from geocoding etc...
SinglePair_FlowChart_Part4_GEOCODE_LOG.graffle
V Distro 2.5.0 – Aug 11 2020

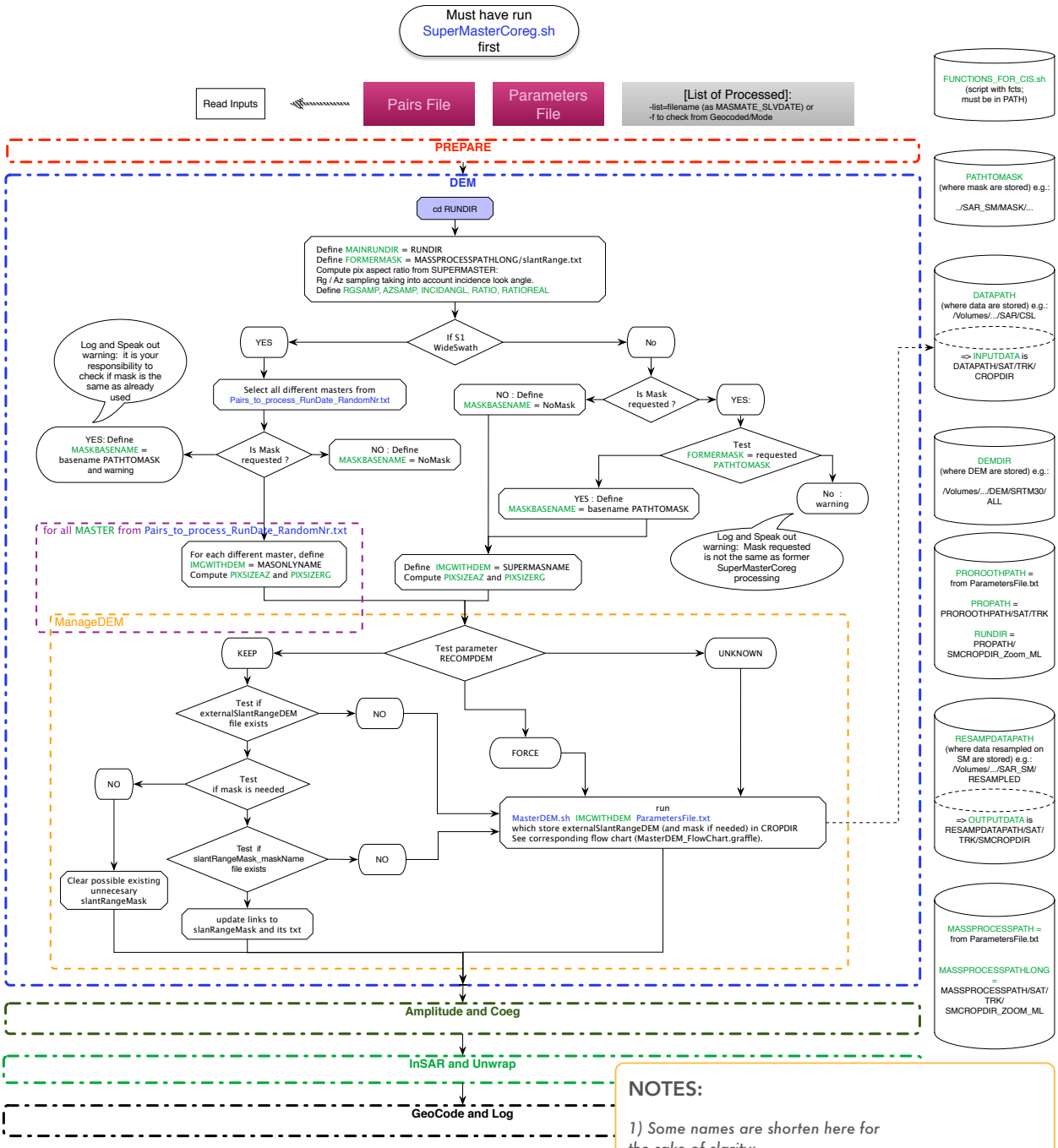
A.4) Flow chart of *SuperMasterCoreg.sh*

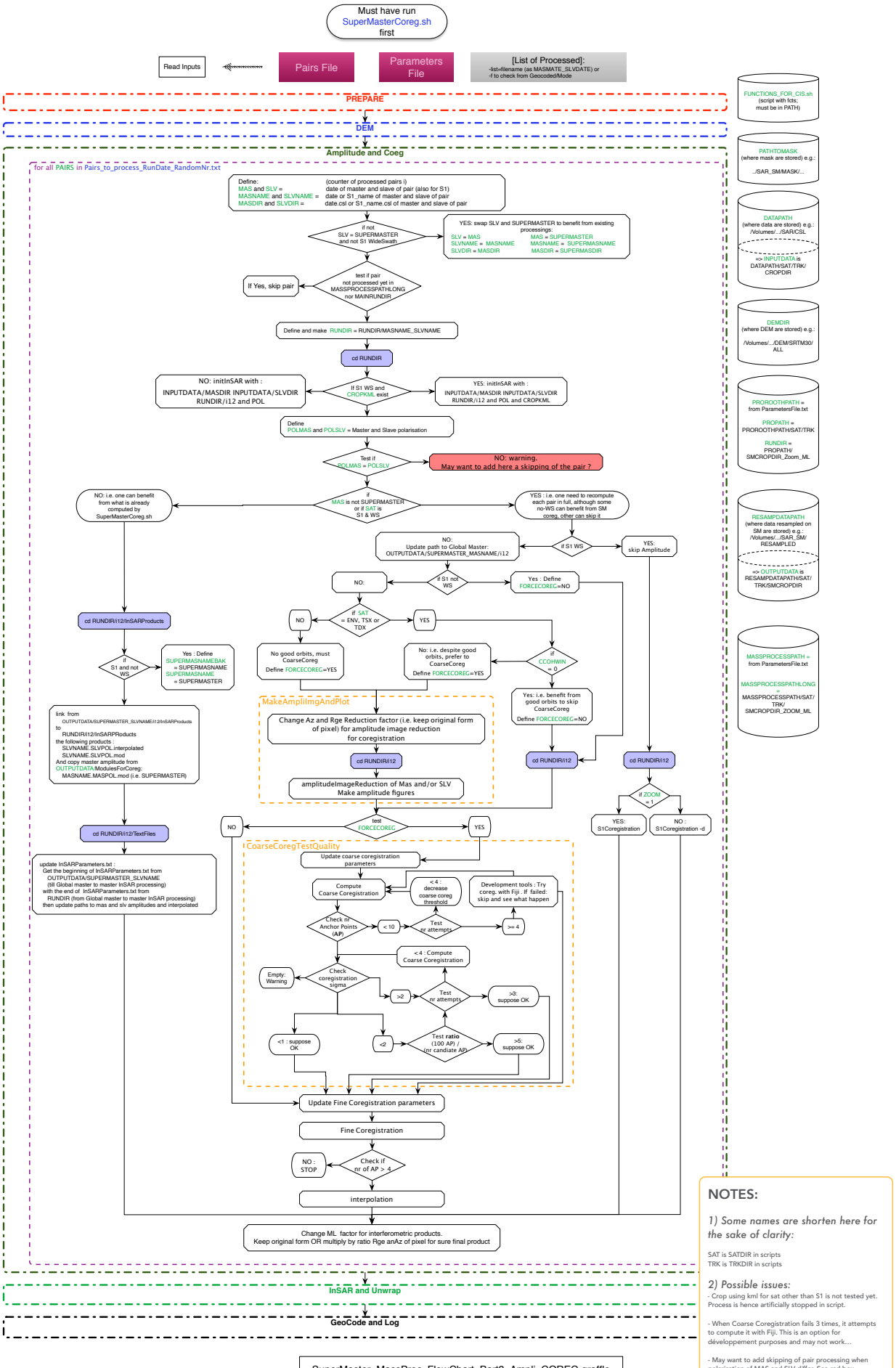


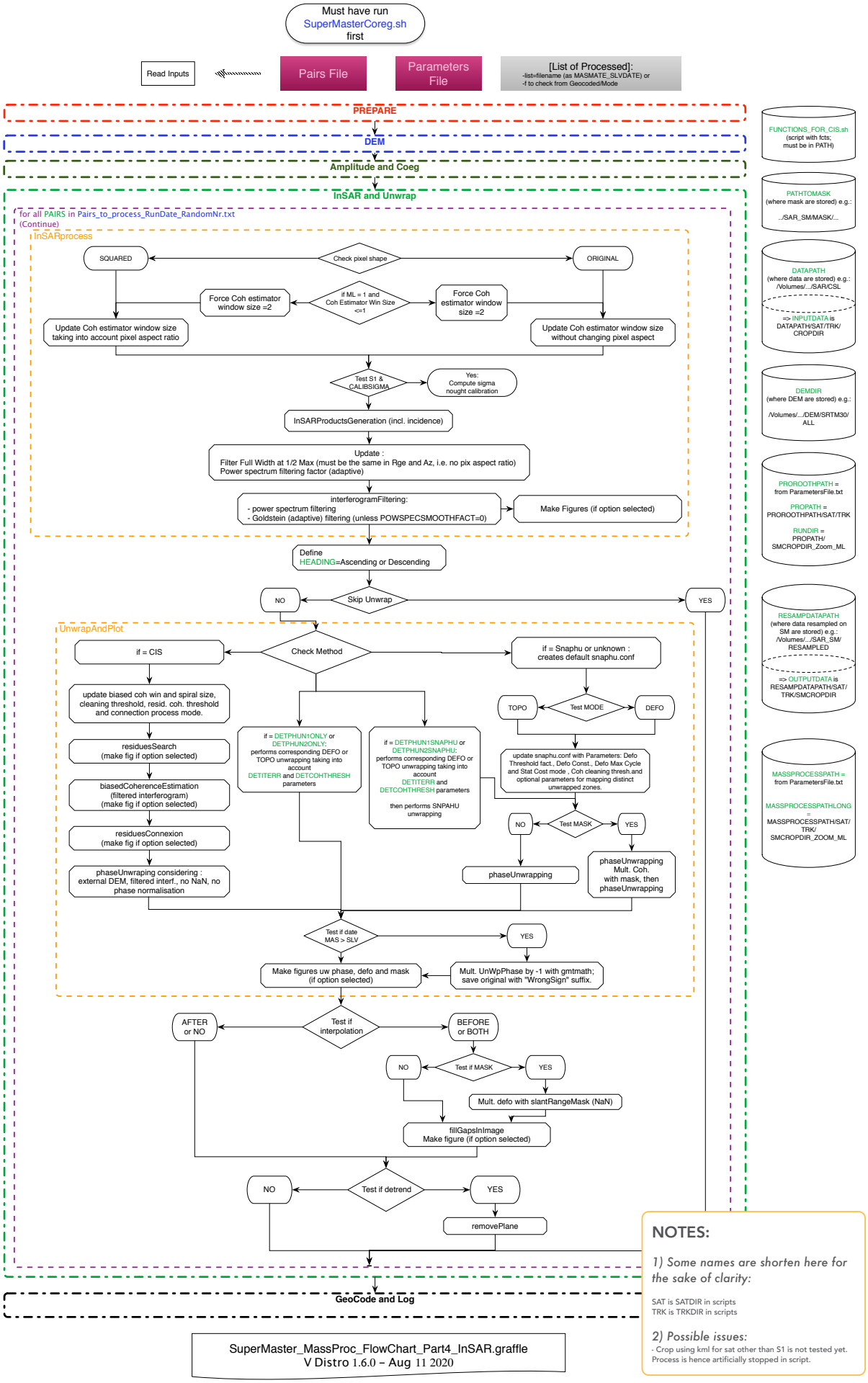

PREPARE and READ
Read Inputs
Parameters File
[FORCE]
to recompute DEM for each S1
DEM
AMPLITUDE and COREG
Define RUNDIR = MAINRUNDIR (i.e. PROPATH/COREG/SMCROPDIR_ZOOM_ML)
for all SLVDIR in New_Slaves_tp_process.txt
For all SLVDIR in New_slaves_to_process_RunDate_RandomNr.txt :
Define :
SLV = date only
SLVNAME = date or S1 name
SLVDIR = date.csl or S1_name.csl
RUNDIR=MAINRUNDIR/SUPERMASTER_SLVNAME
MASDIR = SUPERMASDIR
Create RUNDIR
cd RUNDIR
initInSAR – create
update path to SUPERMASTER, SLV, Pol and processing dir
MakeInitInSAR
test if Crop is CROPKML
NO
YES
initInSAR with :
INPUTDATA/MASDIR INPUTDATA/SLVDIR
RUNDIR/i12 and POL
initInSAR with :
INPUTDATA/MASDIR INPUTDATA/SLVDIR
RUNDIR/i12 CROPKML and POL
test if S1
NO
YES
Make dir (if not done yet) OUTPUTDATA/ModulesForCoreg
cd RUNDIR/i12
test if OUTPUTDATA/ModulesForCoreg/SUPERMASTER* exists
NO
YES
MakeAmpliImgAndPlot
Change Az and Rge Reduction factor (i.e. keep original form of pixel) for amplitude image reduction for coregistration
Make amplitude figures of SUPERMASTER and SLV
Check if SUPERMAS mod file is empty
Warning : crop is out of img or CSL corrupted ?
Make dir (if not done yet) OUTPUTDATA/_SUPERMASTER_Ampli_Img_Reduc/TextFiles and OUTPUTDATA/_SUPERMASTER_Ampli_Img_Reduc/InSARProducts and copy corresponding InSARParameters.txt and SUPERMASTER module. Update path to Reduced Master ampli img path in InSARParameters.txt
MakeAmpliImgAndPlot
Change Az and Rge Reduction factor (i.e. keep original form of pixel) for amplitude image reduction for coregistration
Make amplitude figures of SLV
Update path to Reduced SUPERMaster ampli img dimensions and path in InSARParameters.txt
copy module of SLV in OUTPUTDATA/ModulesForCoreg
Check if SLV mod file is empty
Warning : crop is out of img or CSL corrupted ?
CoarseCoregTestQuality
Update coarse coregistration parameters
Compute Coarse Coregistration
< 4 : decrease coarse coreg threshold
Development tools : Try coreg. with Fiji . If failed: skip and see what happen
Check nr Anchor Points (AP)
< 10
Test nr attempts
>= 4
< 4 : Compute Coarse Coregistration
Empty : Warning
Check coregistration sigma
>2
Test nr attempts
>3: suppose OK
<1 : suppose OK
<2
Test ratio (100 AP) / (nr candiate AP)
>5: suppose OK
Update Fine Coregistration parameters
Fine Coregistration
cd RUNDIR/i12
S1Coregistration
NO : STOP
Check if nr of Anchor Pt > 4
interpolation
Make figures of slv interpolated mod
cd MAINRUNDIR
RUNDIR=MAINRUNDIR
cd MAINRUNDIR
Dump scripts and all the list of images in log file
Copy everything in OUTDATA and check if everything was copied (do not use mv because of permission prblm while changing OS). If OK, clean RUNDIR
cd OUTPUTDATA
update paths in OUTDATA using script: _RenamePathAfterMove.sh SAT
Speak out end of processing
End
PATHTOMASK
(where mask are stored)
e.g.:
../SAR_SM/MASK/...
DATAPATH
(where data are stored)
e.g.:
/Volumes/.../SAR/CSL
=> INPUTDATA is DATAPATH/SAT/TRK/CROPDIR
DEMDIR
(where DEM are stored)
e.g.:
/Volumes/.../DEM/SRTM3i/ALL
RESAMPDATAPATH
(where data resampled on SM are stored) e.g.:
/Volumes/.../SAR_SM/RESAMPLED
=> OUTPUTDATA is RESAMPDATAPATH/SAT/TRK/SMCROPDIR
NOTES:
1) Some names are shorten here for the sake of clarity:
SAT is SATDIR in scripts
TRK is TRKDIR in scripts
2) Possible issue:
- Crop using kml for sat other than S1 is not tested yet. Process is hence artificially stopped in script.
SuperMasterCoreg_FlowChart_Part2_AMPL_COREG.graffle
V Distro 1.4.0 – Aug 11 2020

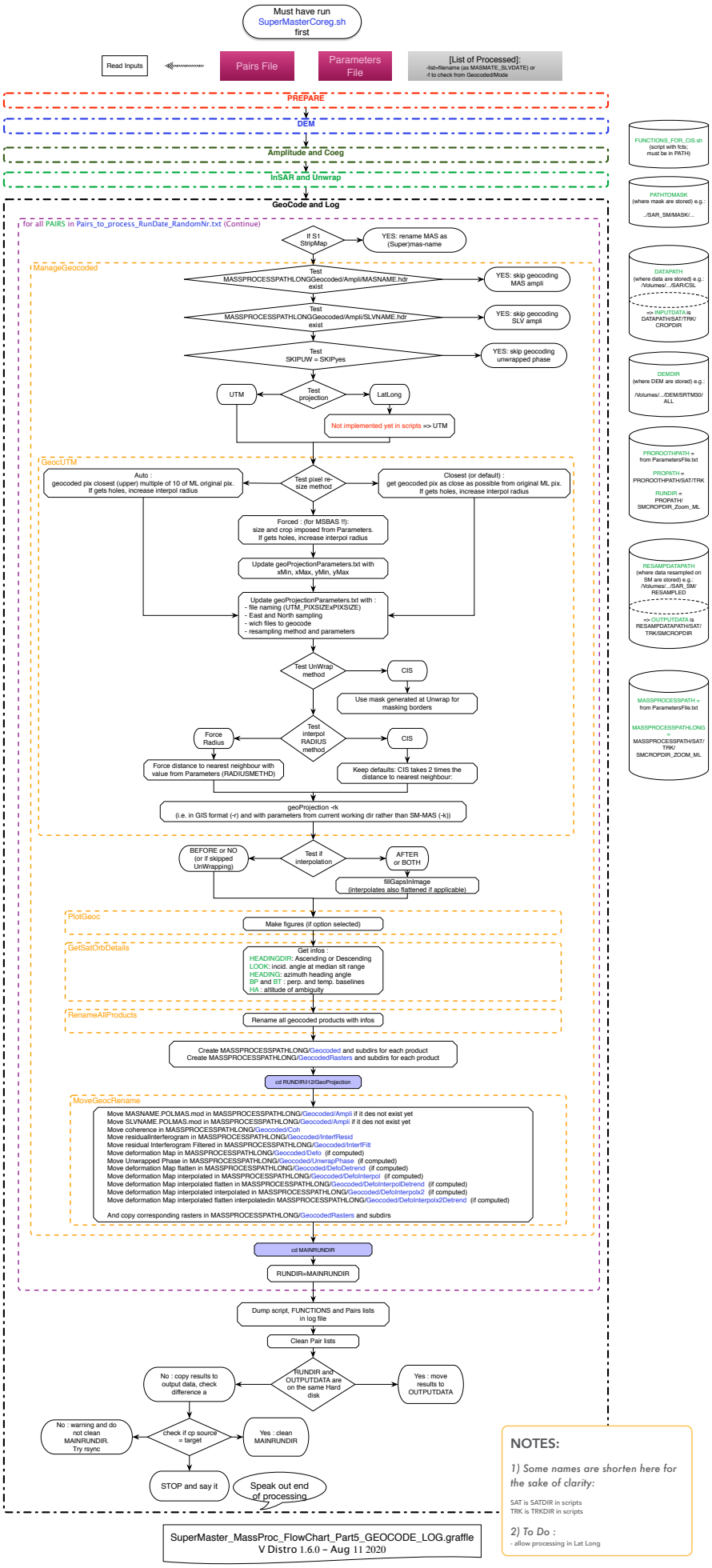
A.5) Flow chart of *SuperMaster_MassProc.sh*




Must have run SuperMasterCoreg.sh first
Read Inputs
Pairs File
Parameters File
[List of Processed]:
-list=filename (as MASMATE_SLVDATE) or
-f to check from Geocoded/Mode
FUNCTIONS_FOR_CIS.sh
(script with fcts;
must be in PATH)
PREPARE
DEM
cd RUNDIR
Define MAINRUNDIR = RUNDIR
Define FORMERMASK = MASSPROCESSPATHLONG/slantRange.txt
Compute pix aspect ratio from SUPERMASTER:
Rg / Az sampling taking into account incidence look angle.
Define RGSAMP, AZSAMP, INCIDANGL, RATIO, RATIOREAL
If S1 WideSwath
YES
No
Log and Speak out warning: it is your responsibility to check if mask is the same as already used
Select all different masters from Pairs_to_process_RunDate_RandomNr.txt
NO : Define MASKBASENAME = NoMask
Is Mask requested ?
YES:
YES: Define MASKBASENAME = basename PATHTOMASK and warning
Is Mask requested ?
NO : Define MASKBASENAME = NoMask
Test FORMERMASK = requested PATHTOMASK
YES : Define MASKBASENAME = basename PATHTOMASK
No : warning
Log and Speak out warning: Mask requested is not the same as former SuperMasterCoreg processing
for all MASTER from Pairs_to_process_RunDate_RandomNr.txt
For each different master, define IMGWITHDEM = MASONLYNAME
Compute PIXSIZEAZ and PIXSIZERG
Define IMGWITHDEM = SUPERMASNAME
Compute PIXSIZEAZ and PIXSIZERG
ManageDEM
Test parameter RECOMPDEM
KEEP
UNKNOWN
Test if externalSlantRangeDEM file exists
NO
FORCE
Test if mask is needed
NO
run
MasterDEM.sh IMGWITHDEM ParametersFile.txt
which store externalSlantRangeDEM (and mask if needed) in CROPDIR
See corresponding flow chart (MasterDEM_FlowChart.graffle).
Test if slantRangeMask_maskName file exists
NO
Clear possible existing unnecesary slantRangeMask
update links to slanRangeMask and its txt
Amplitude and Coeg
InSAR and Unwrap
GeoCode and Log
PATHTOMASK
(where mask are stored) e.g.:
../SAR_SM/MASK/...
DATAPATH
(where data are stored) e.g.:
/Volumes/.../SAR/CSL
=> INPUTDATA is DATAPATH/SAT/TRK/CROPDIR
DEMDIR
(where DEM are stored) e.g.:
/Volumes/.../DEM/SRTM30/ALL
PROROOTHPATH = from ParametersFile.txt
PROPATH = PROROOTHPATH/SAT/TRK
RUNDIR = PROPATH/SMCROPDIR_Zoom_ML
RESAMPDATAPATH
(where data resampled on SM are stored) e.g.:
/Volumes/.../SAR_SM/RESAMPLED
=> OUTPUTDATA is RESAMPDATAPATH/SAT/TRK/SMCROPDIR
MASSPROCESSPATH = from ParametersFile.txt
MASSPROCESSPATHLONG = MASSPROCESSPATH/SAT/TRK/SMCROPDIR_ZOOM_ML
NOTES:
1) Some names are shorten here for the sake of clarity:
SAT is SATDIR in scripts
TRK is TRKDIR in scripts
2) Possible issue:
- Crop using kml for sat other than S1 is not tested yet. Process is hence artificially stopped in script.
- Search for existing pairs using option -f is made in MASSPROCESSEDPATHLONG/Geocoded/DefoInterpolx2Detrend. If not available, you may need to check from another mode dir (eg. Defo.
SuperMaster_MassProc_FlowChart_Part1_DEM.graffle
V Distro 1.6.0 – Aug 11 2020


Must have run SuperMasterCoreg.sh first
Read Inputs
Pairs File
Parameters File
[List of Processed]:
-list=filename (as MASMATE_SLVDATE) or
-f to check from Geocoded/Mode
FUNCTIONS_FOR_GIS.sh (script with fcts: must be in PATH)
PATHTOMASK (where mask are stored) e.g.: ../SAR_SM/MASK/...
DATAPATH (where data are stored) e.g.: /Volumes/.../SARCSL
=> INPUTDATA is DATAPATH/SAT/TRK/CROPDIR
DEMDIR (where DEM are stored) e.g.: /Volumes/.../DEM/SRTM30/ALL
PROROOTHPATH = from ParametersFile.txt
PROPATH = PROROOTHPATH/SAT/TRK
RUNDIR = PROPATH/SMCROPDIR_Zoom_ML
RESAMPDATAPATH (where data resampled on SM are stored) e.g.: /Volumes/.../SAR_SM/RESAMPLED
=> OUTPUTDATA is RESAMPDATAPATH/SAT/TRK/SMCROPDIR
MASSPROCESSPATH = from ParametersFile.txt
MASSPROCESSPATHLONG = MASSPROCESSPATH/SAT/TRK/SMCROPDIR_ZOOM_ML
PREPARE
DEM
Amplitude and Coeg
for all PAIRS in Pairs_to_process_RunDate_RandomNr.txt
Define:
MAS and SLV =
MASNAME and SLVNAME =
MASDIR and SLVDIR =
(counter of processed pairs i)
date of master and slave of pair (also for S1)
date or S1_name of master and slave of pair
date.csl or S1_name.csl of master and slave of pair
if not SLV = SUPERMASTER and not S1 WideSwath
YES: swap SLV and SUPERMASTER to benefit from existing processings:
SLV = MAS
SLVNAME = MASNAME
SLVDIR = MASDIR
MAS = SUPERMASTER
MASNAME = SUPERMASNAME
MASDIR = SUPERMASDIR
test if pair not processed yet in MASSPROCESSPATHLONG nor MAINRUNDIR
If Yes, skip pair
Define and make RUNDIR = RUNDIR/MASNAME_SLVNAME
cd RUNDIR
If S1 WS and CROPKML exist
NO: initInSAR with : INPUTDATA/MASDIR INPUTDATA/SLVDIR RUNDIR/i12 and POL
YES: initInSAR with : INPUTDATA/MASDIR INPUTDATA/SLVDIR RUNDIR/i12 and POL and CROPKML
Define POLMAS and POLSLV = Master and Slave polarisation
Test if POLMAS = POLSLV
NO: warning. May want to add here a skipping of the pair ?
if MAS is not SUPERMASTER or if SAT is S1 & WS
NO: i.e. one can benefit from what is already computed by SuperMasterCoreg.sh
YES : i.e. one need to recompute each pair in full, although some no-WS can benefit from SM coreg, other can skip it
NO: Update path to Global Master: OUTPUTDATA/SUPERMASTER_MASNAME/i12
if S1 WS
YES: skip Amplitude
NO:
if S1 not WS
Yes : Define FORCECOREG=NO
cd RUNDIR/i12/InSARProducts
NO
if SAT = ENV, TSX or TDX
YES
if S1 and not WS
Yes : Define SUPERMASNAMEBAK = SUPERMASNAME SUPERMASNAME = SUPERMASTER
No good orbits, must CoarseCoreg Define FORCECOREG=YES
No: i.e. despite good orbits, prefer to CoarseCoreg Define FORCECOREG=YES
if CCOHWIN = 0
Yes: i.e. benefit from good orbits to skip CoarseCoreg Define FORCECOREG=NO
MakeAmpliImgAndPlot
Change Az and Rge Reduction factor (i.e. keep original form of pixel) for amplitude image reduction for coregistration
cd RUNDIR/i12
amplitudeImageReduction of Mas and/or SLV Make amplitude figures
cd RUNDIR/i12
cd RUNDIR/i12
link from OUTPUTDATA/SUPERMASTER_SLVNAME/i12/InSARProducts to RUNDIR/i12/InSARPRoducts the following products : SLVNAME.SLVPOL.interpolated SLVNAME.SLVPOL.mod And copy master amplitude from OUTPUTDATA/ModulesForCoreg: MASNAME.MASPOL.mod (i.e. SUPERMASTER)
cd RUNDIR/i12/TextFiles
update InSARParameters.txt : Get the beginning of InSARParameters.txt from OUTPUTDATA/SUPERMASTER_SLVNAME (till Global master to master InSAR processing) with the end of InSARParameters.txt from RUNDIR (from Global master to master InSAR processing) then update paths to mas and slv amplitudes and interpolated
test FORCECOREG
NO
YES
CoarseCoregTestQuality
Update coarse coregistration parameters
Compute Coarse Coregistration
< 4 : decrease coarse coreg threshold
Development tools : Try coreg, with Fiji. If failed: skip and see what happen
Check nr Anchor Points (AP)
< 10
Test nr attempts
>= 4
< 4 : Compute Coarse Coregistration
Empty: Warning
Check coregistration sigma
>2
Test nr attempts
>3: suppose OK
<1 : suppose OK
<2
Test ratio (100 AP) / (nr candiate AP)
>5: suppose OK
Update Fine Coregistration parameters
Fine Coregistration
NO : STOP
Check if nr of AP > 4
interpolation
if ZOOM = 1
YES: S1Coregistration
NO : S1Coregistration -d
Change ML factor for interferometric products. Keep original form OR multiply by ratio Rge anAz of pixel for sure final product
InSAR and Unwrap
GeoCode and Log
NOTES:
1) Some names are shorten here for the sake of clarity:
SAT is SATDIR in scripts
TRK is TRKDIR in scripts
2) Possible issues:
- Crop using kml for sat other than S1 is not tested yet. Process is hence artificially stopped in script.
- When Coarse Coregistration fails 3 times, it attempts to compute it with Fiji. This is an option for développement purposes and may not work...
- May want to add skipping of pair processing when polarisation of MAS and SLV differ. See red box.
SuperMaster_MassProc_FlowChart_Part3_Ampli_COREG.graffle
V Distro 1.6.1 – Sep 1 2020


Must have run SuperMasterCoreg.sh first
Read Inputs
Pairs File
Parameters File
[List of Processed]:
-list=filename (as MASMATE_SLVDATE) or
-f to check from Geocoded/Mode
PREPARE
DEM
Amplitude and Coeg
InSAR and Unwrap
for all PAIRS in Pairs_to_process_RunDate_RandomNr.txt
(Continue)
InSARprocess
SQUARED
Check pixel shape
ORIGINAL
Force Coh estimator window size =2
if ML = 1 and Coh Estimator Win Size <=1
Force Coh estimator window size =2
Update Coh estimator window size taking into account pixel aspect ratio
Update Coh estimator window size without changing pixel aspect
Test S1 & CALIBSIGMA
Yes: Compute sigma nought calibration
InSARProductsGeneration (incl. incidence)
Update :
Filter Full Width at 1/2 Max (must be the same in Rge and Az, i.e. no pix aspect ratio)
Power spectrum filtering factor (adaptive)
interferogramFiltering:
- power spectrum filtering
- Goldstein (adaptive) filtering (unless POWSPECSMOOTHFACT=0)
Make Figures (if option selected)
Define HEADING=Ascending or Descending
NO
Skip Unwrap
YES
UnwrapAndPlot
if = CIS
Check Method
if = Snaphu or unknown : creates default snaphu.conf
update biased coh win and spiral size, cleaning threshold, resid. coh. threshold and connection process mode.
residuesSearch (make fig if option selected)
biasedCoherenceEstimation (filtered interferogram) (make fig if option selected)
residuesConnexion (make fig if option selected)
phaseUnwraping considering : external DEM, filtered interf., no NaN, no phase normalisation
if = DETPHUN1ONLY or DETPHUN2ONLY: performs corresponding DEFO or TOPO unwrapping taking into account DETITERR and DETCOHTHRESH parameters
if = DETPHUN1SNAPHU or DETPHUN2SNAPHU: performs corresponding DEFO or TOPO unwrapping taking into account DETITERR and DETCOHTHRESH parameters
then performs SNPAHU unwrapping
TOPO
Test MODE
DEFO
update snaphu.conf with Parameters: Defo Threshold fact., Defo Const., Defo Max Cycle and Stat Cost mode , Coh cleaning thresh.and optional parameters for mapping distinct unwrapped zones.
NO
Test MASK
YES
phaseUnwrapping
phaseUnwrapping Mult. Coh. with mask, then phaseUnwrapping
Test if date MAS > SLV
YES
Make figures uw phase, defo and mask (if option selected)
Mult. UnWpPhase by -1 with gmtmath; save original with "WrongSign" suffix.
AFTER or NO
Test if interpolation
BEFORE or BOTH
NO
Test if MASK
YES
Mult. defo with slantRangeMask (NaN)
fillGapsInImage
Make figure (if option selected)
NO
Test if detrend
YES
removePlane
GeoCode and Log
FUNCTIONS_FOR_CIS.sh (script with fcts; must be in PATH)
PATHTOMASK (where mask are stored) e.g.: ../SAR_SM/MASK/...
DATAPATH (where data are stored) e.g.: /Volumes/.../ISAR/CSL
=> INPUTDATA is DATAPATH/SAT/TRK/CROPDIR
DEMDIR (where DEM are stored) e.g.: /Volumes/.../DEM/SRTM30/ALL
PROROOTHPATH = from ParametersFile.txt
PROPATH = PROROOTHPATH/SAT/TRK
RUNDIR = PROPATH/SMCROPDIR_Zoom_ML
RESAMPDATAPATH (where data resampled on SM are stored) e.g.: /Volumes/.../SAR_SM/RESAMPLED
=> OUTPUTDATA is RESAMPDATAPATH/SAT/TRK/SMCROPDIR
MASSPROCESSPATH = from ParametersFile.txt
MASSPROCESSPATHLONG = MASSPROCESSPATH/SAT/TRK/SMCROPDIR_ZOOM_ML
NOTES:
1) Some names are shorten here for the sake of clarity:
SAT is SATDIR in scripts
TRK is TRKDIR in scripts
2) Possible issues:
- Crop using kml for sat other than S1 is not tested yet. Process is hence artificially stopped in script.
SuperMaster_MassProc_FlowChart_Part4_InSAR.graffle
V Distro 1.6.0 – Aug 11 2020


Must have run SuperMasterCoreg.sh first
Read Inputs
Pairs File
Parameters File
[List of Processed]:
-list=filename (as MASMATE_SLVDATE) or
-f to check from Geocoded/Mode
PREPARE
DEM
Amplitude and Coeg
InSAR and Unwrap
GeoCode and Log
for all PAIRS in Pairs_to_process_RunDate_RandomNr.txt (Continue)
If S1 StripMap
YES: rename MAS as (Super)mas-name
ManageGeocoded
Test MASSPROCESSPATHLONGGeocoded/Ampli/MASNAME.hdr exist
YES: skip geocoding MAS ampli
Test MASSPROCESSPATHLONGGeocoded/Ampli/SLVNAME.hdr exist
YES: skip geocoding SLV ampli
Test SKIPUW = SKIPyes
YES: skip geocoding unwrapped phase
UTM
Test projection
LatLong
Not implemented yet in scripts => UTM
GeocUTM
Auto : geocoded pix closest (upper) multiple of 10 of ML original pix. If gets holes, increase interpol radius
Test pixel re-size method
Closest (or default) : get geocoded pix as close as possible from original ML pix. If gets holes, increase interpol radius
Forced : (for MSBAS !!): size and crop imposed from Parameters. If gets holes, increase interpol radius
Update geoProjectionParameters.txt with xMin, xMax, yMin, yMax
Update geoProjectionParameters.txt with : - file naming (UTM_PIXSIZExPIXSIZE) - East and North sampling - wich files to geocode - resampling method and parameters
Test UnWrap method
CIS
Use mask generated at Unwrap for masking borders
Test interpol RADIUS method
Force Radius
CIS
Force distance to nearest neighbour with value from Parameters (RADIUSMETHD)
Keep defaults: CIS takes 2 times the distance to nearest neighbour:
geoProjection -rk (i.e. in GIS format (-r) and with parameters from current working dir rather than SM-MAS (-k))
BEFORE or NO (or if skipped UnWrapping)
Test if interpolation
AFTER or BOTH
fillGapsInImage (interpolates also flattened if applicable)
PlotGeoc
Make figures (if option selected)
GetSatOrbDetails
Get infos : HEADINGDIR: Ascending or Descending LOOK: incid. angle at median slt range HEADING: azimuth heading angle BP and BT : perp. and temp. baselines HA : altitude of ambiguity
RenameAllProducts
Rename all geocoded products with infos
Create MASSPROCESSPATHLONG/Geocoded and subdirs for each product Create MASSPROCESSPATHLONG/GeocodedRasters and subdirs for each product
cd RUNDIR/i12/GeoProjection
MoveGeocRename
Move MASNAME.POLMAS.mod in MASSPROCESSPATHLONG/Geocoded/Ampli if it des not exist yet
Move SLVNAME.POLMAS.mod in MASSPROCESSPATHLONG/Geocoded/Ampli if it des not exist yet
Move coherence in MASSPROCESSPATHLONG/Geocoded/Coh
Move residualInterferogram in MASSPROCESSPATHLONG/Geocoded/InterfResid
Move residual Interferogram Filtered in MASSPROCESSPATHLONG/Geocoded/InterfFilt
Move deformation Map in MASSPROCESSPATHLONG/Geocoded/Defo (if computed)
Move Unwrapped Phase in MASSPROCESSPATHLONG/Geocoded/UnwrapPhase (if computed)
Move deformation Map flatten in MASSPROCESSPATHLONG/Geocoded/DefoDetrend (if computed)
Move deformation Map interpolated in MASSPROCESSPATHLONG/Geocoded/DefoInterpol (if computed)
Move deformation Map interpolated flatten in MASSPROCESSPATHLONG/Geocoded/DefoInterpolDetrend (if computed)
Move deformation Map interpolated interpolated in MASSPROCESSPATHLONG/Geocoded/DefoInterpolx2 (if computed)
Move deformation Map interpolated flatten interpolatedin MASSPROCESSPATHLONG/Geocoded/DefoInterpolx2Detrend (if computed)
And copy corresponding rasters in MASSPROCESSPATHLONG/GeocodedRasters and subdirs
cd MAINRUNDIR
RUNDIR=MAINRUNDIR
Dump script, FUNCTIONS and Pairs lists in log file
Clean Pair lists
RUNDIR and OUTPUTDATA are on the same Hard disk
No : copy results to output data, check difference a
Yes : move results to OUTPUTDATA
No : warning and do not clean MAINRUNDIR. Try rsync
check if cp source = target
Yes : clean MAINRUNDIR
STOP and say it
Speak out end of processing
FUNCTIONS_FOR_CIS.sh (script with fcts; must be in PATH)
PATHTOMASK (where mask are stored) e.g.: ../SAR_SM/MASK/...
DATAPATH (where data are stored) e.g.: /Volumes/.../SAR/CSL
=> INPUTDATA is DATAPATH/SAT/TRK/CROPDIR
DEMDIR (where DEM are stored) e.g.: /Volumes/.../DEM/SRTM30/ALL
PROROOTHPATH = from ParametersFile.txt
PROPATH = PROROOTHPATH/SAT/TRK
RUNDIR = PROPATH/SMCROPDIR_Zoom_ML
RESAMPDATAPATH (where data resampled on SM are stored) e.g.: /Volumes/.../SAR_SM/RESAMPLED
=> OUTPUTDATA is RESAMPDATAPATH/SAT/TRK/SMCROPDIR
MASSPROCESSPATH = from ParametersFile.txt
MASSPROCESSPATHLONG = MASSPROCESSPATH/SAT/TRK/SMCROPDIR_ZOOM_ML
NOTES:
1) Some names are shorten here for the sake of clarity:
SAT is SATDIR in scripts
TRK is TRKDIR in scripts
2) To Do :
- allow processing in Lat Long
SuperMaster_MassProc_FlowChart_Part5_GEOCODE_LOG.graffle
V Distro 1.6.0 – Aug 11 2020

A.6) Example of full automation

A.6.1) Automatic data download

Special thanks to Gilles Celli who wrote the following two scripts to automatically download data from several regions (Gilles@ecgs.lu):

***sentinel1_download_all.sh*:**

```
#!/bin/sh
# Log:
#
# 2015.11.0: Added sleep command, or else esa.int will block downloads
# See: https://scihub.esa.int/news/News00040
#
# This script downloads the satellite image files from ESA-Sentinel 1
# requires /opt/local/bin/xmlstarlet and curl (install with macports)
# and the main script: /Users/doris/scripts/sentinel1_downloader_ingestiondate.sh
#
# Always mount the SMB Disc via Applescript osascript, this avoids writing an empty 'DiscData' Folder to
/Volumes/
#echo $mount_value

## RD of Congo
/Users/doris/scripts/sentinel1_downloader_ingestiondate.sh --congo --slc --startdate=30-DAYS-AGO --
enddate=TODAY --skipmd5check --deletezip30days
/bin/sleep 180

[…]

## Tristan
/Users/doris/scripts/sentinel1_downloader_ingestiondate.sh --tristan --slc --startdate=10-DAYS-AGO --
enddate=TODAY --skipmd5check --deletezip30days

## Domyuo
/Users/doris/scripts/sentinel1_downloader_ingestiondate.sh --domuyo18 --slc --startdate=10-DAYS-AGO --
enddate=TODAY --skipmd5check --deletezip30days
/Users/doris/scripts/sentinel1_downloader_ingestiondate.sh --domuyo83 --slc --startdate=10-DAYS-AGO --
enddate=TODAY --skipmd5check --deletezip30days
```

sentinel1_downloader_ingestiondate.sh:

```
#!/bin/bash
#
# Set DEBUG mode with command: set -xv
#set -xv
#
# Define default options and variables
VERSION="2.9.5"

# ------------------------------------------------------------
# Script to download ESA's Sentinel1 data from ESA-Site Scihub
# See latest news here: https://scihub.copernicus.eu/news/

# Based on script 'odata-demo.sh' from Scihub Site:
# See odata-demo.sh from scihub.copernicus.eu Site:
# List first 10 products since last <n> days, by product type and intersecting an AOI
# https://scihub.copernicus.eu/twiki/pub/SciHubUserGuide/5APIsAndBatchScripting/odata-demo.sh
# https://scihub.copernicus.eu/twiki/do/view/SciHubUserGuide/5APIsAndBatchScripting
#
# See ESA's 'APIs and Batch scripting' webpage:
# https://scihub.copernicus.eu/userguide/5APIsAndBatchScripting
# ------------------------------------------------------------
# ------------------------------------------------------------
# Changelog:
# ------------------------------------------------------------
#

[...]

# - Using Macports coreutils 'tee' command instead of macOS System /usr/bin/tee
# macports coreutils installs in /opt/local/libexec/gnubin/tee

[...]

# 2017.07.26 - Version 2.5.1
# - SMB Volumes are no more mounted with mount command:
#   Using macOS "automount" feature to automatically mount SMB Volumes
#   in /Users/doris/NAS-Discs instead of mounting them to /Volumes/SMB_SHARENAME
#   See also: https://useyourloaf.com/blog/using-the-mac-os-x-automounter/
#
# ------------------------------------------------------------

# Default values
# INGESTION Start and End Date
INGEST_START_DATE="TODAY"
```

```
INGEST_END_DATE="TODAY"

FULLHOSTNAME=$(/bin/hostname)
HOSTNAME=$(/bin/hostname -s)
LOG_DATE=$(/bin/date +"%Y%m%d%H%M")
SCRIPT_EXEC_STARTDATE=$(/bin/date +"%A, %d %B %Y @ %H:%M")

# Set the email recipients, separate with comma for multiple recipients
#EMAIL_RECIPIENTS="gilles@ecgs.lu"
EMAIL_RECIPIENTS="infodownload@ecgs.lu,ndo@ecgs.lu"
EMAIL_SUBJECT="[${HOSTNAME}] Sentinel1 Downloads for "

# Maximum days / rows = 100
## "The maximum number of rows to be returned in a single query is set to 100."
## "Requests for more than the maximum supported number of rows will result in an error ( http 500 )."
##  See here: https://scihub.copernicus.eu/userguide/5APIsAndBatchScripting
maxRows=100

# new since 2.5.1b1 - use automount feature of Mac OS X to mount smb network discs
SMB_SHARE_NAME="hp-D3600-Data_Share1" # do not use a slash at beginning and end
SMB_SHARE_MOUNTED_ON_LOCAL_DIR="${HOME}/NAS-Discs/$SMB_SHARE_NAME"
S1_WORKDIR="${SMB_SHARE_MOUNTED_ON_LOCAL_DIR}/SAR_DATA/S1"
echo "Working Dir: S1_WORKDIR= ${S1_WORKDIR}"

################################################################################################

[...]

# Create the local log directory (if not present) to save the log-file locally if Network Disc can't be mounted
if [ ! -e "${HOME}/scripts/logs/" ]
   then
   /bin/mkdir -p $HOME/scripts/logs/
   /bin/mkdir -p $HOME/scripts/logs/sentinel1_disc-mount_error_logs/
fi

# Check if directory exists
if [ -e "${S1_WORKDIR}" ]; then
   echo "${SMB_SHARE_MOUNTED_ON_LOCAL_DIR} mounted, continuing ..."
else
   ERROR_LOG="$HOME/scripts/logs/sentinel1_disc-mount_error_logs/disc-mount_error-$LOG_DATE.log"
   EMAIL_SUBJECT="[${HOSTNAME}] ALERT! Network Disc Mount error ! No Sentinel1 Downloads"

   echo "ALERT! This is script: $0 running on $FULLHOSTNAME" > $ERROR_LOG
   echo "Error: ${SMB_SHARE_MOUNTED_ON_LOCAL_DIR} not mounted! Check the network disc with share-name: $SMB_SHARE_NAME !" >>
$ERROR_LOG
   echo " There was an error mounting this disc." >> $ERROR_LOG

    # send an email using /usr/bin/mail
   /usr/bin/mail -s "${EMAIL_SUBJECT}" "${EMAIL_RECIPIENTS}" < $ERROR_LOG
    exit 1;
fi

# ESA-Site login details
DHUS_SERVER_URL="https://scihub.copernicus.eu/dhus"
DHUS_USER=""
DHUS_PASSWD=""

# Setting default variables
ptype="SLC"
COUNTRY="CONGO"
platformname="platformname:Sentinel-1"

# Default Sensor Mode = IW for Congo, Lux, Tanz - for Tristan: SM
#sensormode="(sensoroperationalmode:IW)"

VERBOSE=false
QUERY_RESULT_LIST=""
SKIPMD5CHECK=false
MAINTENANCE_MODE=false
DELETE_ZIPFILE_30DAYS=false
FORCE_DOWNLOAD=false

# Set variables depending to optional command line arguments
ROOT_URL_ODATA="${DHUS_SERVER_URL}/odata/v1"
ROOT_URL_SEARCH="${DHUS_SERVER_URL}/search"

# Third-party application, provide the correct path for curl, md5 and xmlstarlet
#CURL_PREFIX="/opt/local/bin/curl --limit-rate 3500K --max-time 7200 --silent --show-error -gu ${DHUS_USER}:${DHUS_PASSWD}"
CURL_PREFIX="/opt/local/bin/curl --limit-rate 6000K --max-time 14400 --silent --show-error -gu ${DHUS_USER}:${DHUS_PASSWD}"

#CURL_PREFIX="/opt/local/bin/curl --verbose --show-error -gu ${DHUS_USER}:${DHUS_PASSWD}"

XMLSTARLET_PREFIX="/opt/local/bin/xmlstarlet"
MD5_PREFIX="/sbin/md5 -q"
DU_PREFIX="/opt/local/libexec/gnubin/du -sb"
TEE_PREFIX="/opt/local/libexec/gnubin/tee"

# For unzipping, we're using 7zip with options:
# -aos: Skip extracting of existing files
# x: for extracting files with its directory content
UNZIP_PREFIX="/opt/local/bin/7z -aos x"

# Check if command 'curl' or 'xmlstarlet' is installed
if ! $(type ${CURL_PREFIX} &> /dev/null)
    then echo "Command \"${CURL_PREFIX}\" is missing, please install it or change the path in this script first.";
    exit 1
fi

if ! $(type $XMLSTARLET_PREFIX &> /dev/null)
    then echo "Command \"$XMLSTARLET_PREFIX\" is missing, please install it or change the path in this script first!";
    exit 1
fi

if ! $(type $MD5_PREFIX &> /dev/null)
    then echo "Command \"$MD5_PREFIX\" is missing, please install it or change the path in this script first!";
    exit 1
```

```
fi

if ! $(type $UNZIP_PREFIX &> /dev/null)
     then echo "Command \"$UNZIP_PREFIX\" is missing, please install it or change the path in this script first!";
    exit 1
fi

if ! $(type $DU_PREFIX &> /dev/null)
     then echo "Command \"$DU_PREFIX\" is missing, please install it or change the path in this script first!";
    exit 1
fi

if ! $(type $TEE_PREFIX &> /dev/null)
     then echo "Command \"$TEE_PREFIX\" is missing, please install it or change the path in this script first!";
    exit 1
fi

# Display help
function show_help
{
    echo "USAGE: sentinel1_downloader_ingestiondate.sh [OPTION_1] [OPTION_2] [OPTION_3] ... "
    echo "This script downloads Sentinel1 data of a given country using the ESA's OData interface of the Data Hub Service (DHuS)."
    echo " -h, --help       display this help message"
    echo ""
    echo "Choose Country, only one can be selected:"
    echo ""
    echo "      -1, --belgium    download Sentinel1 data of Belgium"
    echo "      -2, --capvert    download Sentinel1 data of Capvert"
    echo "      -3, --congo      download Sentinel1 data of Congo"
    echo "      -4, --cameroon   download Sentinel1 data of Cameroon"
    echo "      -5, --luxembourg download Sentinel1 data of Luxembourg"
    echo "      -6, --tanzania   download Sentinel1 data of Tanzania"
    echo "      -7, --erta_ale   download Sentinel1 data of Ethiopia (Erta Ale)"
    echo "      -8, --hawaii     download Sentinel1 data of Hawaii"
    echo "      -9, --tristan    download Sentinel1 data of Tristan de Cuna"
    echo "      -10, --domuyo18    download Sentinel1 data of Domuyo - ASCENDING 18"
    echo "      -11, --domuyo83    download Sentinel1 data of Domuyo - DESCENDING 83"
    echo ""
    echo "Set product type:"
    echo "      -g, --grd        set product type to GRD"
    echo "      -o, --ocn        set product type to OCN"
    echo "      -r, --raw        set ptype to RAW"
    echo "      -s, --slc        set ptype to SLC"
    echo ""
    echo "Set ingestion start date:"
    echo "      --startdate=YYYY-MM-DD"
    echo "      or"
    echo "      --startdate=TODAY"
    echo "      --startdate=YESTERDAY"
    echo "      --startdate=xx-DAYS-AGO"
    echo "      --startdate=xx-WEEK-AGO"
    echo "      --startdate=xx-MONTH-AGO"
    echo ""
    echo "Set ingestion end date:"
    echo "      --enddate=YYYY-MM-DD"
    echo "      or"
    echo "      --enddate=TODAY"
    echo ""
    echo "Skip md5 checksum:"
    echo "      --skipmd5check"
    echo ""
    echo "Delete Zip-File older than 30 days:"
    echo "      --deletezip30days"
    echo ""
    echo "Force download Zip-File.Caution: use only if you really want to (re)download the Zip-File(s)."
    echo "      This will not delete Zip-File older than 30 days"
    echo "      --force"
    echo ""
    echo " -v, --verbose    display curl command lines and results"
    echo " -V, --version    display the current version of the script"
    echo ""
}

# Parse command line arguments
for arg in "$@"
do
   case "$arg" in
    -h  | --help)       show_help;
                        exit 0
                        ;;

    -1  | --belgium)    COUNTRY="BELGIUM"
                        polygon="POLYGON((2.2759179687511 49.39733675067,6.5825585937511 49.39733675067,6.5825585937511
51.523048670724,2.2759179687511 51.523048670724,2.2759179687511 49.39733675067))"
                        sensormode="(sensoroperationalmode:IW)"
                        ;;

    -2  | --capvert)    COUNTRY="CAPVERT"
                        polygon="POLYGON((-24.805380859375 14.770554001959,-24.212119140625 14.770554001959,-24.212119140625
15.083713334744,-24.805380859375 15.083713334744,-24.805380859375 14.770554001959))"
                        sensormode="(sensoroperationalmode:IW)"
                        ;;

    -3  | --congo)      COUNTRY="DRCONGO"
                        polygon="POLYGON((28.63761230469 -3.7042857753737,29.725258789065 -3.7042857753737,29.725258789065 -
0.54279843346273,28.63761230469 -0.54279843346273,28.63761230469 -3.7042857753737))"
                        sensormode="(sensoroperationalmode:IW)"
                ;;

    -4  | --cameroon)   COUNTRY="CAMEROON"
                        polygon="POLYGON((8.9940576171883 3.9091132063575,9.460976562501 3.9091132063575,9.460976562501
4.4788700707381,8.9940576171883 4.4788700707381,8.9940576171883 3.9091132063575))"
                        sensormode="(sensoroperationalmode:IW)"
                        ;;

    -5  | --luxembourg) COUNTRY="LUXEMBOURG"
```

```
                        polygon="POLYGON((5.4674462890619 48.418939253948,8.2579736328119 48.418939253948,8.2579736328119
50.307533561486,5.4674462890619 50.307533561486,5.4674462890619 48.418939253948))"
                        sensormode="(sensoroperationalmode:IW)"
                        ;;

    -6  | --tanzania)   COUNTRY="TANZANIA"
                        polygon="POLYGON((35.674355468752 -3.1120759839556,36.959755859378 -3.1120759839556,36.959755859378 -
1.3337017191793,35.674355468752 -1.3337017191793,35.674355468752 -3.1120759839556))"
                         sensormode="(sensoroperationalmode:IW)"
                         ;;

    -7  | --erta_ale)   COUNTRY="ERTA_ALE"
                        polygon="POLYGON((40.47309769928795 13.38637299084182,40.937128609315685 13.38637299084182,40.937128609315685
13.885019945672866,40.47309769928795 13.885019945672866,40.47309769928795 13.38637299084182))"
                        sensormode="(sensoroperationalmode:IW)"
                        ;;

    -8  | --hawaii)     COUNTRY="HAWAII"
                        polygon="POLYGON((-155.43700607299937 19.195655095731155,-154.99082250566502 19.195655095731155,-
154.99082250566502 19.44828597828061,-155.43700607299937 19.44828597828061,-155.43700607299937 19.195655095731155))"
                         sensormode="(sensoroperationalmode:IW)"
                         ;;

   -9   | --tristan)   COUNTRY="TRISTAN"
                   #Request POLYGON((-12.437366939445228 -37.19301009606267,-12.142885785004552 -37.19301009606267,-12.142885785004552
-37.032896613954165,-12.437366939445228 -37.032896613954165,-12.437366939445228 -37.19301009606267)))" ) AND ( beginPosition:[2013-
12-01T00:00:00.000Z TO 2018-05-01T23:59:59.999Z] AND endPosition:[2013-12-01T00:00:00.000Z TO 2018-05-01T23:59:59.999Z] ) AND (
(platformname:Sentinel-1 AND producttype:SLC AND sensoroperationalmode:SM))
                   polygon="POLYGON((-12.437366939445228 -37.19301009606267,-12.142885785004552 -37.19301009606267,-12.142885785004552
-37.032896613954165,-12.437366939445228 -37.032896613954165,-12.437366939445228 -37.19301009606267))"
                   sensormode="(sensoroperationalmode:SM)"
                   ;;

   -10   | --domuyo18)   COUNTRY="DOMUYO"
                   #Request POLYGON((-12.437366939445228 -37.19301009606267,-12.142885785004552 -37.19301009606267,-12.142885785004552
-37.032896613954165,-12.437366939445228 -37.032896613954165,-12.437366939445228 -37.19301009606267)))" ) AND ( beginPosition:[2013-
12-01T00:00:00.000Z TO 2018-05-01T23:59:59.999Z] AND endPosition:[2013-12-01T00:00:00.000Z TO 2018-05-01T23:59:59.999Z] ) AND (
(platformname:Sentinel-1 AND producttype:SLC AND sensoroperationalmode:SM))
                   polygon="POLYGON((-70.6898267839218 -36.90762895618818,-70.03362627002683 -36.89749957901811,-70.09693399654567 -
36.05291395012518,-70.71831691721114 -36.06796170586554,-70.6898267839218 -36.90762895618818))"
                   sensormode="(sensoroperationalmode:IW AND relativeorbitnumber:18)"
                   ;;

   -11   | --domuyo83)   COUNTRY="DOMUYO"
                   #Request POLYGON((-12.437366939445228 -37.19301009606267,-12.142885785004552 -37.19301009606267,-12.142885785004552
-37.032896613954165,-12.437366939445228 -37.032896613954165,-12.437366939445228 -37.19301009606267)))" ) AND ( beginPosition:[2013-
12-01T00:00:00.000Z TO 2018-05-01T23:59:59.999Z] AND endPosition:[2013-12-01T00:00:00.000Z TO 2018-05-01T23:59:59.999Z] ) AND (
(platformname:Sentinel-1 AND producttype:SLC AND sensoroperationalmode:SM))
                   polygon="POLYGON((-70.6898267839218 -36.90762895618818,-70.03362627002683 -36.89749957901811,-70.09693399654567 -
36.05291395012518,-70.71831691721114 -36.06796170586554,-70.6898267839218 -36.90762895618818))"
                   sensormode="(sensoroperationalmode:IW AND relativeorbitnumber:83)"
                   ;;

    -g  |  --grd)       ptype="GRD";;
    -o  |  --ocn)       ptype="OCN";;
    -r  |  --raw)       ptype="RAW";;
    -s  |  --slc)       ptype="SLC";;

    # taken from: http://mywiki.wooledge.org/BashFAQ/035
    -sd |   --startdate)
           # Takes an option argument, ensuring it has been specified.
           if [ -n "$3" ]; then
               INGEST_START_DATE=$3
               echo "1. Start Date: ${INGEST_START_DATE}"
               shift
           else
               printf 'ERROR: "--startdate" requires a non-empty option argument,e.g 2017-12-31 (YYYY-MM-DD).\n' >&2
                    exit 1
           fi
           ;;
    --startdate=?*)
           INGEST_START_DATE=${3#*=} # Delete everything up to "=" and assign the remainder.
           echo "2. Start Date: ${INGEST_START_DATE}"
           ;;

    --startdate=)       # Handle the case of an empty --startdate=
           printf 'ERROR: "--startdate" requires a non-empty option argument, e.g 2017-12-31 (YYYY-MM-DD) \n' >&2
           exit 1
           ;;

      -ed |   --enddate)  # Takes an option argument, ensuring it has been specified.
             if [ -n "$4" ]; then
                   INGEST_END_DATE=$4
                echo "1. End Date: ${INGEST_END_DATE}"
             shift
             else
                 printf 'ERROR: "--enddate" requires a non-empty option argument,e.g 2017-12-31 (YYYY-MM-DD).\n' >&2
             exit 1
             fi
             ;;
    --enddate=?*)
             INGEST_END_DATE=${4#*=} # Delete everything up to "=" and assign the remainder.
             echo "2. End Date: ${INGEST_END_DATE}"
             ;;

    --enddate=) # Handle the case of an empty --enddate=
               printf 'ERROR: "--enddate" requires a non-empty option argument, e.g 2017-12-31 (YYYY-MM-DD) \n' >&2
               exit 1
               ;;

    --skipmd5check)  SKIPMD5CHECK=true
             ;;

                  # Delete the ZIPFILE older than 30days
    --deletezip30days)   DELETE_ZIPFILE_30DAYS=true
             ;;
```

```
          # Force download priority
    --force)   FORCE_DOWNLOAD=true
          ;;

    -v   | --verbose)   VERBOSE=true
          ;;

    -V   | --version)   show_version;
                exit 0
                ;;

    *)
        echo "Invalid option: $arg" >&2;
        show_help;
        exit 1
        ;;
   esac
done


# Check 3 arguments are given #
if [ $# -lt 4 ]
then
    printf "Usage: $0 countryname [--belgium, --capvert, --congo, --cameroon, --luxembourg, --tanzania, --erta_ale, --hawaii] "
    printf "ptype [--slc, --raw, --grd, --ocn] --startdate=YYYY-MM-DD --enddate=YYYY-MM-DD"
    echo ""
    exit
fi

if [[ ${INGEST_START_DATE} == *"-DAY"* ]]; then
    # We cut the string at first '-' e.g: 12-DAYS-AGO > result: 12
    DAYS_AGO=$(echo ${INGEST_START_DATE} | /usr/bin/cut -f 1 -d '-')
    INGEST_START_DATE=$(/bin/date -v -${DAYS_AGO}d '+%Y-%m-%d')
    echo "$DAYS_AGO days ago choosen > calculated ingestion start date: ${INGEST_START_DATE}"

elif [[ ${INGEST_START_DATE} == *"-WEEK"* ]]; then
    # We cut the string at first '-' e.g: 12-WEEK-AGO > result: 12
    WEEKS_AGO=$(echo ${INGEST_START_DATE} | /usr/bin/cut -f 1 -d '-')
    INGEST_START_DATE=$(/bin/date -v -${WEEKS_AGO}w '+%Y-%m-%d')
    echo "$WEEKS_AGO weeks ago choosen > calculated ingestion start date: ${INGEST_START_DATE}"

elif [[ ${INGEST_START_DATE} == *"-MONTH"* ]]; then
    # We cut the string at first '-' e.g: 12-MONTHS-AGO > result: 12
    MONTHS_AGO=$(echo ${INGEST_START_DATE} | /usr/bin/cut -f 1 -d '-')
    INGEST_START_DATE=$(/bin/date -v -${MONTHS_AGO}m '+%Y-%m-%d')
    echo "$MONTHS_AGO months ago choosen > calculated ingestion start date: ${INGEST_START_DATE}"

elif [ "${INGEST_START_DATE}" == "YESTERDAY" ]; then
    INGEST_START_DATE=$(/bin/date -v -1d '+%Y-%m-%d')
    echo "YESTERDAY ago choosen > calculated ingestion start date: ${INGEST_START_DATE}"

elif [ "${INGEST_START_DATE}" == "TODAY" ]; then
     INGEST_START_DATE=$(/bin/date "+%Y-%m-%d")
     echo "Ingestion Start Date: TODAY > calculated start date: ${INGEST_START_DATE}"
fi

if [ "${INGEST_END_DATE}" == "TODAY" ]; then
     INGEST_END_DATE=$(/bin/date "+%Y-%m-%d")
     echo "Ingestion End Date: TODAY > calculated end date: ${INGEST_END_DATE}"
fi

# Set the Working-Dir (were zipped files are downloaded) and the Unzip-Dir
ZIP_DOWNLOAD_DIR=${S1_WORKDIR}"/S1-DATA-"${COUNTRY}-${ptype}
UNZIP_DIR=${ZIP_DOWNLOAD_DIR}.UNZIP/

LOG_PATH=${S1_WORKDIR}"/S1_DOWNLOAD_LOGS/"
LOG_FILE=${LOG_PATH}"S1-${COUNTRY}-${ptype}_${INGEST_START_DATE}-til-${INGEST_END_DATE}_${LOG_DATE}.txt"

#Temp log file
LOG_SUMMARY=${LOG_PATH}"S1-${COUNTRY}-${ptype}_${INGEST_START_DATE}-til-${INGEST_END_DATE}_${LOG_DATE}-SUMMARY.txt"
LOG_ALL=${LOG_PATH}"S1-${COUNTRY}-${ptype}_${INGEST_START_DATE}-til-${INGEST_END_DATE}_${LOG_DATE}-ALL.txt"

QUERY_LOG_PATH=${LOG_PATH}"QUERY_RESULT_LIST/"
QUERY_RESULT_LOG="${QUERY_LOG_PATH}S1-${COUNTRY}-${ptype}_${INGEST_START_DATE}-til-${INGEST_END_DATE}_${LOG_DATE}.xml"

#echo "LOG_FILE: ${LOG_FILE}"
#echo "QUERY_RESULT_LOG: ${QUERY_RESULT_LOG}"

# Create the directories if not present
if [ ! -e ${ZIP_DOWNLOAD_DIR} ]
   then
   /bin/mkdir -p ${ZIP_DOWNLOAD_DIR}
fi

if [ ! -e $UNZIP_DIR ]
    then
    /bin/mkdir -p $UNZIP_DIR
fi

if [ ! -e ${LOG_PATH} ]
    then
    /bin/mkdir -p ${LOG_PATH}
fi

if [ ! -e ${QUERY_LOG_PATH} ]
    then
    /bin/mkdir -p ${QUERY_LOG_PATH}
fi

echo "This is script: $0 v${VERSION} running on $FULLHOSTNAME" | $TEE_PREFIX -a ${LOG_FILE}
echo "Script execution start date: ${SCRIPT_EXEC_STARTDATE}"      | $TEE_PREFIX -a ${LOG_FILE}
echo ""                                                          | $TEE_PREFIX -a ${LOG_FILE}

echo "Command used:"   | $TEE_PREFIX -a ${LOG_FILE}
echo "$0 $@"           | $TEE_PREFIX -a ${LOG_FILE}
```

```
echo ""             | $TEE_PREFIX -a ${LOG_FILE}

echo "Working Dir:"      | $TEE_PREFIX -a ${LOG_FILE}
echo "${ZIP_DOWNLOAD_DIR}"   | $TEE_PREFIX -a ${LOG_FILE}
echo ""             | $TEE_PREFIX -a ${LOG_FILE}

echo "Unzip Dir:"      | $TEE_PREFIX -a ${LOG_FILE}
echo "${UNZIP_DIR}"       | $TEE_PREFIX -a ${LOG_FILE}
echo ""             | $TEE_PREFIX -a ${LOG_FILE}

echo "Log saved to:"    | $TEE_PREFIX -a ${LOG_FILE}
echo "${LOG_FILE}"        | $TEE_PREFIX -a ${LOG_FILE}
echo ""             | $TEE_PREFIX -a ${LOG_FILE}

echo "Scihub Query Result List saved to:"   | $TEE_PREFIX -a ${LOG_FILE}
echo "${QUERY_RESULT_LOG}"                | $TEE_PREFIX -a ${LOG_FILE}
echo ""                           | $TEE_PREFIX -a ${LOG_FILE}

# Always mount the SMB Disc via Applescript osascript, this avoids writing an empty 'DiscData' Folder to /Volumes/
#/usr/bin/osascript -e 'mount volume "smb://Nicolas:data@discdata.ecgs.welter/DiscData"'
#MOUNT_RESULT=$?

#if [ $MOUNT_RESULT -ge 1 ]
#then
#   echo "Problem mounting the Newtwork Disc, mount result: $MOUNT_RESULT, exiting now"
#   exit 1
#else
#   echo "Succesfully mounted Network Disc, discdata.ecgs.welter | Mount result: $MOUNT_RESULT"
#fi

# Redefine Email Subject
EMAIL_SUBJECT=${EMAIL_SUBJECT}"${COUNTRY} product=${ptype} - Ingestion Start:${INGEST_START_DATE} til:${INGEST_END_DATE}"


#function ping_esa_site
#{
#   # -q quiet
#   # -c nb of pings to perform
#    echo "Checking if site scihub.copernicus.eu is up." | $TEE_PREFIX -a ${LOG_FILE}
#    echo "ping command result:" | $TEE_PREFIX -a ${LOG_FILE}
#    echo "" | $TEE_PREFIX -a ${LOG_FILE}
#   /sbin/ping -q -c10 scihub.copernicus.eu > /dev/null >> ${LOG_FILE}
#}

# Display a banner with the passed text (limited to 20 lines)
function show_text
{
   echo  "----------------------------------------"
   echo "$1" | /usr/bin/head -20
   [ $(/bin/echo "$1" | /usr/bin/wc -l) -gt 20 ] && /bin/echo "[Truncated to 20 lines]..."
   echo  "----------------------------------------"

}

# Return list of values for the passed field name from the result file depending on its json or xml format
function get_field_values
{
   /bin/echo "Entering get_field_values()"

   field_name="$1"
   QUERY_RESULT_LIST=$(/bin/cat "${QUERY_RESULT_LOG}" | ${XMLSTARLET_PREFIX} sel -T -t -m "//*[local-name()='entry']//*[local-
name()='$field_name']" -v '.' -n)
}


function query_server
{
   #echo "Entering query_server()"
   #echo "query_server() QUERY_RESULT_LIST_LOG: ${QUERY_RESULT_LOG}"

   # Get URL and filter space characters
   URL="${1// /%20}"
   [ "$VERBOSE" = "true" ] && show_text "${CURL_PREFIX} \"$URL\""
    ${CURL_PREFIX} "$URL" > "${QUERY_RESULT_LOG}"

    [ "$VERBOSE" = "true" ] && show_text "$(${XMLSTARLET_PREFIX} fo "${QUERY_RESULT_LOG}")"
}

function show_numbered_list
{
   # Get number of items in the list
   LIST="$1"

   /bin/echo "--- Query Result List for the following parameters: ---"

   RESULT_QUERY_MAINTENANCE=$(/bin/cat ${QUERY_RESULT_LOG} | grep "maintenance")
   RESULT_QUERY_ERROR=$(/bin/cat ${QUERY_RESULT_LOG} | grep "error")

   if [[ ${RESULT_QUERY_MAINTENANCE} == *"maintenance"* ]]; then
         nb_items=0
         MAINTENANCE_MODE=true
       echo  "----------------------------------------------"
      echo "Maintenance Mode detected for ESA Scihub Site !"
       echo  "----------------------------------------------"
      echo "Here's the content of the query result list:"
         echo ""
         /bin/cat ${QUERY_RESULT_LOG}
         echo ""
        echo  "----------------------------------------"
      echo ""
   elif [[ ${RESULT_QUERY_ERROR} == *"error"* ]]; then
      nb_items=0
       MAINTENANCE_MODE=true
       echo  "------------------------------------------------------"
      echo "Error detected! ESA Scihub Site seems to have an error."
       echo  "------------------------------------------------------"
```

```
      echo "Here's the content of the query result list:"
         echo ""
         /bin/cat ${QUERY_RESULT_LOG}
         echo ""
        echo  "---------------------------------------"
      echo ""
   elif [ ! "$LIST" ]; then
      nb_items=0
      MAINTENANCE_MODE=false
         echo "Empty Query Result List for:"
         echo "Country: $COUNTRY | Product type: $ptype"
         echo "Ingestion start date: ${INGEST_START_DATE}"
         echo "Ingestion End date: ${INGEST_END_DATE}"
         echo "Polygon choosed: ${polygon} "
         echo ""
         echo " Here's the content of the query result list:"
         echo ""
         /bin/cat ${QUERY_RESULT_LOG}
         echo ""
         echo  "---------------------------------------"
      echo ""
   else
    # Loop on list and add number as prefix
    nb_items=$(/bin/echo "$LIST" | /usr/bin/wc -l | /usr/bin/tr -d ' ')
    echo "Country: $COUNTRY"
    echo "Product type: $ptype"
   echo "Ingestion Start date:   ${INGEST_START_DATE}"
   echo "Ingestion End date:   ${INGEST_END_DATE}"
   echo "Polygon choosed: ${polygon}"
   echo ""
   echo  "---------------------------------------"
   echo "Query Result List has ${nb_items} item(s): "
    echo  "---------------------------------------"

   OLD_IFS=$IFS
    IFS=$'\n'

   local i=0

    for item in $LIST
    do
       i=$(expr ${i} + 1)
       #printf "Nr.[%03d] " ${i}
       #printf "Product-ID: ${item} \n"
       printf "Nr.[%03d] Product-ID: ${item} \n"  ${i}
   done

    IFS=$OLD_IFS

    echo  "---------------------------------------"

   fi
}

function verify_md5_checksums
{
    # Get the MD5 value to compare with the zipped file
    MD5_URL="${ROOT_URL_ODATA}/Products('$prodid')/Checksum/Value/\$value"
   md5OnlineUppercase=$(${CURL_PREFIX} "${MD5_URL}")

   # Convert value to lowercase, since the command md5 outputs a lowercase value
   md5Online=`echo "$md5OnlineUppercase" | tr '[:upper:]' '[:lower:]'`
    echo "+--MD5 checksum (scihub site): ${md5Online}"

    # Check MD5 value of zipped file, lowercase value output
    echo "+--Calculating MD5 checksum of Zip file, be patient..."
   md5zipFile=$(${MD5_PREFIX} ${zippedProductFile})
   echo "+--MD5 checksum of file: ${md5zipFile}"

   # Compare MD5 values
   if [ "${md5zipFile}" = "${md5Online}" ]; then
       echo "+--SUCCESS: MD5 checksums matches! Continuing..."
           return 0;
    else
       echo "+--ERROR: MD5 checksums, Zip file seems broken!"
       return -1;
   fi
}

function download_quicklook_and_full_file_list
{
   echo ""
     echo "#### BEGIN PROCESSING ####"
   echo ""
   echo ""

     cd ${ZIP_DOWNLOAD_DIR}

    # Declare an indexed array "downloadArray" and initializes it to be empty.
   # Add the name of the Zip-File downloaded.
   # Will be printed at the end of the script
   declare -a downloadArray
   declare -a unzipArray

   declare -a remoteZipSizeArray
   declare -a localZipSizeArray

   local i=0
      local curlReturnValue="0"

   # Loop on list and add number as prefix
   for prodid in ${LIST}
   do
         prodid=${prodid//\"/}

        ## DEBUG:
```

```
        ## echo "prodid = $prodid"

      # Build URL to get product name
      PRODUCT_NAME_URL="${ROOT_URL_ODATA}/Products('$prodid')/Name/\$value"
      ## DEBUG:
      ## echo "PRODUCT_NAME_URL: ${PRODUCT_NAME_URL}";
      prodname=$(${CURL_PREFIX} "${PRODUCT_NAME_URL}")
      ## DEBUG:
      ## printf "prodname:%s \n" $prodname

      # First check if ESA Site is down, if site is in maintenance mode the Zip-File contains
      # the text: "doctype html" or "maintenance"
      # e.g +-Zip-File name: <!doctype html>
      #   <title>The Sentinels Scientific Data Hub</title>
      #   <link href='https://fonts.googleapis.com/css?family=Open+Sans' rel='stylesheet' type='text/css'>
      #   ....
      #   Sorry for the inconvenience,<br/>
      #   we're performing some maintenance at the moment.<br/>

      if [[ ${prodname} == *"doctype"* ]]; then
         echo "+-ALERT! ESA Download Site seems in maintenance mode, cancelling all downloads!"
         MAINTENANCE_MODE=true
         return 1
      fi # END -- if [[ ${prodname} == *"doctype html"* ]]

      # Check
      # https://scihub.copernicus.eu/dhus/odata/v1/Products?$filter=Name eq
'S1A_IW_SLC__1SDV_20141101T165548_20141101T165616_003091_0038AA_558F'
      # Check if the big zipfile is already locally saved but we don't know if the file is fully downloaded
      # Make some file test on zipfile
      zippedProductFile="$prodname.zip"

      unzippedProductFile="$prodname.SAFE"
      unzippedProductFileDirectory="${UNZIP_DIR}${unzippedProductFile}"

      quicklookfile="$prodname.quick-look.png"
      local downloadRequired=false

        # Increment i to print out the filename number
          i=$(expr ${i} + 1)

      printf "+-Product Nr.[%03d/%03d] ---\n" ${i} ${nb_items}
      printf "+-Product-ID: $prodid corresponds to:\n"
      printf "+-Remote Zip-File name: ${zippedProductFile}\n"

      PRODUCT_ZIPFILE_URL="${ROOT_URL_ODATA}/Products('$prodid')/\$value"

          ## DEBUG
      #remoteZipContent=$(${CURL_PREFIX} -sI "${PRODUCT_ZIPFILE_URL}" )
         #echo "remoteZipContent:${remoteZipContent}"
         #printf "+--DEBUG: Content of remote ZipFile:\n${remoteZipContent}"
         ## END DEBUG

         # Old version: "Content-Length"
         #remoteZipFileSize=$(${CURL_PREFIX} -sI "${PRODUCT_ZIPFILE_URL}" | grep "Content-Length:" | awk '{print $2}')

         remoteZipFileSize=$(${CURL_PREFIX} -sI "${PRODUCT_ZIPFILE_URL}" | grep "content-length:" | awk '{print $2}')
       #echo "remoteZipFileSize: $remoteZipFileSize"

         # Remove the \r from remoteZipFileSize
         remoteZipFileSize=$(echo "$remoteZipFileSize" | /usr/bin/tr -d '\r' )

       echo "+--Remote Zip-File size is ${remoteZipFileSize} bytes "

       if [ "${remoteZipFileSize}" -eq "0" ]; then
         printf "+--Remote Zip-File size is 0 bytes: maybe remote file is unavailable or your download quota is exceeded.\n\n"
         printf "+--Remote Zip-File size is 0 bytes: maybe remote file is unavailable or your download quota is exceeded.\n\n" >>
${LOG_FILE}
         printf "+-Skipping this download\n" >> ${LOG_FILE}
      fi

      # Check if local Zip-File exists and check its content
      if [ "${remoteZipFileSize}" -gt "0" ]; then

         # Check if zip file already exists
         echo "+--Checking if local Zip-File already exists in ${ZIP_DOWNLOAD_DIR} " >> ${LOG_FILE}

         if test -f "${ZIP_DOWNLOAD_DIR}/${zippedProductFile}"; then

            # If Zip-File size is less than 1kB it means that an error occured or
                                      # an error-message was written to it
            echo "+--Checking local Zip-File size" >> ${LOG_FILE}
            localZipfileSize=$(stat -c%s "${ZIP_DOWNLOAD_DIR}/${zippedProductFile}")
            if [ ${localZipfileSize} -ge "0" ] && [ ${localZipfileSize} -le "1000" ]; then

               # if garbled Zip-file (containing the error-message) is older than 1 day
                                             # (=1440 min) we delete it
               if [ test `find ${ZIP_DOWNLOAD_DIR}/${zippedProductFile} -mmin +1440` ]; then
                 echo "+--Deleting garbled zipfile ${ZIP_DOWNLOAD_DIR}/${zippedProductFile}"
                      /bin/rm ${ZIP_DOWNLOAD_DIR}/${zippedProductFile}
               else
                 # If zipfile is less than one day old, we check the content
                 # of the zipfile (if download quota was exceeded)
                 zipFileContent=$(/bin/cat ${ZIP_DOWNLOAD_DIR}/${zippedProductFile})

                 if [[ ${zipFileContent} == *"quota"* ]]; then
                   echo "+--Content of ZipFile: ${zipFileContent}"
                   echo "+--Content of Zip-File: ${zipFileContent}" >> ${LOG_FILE}
                   echo "++ALERT! Quota download exceeded, cancelling all downloads !"
                   echo "++ALERT! Quota download exceeded, cancelling all downloads !" >> ${LOG_FILE}
                   /usr/bin/mail -s "[${HOSTNAME}] Sentinel1 Download Quota exceeded for ${COUNTRY} product=${ptype} - Ingestion
Start Date:${INGEST_START_DATE} til End Date:${INGEST_END_DATE}" "${EMAIL_RECIPIENTS}" < ${LOG_FILE}
                   exit
                 fi # END -- [[ ${zipFileContent} == *"quota"* ]];
               fi # END -- if test 'find....'
```

```
      fi # END -- if [ ${localZipfileSize} -ge "0" ] && [ ${localZipfileSize} -le "1000" ]; then
   else
      echo "+--Local Zip-File doesn't exist...continuing."
      echo "+--Local Zip-File doesn't exist...continuing." >> ${LOG_FILE}
   fi # END -- if test -f "${ZIP_DOWNLOAD_DIR}/${zippedProductFile}"; then

if [ ${FORCE_DOWNLOAD} = false ]; then

    if [ ! -d ${unzippedProductFileDirectory} ] && [ -e "${ZIP_DOWNLOAD_DIR}/${zippedProductFile}" ]; then
      echo "+--Zip-File exists but UnZip-Dir missing !"
      echo "+--Zip-File already downloaded: YES"
      echo "+--UnZip-Dir: ${unzippedProductFile} missing!"
      echo "+--Location: ${ZIP_DOWNLOAD_DIR}/${zippedProductFile}"

      localZipFileSize=$(stat -c%s "${ZIP_DOWNLOAD_DIR}/${zippedProductFile}")
      echo "+--Local Zip-File size in bytes:  ${localZipFileSize}"
      echo "+--Remote Zip-File size in bytes: ${remoteZipFileSize}"

      ## DEBUG only
      #echo "+--Displaying content of ${zippedProductFile}"
      #echo "+--in dir: ${ZIP_DOWNLOAD_DIR}"
      #zipFileContent=$(/bin/cat ${ZIP_DOWNLOAD_DIR}/${zippedProductFile})
      #echo "+--Content of zipFile: ${zipFileContent}"
      #echo "+--Content of zipFile: ${zipFileContent}" > ${LOG_FILE}

      if [ "${remoteZipFileSize}" -eq "${localZipFileSize}" ]; then
         echo "+--Zip-File seems OK. Unzipping file."
         # 7zip Manual: http://7zip.bugaco.com/7zip/MANUAL/index.htm
             # 7zip: Skip extracting of existing files with option: -aos
             ${UNZIP_PREFIX} ${zippedProductFile} -o${UNZIP_DIR}
             unzipReturnValue=$?

             if [ "${unzipReturnValue}" -eq "0" ]; then
                 printf "+--Unzipping succesful for Zip-File Nr.[%03d/%03d]\n" ${i} ${nb_items}
                 downloadRequired=false
                 unzipArray+=("${unzippedProductFile}")
         else
            echo "+--ERROR: Unzipping file, redownload required!"
            downloadRequired=true
            fi
         else
            echo "+--ERROR: File sizes don't match, redownload required!"
            downloadRequired=true
         fi # END -- if [ ${remoteZipFileSize} -eq ${localZipFileSize} ];

      elif [ -d "${unzippedProductFileDirectory}" ] && [ -e "${zippedProductFile}" ]; then
         echo "+--Zip-File & UnZip-Dir exists !"
            downloadRequired=false

         echo "+--UnZip-Dir exists: ${unzippedProductFile}"
         localDirSize=$($DU_PREFIX ${unzippedProductFileDirectory} | cut -f1)
         echo "+--UnZip-Dir size in bytes: ${localDirSize}"

         localZipFileSize=$($DU_PREFIX ${zippedProductFile} | cut -f1)
         echo "+--Local Zip-File size in bytes: ${localZipFileSize}"

           if [ ${localDirSize} -ge ${remoteZipFileSize} ]; then
            echo "+--UnZip dir size >= as remote Zip-File, seems OK."
            downloadRequired=false
            echo "+--Checking if Zip-File is older than 30 days:"

                 if [[ $(/usr/bin/find "${zippedProductFile}" -mtime +30 -print) ]]; then
              echo "+--Zip-File is older than 30 days! Will be deleted!"
                 if [ ${DELETE_ZIPFILE_30DAYS} = true ]; then
                 echo "+--DELETE_ZIPFILE_30DAYS option set, deleting Zip-File!"
                 /bin/rm -rf "${zippedProductFile}"
              else
                 echo "+--DELETE_ZIPFILE_30DAYS option not set, not deleting Zip-File."
              fi
           else
              echo "+--Zip-File is not older than 30 days. Will not be deleted."
         fi # END -- if [ ${localDirSize} -ge ${remoteZipFileSize} ];
         else
         echo "+--UnZip-Dir size is smaller than Zip-File. Trying unzipping file."
         echo "+--Checking if local Zip-File size equals to remote size:"

         if [ ${remoteZipFileSize} -eq ${localZipFileSize} ]; then
            echo "+---Sizes are equal. Trying to unzip file."
            # 7zip Manual: http://7zip.bugaco.com/7zip/MANUAL/index.htm
                   # 7zip: Skip extracting of existing files with option: -aos
                ${UNZIP_PREFIX} ${zippedProductFile} -o${UNZIP_DIR}
                unzipReturnValue=$?

             if [ ${unzipReturnValue} -eq "0" ]; then
                    printf "+---Unzipping done for Zip-File Nr.[%03d/%03d]\n" ${i} ${nb_items}
                    downloadRequired=false
                    unzipArray+=("${unzippedProductFile}")
                 elif [ ${unzipReturnValue} -eq "2"]; then
                    printf "+---Fatal Error unzipping file Nr. [%03d/%03d]\n. Skipping!" ${i} ${nb_items}
                    downloadRequired=false
            else
               echo "+---ERROR: Unzipping file, redownload required!"
               downloadRequired=true
            fi # END -- if [ ${unzipReturnValue} -eq "0" ];
         fi # END -- if [ $remoteZipFileSize -eq localZipFileSize ]; then

      fi # END - if [ ${localDirSize} -ge ${remoteZipFileSize} ];

   elif [ -d "${unzippedProductFileDirectory}" ] && [ ! -e "${zippedProductFile}" ]; then

      echo "+--Zip-File missing but UnZip-Dir exists!"
      echo "+--Checking size of UnZip-Dir: ${unzippedProductFile}"

      localDirSize=$($DU_PREFIX ${unzippedProductFileDirectory} | cut -f1)
      echo "+--UnZip-Dir size in bytes: ${localDirSize}"
      echo "+--Remote Zip-File size in bytes: ${remoteZipFileSize}"
```

```
            if [ ${localDirSize} -ge ${remoteZipFileSize} ]; then
             echo "+--UnZip-Dir size is greater or same as Zip-File, seems OK, manual checking preferred."
             downloadRequired=false
             else
             echo "+--UnZip-Dir size is smaller than Zip-File. Redownload required."
             downloadRequired=true
           fi

        elif [ ! -d "${unzippedProductFileDirectory}" ] && [ ! -e "${zippedProductFile}" ]; then
           echo "+--Zip-File & UnZip-Dir missing. Download required."
           downloadRequired=true
        fi # END -- if [ -d ${unzippedProductFileDirectory} ] && [ -e "${zippedProductFile}" ];

      fi # END -- if [ $FORCE_DOWNLOAD=false ]

        # Download only if downloadRequired is true, or if we Force Download
        # By default: downloadRequired and FORCE_DOWNLOAD are set to false

        if [ "${downloadRequired}" = true ] || [ ${FORCE_DOWNLOAD} = true ]; then

           # Set returnValue to 999:
         # Normally all unix commands (curl, 7zip) return value equals to "0", if everything is OK
           local returnValue="999"

           ZIPDOWNLOAD_STARTDATE=$(/bin/date +"%A, %d %B %Y @ %H:%M (%Y%m%d%H%M)")

           echo "+--Zip-File already downloaded: NO"
           echo "+--Trying to download Product-ID as Zip-File."
          echo "+--Remote Zip-File size in bytes: $remoteZipFileSize"
          echo "+--Zip-File Download Start Date: ${ZIPDOWNLOAD_STARTDATE}"

           # Use "-C -" to tell curl to automatically find out where/how to resume the transfer
               ${CURL_PREFIX} -C - --silent --show-error -o "${zippedProductFile}" "${PRODUCT_ZIPFILE_URL}" 2>&1
               #${CURL_PREFIX} -C - --progress-bar --show-error -o "${zippedProductFile}" "${PRODUCT_ZIPFILE_URL}" 2>&1
               #${CURL_PREFIX} --show-error -o "${zippedProductFile}" "${PRODUCT_ZIPFILE_URL}" 2>&1
               curlReturnValue=$?

         if [ ${curlReturnValue} -ne "0" ]; then
            printf "+--ERROR downloading  Zip-File Nr.[%03d/%03d]\n" ${i} ${nb_items}
            printf "+--ERROR: Maybe Site is down ? Check News at: https://scihub.copernicus.eu/news/ \n"
            return 1
         else
            printf "+--Zip-File download done! Saved as: ${zippedProductFile}\n"
            # Add the element to array
            downloadArray+=("${zippedProductFile}")
            remoteZipSizeArray+=("${remoteZipFileSize}")

            #Check the size of the local file, and store it in the array
            localZipFileSize=$($DU_PREFIX ${zippedProductFile} | cut -f1)
           localZipSizeArray+=("${localZipFileSize}")
         fi

        ZIPDOWNLOAD_ENDDATE=$(/bin/date +"%A, %d %B %Y @ %H:%M (%Y%m%d%H%M)")
         printf "+--Zip-File Download End Date: ${ZIPDOWNLOAD_ENDDATE}\n"

          # Verifying md5 checksums
          # Loop through the checksum verification until the file was correctly downloaded
         while [ ${returnValue} -ne "0" ]
            do
               if [ ${SKIPMD5CHECK} = false ] ; then
                  # Verify md5 checksums of zipped file & scihub site
               verify_md5_checksums
               # If md5 checksums matches => returnValue=0, then we unzip the zipped file.
               returnValue=$?
               echo "md5 checksum returnValue :" ${returnValue}
            else
               printf "+--Skipping MD5 Checksum requested: YES\n"
                  returnValue="0" # set returnValue to 0 to continue unzipping
            fi # -- if [ ${SKIPMD5CHECK} = false ]

           # Check if zip file already exists
           echo "+--Checking if product zipfile already exists" > ${LOG_FILE}
           if [ -e "${ZIP_DOWNLOAD_DIR}/${zippedProductFile}" ]; then
              echo "+--Checking product zipfile size" > ${LOG_FILE}

              # Check zipfileContent if download quoata was exceeded
              zipfileSize=$(stat -c%s "${ZIP_DOWNLOAD_DIR}/${zippedProductFile}")

              if [ ${zipfileSize} -ge "0" ] && [ ${zipfileSize} -le "1000" ]; then
                 echo "++ERROR: It seems your download quoata was exceeded..." >> ${LOG_FILE}
                 zipFileContent=$(/bin/cat ${ZIP_DOWNLOAD_DIR}/${zippedProductFile})

                 if [[ ${zipFileContent} == *"quota"* ]]; then
                    echo "++ALERT! Quota download exceeded, cancelling all downloads !" >> ${LOG_FILE}
                    echo "+--Content of zipFile: ${zipFileContent}" >> ${LOG_FILE}
                    /usr/bin/mail -s "[${HOSTNAME}] Sentinel1 Download Quota exceeded for ${COUNTRY} product=${ptype} - Ingestion
Start Date:${INGEST_START_DATE} til End Date:${INGEST_END_DATE}" "${EMAIL_RECIPIENTS}" < ${LOG_FILE}
                    exit
                    #break # Skip entire rest of loop.
                    #return 1
                 fi # END -- [[ ${zipFileContent} == *"quota"* ]];

              elif [ ${zipfileSize} -eq "0" ]; then
                 printf "+--ERROR: Zip-File Nr.[%03d/%03d] has 0 bytes, will not be unzipped...\n" ${i} ${nb_items}
                 # First we delete the file (or else it can loop infinitely if the file was not correctly downloaded)
                  #echo "+---Deleting the broken product file: ${ZIP_DOWNLOAD_DIR}/${zippedProductFile}"
                  #/bin/rm -v "${ZIP_DOWNLOAD_DIR}/${zippedProductFile}"
                  returnValue="999"
               else
                  returnValue="0"
               fi # END -- if [ ${zipfileSize} -gt "0" && ${zipfileSize} -lt "1000" ]

            else
               returnValue="999"
           fi   # END -- if test -f "${ZIP_DOWNLOAD_DIR}/${zippedProductFile}"; then
```

```
            # If returnValue = 0 means that we got no md5 checksum error
            if [ ${returnValue} -eq "0" ]; then

              printf "+--Unzipping file to dir: ${UNZIP_DIR}/${unzippedProductFile}\n"
              # 7zip Manual: http://7zip.bugaco.com/7zip/MANUAL/index.htm
                    # 7zip: Skip extracting of existing files with option: -aos
                  ${UNZIP_PREFIX} ${zippedProductFile} -o${UNZIP_DIR}
                  unzipReturnValue=$?
                  echo "unzipReturnValue= ${unzipReturnValue}"

                  if [ ${unzipReturnValue} -eq "0" ]; then
                      printf "+--Unzipping done for Zip-File Nr.[%03d/%03d]\n" ${i} ${nb_items}
                      unzipArray+=("${unzippedProductFile}")
                      returnValue="0"
                  elif [ ${unzipReturnValue} -eq "2" ]; then
                     echo "Error unzipping Zip-File"
                     returnValue="999"
                  fi
               fi

               if [ ${returnValue} -eq "999" ]; then

                  printf "+--REDOWNLOAD: Zip-File Nr.[%03d/%03d] seems broken. Trying to redownload file!\n" ${i} ${nb_items}
                  # Since we want a return value from 7Zip, we don't use 'tee' to output to LOG_FILE
                  # If you pipe the curl command to 'tee' you will get the return value, but not the returnvalue from curl itself!
                  PRODUCT_ZIPFILE_URL="${ROOT_URL_ODATA}/Products('$prodid')/\$value"

                  # First we delete the file (or else it can loop infinitely if the file was not correctly downloaded)
                  echo "+---Deleting the broken product file: ${ZIP_DOWNLOAD_DIR}/${zippedProductFile}"
                  /bin/rm -v "${ZIP_DOWNLOAD_DIR}/${zippedProductFile}"

                  echo "+---Trying redownload!"
                   #${CURL_PREFIX} -C - --silent --show-error -o "${zippedProductFile}" "${PRODUCT_ZIPFILE_URL}" 2>&1
                   #${CURL_PREFIX} -C - --show-error -o "${zippedProductFile}" "${PRODUCT_ZIPFILE_URL}" 2>&1
                   ${CURL_PREFIX} --show-error -o "${zippedProductFile}" "${PRODUCT_ZIPFILE_URL}" 2>&1

                  returnValue=$?
                 echo "curl - returnValue=${returnValue}"

                if [ ${returnValue} -ne "0" ]; then
                    printf "+--Error redownloading Zip-File! Maybe Site is down ? Check News here:
https://scihub.copernicus.eu/news/ \n"
                    break
                 else
                    printf "+--Redownloading Zip file [Nr.%03d/%03d] was successful.\n" ${i} ${nb_items}
                    # Setting returnValue to 999 for do-while loop to reverify md5 checksum
                    # and eventually redownload file if md5 checksum does not match.
                    returnValue="0"
                 fi # END -- if [ ${returnValue} -eq "999" ]

               fi # END -- if [ ${unzipReturnValue} -eq "0" ]

          done # END -- while [ ${returnValue} -ne "0" ]

             if [ ${returnValue} -ne "0" ]; then
                printf "+--ERROR downloading Product Zip-File [Nr.%03d/%03d]\n" ${i} ${nb_items}
             else
                  printf "+--Download done for Product Zip-File [Nr.%03d/%03d]\n" ${i} ${nb_items}
            fi

          fi # END -- if [ ! -e "${zippedProductFile}" ] && [ "$downloadRequired" = true ]

         echo "+-Done for Zip-File"
         #echo "+-Sleeping for 1 minute before downloading Quicklook file"
         #sleep 60

      if [ -e "$quicklookfile" ]; then
            printf "+-Quick-look file already downloaded: YES\n"
        elif [ ! -e "$quicklookfile" ]; then
             printf "+-Quick-look file already downloaded: NO\n"
          # Build URL to get quick-look
           printf "+--Downloading the Quick-look PNG file.\n"
             returnValue="999"

             while [ ${returnValue} -ne "0" ]
             do
               QUICKLOOK_URL="${ROOT_URL_ODATA}/Products('$prodid')/Nodes('$prodname.SAFE')/Nodes('preview')/Nodes('quick-
look.png')/\$value"

              # Since we want a return value from curl, we don't use $TEE_PREFIX to output to LOG_FILE
             # If you pipe the curl command to $TEE_PREFIX you will get the return value, but no the one from curl itself!
             #${CURL_PREFIX} -C - --verbose -o "$quicklookfile" "${QUICKLOOK_URL}" 2>&1
             ${CURL_PREFIX} -C - --silent --show-error -o "$quicklookfile" "${QUICKLOOK_URL}" 2>&1
             returnValue=$?
             #echo "Curl Quicklook download return code: " ${returnValue}
          done
       fi # END -- if [ ! -e "$quickLookFile" ]

      echo "+--Quick-look file saved as: $quicklookfile"
        echo "+--Done for Quick-look File"

        printf "+-End Processing Product Nr.[%03d/%03d] ---\n" ${i} ${nb_items}
        echo ""

     fi # END -- if [ $remoteZipFile -gt "0" ]

   done # END -- for prodid in ${LIST}

    echo "" | $TEE_PREFIX -a ${LOG_SUMMARY}
   echo "#### --- SUMMARY --- ####" | $TEE_PREFIX -a ${LOG_SUMMARY}
   echo "" | $TEE_PREFIX -a ${LOG_SUMMARY}

   # Get the array length
   numberOfDownloadedFiles=${#downloadArray[@]}
```

```
   if [ ${numberOfDownloadedFiles} -gt "0" ]; then
      echo "The following $numberOfDownloadedFiles file(s) were downloaded to directory:" | $TEE_PREFIX -a ${LOG_SUMMARY}
      echo "" | $TEE_PREFIX -a ${LOG_SUMMARY}
      echo "$ZIP_DOWNLOAD_DIR" | $TEE_PREFIX -a ${LOG_SUMMARY}
      echo "" | $TEE_PREFIX -a ${LOG_SUMMARY}

      nbrFile=1;

       for (( i = 0 ; i < ${numberOfDownloadedFiles} ; i=${i}+1 ));
       do
          printf "[%03d]: ${downloadArray[${i}]}\n" $nbrFile | $TEE_PREFIX -a ${LOG_SUMMARY}
          printf "          -Remote-FileSize: ${remoteZipSizeArray[${i}]} bytes\n" | $TEE_PREFIX -a ${LOG_SUMMARY}
          printf "          -Local-FileSize:     ${localZipSizeArray[${i}]} bytes \n" | $TEE_PREFIX -a ${LOG_SUMMARY}
          echo "" | $TEE_PREFIX -a ${LOG_SUMMARY}

          nbrFile=$[${nbrFile} + 1]
       done

   else
      echo "*** No Zip-Files were downloaded. ***" | $TEE_PREFIX -a ${LOG_SUMMARY}
   fi # END -- if [ ${numberOfDownloadedFiles} -gt "0" ];
   echo ""  | $TEE_PREFIX -a ${LOG_SUMMARY}

   # Get the array length
   numberOfUnzippedFiles=${#unzipArray[@]}

   if [ ${numberOfUnzippedFiles} -gt "0" ]; then
      echo "The following $numberOfUnzippedFiles file(s) were unzipped to directory:" | $TEE_PREFIX -a ${LOG_SUMMARY}

      echo "$UNZIP_DIR"  | $TEE_PREFIX -a ${LOG_SUMMARY}
      echo "" | $TEE_PREFIX -a ${LOG_SUMMARY}
      nbrFile=1;

      for iUnzipName in "${unzipArray[@]}"
      do
         printf "[%03d]: ${iUnzipName} \n" $nbrFile | $TEE_PREFIX -a ${LOG_SUMMARY}
         nbrFile=$[${nbrFile} +1]
      done
   else
      echo "*** No Zip-Files were unzipped. ***" | $TEE_PREFIX -a ${LOG_SUMMARY}
   fi # END -- if [ ${numberOfUnzippedFiles} -gt "0" ];
   echo "" | $TEE_PREFIX -a ${LOG_SUMMARY}

   echo "#### --- END SUMMARY --- ####" | $TEE_PREFIX -a ${LOG_SUMMARY}
   echo ""  | $TEE_PREFIX -a ${LOG_SUMMARY}
   echo ""  | $TEE_PREFIX -a ${LOG_SUMMARY}
   echo "---- Detailed Logs ----" | $TEE_PREFIX -a ${LOG_SUMMARY}
   echo ""  | $TEE_PREFIX -a ${LOG_SUMMARY}


} # -- END -- function download_quicklook_and_full_file_list()


## Main Routine

# Pinging the site doesn't seem to work reliably, so we just turn it OFF
# Note: don't add the 'tee command' after calling function ping_esa_site
# or else the if conditional will be truncated to the 'tee' result value
# and not the one from the ping function command

##ping_esa_site.OFF
##if [ $? -ne 0 ]; then
#   echo "" | $TEE_PREFIX -a ${LOG_FILE}
#   echo "Ping test of site scihub.copernicus.eu seems down, continuing anyway..." | $TEE_PREFIX -a ${LOG_FILE}
#   echo "Check latest News here: https://scihub.copernicus.eu/news/" | $TEE_PREFIX -a ${LOG_FILE}
#   echo "" | $TEE_PREFIX -a ${LOG_FILE}
#   echo "" | $TEE_PREFIX -a ${LOG_FILE}

   # Send an email using /usr/bin/mail
#   /usr/bin/mail -s "${EMAIL_SUBJECT}" "${EMAIL_RECIPIENTS}" < ${LOG_FILE}
#   exit -1
#else
#   echo "ESA's scihub.copernicus.eu seems OK, continuing..." | $TEE_PREFIX -a ${LOG_FILE}
#   echo "" | $TEE_PREFIX -a ${LOG_FILE}
#fi

## Build query and replace blanks spaces by '+'
## This is the correct query, from odata-demos.sh:
## query="ingestiondate:[NOW-${maxRows}DAYS TO NOW] AND producttype:${ptype} AND footprint:\"Intersects(${polygon})\""
# Example for 2017-04-01 til 2017-05-01
# query="ingestiondate:[2017-04-01T00:00:00.000Z TO 2017-05-01T00:00:00.000Z] AND producttype:${ptype} AND
footprint:\"Intersects(${polygon})\""
#
# Making queries with ingestion start date and ingestion end date
#query="ingestiondate:[${INGEST_START_DATE}T00:00:00:000Z TO ${INGEST_END_DATE}T00:00:00.000Z] AND producttype:${ptype} AND
${sensormode} AND footprint:\"Intersects(${polygon})\""
query="ingestiondate:[${INGEST_START_DATE}T00:00:00:000Z TO ${INGEST_END_DATE}T00:00:00.000Z] AND producttype:${ptype} AND
${platformname} AND ${sensormode} AND footprint:\"Intersects(${polygon})\""
#query="ingestiondate:[${INGEST_START_DATE}T00:00:00:000Z TO ${INGEST_END_DATE}T00:00:00.000Z] AND producttype:${ptype} AND
${platformname} AND ${sensormode} AND footprint:\"Intersects(${polygon})\""

query_server "${ROOT_URL_SEARCH}?q=${query// /+}&rows=$maxRows&start=0"
QUERY_RESULT_LIST=$(/bin/cat "${QUERY_RESULT_LOG}" | ${XMLSTARLET_PREFIX} sel -T -t -m '//_:entry/_:id/text()' -v '.' -n)

#echo "----------------------------------------" | $TEE_PREFIX -a ${LOG_FILE}
#echo "Query sent:" | $TEE_PREFIX -a ${LOG_FILE}
#echo "$query" | $TEE_PREFIX -a ${LOG_FILE}
#echo "----------------------------------------" | $TEE_PREFIX -a ${LOG_FILE}
#echo "" | $TEE_PREFIX -a ${LOG_FILE}

# Display result list

# Get nb_items from show_numbered_list
# Note when using "tee" to write to logfile we don't get the result of nb_items
# so we are redirecting directly to ${LOG_FILE}
```

```
show_numbered_list "${QUERY_RESULT_LIST}" >> ${LOG_FILE}

# Get the result from show_number_list and store it in variable "nb_items"
if [ ${nb_items} -gt "0" ]; then
   echo "" | $TEE_PREFIX -a ${LOG_FILE}
    download_quicklook_and_full_file_list "${QUERY_RESULT_LIST}" | $TEE_PREFIX -a ${LOG_FILE}
    echo "" | $TEE_PREFIX -a ${LOG_FILE}

elif [ "$MAINTENANCE_MODE" = true ]; then
   echo "----------------------------------------"   | $TEE_PREFIX -a ${LOG_FILE}
   echo "ESA-Site seems to be in Maintenance Mode !" | $TEE_PREFIX -a ${LOG_FILE}
   echo "----------------------------------------"   | $TEE_PREFIX -a ${LOG_FILE}
   echo "" | $TEE_PREFIX -a ${LOG_FILE}
else
   echo "-----------------------------------------------" | $TEE_PREFIX -a ${LOG_FILE}
   echo "The Query Result List is empty. Nothing done!" | $TEE_PREFIX -a ${LOG_FILE}
   echo "-----------------------------------------------" | $TEE_PREFIX -a ${LOG_FILE}
   echo "" | $TEE_PREFIX -a ${LOG_FILE}
fi

SCRIPT_EXEC_ENDDATE=$(/bin/date +"%A, %d %B %Y @ %H:%M")
echo "Script Start Date:   ${SCRIPT_EXEC_STARTDATE}"   | $TEE_PREFIX -a ${LOG_FILE}
echo "Script End Date:   ${SCRIPT_EXEC_ENDDATE}"         | $TEE_PREFIX -a ${LOG_FILE}
echo "## End of script ${0} ${VERSION} ###"             | $TEE_PREFIX -a ${LOG_FILE}

# Send an email using /usr/bin/mail
if [ "$MAINTENANCE_MODE" = false ]; then
    # Combine the summary log (before) and the log file together
    /bin/cat ${LOG_SUMMARY} ${LOG_FILE} > ${LOG_ALL}

   #/usr/bin/mail -s "${EMAIL_SUBJECT}" "${EMAIL_RECIPIENTS}" < ${LOG_FILE}
   /usr/bin/mail -s "${EMAIL_SUBJECT}" "${EMAIL_RECIPIENTS}" < ${LOG_ALL}
else
   /usr/bin/mail -s "[${HOSTNAME}] Sentinel1 Maintenance Mode detected for ${COUNTRY} product=${ptype} - Ingestion Start
Date:${INGEST_START_DATE} til End Date:${INGEST_END_DATE}" "${EMAIL_RECIPIENTS}" < ${LOG_FILE}
fi

# Exit
exit 0
```

A.6.2) *Domuyo_S1_Step1_Read_SMCoreg_Pairs.sh*

```
#!/bin/bash
# Script to run in cronjob for processing DOMUYO images:
# Read images, check if nr of bursts and corners coordinates are OK,
# corigister them on a Global Primary (super master) and compute the compatible pairs.
# It also creates a common baseline plot for  ascending and descending modes.

# New in Distro V 2.0.0 20220602 :  - use new Prepa_MSBAS.sh compatible with D Derauw and L. Libert tools for Baseline Ploting
# New in Distro V 3.0.0 20230104 :  - Use Read_All_Img.sh V3 which requires 3 more parameters (POl + paths  to RESAMPLED and to SAR_MASSPROCESS)
# New in Distro V 3.1.0 20230626 :  - Color tables are now in TemplatesForPlots
# New in Distro V 4.0.0 20230830:   - Rename SCRIPTS_OK directory as SCRIPTS_MT
#                                   - Replace CIS by MT in names
#                                   - Renamed FUNCTIONS_FOR_MT.sh
# New in Distro V 5.0.0 20231030:   - Rename MasTer Toolbox as AMSTer Software
#                                   - rename Master and Slave as Primary and Secondary (though not possible in some variables and files)
#
# AMSTer: SAR & InSAR Automated Mass processing Software for Multidimensional Time series
# NdO (c) 2016/03/25 - could make better... when time.
# ------------------------------------------------------------------------------------------
source $HOME/.bashrc

echo "Starting $0" > $PATH_1650/SAR_CSL/S1/ARG_DOMU_LAGUNA/Last_Run_Cron_Step1.txt
date >> $PATH_1650/SAR_CSL/S1/ARG_DOMU_LAGUNA/Last_Run_Cron_Step1.txt

BP=20
NEWASCPATH=$PATH_1650/SAR_SM/RESAMPLED/S1/ARG_DOMU_LAGUNA_A_18/SMNoCrop_SM_20180512
NEWDESCPATH=$PATH_1650/SAR_SM/RESAMPLED/S1/ARG_DOMU_LAGUNA_D_83/SMNoCrop_SM_20180222

# Read all S1 images for that footprint
#######################################
$PATH_SCRIPTS/SCRIPTS_MT/Read_All_Img.sh $PATH_3600/SAR_DATA/S1/S1-DATA-DOMUYO-SLC.UNZIP $PATH_1650/SAR_CSL/S1/ARG_DOMU_LAGUNA/NoCrop S1
$PATH_1650/SAR_CSL/S1/ARG_DOMU_LAGUNA/DomuyoYLagunaFea.kml VV ${PATH_1650}/SAR_SM/RESAMPLED/ ${PATH_3601}/SAR_MASSPROCESS/ > /dev/null 2>&1

# Check nr of bursts and coordinates of corners. If not as expected, move img in temp quatantine and log that. Check regularily: if not updated after few days,
it means that image is bad or zip file not correctly downloaded
################################################
# Asc ; bursts size and coordinates are obtained by running e.g.:  _Check_S1_SizeAndCoord.sh /Volumes/hp-1650-
Data_Share1/SAR_CSL/S1/ARG_DOMU_LAGUNA_A_18/NoCrop/S1B_18_20211210_A.csl Dummy
_Check_ALL_S1_SizeAndCoord_InDir.sh $PATH_1650/SAR_CSL/S1/ARG_DOMU_LAGUNA_A_18/NoCrop 14 -71.1264 -37.3038 -69.1902 -36.8461 -71.5447 -36.0797 -69.6394 -
35.6292

# Desc ; bursts size and coordinates are obtained by running e.g.: _Check_S1_SizeAndCoord.sh /Volumes/hp-1650-
Data_Share1/SAR_CSL/S1/ARG_DOMU_LAGUNA_D_83/NoCrop/S1B_83_20211109_D.csl Dummy
# Beware D83 with S1B after Jan 2020 are shorter on the Western side, hence check first with large coordinate, then check the images in __TMP_QUARANTINE with
smaller coordinates.
#        If OK, put them back in NoCrop dir. If not, keep them in original __TMP_QUARANTINE

# consistent with S1B before Jan 2020
_Check_ALL_S1_SizeAndCoord_InDir.sh $PATH_1650/SAR_CSL/S1/ARG_DOMU_LAGUNA_D_83/NoCrop 14 -69.0318 -36.1361 -70.9962 -35.6524 -69.4497 -37.3619 -71.4464 -
36.8704
# consistent with S1B after Jan 2020
_Check_ALL_S1_SizeAndCoord_InDir.sh $PATH_1650/SAR_CSL/S1/ARG_DOMU_LAGUNA_D_83/NoCrop/__TMP_QUARANTINE 14 -68.9461 -36.1361 -70.9962 -35.6524 -69.3630 -37.3619
-71.4464 -36.8704
            mv $PATH_1650/SAR_CSL/S1/ARG_DOMU_LAGUNA_D_83/NoCrop/__TMP_QUARANTINE/*.csl $PATH_1650/SAR_CSL/S1/ARG_DOMU_LAGUNA_D_83/NoCrop/ 2>/dev/null
            mv $PATH_1650/SAR_CSL/S1/ARG_DOMU_LAGUNA_D_83/NoCrop/__TMP_QUARANTINE/__TMP_QUARANTINE/*.csl
$PATH_1650/SAR_CSL/S1/ARG_DOMU_LAGUNA_D_83/NoCrop/__TMP_QUARANTINE/ 2>/dev/null
            mv $PATH_1650/SAR_CSL/S1/ARG_DOMU_LAGUNA_D_83/NoCrop/__TMP_QUARANTINE/*.txt $PATH_1650/SAR_CSL/S1/ARG_DOMU_LAGUNA_D_83/NoCrop 2>/dev/null
            rm -R $PATH_1650/SAR_CSL/S1/ARG_DOMU_LAGUNA_D_83/NoCrop/__TMP_QUARANTINE/__TMP_QUARANTINE 2>/dev/null


# Coregister all images on the Global Primary (Super Master)
############################################################
# in Ascending mode
$PATH_SCRIPTS/SCRIPTS_MT/SuperMasterCoreg.sh $PATH_1650/Param_files/S1/ARG_DOMU_LAGUNA_A_18/LaunchMTparam_S1_Arg_Domu_Laguna_A_18_Zoom1_ML4_MassProc_Coreg.txt
&
# in Descending mode
$PATH_SCRIPTS/SCRIPTS_MT/SuperMasterCoreg.sh $PATH_1650/Param_file/S1/ARG_DOMU_LAGUNA_D_83/LaunchMTparam_S1_Arg_Domu_Laguna_D_83_Zoom1_ML4_MassProc_Coreg.txt &

# Search for pairs
##################
# Link all images to corresponding set dir
$PATH_SCRIPTS/SCRIPTS_MT/lns_All_Img.sh $PATH_1650/SAR_CSL/S1/ARG_DOMU_LAGUNA_A_18/NoCrop $PATH_1650/SAR_SM/MSBAS/ARGENTINE/set1 S1 > /dev/null 2>&1  &
$PATH_SCRIPTS/SCRIPTS_MT/lns_All_Img.sh $PATH_1650/SAR_CSL/S1/ARG_DOMU_LAGUNA_D_83/NoCrop $PATH_1650/SAR_SM/MSBAS/ARGENTINE/set2 S1 > /dev/null 2>&1 &
wait

# Compute pairs
# Compute pairs only if new data is identified
if [ ! -s ${NEWASCPATH}/_No_New_Data_Today.txt ] ; then
            echo "n" | Prepa_MSBAS.sh $PATH_1650/SAR_SM/MSBAS/ARGENTINE/set1 ${BP} 450 20180512 > /dev/null 2>&1  &
fi
if [ ! -s ${NEWDESCPATH}/_No_New_Data_Today.txt ] ; then
            echo "n" | Prepa_MSBAS.sh $PATH_1650/SAR_SM/MSBAS/ARGENTINE/set2 ${BP} 450 20180222 > /dev/null 2>&1  &
fi
wait

# Plot baseline plot with both modes
if [ ! -s ${NEWASCPATH}/_No_New_Data_Today.txt ] || [ ! -s ${NEWDESCPATH}/_No_New_Data_Today.txt ] ; then
                        if [ `baselinePlot | wc -l` -eq 0 ]
                                    then
                                                # use AMSTer Engine before May 2022

                                                mkdir -p $PATH_1650/SAR_SM/MSBAS/ARGENTINE/BaselinePlots_S1_set_1_2
                                                cd $PATH_1650/SAR_SM/MSBAS/ARGENTINE/BaselinePlots_S1_set_1_2

                                                echo "$PATH_1650/SAR_SM/MSBAS/ARGENTINE/set1" > ModeList.txt
                                                echo "/$PATH_1650/SAR_SM/MSBAS/ARGENTINE/set2" >> ModeList.txt

                                                $PATH_SCRIPTS/SCRIPTS_MT/plot_Multi_span.sh ModeList.txt 0 ${BP} 0 450
$PATH_SCRIPTS/SCRIPTS_MT/TemplatesForPlots/ColorTable_AD.txt
                                    else
                                                # use AMSTer Engine > May 2022
                                                mkdir -p $PATH_1650/SAR_SM/MSBAS/ARGENTINE/BaselinePlots_set1_set2
                                                cd $PATH_1650/SAR_SM/MSBAS/ARGENTINE/BaselinePlots_set1_set2

                                                echo "$PATH_1650/SAR_SM/MSBAS/ARGENTINE/set1" > ModeList.txt
                                                echo "/$PATH_1650/SAR_SM/MSBAS/ARGENTINE/set2" >> ModeList.txt

                                                plot_Multi_BaselinePlot.sh $PATH_1650/SAR_SM/MSBAS/ARGENTINE/BaselinePlots_set1_set2/ModeList.txt

                        fi
fi

echo "Ending $0" >> $PATH_1650/SAR_CSL/S1/ARG_DOMU_LAGUNA/Last_Run_Cron_Step1.txt
date >> $PATH_1650/SAR_CSL/S1/ARG_DOMU_LAGUNA/Last_Run_Cron_Step1.txt
```

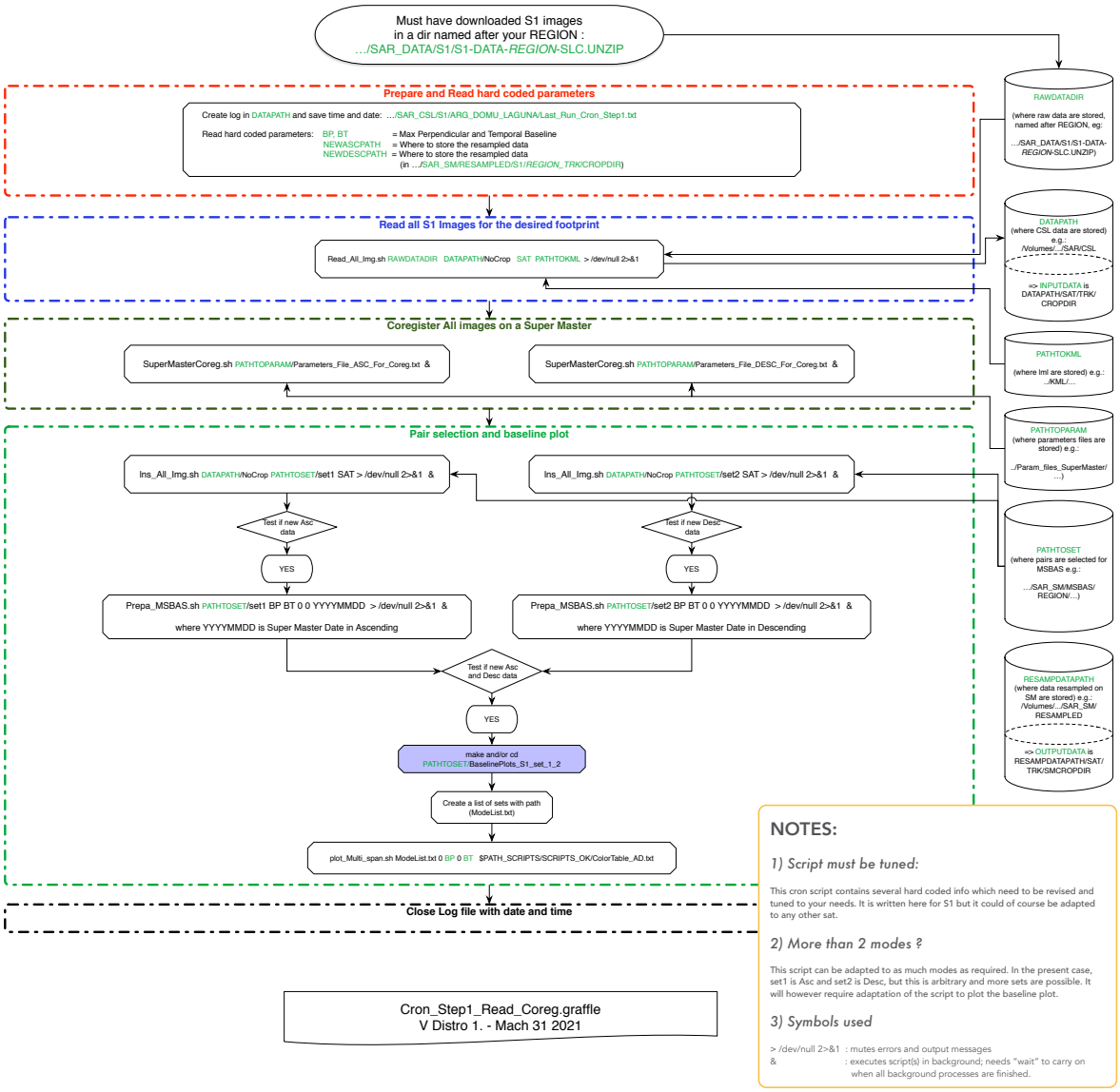


Figure 25: Flow chart of cron job step 1 (read and coregister images) – Beware, this corresponds to an old version, though it must give enough hints to understand the flow.

A.6.3) *Domuyo_S1_Step2_MassProc.sh*

```
#!/bin/bash
# Script to run in cronjob for processing Domuyo images:
# Runs a mass processing after having checked that no other process is using the same param file (on this computer).
#
# NOTE: usualy by running the reading and coregistration at 1 am, it is finished around 1am30
#       hence this script should be safely launched around 2 am for instance.
#       Nevertheless because VVP_S1_Step1_Read_SMCoreg_Pairs.sh uses RadAll_Img.sh, which also move updated prelim orbit images at all levels in _CLN dir,
#       and coregister images on Global Primary (super masters), one check that it is not running anymore before starting.
#
# New in Distro V 2.0 20230830:     - Rename SCRIPTS_OK directory as SCRIPTS_MT
#                                   - Replace CIS by MT in names
#                                   - Renamed FUNCTIONS_FOR_MT.sh
# New in Distro V 3.0 20231030:     - Rename MasTer Toolbox as AMSTer Software
#                                   - rename Master and Slave as Primary and Secondary (though not possible in some variables and files)
#
# AMSTer: SAR & InSAR Automated Mass processing Software for Multidimensional Time series
# NdO (c) 2016/03/25 - could make better... when time.
# ------------------------------------------------------------------------------------------

source $HOME/.bashrc

echo "Starting $0"
cd

BP=20

# some files
############
TABLEASC=$PATH_1650/SAR_SM/MSBAS/ARGENTINE/set1/table_0_${BP}_0_450.txt
TABLEDESC=$PATH_1650/SAR_SM/MSBAS/ARGENTINE/set2/table_0_${BP}_0_450.txt

PARAMPROCESSASC=$PATH_1650/Param_files/S1/ARG_DOMU_LAGUNA_A_18/LaunchMTparam_S1_Arg_Domu_Laguna_A_18_Zoom1_ML4_MassProc_MaskCohWater.txt
PARAMPROCESSDESC=$PATH_1650/Param_files/S1/ARG_DOMU_LAGUNA_D_83/LaunchMTparam_S1_Arg_Domu_Laguna_D_83_Zoom1_ML4_MassProc_Snaphu_WaterCohMask.txt

PARAMASCNAME=`basename ${PARAMPROCESSASC}`
PARAMDESCNAME=`basename ${PARAMPROCESSDESC}`

TODAY=`date`

## first restric pair table to last data
#RemovePairsFromFlist_WithImagesBefore.sh $PATH_1650/SAR_SM/MSBAS/ARGENTINE/set1/table_0_20_0_450.txt 20190425
#RemovePairsFromFlist_WithImagesBefore.sh $PATH_1650/SAR_SM/MSBAS/ARGENTINE/set2/table_0_20_0_450.txt 20190430
#TABLEASC=$PATH_1650/SAR_SM/MSBAS/ARGENTINE/set1/table_0_20_0_450.txt_Below20190425_NoBaselines_${TODAY}.txt
#TABLEDESC=$PATH_1650/SAR_SM/MSBAS/ARGENTINE/set2/table_0_20_0_450.txt_Below20190430_NoBaselines_${TODAY}.txt


# Check that Domuyo_S1_Step1_Read_SMCoreg_Pairs.sh is finished
#CHECKREAD=`ps -eaf | ${PATHGNU}/grep Domuyo_S1_Step1_Read_SMCoreg_Pairs.sh | ${PATHGNU}/grep -v "grep " | wc -l`
# below will be 0 if no run and 2 if script is running (3 if two runs are in preogress etc...)
CHECKREAD=`ps -eaf | ${PATHGNU}/grep Domuyo_S1_Step1_Read_SMCoreg_Pairs.sh | ${PATHGNU}/grep -v "grep " | ${PATHGNU}/grep -v "dev/null" | wc -l`

if [ ${CHECKREAD} -eq 0 ]
            then
                        # OK, no more Domuyo_S1_Step1_Read_SMCoreg_Pairs.sh is running:
                        # Check that no other Global Primary (SuperMaster) automatic Ascending and Desc mass processing uses the LaunchMTparam_.txt yet
                        CHECKASC=`ps -eaf | ${PATHGNU}/grep SuperMaster_MassProc.sh | ${PATHGNU}/grep -v "grep "  | ${PATHGNU}/grep ${PARAMASCNAME} | wc -l`
                        CHECKDESC=`ps -eaf | ${PATHGNU}/grep SuperMaster_MassProc.sh | ${PATHGNU}/grep -v "grep " | ${PATHGNU}/grep ${PARAMDESCNAME} | wc -l`
                        if [ ${CHECKASC} -lt 1 ]
                                    then
                                                # No process running yet
                                                echo              "Asc              run              on              ${TODAY}"                              >>
$PATH_3601/SAR_MASSPROCESS/S1/ARG_DOMU_LAGUNA_A_18/SMNoCrop_SM_20180512_Zoom1_ML4/_Asc_last_MassRun.txt
                                                $PATH_SCRIPTS/SCRIPTS_MT/SuperMaster_MassProc.sh ${TABLEASC} ${PARAMPROCESSASC} > /dev/null 2>&1 &
                                    else
                                                echo   "Asc   attempt   aborted   on   ${TODAY}   because   other   Mass   Process   in   progress"      >>
$PATH_3601/SAR_MASSPROCESS/S1/ARG_DOMU_LAGUNA_A_18/SMNoCrop_SM_20180512_Zoom1_ML4/_Asc_last_aborted.txt
                        fi
                        # if riunning yet we will try egain tomorrow

                        if [ ${CHECKDESC} -lt 1 ]
                                    then
                                                # No process running yet
                                                echo              "Desc              run              on              ${TODAY}"                              >>
$PATH_3601/SAR_MASSPROCESS/S1/ARG_DOMU_LAGUNA_D_83/SMNoCrop_SM_20180222_Zoom1_ML4/_Desc_last_Mass.txt
                                                $PATH_SCRIPTS/SCRIPTS_MT/SuperMaster_MassProc.sh ${TABLEDESC} ${PARAMPROCESSDESC} > /dev/null 2>&1 &
                                    else
                                                echo   "Desc   attempt   aborted   on   ${TODAY}   because   other   Mass   Process   in   progress"      >>
$PATH_3601/SAR_MASSPROCESS/S1/ARG_DOMU_LAGUNA_D_83/SMNoCrop_SM_20180222_Zoom1_ML4/_Desc_last_aborted.txt
                        fi
            else
                        # VVP_S1_Step1_Read_SMCoreg_Pairs.sh is still running: abort and wait for tomorrow
                        echo  "Step2  aborted  on  ${TODAY}  because  DOMUYO_S1_Step1_Read_SMCoreg_Pairs.sh  is  still  running:  wait  for  tomorrow"    >>
$PATH_3601/SAR_MASSPROCESS/S1/ARG_DOMU_LAGUNA_A_18/SMNoCrop_SM_20180512_Zoom1_ML4/_aborted_because_Read_inProgress.txt
                        echo  "Step2  aborted  on  ${TODAY}  because  DOMUYO_S1_Step1_Read_SMCoreg_Pairs.sh  is  still  running:  wait  for  tomorrow"    >>
$PATH_3601/SAR_MASSPROCESS/S1/ARG_DOMU_LAGUNA_D_83/SMNoCrop_SM_20180222_Zoom1_ML4/_aborted_because_Read_inProgress.txt

                        exit 0
fi
```

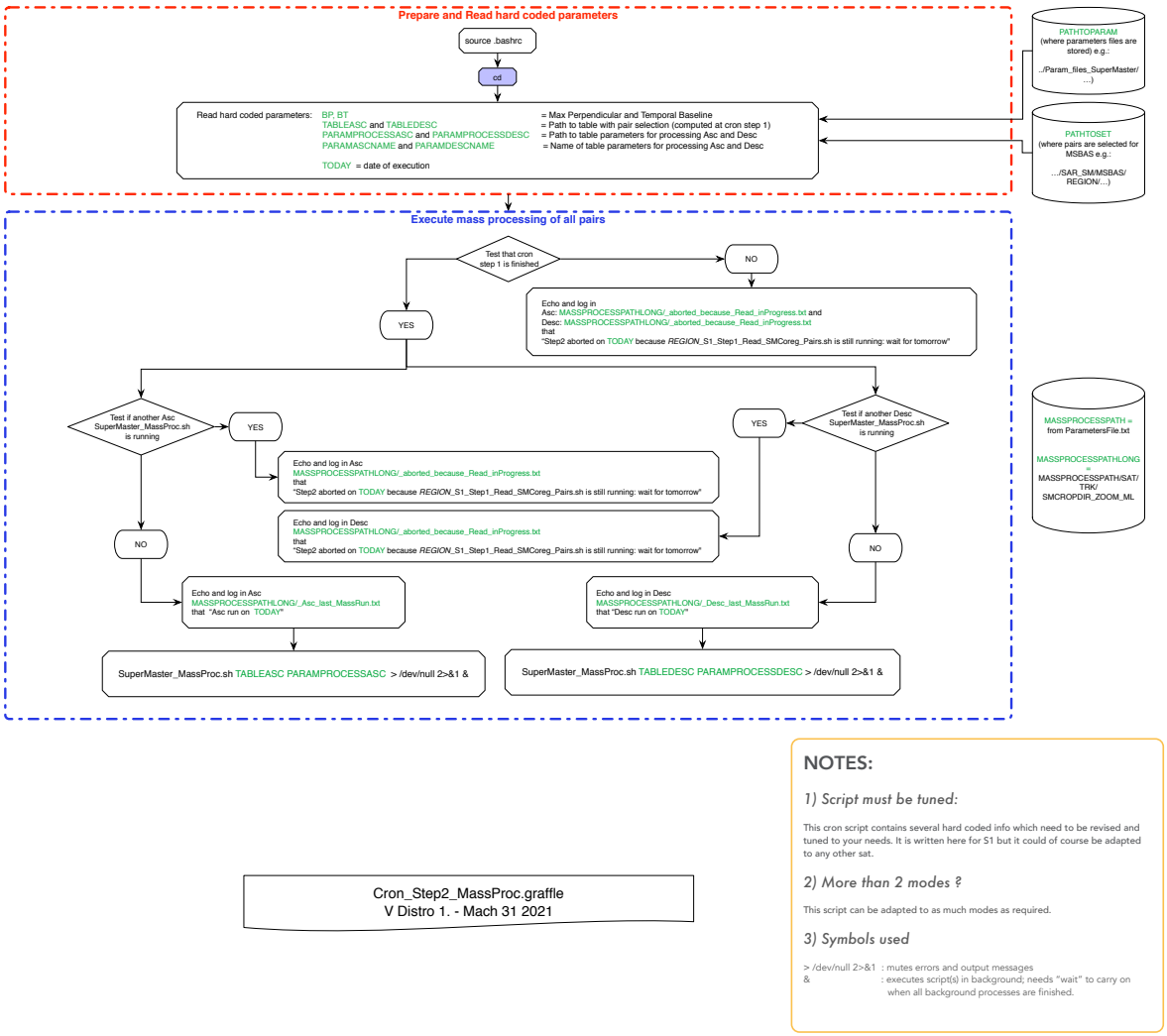


Figure 26: Flow chart of cron job step 2 (mass processing of all pairs) – Beware, this corresponds to an old version, though it must give enough hints to understand the flow.

A.6.4) *Domuyo_S1_Step3_MSBAS.sh*

```
#!/bin/bash
# Script intends to run in a cronjob an automatic systematic (re)processing of msbas time
# series when new images were made available. If orbits were updated, corresponding products
# will be taken into account at the time of processing with new images.
#
# It will prepare and run MSBAS only if no other mass process is in progress.
# It also plots several time series and double differences based on provided list of points.
#
# Optional : perform a selection of pairs based on a mean coh computed on a provided footprint.
#                                This might be useful for regions known to be affected by strong seasonal decorrelation.
#                                For instance, ensuring a mean coh of at least 0.235 on the Laguna_Maule area (Chile)
#                                ensured a proper estimation of the deformation. Not performing that selection based
#                                on the coh underestimated the defo up to 60%.
#
# NOTE: - MSBAS Calibration is disabled because deformation maps are detrended at processing.
#
# Parameters: - none
#
# Hardcoded: - a lot... se below paragraph named HARD CODED but also adapt below depending on the number of modes
#                                 - suppose everywhere that modes are DefoInterpolx2Detrend
#
# Dependencies:         -
#
# New in Distro V 2.0:  - based on beta version used for VVP, Domuyo, Lux and PF processing at ECGS
# New in Distro V 2.1:  - updated to use new PlotTS_all_comp.sh that computes as well the location and explanation tags etc...
# New in Distro V 2.2:  - correct syntax bug in testing EXCLUDEi
# New in Distro V 2.3:  - rm jpg images before moving to __Combi
# New in Distro V 2.4:  - change all _combi as _Combi for uniformisation
# New in Distro V 2.5:  - search for latest deg file using find instead of ls, which crashes when too many files
# New in Distro V 2.6:  - correct bug in searching for latest deg file
# New in Distro V 2.7: - replace if -s as -f -s && -f to be compatible with mac os if
# New in Distro V 3.0 20230830:     - Rename SCRIPTS_OK directory as SCRIPTS_MT
#                                   - Replace CIS by MT in names
#                                   - Renamed FUNCTIONS_FOR_MT.sh
# New in Distro V 4.0 20231030:     - Rename MasTer Toolbox as AMSTer Software
#                                   - rename Master and Slave as Primary and Secondary (though not possible in some variables and files)
#
# AMSTer: SAR & InSAR Automated Mass processing Software for Multidimensional Time series
# NdO (c) 2016/03/07 - could make better with more functions... when time.
# ------------------------------------------------------------------------------------------
PRG=`basename "$0"`
VER="Distro V4.0 AMSTer script utilities"
AUT="Nicolas d'Oreye, (c)2016-2019, Last modified on Aug 30, 2023"
echo " "
echo "${PRG} ${VER}, ${AUT}"
echo " "

source $HOME/.bashrc

cd

TODAY=`date`

# vvvvvvvvv Hard coded lines vvvvvvvvvvvvvv
            # some parameters
            #################
                        # Max baselines (used for all the mode in present case but you can change)
                        BP=20                                   # max perpendicular baseline
                        BT=450                                  # max temporal baseline

                        LABEL=Domuyo            # Label for file naming (used for naming zz_ dirs with results and figs etc)

                        #R_FLAG
                        # Order
                        ORDER=3
                        # Lambda
                        LAMBDA=0.04

            # some files and PATH for each mode
            ###################################
                        # Path to Pair Dirs and Geocoded files to use (need one for each mode)
                        S1ASC=${PATH_3601}/SAR_MASSPROCESS/S1/ARG_DOMU_LAGUNA_A_18/SMNoCrop_SM_20180512_Zoom1_ML4
                        S1DESC=${PATH_3601}/SAR_MASSPROCESS/S1/ARG_DOMU_LAGUNA_D_83/SMNoCrop_SM_20180222_Zoom1_ML4

                        # Path to dir where list of compatible pairs files are computed (need one for each mode)
                        SET1=${PATH_1650}/SAR_SM/MSBAS/ARGENTINE/set1
                        SET2=${PATH_1650}/SAR_SM/MSBAS/ARGENTINE/set2

                        # Path to LaunchParameters.txt files for each mode (need one for each mode)
                        LAUNCHPARAMASC=LaunchMTparam_S1_Arg_Domu_Laguna_A_18_Zoom1_ML4_MassProc_MaskCohWater.txt
                        LAUNCHPARAMDESC=LaunchMTparam_S1_Arg_Domu_Laguna_D_83_Zoom1_ML4_MassProc_Snaphu_WaterCohMask.txt

            # Events tables
            ###############
                        #EVENTS=${PATH_1650}/EVENTS_TABLES/${LABEL}

            # Path to dir where MSBAS will be computed
            ##########################################
                        MSBASDIR=${PATH_3602}/MSBAS/_${LABEL}_S1_Auto_${BP}m_${BT}days

            # Coherence restriction
            #######################
                        IFCOH="YES"             # YES or NO

                        if [ ${IFCOH} == "YES" ]
                                    then

                                                # Path to kml zone used to check coherence
                                                KMLCOH=${PATH_1650}/kml/ARGENTINA/Laguna_Maule.kml

                                                # Coherence restriction threshold (to be compared to mean coh computed on KMLCOH)
                                                COHRESTRICT=0.235

                                                # Exclude pairs from modes: If pairs are incidentally above Coh Threshold,
                                                # they can be excluded if they are stored as DATE_DATE in a list named
                                                # ${MSBASDIR}/DefoInterpolx2Detrendi/_EXCLUDE_PAIRS_ALTHOUGH_CRITERIA_OK.txt
                                                # and parameter below set to YES
                                                EXCLUDE1="NO"           # YES or NO
                                                EXCLUDE2="NO"           # YES or NO

                                                if [ ! -s ${KMLCOH} ] ; then echo "Missing kml for coherence estimation. Please Check" ; exit ; fi
                                    else
                                                EXCLUDE1="NO"           # always NO of course
                                                EXCLUDE2="NO"           # always NO of course

                        fi
```

```
            # Path to list of points for plotting time series
            ################################################
                        # List of SINGLE points for plotting time series with error bars
                        TIMESERIESPTSDESCR=${PATH_SCRIPTS}/SCRIPTS_MT/_cron_scripts/Points_TS_${LABEL}.txt

                        # List of PAIRS of points for plotting double difference (i.e. without error bar) in EW and UD, ASC and Desc...
                        #           Note: if pixels are coherent in all modes, these can be the same list
                        DOUBLEDIFFPAIRSEWUD=${PATH_SCRIPTS}/SCRIPTS_MT/_cron_scripts/List_DoubleDiff_EW_UD_${LABEL}.txt
                        DOUBLEDIFFPAIRSASC=${PATH_SCRIPTS}/SCRIPTS_MT/_cron_scripts/List_DoubleDiff_EW_UD_${LABEL}.txt
                        DOUBLEDIFFPAIRSDESC=${PATH_SCRIPTS}/SCRIPTS_MT/_cron_scripts/List_DoubleDiff_EW_UD_${LABEL}.txt

#                       # Path to dir where jpg figs of location of each pair of points are stored
#                       PATHLOCA=${PATH_3602}/MSBAS/zz_insets/${LABEL}

            # Name of previous cron jobs for the automatic processing of that target (used to check that no other process is runing)
            ########################################################################
            CRONJOB2=Domuyo_S1_Step2_MassProc.sh

# ^^^^^^^^^^ Hard coded lines ^^^^^^^^^^^^

# Prepare directories
####################
            mkdir -p ${MSBASDIR}

            mkdir -p ${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}
            mkdir -p ${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}/_Time_series
            mkdir -p ${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}/__Combi/

            mkdir -p ${MSBASDIR}/zz_LOS_TS_Asc_Auto_${ORDER}_${LAMBDA}_${LABEL}
            mkdir -p ${MSBASDIR}/zz_LOS_TS_Asc_Auto_${ORDER}_${LAMBDA}_${LABEL}/__Combi/
            mkdir -p ${MSBASDIR}/zz_LOS_TS_Desc_Auto_${ORDER}_${LAMBDA}_${LABEL}
            mkdir -p ${MSBASDIR}/zz_LOS_TS_Desc_Auto_${ORDER}_${LAMBDA}_${LABEL}/__Combi/

            # in Coh threshold restriction
            if [ ${IFCOH} == "YES" ] ; then
                        mkdir -p ${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh/
                        mkdir -p ${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh/__Combi/
                        mkdir -p ${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh/_Time_series
            fi

            cd ${MSBASDIR}

# prepare points lists
######################
            TIMESERIESPTNAME=$(basename "${TIMESERIESPTSDESCR}")
            cp -f ${TIMESERIESPTSDESCR}  ${MSBASDIR}/${TIMESERIESPTNAME}
            TIMESERIESPTSDESCR=${MSBASDIR}/${TIMESERIESPTNAME}
            #cp -f ${TIMESERIESPTSDESCR} ${MSBASDIR}/${TIMESERIESPTNAME}.tmp  #.tmp is now as the original; the original will be cut from first line (title)
            # Remove header and naming in 1st col from Pts list
            ${PATHGNU}/gsed '1d' "${TIMESERIESPTSDESCR}" > ${MSBASDIR}/Cln_${TIMESERIESPTNAME}
            ${PATHGNU}/gsed  -i -r 's/(\s+)?\S+//1' ${MSBASDIR}/Cln_${TIMESERIESPTNAME}
            # remove 3rd col
            #${PATHGNU}/gsed  -i -r 's/(\s+)?\S+//3' /Users/doris/PROCESS/SCRIPTS_MT/_cron_scripts/Cln_${LABEL}.txt
            TIMESERIESPTS=${MSBASDIR}/Cln_${TIMESERIESPTNAME}

# functions
###########
            function PlotAll()
                        {
                        unset X1 Y1 X2 Y2 DESCRIPTION
                        local X1=$1
                        local Y1=$2
                        local X2=$3
                        local Y2=$4
                        local DESCRIPTION=$5

                        if [ "${EVENTS}" == "" ]
                                    then
                                                ${PATH_SCRIPTS}/SCRIPTS_MT/PlotTS_all_comp.sh _Auto_${ORDER}_${LAMBDA}_${LABEL} ${X1} ${Y1} ${X2} ${Y2} -f -r -t
-g   # remove -f if does not want the linear fit etc..
                                    else
                                                ${PATH_SCRIPTS}/SCRIPTS_MT/PlotTS_all_comp.sh _Auto_${ORDER}_${LAMBDA}_${LABEL} ${X1} ${Y1} ${X2} ${Y2} -f -r -t
-g -events=${EVENTS}  # remove -f if does not want the linear fit etc..
                        fi
                        mv                                                                  ${MSBASDIR}/timeLines_${X1}_${Y1}_Auto_${ORDER}_${LAMBDA}_${LABEL}.eps
${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}/${DESCRIPTION}_timeLines_${X1}_${Y1}_Auto_${ORDER}_${LAMBDA}_${LABEL}.eps
                        mv                                                                  ${MSBASDIR}/timeLines_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}.eps
${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}/${DESCRIPTION}_timeLines_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}.eps

                        mv                                                              ${MSBASDIR}/timeLines_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}.eps
${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}/${DESCRIPTION}_timeLines_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}.eps

#                       # add map tag in fig
#                       convert                     -density                    300                     -rotate                     90                      -trim
${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}/${DESCRIPTION}_timeLines_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}.eps
${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}/${DESCRIPTION}_timeLines_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}.jpg
#                       convert
${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}/${DESCRIPTION}_timeLines_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}.jpg
${PATHLOCA}/Loca_${X1}_${Y1}_${X2}_${Y2}.jpg          -gravity              northwest             -geometry             +250+150              -composite
${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}/${DESCRIPTION}_timeLines_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}_Combi.jpg
                        rm                                                                                                                                  -f
${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}/${DESCRIPTION}_timeLines_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}_Combi.jpg
                        mv                                                      ${MSBASDIR}/timeLines_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}_Combi.jpg
${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}/${DESCRIPTION}_timeLines_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}_Combi.jpg

                        mv                                                  ${MSBASDIR}/timeLine_UD_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}.txt
${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}/${DESCRIPTION}_timeLines_UD_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}.txt
                        mv                                                  ${MSBASDIR}/timeLine_EW_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}.txt
${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}/${DESCRIPTION}_timeLines_EW_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}.txt

#                       rm                                                                                                                                  -f
${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}/${DESCRIPTION}_timeLines_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}.jpg
                        }

            function PlotAllNoCoh()
                        {
                        unset X1 Y1 X2 Y2 DESCRIPTION
                        local X1=$1
                        local Y1=$2
                        local X2=$3
                        local Y2=$4
                        local DESCRIPTION=$5


                        if [ "${EVENTS}" == "" ]
                                    then
                                                ${PATH_SCRIPTS}/SCRIPTS_MT/PlotTS_all_comp.sh  _Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh  ${X1}  ${Y1}  ${X2}
${Y2} -f -r -t -g   # remove -f if does not want the linear fit etc..
                                    else
                                                ${PATH_SCRIPTS}/SCRIPTS_MT/PlotTS_all_comp.sh  _Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh  ${X1}  ${Y1}  ${X2}
${Y2} -f -r -t -g -events=${EVENTS}  # remove -f if does not want the linear fit etc..
                        fi
#                       rm plotTS*.gnu timeLines_*.png

                        if      [      -f      "${MSBASDIR}/timeLines_${X1}_${Y1}_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh.eps"      ]      &&      [      -s
"${MSBASDIR}/timeLines_${X1}_${Y1}_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh.eps" ] ; then
```

```
                                mv                                                  ${MSBASDIR}/timeLines_${X1}_${Y1}_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh.eps
${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh/${DESCRIPTION}_timeLines_${X1}_${Y1}_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh.eps
                        fi
                        if      [       -f      "${MSBASDIR}/timeLines_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh.eps"      ]       &&      [       -s
"${MSBASDIR}/timeLines_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh.eps" ] ; then
                                mv                                                  ${MSBASDIR}/timeLines_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh.eps
${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh/${DESCRIPTION}_timeLines_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh.eps
                        fi
                        if      [       -f      "${MSBASDIR}/timeLines_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh.eps"      ]       &&      [       -s
"${MSBASDIR}/timeLines_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh.eps" ] ; then
                                mv                                  ${MSBASDIR}/timeLines_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh.eps
${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh/${DESCRIPTION}_timeLines_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThre
sh.eps

#                               # add map tag in fig
#                               convert                         -density                300                 -rotate                 90                  -trim
${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh/${DESCRIPTION}_timeLines_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThre
sh.eps
${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh/${DESCRIPTION}_timeLines_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThre
sh.jpg
#                               # get location from dir with coh threshold (where it was added manually)
#                               convert
${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh/${DESCRIPTION}_timeLines_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThre
sh.jpg          ${PATHLOCA}/Loca_${X1}_${Y1}_${X2}_${Y2}.jpg          -gravity          northwest          -geometry          +250+150          -composite
${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh/${DESCRIPTION}_timeLines_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}_Combi_NoC
ohThresh.jpg
                                rm                                                                                                                          -f
${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh/${DESCRIPTION}_timeLines_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThre
sh_Combi.jpg
                                mv                          ${MSBASDIR}/timeLines_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh_Combi.jpg
${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh/${DESCRIPTION}_timeLines_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThre
sh_Combi.jpg

                                mv                              ${MSBASDIR}/timeLine_UD_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh.txt
${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh/${DESCRIPTION}_timeLines_UD_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohT
hresh.txt
                                mv                              ${MSBASDIR}/timeLine_EW_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh.txt
${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh/${DESCRIPTION}_timeLines_EW_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohT
hresh.txt
                        fi

#                               rm                                                                                                                          -f
${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh/${DESCRIPTION}_timeLines_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThre
sh.jpg
                        }


            function PlotAllLOSasc()
                        {
                        unset X1 Y1 X2 Y2 DESCRIPTION
                        local X1=$1
                        local Y1=$2
                        local X2=$3
                        local Y2=$4
                        local DESCRIPTION=$5

                        cd ${MSBASDIR}/zz_LOS_Asc_Auto_${ORDER}_${LAMBDA}_${LABEL}/
                        mkdir -p ${MSBASDIR}/zz_LOS_TS_Asc_Auto_${ORDER}_${LAMBDA}_${LABEL}/_Time_series

                        if [ "${EVENTS}" == "" ]
                                then
                                        ${PATH_SCRIPTS}/SCRIPTS_MT/PlotTS.sh ${X1} ${Y1} ${X2} ${Y2} -f -r -t -g # remove -f if does not want the linear
fit
                                else
                                        ${PATH_SCRIPTS}/SCRIPTS_MT/PlotTS.sh ${X1} ${Y1} ${X2} ${Y2} -f -r -t -g -events=${EVENTS}  # remove -f if does
not want the linear fit etc..
                        fi
#                       rm plotTS*.gnu timeLine*.png

                        mv                                                                                                                  timeLine${X1}_${Y1}.eps
${MSBASDIR}/zz_LOS_TS_Asc_Auto_${ORDER}_${LAMBDA}_${LABEL}/${DESCRIPTION}_timeLine_${X1}_${Y1}_Auto_${ORDER}_${LAMBDA}_${LABEL}.eps
                        mv                                                                                                                  timeLine${X2}_${Y2}.eps
${MSBASDIR}/zz_LOS_TS_Asc_Auto_${ORDER}_${LAMBDA}_${LABEL}/${DESCRIPTION}_timeLine_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}.eps
                        mv                                                                                                          timeLine${X1}_${Y1}_${X2}_${Y2}.eps
${MSBASDIR}/zz_LOS_TS_Asc_Auto_${ORDER}_${LAMBDA}_${LABEL}/${DESCRIPTION}_timeLine_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}.eps

                        mv timeLine${X1}_${Y1}.txt ${MSBASDIR}/zz_LOS_TS_Asc_Auto_${ORDER}_${LAMBDA}_${LABEL}/_Time_series/
                        mv timeLine${X2}_${Y2}.txt ${MSBASDIR}/zz_LOS_TS_Asc_Auto_${ORDER}_${LAMBDA}_${LABEL}/_Time_series/
                        mv timeLine${X1}_${Y1}_${X2}_${Y2}.txt ${MSBASDIR}/zz_LOS_TS_Asc_Auto_${ORDER}_${LAMBDA}_${LABEL}/_Time_series/

#                       # add map tag in fig
#                       convert                         -density                300                 -rotate                 90                  -trim
${MSBASDIR}/zz_LOS_TS_Asc_Auto_${ORDER}_${LAMBDA}_${LABEL}/${DESCRIPTION}_timeLine_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}.eps
${MSBASDIR}/zz_LOS_TS_Asc_Auto_${ORDER}_${LAMBDA}_${LABEL}/${DESCRIPTION}_timeLine_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}.jpg
#                       convert
${MSBASDIR}/zz_LOS_TS_Asc_Auto_${ORDER}_${LAMBDA}_${LABEL}/${DESCRIPTION}_timeLine_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}.jpg
${PATHLOCA}/Loca_${X1}_${Y1}_${X2}_${Y2}.jpg          -gravity          northwest          -geometry          +250+150          -composite
${MSBASDIR}/zz_LOS_TS_Asc_Auto_${ORDER}_${LAMBDA}_${LABEL}/${DESCRIPTION}_timeLine_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}_Combi_Asc.jpg
                        rm                                                                                                                          -f
${MSBASDIR}/zz_LOS_TS_Asc_Auto_${ORDER}_${LAMBDA}_${LABEL}/${DESCRIPTION}_timeLine_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}_Combi_Asc.jpg
                        mv                                  ${MSBASDIR}/zz_LOS_Asc_Auto_${ORDER}_${LAMBDA}_${LABEL}/timeLine${X1}_${Y1}_${X2}_${Y2}_Combi.jpg
${MSBASDIR}/zz_LOS_TS_Asc_Auto_${ORDER}_${LAMBDA}_${LABEL}/${DESCRIPTION}_timeLine_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}_Combi_Asc.jpg

#                       rm                                                                                                                          -f
${MSBASDIR}/zz_LOS_TS_Asc_Auto_${ORDER}_${LAMBDA}_${LABEL}/${DESCRIPTION}_timeLine_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}.jpg
                        }

            function PlotAllLOSdesc()
                        {
                        unset X1 Y1 X2 Y2 DESCRIPTION
                        local X1=$1
                        local Y1=$2
                        local X2=$3
                        local Y2=$4
                        local DESCRIPTION=$5

                        cd ${MSBASDIR}/zz_LOS_Desc_Auto_${ORDER}_${LAMBDA}_${LABEL}/
                        mkdir -p ${MSBASDIR}/zz_LOS_TS_Desc_Auto_${ORDER}_${LAMBDA}_${LABEL}/_Time_series

                        if [ "${EVENTS}" == "" ]
                                then
                                        ${PATH_SCRIPTS}/SCRIPTS_MT/PlotTS.sh ${X1} ${Y1} ${X2} ${Y2} -f -r -t -g # remove -f if does not want the linear
fit
                                else
                                        ${PATH_SCRIPTS}/SCRIPTS_MT/PlotTS.sh ${X1} ${Y1} ${X2} ${Y2} -f -r -t -g -events=${EVENTS}  # remove -f if does
not want the linear fit etc..
                        fi

#                       rm plotTS*.gnu timeLine*.png

                        mv                                                                                                                  timeLine${X1}_${Y1}.eps
${MSBASDIR}/zz_LOS_TS_Desc_Auto_${ORDER}_${LAMBDA}_${LABEL}/${DESCRIPTION}_timeLine_${X1}_${Y1}_Auto_${ORDER}_${LAMBDA}_${LABEL}.eps
                        mv                                                                                                                  timeLine${X2}_${Y2}.eps
${MSBASDIR}/zz_LOS_TS_Desc_Auto_${ORDER}_${LAMBDA}_${LABEL}/${DESCRIPTION}_timeLine_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}.eps
                        mv                                                                                                          timeLine${X1}_${Y1}_${X2}_${Y2}.eps
${MSBASDIR}/zz_LOS_TS_Desc_Auto_${ORDER}_${LAMBDA}_${LABEL}/${DESCRIPTION}_timeLine_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}.eps
```

```
                        mv timeLine${X1}_${Y1}.txt ${MSBASDIR}/zz_LOS_TS_Desc_Auto_${ORDER}_${LAMBDA}_${LABEL}/_Time_series/
                        mv timeLine${X2}_${Y2}.txt ${MSBASDIR}/zz_LOS_TS_Desc_Auto_${ORDER}_${LAMBDA}_${LABEL}/_Time_series/
                        mv timeLine${X1}_${Y1}_${X2}_${Y2}.txt ${MSBASDIR}/zz_LOS_TS_Desc_Auto_${ORDER}_${LAMBDA}_${LABEL}/_Time_series/

#                       # add map tag in fig
#                       convert                 -density                300                 -rotate                 90                  -trim
${MSBASDIR}/zz_LOS_TS_Desc_Auto_${ORDER}_${LAMBDA}_${LABEL}/${DESCRIPTION}_timeLine_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}.eps
${MSBASDIR}/zz_LOS_TS_Desc_Auto_${ORDER}_${LAMBDA}_${LABEL}/${DESCRIPTION}_timeLine_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}.jpg
#                       convert
${MSBASDIR}/zz_LOS_TS_Desc_Auto_${ORDER}_${LAMBDA}_${LABEL}/${DESCRIPTION}_timeLine_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}.jpg
${PATHLOCA}/Loca_${X1}_${Y1}_${X2}_${Y2}.jpg            -gravity            northwest            -geometry            +250+150            -composite
${MSBASDIR}/zz_LOS_TS_Desc_Auto_${ORDER}_${LAMBDA}_${LABEL}/${DESCRIPTION}_timeLine_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}_Combi_Desc.jpg
                        rm                                                                                                                                                                                                  -f
${MSBASDIR}/zz_LOS_TS_Desc_Auto_${ORDER}_${LAMBDA}_${LABEL}/${DESCRIPTION}_timeLine_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}_Combi_Desc.jpg
                        mv                                      ${MSBASDIR}/zz_LOS_Desc_Auto_${ORDER}_${LAMBDA}_${LABEL}/timeLine${X1}_${Y1}_${X2}_${Y2}_Combi.jpg
${MSBASDIR}/zz_LOS_TS_Desc_Auto_${ORDER}_${LAMBDA}_${LABEL}/${DESCRIPTION}_timeLine_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}_Combi_Desc.jpg

                        #mv                                             ${MSBASDIR}/timeLine_UD_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}.txt
${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}/${DESCRIPTION}_timeLines_UD_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}.txt
                        #mv                                             ${MSBASDIR}/timeLine_EW_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}.txt
${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}/${DESCRIPTION}_timeLines_EW_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}.txt

#                       rm                                                                                                                                                                                                  -f
${MSBASDIR}/zz_LOS_TS_Desc_Auto_${ORDER}_${LAMBDA}_${LABEL}/${DESCRIPTION}_timeLine_${X1}_${Y1}_${X2}_${Y2}_Auto_${ORDER}_${LAMBDA}_${LABEL}.jpg
                        }


# Check that there is no other cron (Step 2 or 3) or manual SuperMaster_MassProc.sh running
################################################################################
            # Check that no other cron job step 3 (MSBAS) or manual SuperMaster_MassProc.sh is running
            CHECKMB=`ps -Af | ${PATHGNU}/grep ${PRG} | ${PATHGNU}/grep -v "grep " | ${PATHGNU}/grep -v "/dev/null" | wc -l`
                        #### For Debugging
                        # echo "ps -Af | ${PATHGNU}/grep ${PRG} | ${PATHGNU}/grep -v ${PATHGNU}/grep | ${PATHGNU}/grep -v /dev/null | wc -l" > CheckRun.txt
                        # echo ${CHECKMB} >> CheckRun.txt
                        # ps -Af | ${PATHGNU}/grep ${PRG} | ${PATHGNU}/grep -v "grep " | ${PATHGNU}/grep -v "/dev/null" >> CheckRun.txt

            if [ ${CHECKMB} -gt 3 ] ; then # use ${PATHGNU}/grep -v "grep "  instead of ${PATHGNU}/grep -v "grep ${PRG}" because depending on environment, it may
miss the second version
                                    REASON=" another ${PRG} is running"
                                    STOPRUN="YES"
                        else
                                    # Check that no other SuperMaster_MassProc.sh automatic Ascending and Desc mass processing uses the LaunchMTparam_.txt yet
                                    CHECKASC=`ps -eaf | ${PATHGNU}/grep SuperMaster_MassProc.sh | ${PATHGNU}/grep -v "grep "  | ${PATHGNU}/grep ${LAUNCHPARAMASC}
| ${PATHGNU}/grep -v "/dev/null" | wc -l`
                                    CHECKDESC=`ps -eaf | ${PATHGNU}/grep SuperMaster_MassProc.sh | ${PATHGNU}/grep -v "grep " | ${PATHGNU}/grep ${LAUNCHPARAMDESC}
| ${PATHGNU}/grep -v "/dev/null" | wc -l`
                                    # For unknown reason it counts 1 even when no process is running
                                    if [ ${CHECKASC} -ne 0 ] || [ ${CHECKDESC} -ne 0 ] ; then REASON="  SuperMaster_MassProc.sh in progress (probably manual)" ;
STOPRUN="YES" ; else STOPRUN="NO" ; fi
            fi

            # Check that no other cron job step 2 (SuperMaster_MassProc.sh) is running
            CHECKMP=`ps -eaf | ${PATHGNU}/grep ${CRONJOB2} | ${PATHGNU}/grep -v "grep " | ${PATHGNU}/grep -v "/dev/null" | wc -l`
            if [ ${CHECKMP} -ne 0 ] ; then REASON=" SuperMaster_MassProc.sh in progress (from ${CRONJOB2})" ; STOPRUN="YES" ; else STOPRUN="NO" ; fi

            if [ "${STOPRUN}" == "YES" ]
                        then
                                    echo "MSBAS attempt aborted on ${TODAY} because ${REASON}" >>  ${MSBASDIR}/_last_MSBAS_process.txt
                                    echo "MSBAS attempt aborted on ${TODAY} because ${REASON}"
                                    #mv -f  ${MSBASDIR}/${TIMESERIESPTSDESCR}.tmp  ${MSBASDIR}/${TIMESERIESPTSDESCR}
                                    exit
            fi

# Check defo maps in SAR_MASSPROCESS
####################################
# Remove possible duplicate geocoded products in SAR_MASSPROCESS/.../Geocoded/...
# i.e. remove in each MODE (but Ampl) possible products from same pair of dates but with different Bp, Ha etc.. that would results from
# reprocessing with updated orbits. If duplicated product detected, it keeps only the most recent product.

            cd ${S1ASC}
            Remove_Duplicate_Pairs_File_All_Modes_But_Ampl.sh &
            cd ${S1DESC}
            Remove_Duplicate_Pairs_File_All_Modes_But_Ampl.sh &
            wait

# Get date (in sec) of last available processed pairs in each MODE
####################################################################
            # get the name of last available processed pair in each MODE

    # ls crashes when too many files
            #LASTASC=`ls -lt ${S1ASC}/Geocoded/DefoInterpolx2Detrend | head -n 2 | tail -n 1 | sed 's/.* //'` # may be messing up if txt files are created for
any other purpose in the dir...
            #LASTDESC=`ls -lt ${S1DESC}/Geocoded/DefoInterpolx2Detrend | head -n 2 | tail -n 1 | sed 's/.* //'`
            LASTASC=`find ${S1ASC}/Geocoded/DefoInterpolx2Detrend/ -maxdepth 1 -type f -name "*deg" -printf "%T+ %p\n" | sort -r | head -1 | ${PATHGNU}/gawk
'{print $2}'`
            LASTDESC=`find ${S1DESC}/Geocoded/DefoInterpolx2Detrend/ -maxdepth 1 -type f -name "*deg" -printf "%T+ %p\n" | sort -r | head -1 | ${PATHGNU}/gawk
'{print $2}'`
            # get date in sec of last available processed pairs in each MODE
            LASTASCTIME=`stat -c %Y ${LASTASC}`
            LASTDESCTIME=`stat -c %Y ${LASTDESC}`

# Check if first run and if  appropriate, get time of last images in time series
################################################################################
            if [ -f "${MSBASDIR}/_Last_MassProcessed_Pairs_Time.txt" ] && [ -s "${MSBASDIR}/_Last_MassProcessed_Pairs_Time.txt" ]
                        then
                                    echo "Existing ${MSBASDIR}/_Last_MassProcessed_Pairs_Time.txt, hence not the first run"
                                    FIRSTRUN=NO
                                    FORMERLASTASCTIME=`head -1 ${MSBASDIR}/_Last_MassProcessed_Pairs_Time.txt`
                                    FORMERLASTDESCTIME=`head -2 ${MSBASDIR}/_Last_MassProcessed_Pairs_Time.txt | tail -1` # tail -1 ok also but this is ready for
case where more than 2 lines are present in _Last_MassProcessed_Pairs_Time.txt

                                    if [ ${FORMERLASTASCTIME} -eq ${LASTASCTIME} ] && [ ${FORMERLASTDESCTIME} -eq ${LASTDESCTIME} ]  # if no more recent file is
available since the last cron processing
                                                then
                                                            echo "MSBAS finished on ${TODAY} without new pairs to process"  >>  ${MSBASDIR}/_last_MSBAS_process.txt
                                                            echo "MSBAS finished on ${TODAY} without new pairs to process"
                                                            #mv -f  ${MSBASDIR}/${TIMESERIESPTSDESCR}.tmp  ${MSBASDIR}/${TIMESERIESPTSDESCR}
                                                            exit
                                    fi
                        else
                                    echo "No ${MSBASDIR}/_Last_MassProcessed_Pairs_Time.txt, hence first run"
                                    FIRSTRUN=YES
            fi

# Remove possible broken links in MSBAS/.../MODEi and clean corresponding files
################################################################################
# (clean if required MODEi.txt and Checked_For_CohThreshold_To_Be_Ignored_At_Next_Rebuild_msbas_Header.txt if any)
            if [ "${FIRSTRUN}" == "NO" ] ; then
                        echo "Remove Broken Links and Clean txt file in existing ${MSBASDIR}/DefoInterpolx2Detrend"
                        Remove_BrokenLinks_and_Clean_txt_file.sh ${MSBASDIR}/DefoInterpolx2Detrend1 &
                        Remove_BrokenLinks_and_Clean_txt_file.sh ${MSBASDIR}/DefoInterpolx2Detrend2 &
                        wait
                        echo "Possible broken links in former existing MODEi dir are cleaned"
                        echo ""

                        #Need also for the _Full ones (that is without coh threshold)
                        if [ ${IFCOH} == "YES" ] ; then
```

```
                                    Remove_BrokenLinks_and_Clean_txt_file.sh ${MSBASDIR}/DefoInterpolx2Detrend1_Full &
                                    Remove_BrokenLinks_and_Clean_txt_file.sh ${MSBASDIR}/DefoInterpolx2Detrend2_Full &
                                    wait
                                    echo "Possible broken links in former existing MODEi_Full dir are cleaned"
                                    echo ""
                        fi
            fi

# Check MSBAS/.../MODEi.txt file
################################
cd ${MSBASDIR}

# Remove possible lines with less that 4 columns
            if [ "${FIRSTRUN}" == "NO" ] ; then
                        mv DefoInterpolx2Detrend1.txt DefoInterpolx2Detrend1_all4col.txt
                        mv DefoInterpolx2Detrend2.txt DefoInterpolx2Detrend2_all4col.txt
                        ${PATHGNU}/gawk 'NF>=4' DefoInterpolx2Detrend1_all4col.txt > DefoInterpolx2Detrend1.txt
                        ${PATHGNU}/gawk 'NF>=4' DefoInterpolx2Detrend2_all4col.txt > DefoInterpolx2Detrend2.txt
                        rm -f DefoInterpolx2Detrend1_all4col.txt DefoInterpolx2Detrend2_all4col.txt
                        echo "All lines in former existing MODEi.txt have 4 columns"
                        echo ""

                        #Need also for the _Full ones (that is without coh threshold)
                        if [ ${IFCOH} == "YES" ] ; then
                                    mv                                                  ${MSBASDIR}/DefoInterpolx2Detrend1_Full/DefoInterpolx2Detrend1_Full.txt
${MSBASDIR}/DefoInterpolx2Detrend1_Full/DefoInterpolx2Detrend1_Full_all4col.txt
                                    mv                                                  ${MSBASDIR}/DefoInterpolx2Detrend2_Full/DefoInterpolx2Detrend2_Full.txt
${MSBASDIR}/DefoInterpolx2Detrend2_Full/DefoInterpolx2Detrend2_Full_all4col.txt
                                    ${PATHGNU}/gawk       'NF>=4'       ${MSBASDIR}/DefoInterpolx2Detrend1_Full/DefoInterpolx2Detrend1_Full_all4col.txt       >
${MSBASDIR}/DefoInterpolx2Detrend1_Full/DefoInterpolx2Detrend1_Full.txt
                                    ${PATHGNU}/gawk       'NF>=4'       ${MSBASDIR}/DefoInterpolx2Detrend2_Full/DefoInterpolx2Detrend2_Full_all4col.txt       >
${MSBASDIR}/DefoInterpolx2Detrend2_Full/DefoInterpolx2Detrend2_Full.txt
                                    rm                  -f                        ${MSBASDIR}/DefoInterpolx2Detrend1_Full/DefoInterpolx2Detrend1_Full_all4col.txt
${MSBASDIR}/DefoInterpolx2Detrend2_Full/DefoInterpolx2Detrend2_Full_all4col.txt
                                    echo "All lines in former existing MODEi_Full.txt have 4 columns"
                                    echo ""
                        fi

# Remove lines in MSBAS/MODEi.txt file associated to possible broken links or duplicated lines with same name though wrong BP (e.g. after S1 orb update)
                        cd ${MSBASDIR}
                        echo "Remove lines in existing MSBAS/MODEi.txt file associated to possible broken links or duplicated lines"
                        _Check_bad_DefoInterpolx2Detrend.sh DefoInterpolx2Detrend1 ${PATH_3601}/SAR_MASSPROCESS &
                        _Check_bad_DefoInterpolx2Detrend.sh DefoInterpolx2Detrend2 ${PATH_3601}/SAR_MASSPROCESS &
                        wait
                        echo "All lines in former existing MODEi.txt are ok"
                        echo ""

                        #Need also for the _Full ones (that is without coh threshold)
                        if [ ${IFCOH} == "YES" ] ; then
                                    _Check_bad_DefoInterpolx2Detrend.sh DefoInterpolx2Detrend1_Full ${PATH_3601}/SAR_MASSPROCESS &
                                    _Check_bad_DefoInterpolx2Detrend.sh DefoInterpolx2Detrend2_Full ${PATH_3601}/SAR_MASSPROCESS &
                                    wait
                                    echo "All lines in former existing MODEi_Full.txt are ok"
                                    echo ""
                        fi

            fi

# Prepare MSBAS
###############
            ${PATH_SCRIPTS}/SCRIPTS_MT/build_header_msbas_criteria.sh DefoInterpolx2Detrend 2 ${BP} ${BT} ${S1ASC} ${S1DESC}

            # update here the R_FLAG if needed
            ${PATHGNU}/gsed -i "s/R_FLAG = 2, 0.02/R_FLAG = ${ORDER}, ${LAMBDA}/"  ${MSBASDIR}/header.txt
            # because interferos are detreneded, i.e. averaged to zero, there is no need to calibrate again
            ${PATHGNU}/gsed -i 's/C_FLAG = 10/C_FLAG = 0/' ${MSBASDIR}/header.txt

            # Check again that files are OK
                        # ensure that format is ok, that is with 4 columns
                        mv DefoInterpolx2Detrend1.txt DefoInterpolx2Detrend1_all4col.txt
                        mv DefoInterpolx2Detrend2.txt DefoInterpolx2Detrend2_all4col.txt
                        ${PATHGNU}/gawk 'NF>=4' DefoInterpolx2Detrend1_all4col.txt > DefoInterpolx2Detrend1.txt
                        ${PATHGNU}/gawk 'NF>=4' DefoInterpolx2Detrend2_all4col.txt > DefoInterpolx2Detrend2.txt
                        # keep track of prblms
                        ${PATHGNU}/gawk 'NF<4' DefoInterpolx2Detrend1_all4col.txt > DefoInterpolx2Detrend1_MissingCol.txt
                        ${PATHGNU}/gawk 'NF<4' DefoInterpolx2Detrend2_all4col.txt > DefoInterpolx2Detrend2_MissingCol.txt
                        rm -f DefoInterpolx2Detrend1_all4col.txt DefoInterpolx2Detrend2_all4col.txt

                        # Need again to check for duplicated lines with different Bp in Col 2 resulting from orbit update
                        if [ ${IFCOH} == "YES" ] ; then
                                    echo "Remove lines in newly created MSBAS/MODEi.txt file associated to possible broken links or duplicated lines"
                                    _Check_bad_DefoInterpolx2Detrend.sh DefoInterpolx2Detrend1 ${PATH_3601}/SAR_MASSPROCESS &
                                    _Check_bad_DefoInterpolx2Detrend.sh DefoInterpolx2Detrend2 ${PATH_3601}/SAR_MASSPROCESS &
                                    wait
                                    echo "All lines in new MODEi.txt should be ok"
                                    echo ""
                        fi

# Let's go
##########
            cd ${MSBASDIR}
            cp -f header.txt header_back.txt

            # EW-UD without coh threshold restriction
            #----------------------------------------
                        case ${FIRSTRUN} in
                                    "YES")
                                                # one have only the newly created MODEi dir and MODEi.txt
                                                mkdir -p ${MSBASDIR}/DefoInterpolx2Detrend1_Full
                                                cp -R ${MSBASDIR}/DefoInterpolx2Detrend1/* ${MSBASDIR}/DefoInterpolx2Detrend1_Full/
                                                cp                                          -f                                  ${MSBASDIR}/DefoInterpolx2Detrend1.txt
${MSBASDIR}/DefoInterpolx2Detrend1_Full/DefoInterpolx2Detrend1_Full.txt
                                                cp -f ${MSBASDIR}/DefoInterpolx2Detrend1.txt ${MSBASDIR}/DefoInterpolx2Detrend1_Full.txt

                                                mkdir -p ${MSBASDIR}/DefoInterpolx2Detrend2_Full
                                                cp -R ${MSBASDIR}/DefoInterpolx2Detrend2/* ${MSBASDIR}/DefoInterpolx2Detrend2_Full/
                                                cp                                          -f                                  ${MSBASDIR}/DefoInterpolx2Detrend2.txt
${MSBASDIR}/DefoInterpolx2Detrend2_Full/DefoInterpolx2Detrend2_Full.txt
                                                cp -f ${MSBASDIR}/DefoInterpolx2Detrend2.txt ${MSBASDIR}/DefoInterpolx2Detrend2_Full.txt
                                                ;;
                                    "NO")
                                                # one must merge the newly created MODEi dir and MODEi.txt with former _Full ones
                                                sort ${MSBASDIR}/DefoInterpolx2Detrend1.txt | uniq > ${MSBASDIR}/DefoInterpolx2Detrend1_tmp.txt
                                                sort ${MSBASDIR}/DefoInterpolx2Detrend2.txt | uniq > ${MSBASDIR}/DefoInterpolx2Detrend2_tmp.txt

                                                sort        ${MSBASDIR}/DefoInterpolx2Detrend1_Full/DefoInterpolx2Detrend1_Full.txt        |        uniq        >
${MSBASDIR}/DefoInterpolx2Detrend1_Full_tmp.txt
                                                sort        ${MSBASDIR}/DefoInterpolx2Detrend2_Full/DefoInterpolx2Detrend2_Full.txt        |        uniq        >
${MSBASDIR}/DefoInterpolx2Detrend2_Full_tmp.txt

                                                cat ${MSBASDIR}/DefoInterpolx2Detrend1_tmp.txt ${MSBASDIR}/DefoInterpolx2Detrend1_Full_tmp.txt | sort | uniq >
${MSBASDIR}/DefoInterpolx2Detrend1_Full.txt
                                                cat ${MSBASDIR}/DefoInterpolx2Detrend2_tmp.txt ${MSBASDIR}/DefoInterpolx2Detrend2_Full_tmp.txt | sort | uniq >
${MSBASDIR}/DefoInterpolx2Detrend2_Full.txt

                                                cp -R -n ${MSBASDIR}/DefoInterpolx2Detrend1/* ${MSBASDIR}/DefoInterpolx2Detrend1_Full/
                                                cp -R -n ${MSBASDIR}/DefoInterpolx2Detrend2/* ${MSBASDIR}/DefoInterpolx2Detrend2_Full/
```

```
                                        cp                              -f                              ${MSBASDIR}/DefoInterpolx2Detrend1_Full.txt
${MSBASDIR}/DefoInterpolx2Detrend1_Full/DefoInterpolx2Detrend1_Full.txt
                                        cp                              -f                              ${MSBASDIR}/DefoInterpolx2Detrend2_Full.txt
${MSBASDIR}/DefoInterpolx2Detrend2_Full/DefoInterpolx2Detrend2_Full.txt

                                        rm -f ${MSBASDIR}/DefoInterpolx2Detrend1_tmp.txt ${MSBASDIR}/DefoInterpolx2Detrend1_Full_tmp.txt
                                        rm -f ${MSBASDIR}/DefoInterpolx2Detrend2_tmp.txt ${MSBASDIR}/DefoInterpolx2Detrend2_Full_tmp.txt

                                        #       because       ${MSBASDIR}/DefoInterpolx2Detrendi_Full.txt       was       built       with       unclneaned
${MSBASDIR}/DefoInterpolx2Detrend2_Full/DefoInterpolx2Detrend2_Full.txt, let's clean it again
                                        echo "Remove again lines in MSBAS/MODEi_Full.txt file associated to possible broken links or duplicated lines"
                                        _Check_bad_DefoInterpolx2Detrend.sh DefoInterpolx2Detrend1_Full ${PATH_3601}/SAR_MASSPROCESS &
                                        _Check_bad_DefoInterpolx2Detrend.sh DefoInterpolx2Detrend2_Full ${PATH_3601}/SAR_MASSPROCESS &
                                        wait
                                        echo "All lines in new MODEi_Full.txt should be ok"
                                        ;;
                        esac
                        # trick the header file
                        ${PATHGNU}/gsed -i 's/DefoInterpolx2Detrend1.txt/DefoInterpolx2Detrend1_Full.txt/' ${MSBASDIR}/header.txt
                        ${PATHGNU}/gsed -i 's/DefoInterpolx2Detrend2.txt/DefoInterpolx2Detrend2_Full.txt/' ${MSBASDIR}/header.txt

                        ${PATH_SCRIPTS}/SCRIPTS_MT/MSBAS.sh _Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh ${TIMESERIESPTS}

                        # Make baseline plot
                        PlotBaselineGeocMSBASmodeTXT.sh ${SET1} ${MSBASDIR}/DefoInterpolx2Detrend1_Full.txt
                        PlotBaselineGeocMSBASmodeTXT.sh ${SET2} ${MSBASDIR}/DefoInterpolx2Detrend2_Full.txt

                        # Now msbas single points (with error bars) times series and plots are in dir. Let's add the description to the naming
                        cp ${TIMESERIESPTSDESCR} ${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh/
                        # remove header line to avoid error message
                        #TIMESERIESPTSDESCRNOHEADER=`tail -n +2 ${TIMESERIESPTSDESCR}`
                        while read -r DESCR X Y RX RY
                                do
                                        echo "Rename time series of ${X}_${Y} as ${X}_${Y}_${RX}_${RY}_${DESCR}"
                                        mv          ${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh/MSBAS_${X}_${Y}_${RX}_${RY}.txt
${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh/MSBAS_${X}_${Y}_${RX}_${RY}_${DESCR}.txt
                                        mv          ${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh/MSBAS_${X}_${Y}_${RX}_${RY}.pdf
${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh/MSBAS_${X}_${Y}_${RX}_${RY}_${DESCR}.pdf
                        done < ${TIMESERIESPTSDESCR} | tail -n +2  # ignore header

                        # Why not some double difference plotting
                        while read -r X1 Y1 X2 Y2 DESCR
                                do
                                        PlotAllNoCoh ${X1} ${Y1} ${X2} ${Y2} ${DESCR}
                        done < ${DOUBLEDIFFPAIRSEWUD}

                        # move all plots in same dir
                        rm -f ${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh/__Combi/*.jpg
                        mv                                      ${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh/*_NoCohThresh_Combi.jpg
${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh/__Combi/

                        # move all time series in dir
                        mv                                                              ${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh/*.txt
${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}_NoCohThresh/_Time_series/

           # EW-UD with coh threshold restriction
           #-------------------------------------
         cd ${MSBASDIR}
         cp -f header_back.txt header.txt

        # run restrict_msbas_to_Coh.sh
         restrict_msbas_to_Coh.sh DefoInterpolx2Detrend1 ${COHRESTRICT} ${KMLCOH} ${S1ASC}/Geocoded/Coh
         restrict_msbas_to_Coh.sh DefoInterpolx2Detrend2 ${COHRESTRICT} ${KMLCOH} ${S1DESC}/Geocoded/Coh

                        # Force pair exclusion
                                if [ ${EXCLUDE1} == "YES" ] ; then
                                        ${PATH_SCRIPTS}/SCRIPTS_MT/zz_Utilities_MT/Exclude_Pairs_From_Mode.txt.sh ${MSBASDIR}/DefoInterpolx2Detrend1
                                fi
                                if [ ${EXCLUDE2} == "YES" ] ; then
                                        ${PATH_SCRIPTS}/SCRIPTS_MT/zz_Utilities_MT/Exclude_Pairs_From_Mode.txt.sh ${MSBASDIR}/DefoInterpolx2Detrend2
                                fi

                        cd ${MSBASDIR}
                        ${PATH_SCRIPTS}/SCRIPTS_MT/MSBAS.sh _Auto_${ORDER}_${LAMBDA}_${LABEL} ${TIMESERIESPTS}

                        # test if MSBAS_log.txt contains "completed 100%" ; if not log error
#                       if ${PATHGNU}/grep -q "writing results to a disk" ${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}/MSBAS_LOG.txt
#                               then
#                                       echo "MSBAS ok"
#                               else
#                                       # try again after cleaning DefoInterpolx2Detrendi.txt
#                                       _Check_bad_DefoInterpolx2Detrend.sh DefoInterpolx2Detrend1 ${PATH_3601}/SAR_MASSPROCESS &
#                                       _Check_bad_DefoInterpolx2Detrend.sh DefoInterpolx2Detrend2 ${PATH_3601}/SAR_MASSPROCESS &
#                                       wait
#
#                                       ${PATH_SCRIPTS}/SCRIPTS_MT/MSBAS.sh _Auto_${ORDER}_${LAMBDA}_${LABEL} ${TIMESERIESPTS}
#                                       if          ${PATHGNU}/grep          -q          "writing          results          to          a          disk"
${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}/MSBAS_LOG.txt ; then echo "Solved after cleaning DefoInterpolx2Detrend's txt"; else  echo "!! MSBAS
crashed on ${TODAY}"  >>  ${MSBASDIR}/_last_MSBAS_process.txt ; fi
#                       fi

                        # Make baseline plot
                        PlotBaselineGeocMSBASmodeTXT.sh ${SET1} ${MSBASDIR}/DefoInterpolx2Detrend1.txt
                        PlotBaselineGeocMSBASmodeTXT.sh ${SET2} ${MSBASDIR}/DefoInterpolx2Detrend2.txt

                        # Now msbas single points (with error bars) times series and plots are in dir. Let's add the description to the naming
                        cp ${TIMESERIESPTSDESCR} ${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}/

                        while read -r DESCR X Y RX RY
                                do
                                        echo "Rename time series of ${X}_${Y} as ${X}_${Y}_${RX}_${RY}_${DESCR}"
                                        mv                      ${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}/MSBAS_${X}_${Y}_${RX}_${RY}.txt
${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}/MSBAS_${X}_${Y}_${RX}_${RY}_${DESCR}.txt
                                        mv                      ${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}/MSBAS_${X}_${Y}_${RX}_${RY}.pdf
${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}/MSBAS_${X}_${Y}_${RX}_${RY}_${DESCR}.pdf
                        done < ${TIMESERIESPTSDESCR} | tail -n +2  # ignore header

                        # Why not some double difference plotting
                        #WhichPlots
                        while read -r X1 Y1 X2 Y2 DESCR
                                do
                                        PlotAll ${X1} ${Y1} ${X2} ${Y2} ${DESCR}
                        done < ${DOUBLEDIFFPAIRSEWUD}

                        # move all plots in same dir
                        rm -f ${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}/__Combi/*.jpg
                        mv ${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}/*_Combi.jpg ${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}/__Combi/

                        # move all time series in dir
                        mv ${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}/*.txt ${MSBASDIR}/zz_UD_EW_TS_Auto_${ORDER}_${LAMBDA}_${LABEL}/_Time_series/


           # Asc and Desc (with coh threshold restriction)
           #--------------------------------------
                        # Prepare header files
                        # Change  second occurrence of "SET = " with "#SET = "
                        ${PATHGNU}/gsed  '0,/SET = /! {0,/SET = / s/SET = /#SET = /}' ${MSBASDIR}/header.txt > ${MSBASDIR}/header_Asc.txt
```

```
                        # Change  first occurrence of "SET = " with "#SET = "
                        ${PATHGNU}/gsed '0,/SET =/{s/SET =/#SET =/}' ${MSBASDIR}/header.txt > ${MSBASDIR}/header_Desc.txt

                        # ASCENDING
                                    cd ${MSBASDIR}
                                    cp -f header_Asc.txt header.txt

                                    ${PATH_SCRIPTS}/SCRIPTS_MT/MSBAS.sh _Asc_Auto_${ORDER}_${LAMBDA}_${LABEL} ${TIMESERIESPTS}

                                    cp ${TIMESERIESPTSDESCR} ${MSBASDIR}/zz_LOS_TS_Asc_Auto_${ORDER}_${LAMBDA}_${LABEL}/
                                    while read -r DESCR X Y RX RY
                                                do
                                                            echo "Rename time series of ${X}_${Y} as ${X}_${Y}_${RX}_${RY}_${DESCR}"
                                                            mv        ${MSBASDIR}/zz_LOS_TS_Asc_Auto_${ORDER}_${LAMBDA}_${LABEL}/MSBAS_${X}_${Y}_${RX}_${RY}.txt
${MSBASDIR}/zz_LOS_TS_Asc_Auto_${ORDER}_${LAMBDA}_${LABEL}/MSBAS_${X}_${Y}_${RX}_${RY}_${DESCR}.txt
                                                            # there is no automatic plotting by msbas when only in LOS
                                    done < ${TIMESERIESPTSDESCR} | tail -n +2  # ignore header

                                    # Why not some double difference plotting
                                    while read -r X1 Y1 X2 Y2 DESCR
                                                do
                                                            PlotAllLOSasc ${X1} ${Y1} ${X2} ${Y2} ${DESCR}
                                    done < ${DOUBLEDIFFPAIRSASC}

                                    # move all plots in same dir
                                    rm -f ${MSBASDIR}/zz_LOS_TS_Asc_Auto_${ORDER}_${LAMBDA}_${LABEL}/__Combi/*.jpg
                                    mv                                                                  ${MSBASDIR}/zz_LOS_TS_Asc_Auto_${ORDER}_${LAMBDA}_${LABEL}/*_Combi*.jpg
${MSBASDIR}/zz_LOS_TS_Asc_Auto_${ORDER}_${LAMBDA}_${LABEL}/__Combi/
                                    # move all time series in dir
                                    #mv                                                                     ${MSBASDIR}/zz_LOS_TS_Asc_Auto_${ORDER}_${LAMBDA}_${LABEL}/*.txt
${MSBASDIR}/zz_LOS_TS_Asc_Auto_${ORDER}_${LAMBDA}_${LABEL}/_Time_series/

                        # DESCENDING
                                    cd ${MSBASDIR}
                                    cp -f header_Desc.txt header.txt

                                    ${PATH_SCRIPTS}/SCRIPTS_MT/MSBAS.sh _Desc_Auto_${ORDER}_${LAMBDA}_${LABEL} ${TIMESERIESPTS}

                                    cp ${TIMESERIESPTSDESCR} ${MSBASDIR}/zz_LOS_TS_Desc_Auto_${ORDER}_${LAMBDA}_${LABEL}/
                                    while read -r DESCR X Y RX RY
                                                do
                                                            echo "Rename time series of ${X}_${Y} as ${X}_${Y}_${RX}_${RY}_${DESCR}"
                                                            mv        ${MSBASDIR}/zz_LOS_TS_Desc_Auto_${ORDER}_${LAMBDA}_${LABEL}/MSBAS_${X}_${Y}_${RX}_${RY}.txt
${MSBASDIR}/zz_LOS_TS_Desc_Auto_${ORDER}_${LAMBDA}_${LABEL}/MSBAS_${X}_${Y}_${RX}_${RY}_${DESCR}.txt
                                                            # there is no automatic plotting by msbas when only in LOS
                                    done < ${TIMESERIESPTSDESCR} | tail -n +2  # ignore header

                                    # Why not some double difference plotting
                                    while read -r X1 Y1 X2 Y2 DESCR
                                                do
                                                            PlotAllLOSdesc ${X1} ${Y1} ${X2} ${Y2} ${DESCR}
                                    done < ${DOUBLEDIFFPAIRSDESC}

                                    # move all plots in same dir
                                    rm -f ${MSBASDIR}/zz_LOS_TS_Desc_Auto_${ORDER}_${LAMBDA}_${LABEL}/__Combi/*.jpg
                                    mv                                                                  ${MSBASDIR}/zz_LOS_TS_Desc_Auto_${ORDER}_${LAMBDA}_${LABEL}/*_Combi*.jpg
${MSBASDIR}/zz_LOS_TS_Desc_Auto_${ORDER}_${LAMBDA}_${LABEL}/__Combi/
                                    # move all time series in dir
                                    #mv                                                                     ${MSBASDIR}/zz_LOS_TS_Desc_Auto_${ORDER}_${LAMBDA}_${LABEL}/*.txt
${MSBASDIR}/zz_LOS_TS_Desc_Auto_${ORDER}_${LAMBDA}_${LABEL}/_Time_series/

            # Back to normal for next run and get out
                        cp -f ${MSBASDIR}/header_back.txt ${MSBASDIR}/header.txt

                        TODAY=`date`
                        echo "MSBAS finished on ${TODAY}"  >>  ${MSBASDIR}/_last_MSBAS_process.txt

                        echo "${LASTASCTIME}" > ${MSBASDIR}/_Last_MassProcessed_Pairs_Time.txt
                        echo "${LASTDESCTIME}" >> ${MSBASDIR}/_Last_MassProcessed_Pairs_Time.txt

                        #mv -f ${MSBASDIR}/${TIMESERIESPTSDESCR}.tmp ${MSBASDIR}/${TIMESERIESPTSDESCR}

# All done...
```

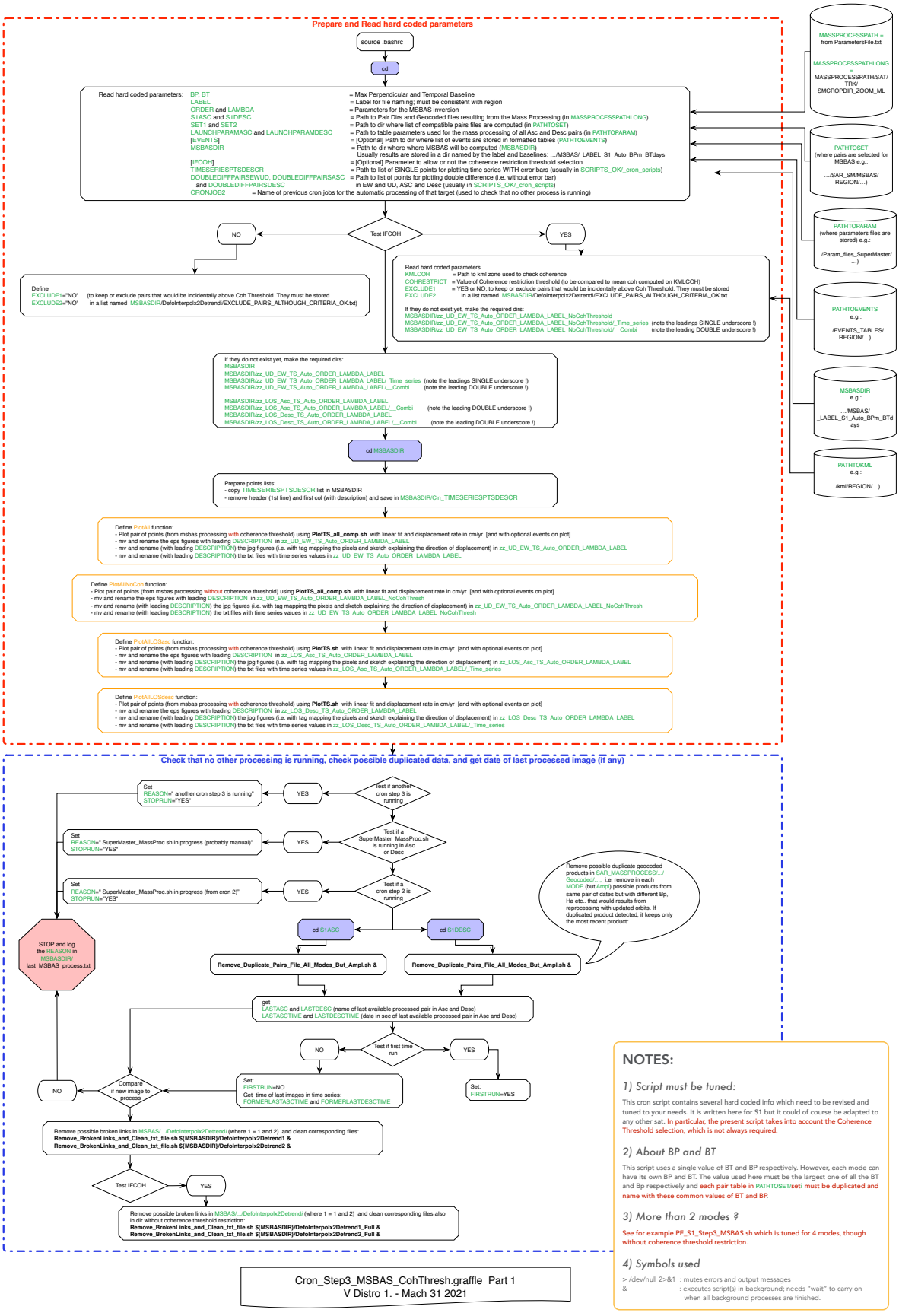


Figure 27: Flow chart of cron job step 3 - part 1/2 (msbas inversion and plot time series) – Beware, this corresponds to an old version, though it must give enough hints to understand the flow.

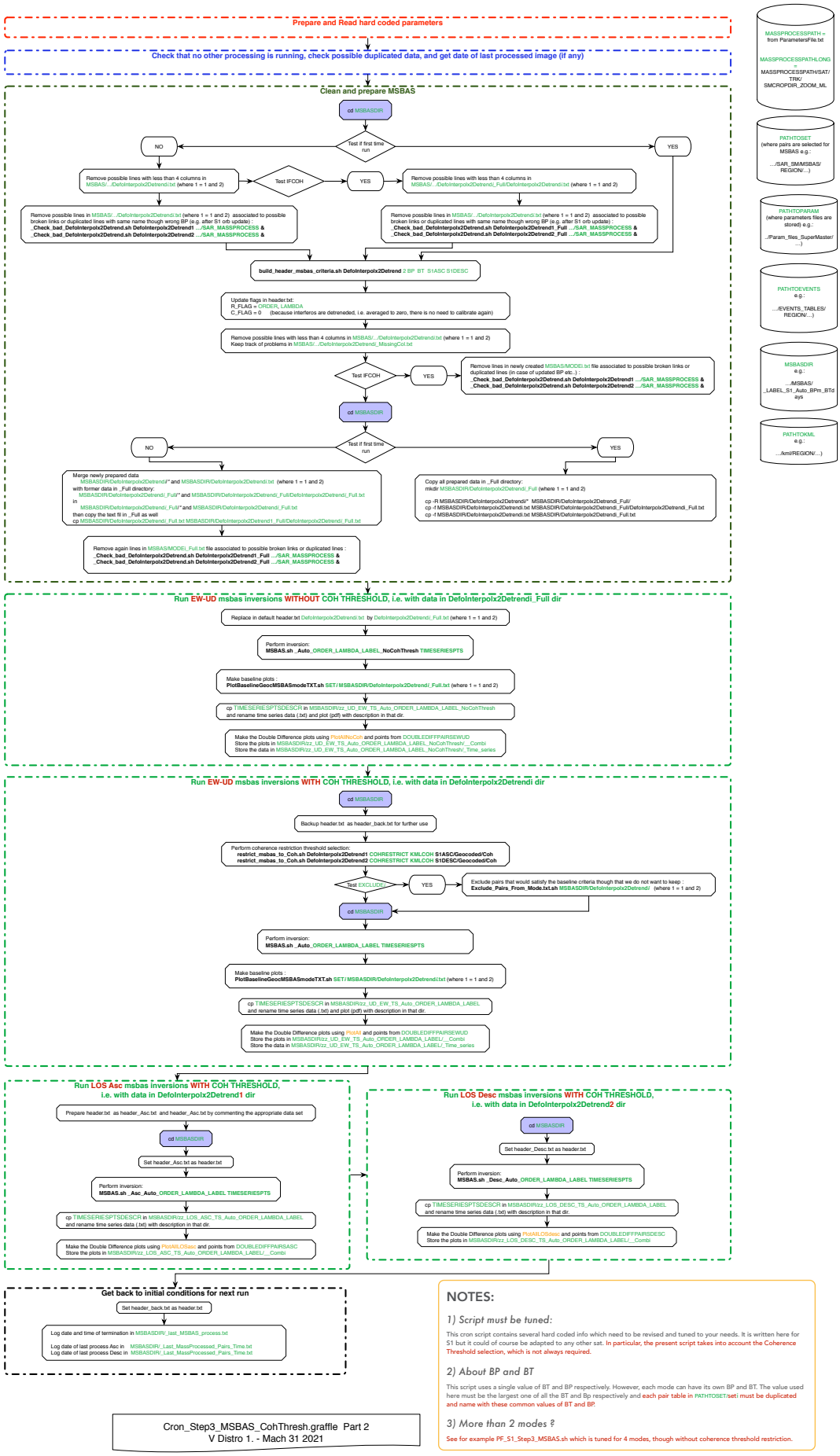


Figure 28: Flow chart of cron job step 3 - part 2/2 (msbas inversion and plot time series) – Beware, this corresponds to an old version, though it must give enough hints to understand the flow.

A.7) Figures

A.8) Index of scripts, main state variables and main files

D

S

W

Z

References

Derauw & Moxhet (1996). Multiple images SAR interferometry (1996), FRINGE'96 workshop, Edinburg 30 Sept - 2 Oct. 1996

Derauw D. (1999)
Phasimétrie par Radar à Synthèse d'Ouverture; théorie et applications.
PhD Thesis, Université de Liège, pp. 1-141

Derauw D., **d'Oreye N.**, **Jaspard M., Caselli A. and Samsonov S. (2020)**
Ongoing automated Ground Deformation monitoring of Domuyo – Laguna del Maule area (Argentina) using Sentinel-1 MSBAS time series: Methodology description and first observations for the period 2015 – 2020.
J. South Am. Earth Sc., Vol. 104, 102850. doi.org/10.1016/j.jsames.2020.102850
See open access: https://www.sciencedirect.com/science/article/pii/S089598112030393X?via%3Dihub

d'Oreye N., Derauw, D., S. Samsonov, M. Jaspard, and D. Smittarello (2021)
MasTer: a full automatic multi-satellite InSAR mass processing tool for rapid incremental 2D ground deformation time series. Proc. IEEE IGARSS21, Brussels, July 2021.

Libert, L., Derauw, D., d'Oreye, N., Barbier, C., Orban, A., (2017).
Split-band interferometry-assisted phase unwrapping for the phase ambiguities correction. Remote Sensing, 9(9), 879.

Samsonov S., N. d'Oreye (2012)
Multidimensional time series analysis of ground deformation from multiple InSAR data sets applied to Virunga Volcanic Province. Geophysical Journal International, vol. 191, Iss, 3, pp. 1095-1108

Samsonov S., N. d'Oreye (2017)
Multidimensional small baseline Subset (MSBAS) for two-dimensional deformation analysis: case study Mexico city. Can. J. Rem. Sens., 43 (4) (2017), pp. 318-329, 10.1080/07038992.2017.1344926

Samsonov S., W. Feng, A. Peltier, H. Geirsson, N. d'Oreye, KK. F.Tiampo (2017). Multidimensional Small Baseline Subset (MSBAS) for volcano monitoring in two dimensions: opportunities and challenges. Case study Piton de la Fournaise volcano. Journal of Volcanology and Geothermal Research, Vol 344, 121-138, https://doi.org/10.1016/j.jvolgeores.2017.04.017

Samsonov S., Dille A., Dewitte O., Kervyn F., **d'Oreye N. (2020)**
Satellite interferometry for mapping surface deformation time series in one, two and three dimensions.
Eng. Geol., 266, 105471. doi.org/10.1016/j.enggeo.2019.105471.
See open access: https://www.sciencedirect.com/science/article/pii/S0013795219316916?via%3Dihub

Smittarello, D., d'Oreye, N., Jaspard, M., Derauw, D., & Samsonov, S. (2022).
Pair selection optimization for InSAR time series processing.
Journal of Geophysical Research: Solid Earth, 127, e2021JB022825.
See open access: https://doi.org/10.1029/2021JB022825