Release Notes GAMMA Software, 20221201

Urs Wegmüller, Christophe Magnard, Charles Werner, Andreas Wiesmann
Gamma Remote Sensing AG
Worbstrasse 225, CH-3073 Gümligen
http://www.gamma-rs.ch
1-Dec-2022

Introduction

This information is provided to users of the GAMMA software. It is also available online at https://www.gamma-rs.ch/uploads/media/GAMMA_Software_upgrade_information.pdf.

This release of the Gamma software includes new programs that provide new capability, additional features to existing programs and bug fixes.

Gamma Software on Linux, macOS, and Windows

The Gamma software has been compiled and tested on Linux (different distributions), Apple macOS Monterey (12.6.1), and Windows 10 and 11. Computationally intensive programs such as used in co-registration and resampling and geocoding have been parallelized using the OpenMP API built into the GCC compiler. Processing speed on Linux, macOS, and Windows systems is comparable.

Linux Distribution:

The Gamma software is developed on Ubuntu 22.04 LTS 64-bit Linux and is tested extensively with this distribution. The Gamma software is also available for Ubuntu 20.04 LTS.

Announcement: Support for Ubuntu 20.04 LTS will be available until the end-of-2023 upgrade.

Versions of the Software will also be uploaded for RHEL7 based on CentOS7 and RHEL8 based on CentOS8. Note that Red Hat has ended support for CentOS8 at the end of 2021. Consequently, the Gamma software built for RHEL8 / CentOS8 uses the final CentOS8 release.

For installation instructions for the binary LINUX distributions see the HTML file INSTALL_linux.html (provided with the distribution E-mail or found in the main directory of the distribution).

Apple MacOS Distribution:

The software in this version has been compiled using macOS Monterey (12.6.1). You will need to install libraries such as GDAL using MacPorts. The build uses GCC 12 compiler on macOS Monterey. A macOS Monterey (12.6.1) M1/M2 version is also available.

Announcement: MacOS Monterey will no longer be supported after the mid-2023 upgrade.

For installation instructions for the binary macOS distributions see the HTML file INSTALL_macOS.html (provided with the distribution E-mail or found in the main directory of the distribution).

Windows Distribution:

The Windows 10 and 11 version of the Gamma software is compiled with 64-bit support and multi-threaded. The build uses the MINGW64 GCC 12 compiler.

For installation instructions for the binary Windows distributions see the HTML file INSTALL_win64.html (provided with the distribution E-mail or found in the main directory of the distribution). Notice that installing the latest GAMMA_LOCAL_w64 version is mandatory because a new GCC compiler and new libraries were used to build the software. Furthermore, the .bashrc file needs to be updated following the installation instructions.

On Windows 11, it is now also possible to install the Windows Subsystem for Linux (WSL) and run a Linux distribution of the Gamma software on that environment. Instructions for this setup are available in the HTML file INSTALL_win11_wsl.html located in the main directory of the distribution.

Documentation and Program List

The Gamma documentation browser is an HTML based system for viewing the web pages and pdf documents. The documentation browser includes for each module a Contents sidebar on the right side of the screen and a search functionality.

The program *gamma_doc* facilitates the access to the documentation related to a given module or program:

gamma_doc	Opens the main page of the Gamma documentation browser and shows the program list.
gamma_doc DIFF	Opens the DIFF&GEO documentation.
gamma_doc gc_map2	Opens the reference manual web page for *gc_map2*.

Further information related to the GAMMA Software is available online:

General information:
gamma-rs.ch/uploads/media/GAMMA_Software_information.pdf
Technical reports, conference and journal papers:
gamma-rs.ch/uploads/media/GAMMA_Software_references.pdf
Release notes / upgrade information:
gamma-rs.ch/uploads/media/GAMMA_Software_upgrade_information.pdf

In case the program list is incomplete, run the python script `program_list.py` after successful installation of the Gamma Software in the main folder of the Gamma Software distribution:

```
./program_list.py Gamma_documentation_base.html Gamma_documentation_contents_sidebar.html -a
```

Python and Matlab wrappers

The Gamma Software is integrated into Python and Matlab through wrappers.

The *py_gamma* Python module permits a smooth usage of the Gamma Software within Python scripts as well as within a Python Interactive Development Environment (IDE) such as Spyder or PyCharm or using Jupyter Notebooks.

In the same way, the Matlab (and Octave) wrapper, composed of *mat_gamma* and *par_file* classes, permits a smooth usage of the Gamma Software within an interactive use of Matlab as well as within Matlab scripts.

Hardware Recommendations

Using multi-core processors (6 or more cores) will bring substantial improvement in processing speed due to parallelization of the code base. There should be at least 8 GB RAM available for each processor core with 16 GB per core recommended. Disk storage requirements for using the Gamma Software effectively depend on the amount of input data and data products that will be produced. Based on our experience we recommend considering at least 16 TB space, especially when working with stacks of Sentinel-1 or very high-resolution data (TerraSAR-X, Cosmo-Skymed) data. The current trend towards larger data products requires substantially increased storage capacities.

GAMMA Software Training Courses

A SAR/INSAR (MSP/ISP/DIFF&GEO/LAT) training at GAMMA (near Bern, Switzerland) is planned for 8th - 11th May 2023. A PSI (IPTA) training is planned for 2nd - 5th May 2023. See also our website under http://www.gamma-rs.ch/courses/training-courses.html.

Significant Changes in the Gamma Software Modules since the mid 2022 Release

Multiple illumination orientation shaded relief

An option to generate a multiple illumination shadad relief was added to programs that use a shaded relief for the visualization or that generate a shaded relief, to be used for the visualization of results in map geometry (*disdem_par*, *disshd*, *mapshd*, *rasshd*). Using a "multiple illumination shadad relief" is an attractive alternative to a single illumination shaded relief as it offers a more homogeneous visibility of the terrain morphology and less dependence of the "visualization quality" on the terrain slope angle.

Estimation of atmospheric path delay

The programs *atm_mod_2d* and *atm_mod_2d_pt* permit the estimate spatially adaptive atmospheric path delay models based on unwrapped InSAR phases in 2d or IPTA vector data format. An often used intermediary step between the estimation of local phase model parameters and the use of these to calculate the simulate atmospheric phase using *atm_sim_2d* or *atm_sim_2d_pt* is spatial filtering of the estimated model parameter. For IPTA stacks this filtering is now supported by the script *atm_mod_2d_pt_filter*.

SLC deskew

For most sensors SLC data are provided in "deskewed" (or "zero-Doppler") geometry. This means the SLC data is resampled to the geometry of a SAR looking in the "zero-Doppler", which is the direction perpendicular to the orbit track – this in spite of the actual sensor looking in a direction that may slightly differ from the "zero Doppler" direction. Furthermore, the phase is typically adjusted so that it corresponds to the "zero Doppler" geometry.

For some sensors, such as PALSAR-1, some data providers (in this case JAXA and ASF) provide SLC data in "non-deskewed" geometry. The new program *SLC_deskew* permits to transform the image from the "non-deskewed" (or skewed) geometry to the "deskewed" (or "zero-Doppler") geometry. In the case of PALSAR-1 doing this transformation is required to be able to correctly concatenate subsequent SLC scenes acquired in the same orbit. In the same step the program permits to (optionally) correct the phase so that it corresponds to the "zero-Doppler" geometry. Furthermore, it also supports the inverse transformation.

Concatenation of multiple consecutive SLCs

The new script *SLC_cat_list.py* concatenates consecutive SLCs specified in a list. The concatenation process is iterative: the first and second SLC are first concatenated, then the resulting SLC is concatenated with the third SLC, etc.

In each iteration, the following process is carried out: based on the metadata in the parameter files, an offset between the two SLC to concatenate is calculated using *init_offset_orbit*. Provided there is enough overlap, a refinement process is then performed: constant offsets in range and azimuth directions are calculated on the overlap area by running two iterations of *offset_pwr_tracking* and *offset_fit*. The two SLCs are then concatenated using *SLC_cat*.

The overlap areas can optionally be used to calculate and apply a constant phase correction to the concatenated SLCs relative to the first SLC. A gain correction based on the *calibration_gain* values provided in the parameter files can optionally be applied to the concatenated SLCs relative to the first SLC.

Coordinates transformation using "coord_trans" and "coord_trans_list"

Coordinates transformation can be performed either interactively using the program *coord_trans* or for a list of points using the new program *coord_trans_list*.

coord_trans:

The program *coord_trans* was updated and now permits using heights relative to a geoid both for the input and in the output. In addition, the interactive navigation has been considerably improved. It is now possible to return to the previous step by typing "back". This can be very useful in case of input / typing error, there's no need to restart the program anymore.

coord_trans_list:

The new program *coord_trans_list* performs coordinate transformation between map projections for a list of points.

The input list of coordinates is a text file, where each line corresponds to one point and each point is defined by a triplet of coordinates. The coordinates on each line are either `northing easting altitude` in meter for map projections or `latitude longitude altitude` for geographic coordinate systems, with the latitude and longitude in decimal degrees and the altitude in meters. Note that SCH coordinates are also supported, in that case the coordinates are arranged in the `S C H` order and the units are in meters. The list of output coordinates is generated by *coord_trans_list* and has the same format as the input list.

The input and output coordinates systems are defined in the input and output DEM parameter files, respectively. Only the definitions of the coordinate systems are used. The parameters about the width and height, edge coordinates and posting are ignored; these fields don't need to be filled, they can be set to 0. The DEM parameter files can be generated using *create_dem_par*. Alternatively, they may be generated when reading GeoTIFF or GDAL-supported files using *dem_import* or *par_data_geo*.

Altitude measurements can be relative to the sea level, i.e. above the geoid or a geoid approximation, or relative to the ellipsoid of the local coordinate system. Global geoid approximations of the Earth geoid are e.g. the EGM96 or EGM2008. Local geoid approximations are also typically available for individual countries, e.g. the LN02 or CHGeo2004 models in Switzerland, GEOID12B in USA, or the GCG2016 geoid approximation in Germany. Geoid raster files or constant geoid heights can be used to convert the coordinates from an altitude relative to the sea level (i.e. above the geoid) to an altitude relative to the ellipsoid of the coordinate system and vice versa.

The (program internal) coordinate transformation workflow is the following:

1. Conversion of input height from an altitude relative to a geoid to an altitude relative to the ellipsoid of the input coordinate system. (optional)
2. Conversion of input coordinates to global (WGS84) Cartesian coordinates.
3. Conversion of global (WGS84) Cartesian coordinates to local latitude/longitude/altitude coordinates of the output coordinate system (use of the ellipsoid of the output coordinate system and datum shift)
4. Conversion of output height from an altitude relative to the ellipsoid of the output coordinate system to an altitude relative to a geoid. (optional)
5. For map projections, conversion to northing/easting/altitude coordinates.

The global (WGS84) Cartesian coordinates may also be of interest and can therefore, optionally, be written to a text file.

Geocoding using very large DEMs

For very large DEMs, or when the memory resources of the computer are limited, running *gc_map2* with generation of several output maps may exceed the available memory.

To overcome this limitation, the script *gc_map2_large.py* has been added. It is first only generating the look-up table using *gc_map2* and then splitting the data in blocks to calculate the other outputs. Once all the blocks have been processed, the results are mosaicked together. The results should be almost the same as when running *gc_map2* for the areas within the SAR image swath. As a result, the memory (RAM) usage is limited to ~3 times the DEM or DEM segment file size.

The intermediate results of the various blocks are stored in a temporary directory called `tmp_dir_gc_map2_XXX` with `XXX` a time stamp in microseconds. The temporary directory is deleted at the end of the processing.

The script *geocoding.py* now includes the option *–large*; when that option is used, it uses *gc_map2_large.py* instead of *gc_map2*.

Finally, *gc_map2* has also been optimized to use slightly less memory.

Update of create_dem_par for automatically defined DEM bounds

Two new options are now available in the interactive mode flag (*iflg* = 2 and *iflg* = 3). These new options are very similar to option *iflg* = 0 but make sure that the grid used is aligned to a standard grid: the grid will pass through full degrees for latitude / longitude coordinates, and through full kilometers for the map projections in meter units. The pixel spacing will be automatically adapted, if needed, to make it compatible with a regular grid. The difference between *iflg* = 2 and *iflg* = 3 is that for *iflg* = 2, full degrees / kilometers will be located at pixel centers, while for *iflg* = 3, full degrees / kilometers will be located at pixel edges.

In addition, *create_dem_par* was also updated to allow a GPRI2 parameter file to be used to automatically define the bounds of the DEM.

Median filtering option added to spatial filters in IPTA

The spatial filter programs for vector data format stack *spf_pt* and *fspt_pt* (in the IPTA) support now also median filtering. Median filtering is of interest to reduce phase noise in cases where significant phase gradients occur (e.g. related to ground motion). Median filtering is more “edge preserving” than distance weighted spatial filtering. It is not expected that the median filtering will now be used very often. In particular for the estimation of atmospheric phases, distance weighted

filtering is typically preferred, but as an option having also median filtering available is valuable, e.g. to filter a displacement rate map.

Doppler parameter estimation from SLC

The program *doppler_2d_SLC* was added to estimate Doppler parameters based on SLC data. Especially for new sensors it is not always clear if the Doppler information (Doppler Centroid, its range dependence and its variation with time) read from the meta data is correct – or it may not be available. The program *doppler_2d_SLC* estimated the Doppler Centroid for many patches over the SLC scene and determines the selected Doppler parameters (e.g. the range dependence and the doppler rate) and writes these into the SLC parameter file.

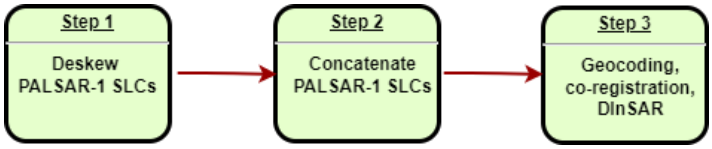
Correct Doppler information is relevant when resampling or oversampling SLC data (e.g. in the co-registration of an SLC to a reference SLC). We used this new functionality, for example, in the case of StriX Spotlight mode data. The program is also helpful to determine if an azimuth phase ramp has been subtracted to "deramp" the azimuth spectrum of an SLC.

New and updated SAR data readers

New / updated SAR data reader	Short description
par_EORC_PALSAR_geo	The program supports the reading of detected geocoded PALSAR-1 and PALSAR-2 data products. In addition. georeferenced data are now also supported: they are resampled to map coordinates using the polynomial provided in the CEOS metadata.
par_TX_ScanSAR, TX_ScanSAR_import_SLC.py	The program *par_TX_ScanSAR* reads in TX ScanSAR data as a "burst SLC data set" with a burst SLC, burst SLC par and "TOPS_par" file. The resulting format permits then to use the "ScanSAR data tools" for steps as co-registration or SLC mosaic generation. The new option [dtype] permits writing FCOMPLEX data when input data are SCOMPLEX. For data written in FCOMPLEX format, the calibration factor (calibration gain) is now automatically applied to the SLC data. *TX_ScanSAR_import_SLC.py* is a script to import all selected sub-swaths and polarizations of a TX ScanSAR SLC data set and ensure all the subswaths share a common grid.

Gamma Software Demo examples

In this period again a Gamma Software Demo example was added. The access to the Gamma Software Demo examples is limited to Gamma Software users with a valid license. The access information is provided with the software delivery.

New / modified demo example:	Contents
PALSAR-1 SLC deskew and concatenation Gamma_demo_PALSAR1_deskew_cat.tar.gz	PALSAR-1 SLC data (level 1.1) retrieved from the ASF platform are in a Doppler centroid (non-deskewed) geometry, and each scene is typically processed using a slightly different Doppler centroid. Concatenation of consecutive SLC images (from the same orbit) requires prior deskewing of the data, i.e., converting them to a zero-Doppler geometry. This demo example demonstrates the reading, deskewing and concatenation of consecutive PALSAR-1 SLC data. This process is performed for two different dates. The reference scene is then geocoded. Finally, the two scenes are coregistered and an interferogram and a coherence map are generated. 

MSP

JERS_PROC_SCRIPT, SIRC_PROC_SCRIPT: The script *JERS_PROC* was renamed to *JERS_PROC_SCRIPT* and *SIRC_PROC* to *SIRC_PROC_SCRIPT* to avoid conflicts with the binary raw data reader programs *JERS_proc* and *SIRC_proc*.

ISP

SLC_intf2: Corrected calculation of the number of lines in the last block.

par_TX_ScanSAR: Added new option [dtype] that permits writing FCOMPLEX data when input data are SCOMPLEX. For data written in FCOMPLEX format, the calibration factor (calibration gain) is now automatically applied to the SLC data.

TX_ScanSAR_import_SLC.py: New script to import TX ScanSAR SLC data. It permits selecting which polarization and sub-swath(s) will be imported. When a reference parameter file or a reference sub-swath is specified, the script resamples the data in range direction according to the range pixel spacing of the reference and makes sure all the sub-swaths share the same spatial grid.

ave_cpx, ave_image: The limit on the maximum number of averaged files has been removed.

ISP_io.c, typedef_ISP.h: Added new SLC parameter DEM_par_file to *typedef_ISP.h.* Updated write_to_SLC_par() routine to write the DEM_par_file parameter when image_geometry is set to GEOCODED (instead of SLANT_RANGE). Updated read_SLC_par_section() to read the DEM_par_file parameter when the image_geometry parameter is set to GEOCODED.

make_tab: The shell script *make_tab* was changed to a perl script for faster execution.

run_all, run_all.pl: *run_all.pl* has been renamed to *run_all* (it replaces the previous csh script *run_all*).

SLC_deskew: New program to change geometry from skewed (or Doppler-centroid) geometry to zero-Doppler (or deskewed) geometry; or vice-versa. Besides the geometry transformation the phase can (optionally) also be changed accordingly.

SLC_cat_list.py: New script to concatenate multiple consecutive SLC images specified in a list.

doppler_2d_SLC: New program to calculate the Doppler centroid of an SLC as a function of slant range and azimuth position using line to line cross-correlation measurements of SLC image data.

SLC_cat: New options were added permitting to select the interpolation method between Lanczos and B-spline as well as the Lanczos interpolator order / B-spline degree. The phase correction option now estimates and applies a constant phase correction (it was previously a range-dependent phase correction) for improved robustness.

DIFF&GEO

multi_cpx: Added functionality to support GPRI2 data by reading the GPRI2 parameter block in SLC or MLI parameter files.

pol2rec: B-spline and Lanczos interpolation methods were added. Parallelized using OpenMP.

dem_gradient: The terrain gradient is now calculated by interpolating the height at locations around (very close to) the pixels. The gradient sharpness can be set using the new [sharpness] option. The new [edge] option avoids ringing artifacts at the edges of the dataset and close to areas with no-data values.

phase_sum: The limit on the maximum number of summed files has been removed.

srtm2dem, vrt2dem, vrt2dem_latlon: The scripts can now also use geoid models in Gamma format (binary file and associated parameter file).

create_dem_par: New options iflg = 2 and iflg = 3: very similar to option iflg = 0 but assuring that the grid used is aligned to a standard grid: the grid will pass through full degrees for lat / lon coordinates, and through full kilometers for the map projections in meter units. The pixel spacing will be automatically adapted, if needed, to make it compatible with a regular grid. The difference between iflg = 2 and iflg = 3 is that for iflg = 2, full degrees / kilometers will be located at pixel centers, while for iflg = 3, full degrees / kilometers will be located at pixel edges. A GPRI parameter file can now also be used to automatically define the bounds of the DEM.

coord_trans: The coordinate transformation now also supports altitudes relative to geoids, both for the input and the output coordinates. The text-based interface is now easier to navigate with clearer instructions. Typing "back" now goes back to the previous step.

coord_trans_list: New program to transform lists of points from one map projection to another one. An option permits also writing the corresponding global Cartesian coordinates. Supports altitudes relative to geoids, both for the input and output coordinates.

par_EORC_PALSAR_geo: Reading georeferenced PALSAR-1 and PALSAR-2 data are now supported, they are resampled to map coordinates using the polynomial provided in the CEOS data.

par_JERS_geo: Reading georeferenced JERS data. Interpolation is now performed using B-spline degree 3 method (on amplitude). Parallelized using OpenMP.

gc_map2: Now uses slightly less memory.

gc_map2_large.py: New script to calculate lookup table and DEM related products (layover and shadow, incidence angle, local resolution, offnadir angle) for very large DEMs. Uses block

processing to limit memory usage. Should be used when memory usage of *gc_map2* is exceeding the available memory (RAM) resources.

LAT

temp_filt, temp_filt_ad, temp_lin_var, temp_log_var, multi_stat, mt_lee_filt, mt_lee_filt_cpx: For all these programs the limit on the maximum number of files has been removed.

*stokes, stokes*_qm: Changed sign of s3 output in stokes and s2chi output in *stokes_qm*.

DISP

replace_values: The input and output data files can now be the same file.

disdem_par, disshd, mapshd, rasshd: A new [illum_mode] option permits the generation of a multiple illumination direction shaded relief. The terrain slope (or gradient) is now calculated by interpolating the height at locations around (very close to) the pixels. The shaded relief sharpness can be set using the new [sharpness] option. The new [edge] option avoids ringing artifacts at the edges of the dataset and close to areas with no-data values.

rasshd: New option to write the shaded relief intensity image in FLOAT format (for later use as brightness in visualization programs.

IPTA

pt_density_reduction: Maximum radius (in slant range pixels) is now a floating point value (previously integer), new defaults were set for the radius and target point density.

def_mod, multi_def: Now also support GPRI2 polar format data.

DEF_MOD_PT_SCRIPT: The script *DEF_MOD_PT* was renamed to *DEF_MOD_PT_SCRIPT* to avoid conflicts under Windows and macOS.

atm_mod_2d_pt: The program has been modified to check if data is available at a point and works with a reduced list of valid data points. In the case of data that use a geocoded point list, it may occur that there are multiple points at the same coordinate and hence exceed the allocated data storage for point data. Data are read until the storage is filled and only these points are used for calculating the atmosphere phase model. Corrected errors for writing out the singular values and reading svd_tol.

atm_mod_2d_pt_filter: New spatial filter for a0 and a1 parameter files estimated with *atm_mod_2d_pt* and then used in *atm_sim_2d_pt.*

base_calc, base_plot: *base_calc* and *base_plot* were removed from the IPTA (they are available in the DIFF&GEO).

spf_pt, fspf_pt: Median filtering method added to [spf_type] option.

mb, mb_pt: Now checking if time-series is complete for reference point / center of reference area, an error message is given if it is not the case.

mk_sp_all: New option added to manually specify the minimum number of valid image values required to calculate the average spectral diversity correlation and the average mean/sigma ratio.

pwr_stat: New option to manually specify the minimum number of valid image values required to calculate the mean/sigma ratio.

Python wrapper

py_gamma.py: The Python wrapper now also support Python scripts for GS-L and GPRI2. To enable this support, the environment variables "GSL_HOME" and "GPRI2_HOME" must have been defined.

py_gamma.py: When both "python" and "python3" calls exist, and "python" points to Python 2, the wrappers will execute Python scripts using the Python version specified in the shebang.

Matlab wrapper

mat_gamma_base.m: The Matlab wrapper now also support Python scripts for GS-L and GPRI2. To enable this support, the environment variables "GSL_HOME" and "GPRI2_HOME" must have been defined.

All packages

INSTALL_win11_wsl.html: Installation instructions for running the Gamma Software under WSL in Windows 11.

**/html/*_documentation.html*: renamed *_documentation_contents_sidebar.html to _documentation.html for all modules (DIFF, DISP, GEO, ISP, IPTA, LAT, MSP, TDBP). The black frame around the iframe has been removed to improve the readability of the help system.

**/html/*.html*: The GAMMA Software html documentation uses now the new css style sheet *gamma_documentation_style.css* in the uppermost directory of the software, to improve readability of the help system.