Release Notes GAMMA Software, 20211201

Urs Wegmüller, Charles Werner, Christophe Magnard, Andreas Wiesmann,
Othmar Frey, Oliver Cartus
Gamma Remote Sensing AG
Worbstrasse 225, CH-3073 Gümligen
http://www.gamma-rs.ch

1-Dec-2021

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 BigSur (11.6.1) and Monterey (12.0.1), 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 20.04 LTS 64-bit Linux and is tested extensively with this distribution. Hence it is highly recommended to run the software on this distribution.

Announcement: Support for Ubuntu 18.04 LTS has been extended for this release, but will no longer be available in the mid-2022 upgrade.

Versions of the Software will also be uploaded for RHEL7 based on CentOS7 and RHEL8 based on CentOS8.

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 BigSur (11.6.1) and macOS Monterey (12.0.1). You will need to install libraries such as GDAL using MacPorts. The build uses the GCC 10 compiler on macOS BigSur and GCC 11 compiler on macOS Monterey.

Announcement: Support for macOS BigSur (11.6.1) will end after the mid-2022 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 10 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).

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 to consider 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 spring 2022. A PSI (IPTA) training is planned for Feb. 2022. See also our website under http://www.gamma-rs.ch/courses/training-courses.html.

Significant Changes in the Gamma Software Modules since the Mid 2021 Release

Script to co-register a stack of ScanSAR or TOPS mode burst SLCs to a reference burst SLC

The new script *ScanSAR_coreg_stack.py* supports the co-registration of a stack of ScanSAR or TOPS mode burst SLCs to a reference burst SLC. It was developed to simplify the co-registration of time-series spanning long periods of time.

Co-registration of ScanSAR and TOPS mode burst SLCs is performed as follows with the Gamma Software using the script *ScanSAR_coreg.py*:

- The second scene is first transformed into the geometry of the reference scene using the orbit data and the topography information.
- The co-registration is refined using an intensity-based cross-correlation method. There, an offset field is generated by computing the cross-correlation between patches spread all over the two scenes. A polynomial fit is then calculated from the offset field and combined with the orbit data and topography transformation.
- A second refinement is then calculated using the azimuth overlap areas between the bursts. There, we make sure that the differential phase in the overlap areas is the same for both pairs of bursts. For this, the co-registration program calculates the difference between the two differential phases in the overlap areas. A non-zero double difference phase can be converted into a corresponding azimuth offset. Double difference phases are computed for all overlap areas and averaged to compute a single azimuth offset for the whole scene.

The azimuth overlap co-registration method (or spectral diversity method) is extremely precise, but it only works if there is interferometric coherence. In areas covered by vegetation, interferometric coherence typically fades rapidly during the growing season(s), while in mountainous areas, snowfalls drastically decrease the coherence. Hence, precise co-registration of scenes with a long temporal baseline may not be possible using the azimuth overlap co-registration method.

To overcome this issue, we can use a third scene. That scene was already successfully co-registered with the reference, and can be used as a secondary reference. The coherence between the second scene and that secondary reference is then high enough to perform the azimuth overlap co-registration.

For the co-registration of stacks of data, we need to make sure that every scene can be precisely co-registered to the reference using appropriate secondary references. In the new script *ScanSAR_coreg_stack.py*, which internally calls *ScanSAR_coreg.py*, 4 co-registration templates are available:

1. No secondary reference for azimuth overlap method.

2. Yearly secondary reference for azimuth overlap method (keyword "yearly"). This template co-registers directly with the reference one scene per year acquired around the same date. The other scenes are then co-registered with the help of the closest (yearly) secondary reference.

3. Temporally adjacent scene as secondary reference for azimuth overlap method (keyword "adjacent"). The temporally adjacent scene(s) to the reference scene is (are) co-registered first. The other scenes are then co-registered by sequentially increasing the temporal baseline with the reference scene and using their already co-registered temporally adjacent scene as secondary reference.
4. Temporally adjacent scene as secondary reference, with yearly direct connection to main reference (keyword "mix"). This template co-registers directly with the reference one scene per year acquired around the same date. The other scenes are then co-registered in the same way as using option "adjacent", but starting from each yearly sub-reference. This ensures that any co-registration error is not propagated over more than 6 months (provided that the yearly reference was successfully co-registered).

The three templates that use secondary references ("yearly", "adjacent", and "mix") are schematically described in Figure 1.

For quality assurance, a file summarizing the co-registration results is generated. It permits to quickly check if all files have been successfully co-registered. A warning is clearly indicated if it is not the case.

When scenes are added over time to a stack of data, *ScanSAR_coreg_stack.py* can be called using the same command to carry on the co-registration of the new scenes without recomputing the previously co-registered scenes (provided the already co-registered scenes have not been deleted).

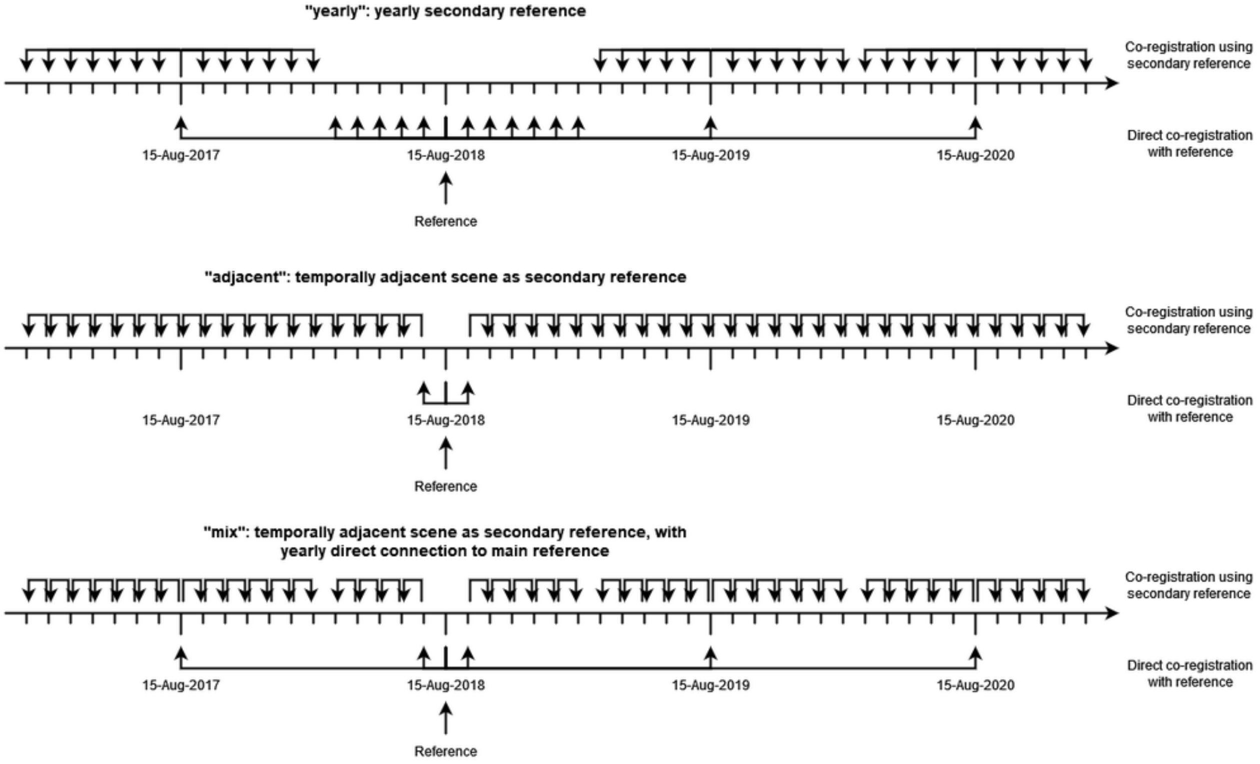
"yearly": yearly secondary reference
Co-registration using secondary reference
Direct co-registration with reference
15-Aug-2017
15-Aug-2018
15-Aug-2019
15-Aug-2020
Reference
"adjacent": temporally adjacent scene as secondary reference
Co-registration using secondary reference
Direct co-registration with reference
15-Aug-2017
15-Aug-2018
15-Aug-2019
15-Aug-2020
Reference
"mix": temporally adjacent scene as secondary reference, with yearly direct connection to main reference
Co-registration using secondary reference
Direct co-registration with reference
15-Aug-2017
15-Aug-2018
15-Aug-2019
15-Aug-2020
Reference

Figure 1 Co-registration templates schematically described. In this example, the (main) reference is a scene acquired on 15-Aug-2018.

Script for terrain geocoding with refinement

The new script *geocoding.py* performs a geocoding sequence of an MLI (multi-looked intensity) image with refinement. It was developed and added to the Gamma Software to simplify a step which is carried out in nearly every SAR processing chain.

The following steps are typically performed:

- Generation of a geocoding look-up table, incidence angle map and layover-shadow maps using *gc_map2*.
- Calculation of the gamma0 normalization area as a simulated backscatter image using *pixel_area*.
- Measurement of an offset field using *offset_pwrm* through cross-correlation between the input MLI image and the gamma0 normalization area image.
- Polynomial fit of the offset field in range and azimuth direction, using *offset_fitm*.
- Refinement of the geocoding look-up table based on the offset polynomials using *gc_map_fine* or *gc_map2*.
- Geocoding of the MLI image using the refined look-up table, using *geocode_back*.
- Calculation of a height map in RDC (slant range / azimuth) geometry using *geocode*.

The output files are named using the *root_name* command line parameter, and follow these conventions:

- Files in or related to RDC geometry will be composed of the root name and an extension.
- Files in or related to MAP geometry (the geometry of the DEM or DEM segment) will be composed of the DEM or DEM segment name, the root name and an extension.

By default, *geocoding.py* will perform the geocoding using the input DEM. Optionally, the area of interest encompassing the area covered by the MLI image will be cropped from the DEM and typically resampled, forming a DEM segment.

Additional options permit generating a KML and/or a GeoTIFF of the geocoded image, as well as manually setting the parameters used in the various programs called by *geocoding.py*.

Options are also available to avoid generating a height map in RDC geometry, generating a geocoded MLI, or performing the geocoding refinement procedure.

Update of phase unwrapping program mcf

The phase unwrapping program *mcf* was updated to improve its reliability and further reduce the probability of phase unwrapping errors.

Two new triangulation modes were added in addition to the previously available filled triangular mesh and Delaunay triangulation modes.

- The first additional mode uses a *filled triangular mesh* and replaces gaps (i.e., masked data or no-data values) by random noise to facilitate the residue discharge at the edge of such areas. That method is now used as default.
- The second additional mode uses again a *filled triangular mesh* and replaces gaps by random noise. In addition, a noise-filled edge is temporarily added around the data during computation to improve phase unwrapping behavior at the edges of valid data.

In addition, new default values have been set for several options:

The number of patches in range and in azimuth is now set to 1 per default. It is strongly recommended to use that default values. Should it not be possible, e.g., due to memory limitations, setting the number of patches to 0 will let *mcf* choose an optimized number of patches. The number of patches can also be manually specified. If a phase unwrapping error is evident on the patch boundary, reducing the number of patches and/or increasing the patch overlap may eliminate the error.

The default overlap between the patches has been increased from 512 to 1024 pixels.

The default reference point is now defined as the center of a rectangular bounding box encompassing the valid input data.

New and updated SAR data readers

New / updated SAR data reader	Short description
par_SICD_SLC	New program to generate SLC parameter and image files for SICD format data (Sensor Independent Complex Data). For example, Capella data can be ordered either in their native format or in SICD format.
par_ESA_PALSAR_GDH	New program to generate MLI and GRD image and parameter files for PALSAR and PALSAR2 level 1.5 GDH data provided by ESA.
par_RCM_MLC	New program to generate parameter and image files for Radarsat Constellation MLC (Multi-Look Complex) data from GeoTIFF or NITF format.
par_Capella_SLC	The following options were added: - *radcal* option to specify output radiometric calibration, either beta0 or sigma0 are available. - *noise* option to perform noise calibration (subtraction of noise floor).
par_NovaSAR_GRD	
par_NovaSAR_SRD	
par_NovaSAR_SLC	

get_data_values

The new program *get_data_values* permits extracting values along a profile, in a polygonal region, or for a set of specified positions, as selected by the user.

It replaces both programs *takethat* and *takethat_dem_par*. These programs were removed from the Gamma Software, hence in case they were used in scripts, their calls should be replaced by a call to *get_data_values*.

In the general case where an SLC/MLI, DIFF/GEO or ISP offset/interferogram parameter file or the image width is provided in the *par/width* command line parameter, the positions text file contains point locations (column and row numbers) to define the profile, the polygon region, or the list of positions where to extract the values from.

In the specific case where a DEM parameter file is provided in the *par/width* command line parameter, the positions text file contains map or geographical coordinates (depending on the coordinate system defines in the DEM parameter file) to define the profile, the polygon region, or the list of positions where to extract the values from.

Several input data file formats are supported. The output values are written to a report text file.

update_par

The new program *update_par* updates parameter files to the latest format version. This can be used to convert parameter files from older Gamma Software versions to their newest format. It can also be used to improve the cosmetics of a parameter file, e.g., when a parameter file has been generated using the Python or Matlab wrappers or for manually-made parameter files.

The parameter file type is automatically detected, *update_par* supports the following parameter file types:

- DEM parameter files
- DIFF/GEO parameter files (DIFF_par)
- GPRI parameter files (SLC/MLI parameter file with additional GPRI-related parameters)
- ISP offset/interferogram parameter files (OFF_par)
- SLC/MLI parameter files
- TOPS parameter files

The output parameter file can be the same as the input parameter file, in which case it will be overwritten.

ScanSAR_ovr.py

The new Python script *ScanSAR_ovr.py* oversamples ScanSAR or TOPS burst data in both range and azimuth directions.

A sampling rate twice the bandwidth is needed to avoid aliasing real data, while a sampling rate equal to the bandwidth is sufficient for complex data. Since the bandwidth of SLC data is typically larger than half the sampling frequency in both range and azimuth directions, detecting (i.e. when calculating intensity images) or computing the differential phase without oversampling causes aliasing. This aliasing may for example slightly skew phase values in interferograms, or cause incoherent areas to spuriously show slightly higher coherence while highly coherent areas may show a decreased coherence.

Oversampling the complex data avoids aliasing them when detecting or when computing interferograms.

The range and azimuth oversampling factors are set by default to 2, only integer factors are accepted. Within *ScanSAR_ovr.py*, a deramping is typically performed before interpolating, and a reramping is performed afterwards (using *SLC_deramp_ScanSAR*). This procedure centers the spectrum and avoids aliasing the data during interpolation. In case the data were already deramped in a previous step, this internal deramping and reramping procedure has to be switched off using the option *--is_deramped*.

Complex multi-looking using multi_cpx

The program *multi_cpx* has been updated and now supports:

- SLC/MLI parameter files
- DEM parameter files (DEM_par)
- DIFF/GEO parameter files (DIFF_par)
- ISP offset/interferogram parameter files (OFF_par)

Note that the selection of an area of interest (using the *loff, nlines*, *roff*, and *nsamp* options) is not supported and will be ignored when DIFF/GEO parameter files are used.

In addition, *multi_cpx* now supports both FCOMPLEX and SCOMPLEX data. It was parallelized using OpenMP. The interpolation is now performed using a 5th degree B-spline method. The program was moved from the ISP to the DIFF package.

Gamma Software Demo examples

In this period again some Gamma Software Demo examples were added/modified. Their access is limited to Gamma Software users with a valid license. The access information is provided with the software delivery.

New / modified demo example:	Contents
Gamma_demo_PALSAR2_ScanSAR_NZ.tar.gz	Demo example on the generation of a large-area differential interferogram using PALSAR-2 ScanSAR SLC (processed using the full-aperture processor, provided in CEOS format). The differential interferograms are generated in slant-range geometry, then geocoded and mosaiced and then unwrapped. Besides the normal co-registration procedure, a procedure that corrects the co-registration for the pixel scale co-seismic offsets is shown (see Figure 2).
Gamma_IPTA_demo_PALSAR2_Ticino.tar.gz	This demo example shows a single reference stack persistent scatterer processing using IPTA for PALSAR-2 stripmap mode data over Switzerland. Figure 3 shows the average deformation rate visualized in Google Earth. For this mountainous site the estimation and subtraction of a height dependent atmospheric path delay is an important processing step.
Gamma_IPTA_demo_PALSAR2_ScanSAR_Mexico.tar.gz	Demonstrates an IPTA processing using PALSAR-2 ScanSAR data (see Figure 4). Point differential interferograms of multiple sub-swaths are mosaiced in the map geometry – so that the main IPTA steps could be done as a single processing. The regression analysis, atmosphere estimation etc. is fully done in the map geometry using the recently added related functionality.
IPTA_demo_S1_Aletsch_from_orig.tar.gz	Now uses the new stack co-registration script *ScanSAR_coreg_stack.py* and the new geocoding script *geocoding.py*. The demo was also slightly simplified.
IPTA_demo_S1_Aletsch_from_rslc.tar.gz	Now uses the new geocoding script *geocoding.py*. The demo was also slightly simplified.

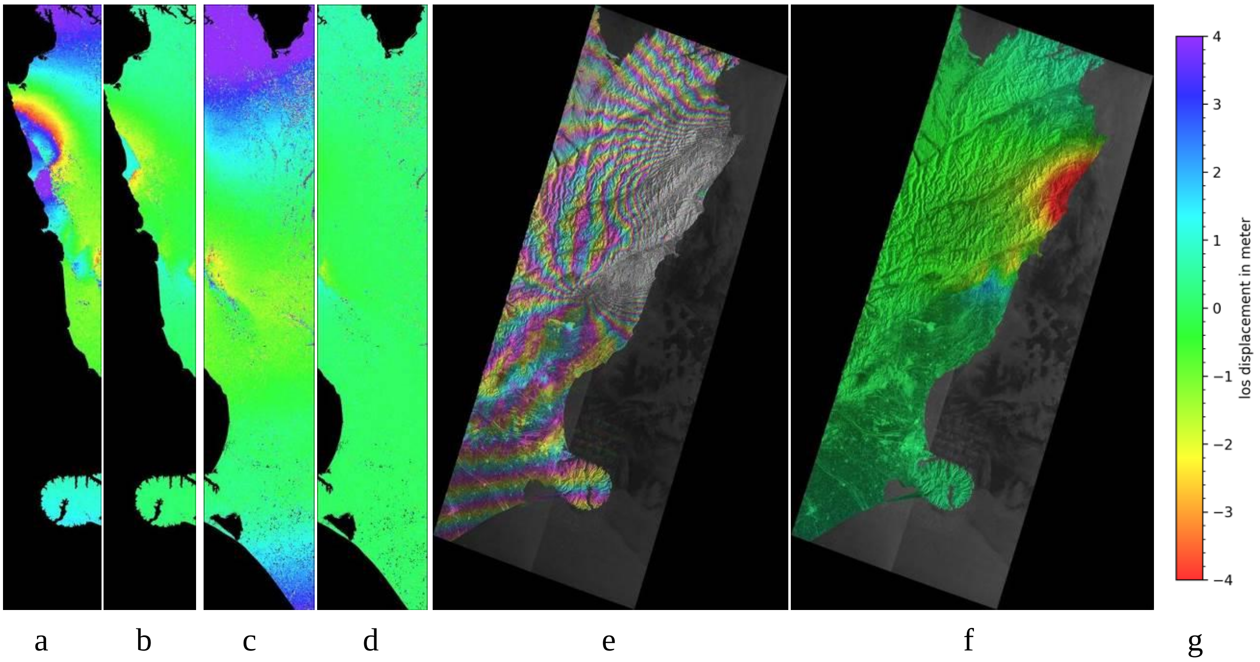

Figure 2 Kaikoura earthquake: DInSAR with PALSAR-2 ScanSAR data (20161018-20161115). a) and b) show the determined azimuth and range offset for sub-swath F4, and c) and d) for F5. e) shows the geocoded complex valued differential interferogram and f) the line-of-sight displacement, using the scale in g).

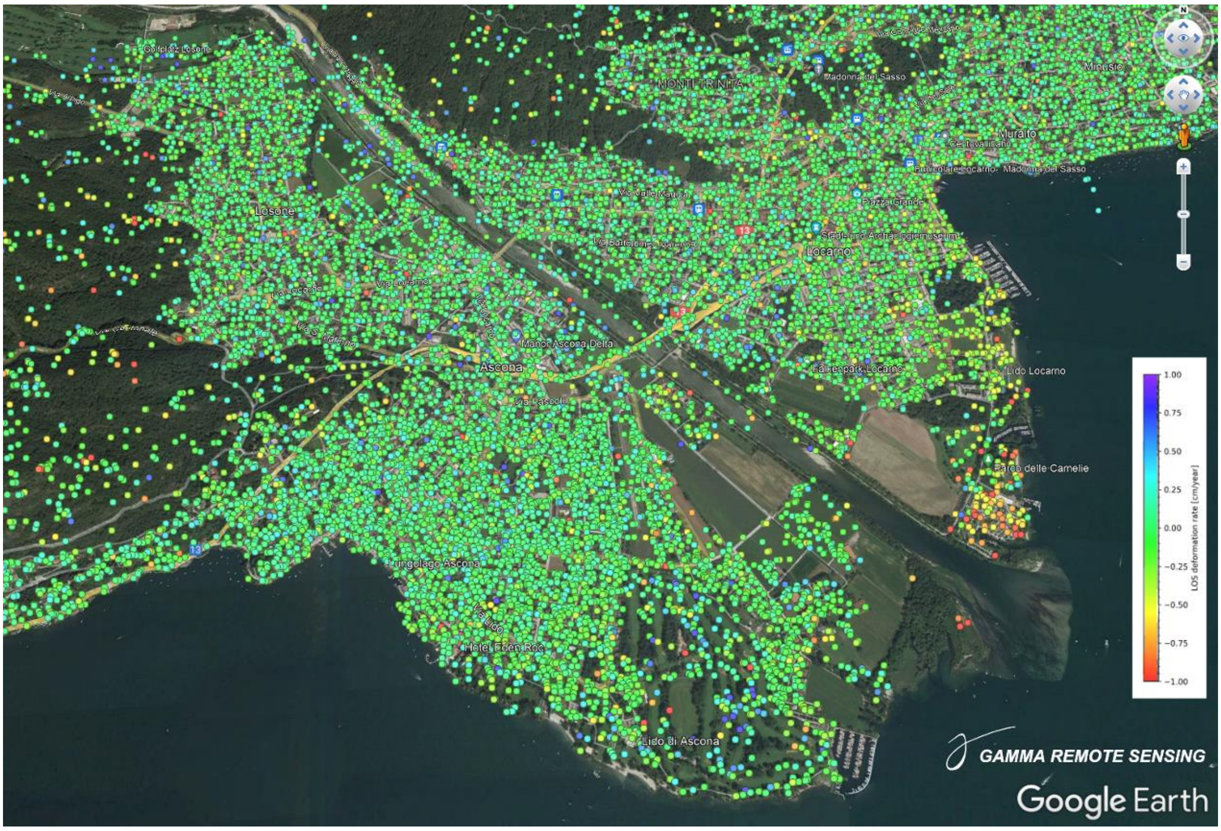

Figure 3 PALSAR-2 Persistent Scatterer Interferometry (PSI) over Locarno, Switzerland visualized in Google Earth.

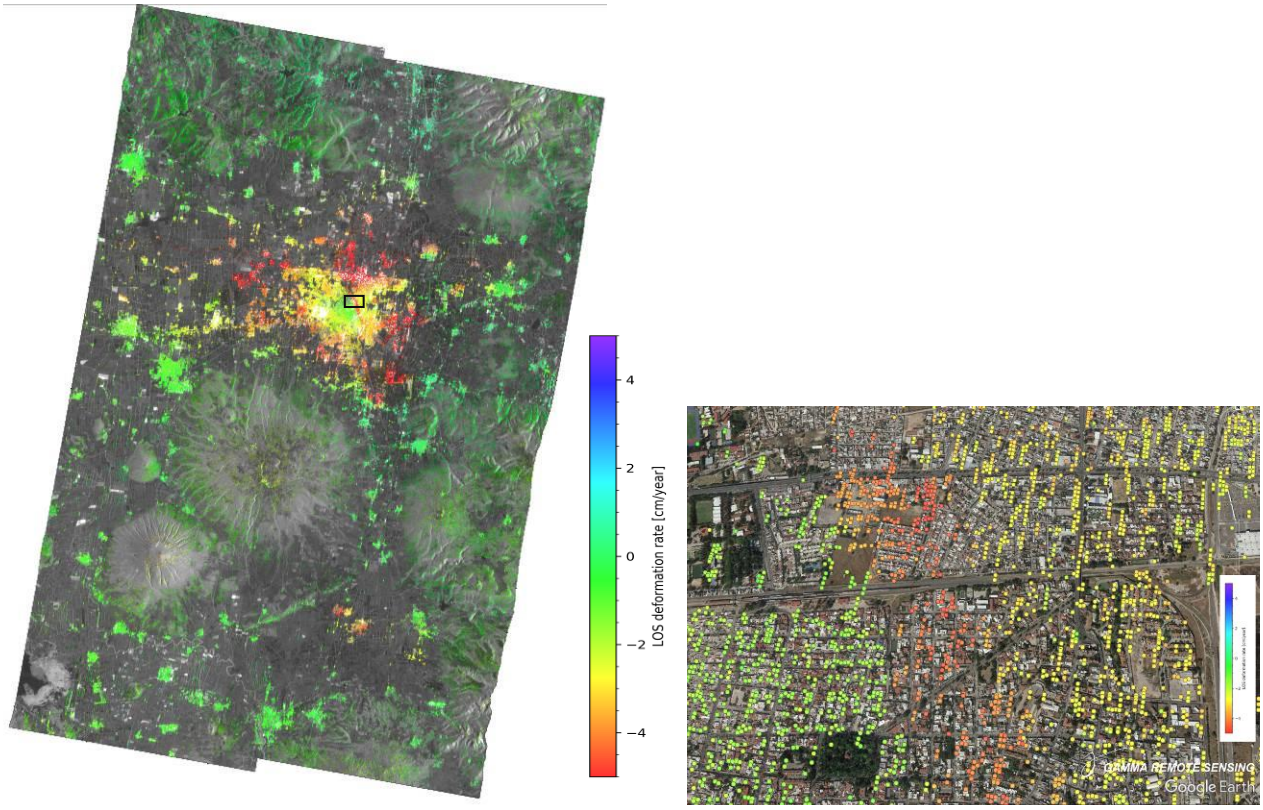

Figure 4 Persistent scatterer interferometry result over Celaya, Mexico, derived from PALSAR-2 ScanSAR data acquired between 2014 and 2018. The image to the left shows an overview that includes data of two adjacent sub-swaths. The image to the right shows the result for a distinct subsidence feature (area indicated by black rectangle) in more detail.

MSP

-

ISP

par_SICD_SLC: New program for generating SLC parameter and image files for SICD SLC data.

par_ESA_PALSAR_GDH: New program for generating MLI and GRD image and parameter files for PALSAR and PALSAR2 level 1.5 GDH data provided by ESA.

mcf: New tri_mode options added where zero-values are replaced by random noise (which allows to connect residues through these no-data zones) and an edge filled with random noise is temporarily added around the data during computation (which may be useful in cases with an overall phase ramp, where connecting residues to the edge is reasonable) to improve phase unwrapping behavior at the edges of valid data. New default values for triangulation mode, number of patches, overlap between patches, and reference point were introduced.

par_EORC_PALSAR_ScanSAR: New program for generating SLC parameter and image files from EORC PALSAR2 ScanSAR burst SLC data in CEOS format. While InSAR with the PALSAR2 ScanSAR burst SLC data becomes functionally possible, using the full-aperture SLC product is recommended (as we observed phase anomalies when using the burst SLC data).

par_Capella_SLC: Radiometric and noise calibration options added to par_Capella_SLC.

GRD_to_SR, SR_to_GRD: Addition of following interpolation methods: bicubic spline sqrt(x), B-spline, and B-spline sqrt(x). Interpolation is now parallelized using OpenMP.

par_RCM_MLC: New program for generating parameter and image files for Radarsat Constellation MLC (Multi-Look Complex) data from GeoTIFF or NITF format.

SLC_ovr: A deramp flag has been added as option to turn ON or OFF the internal deramping and reramping procedures, with its default value set to ON. Deramping is performed to center the spectrum, which avoids aliasing the data during interpolation. The deramp flag needs to be set to OFF for SLC data already deramped.

ScanSAR_ovr.py: New script to oversample ScanSAR or TOPS burst SLC data in range and/or azimuth.

par_NovaSAR_GRD, par_NovaSAR_SRD, par_NovaSAR_SLC : New [radcal] option to specify output radiometric calibration. Either beta0 or sigma0 are available.

par_NovaSAR_GRD, par_NovaSAR_SRD, par_NovaSAR_SLC: New [noise] option to perform noise calibration.

DIFF&GEO

gc_map2: New option [inc_flg] permits writing the incidence angle instead of the local incidence angle.

rdc_trans: Now also supports the resampling of data acquired in an ascending track to a reference in a descending track, and vice-versa.

ScanSAR_coreg_stack.py: New script to co-register a stack of ScanSAR or TOPS mode burst SLCs to a reference burst SLC. Multiple co-registration templates are available, enabling accurate co-registration for burst SLC stacks acquired over long periods of time.

geocoding.py: New script running a terrain geocoding sequence with refinement. Replaces the previous script *GC-MAP*.

multi_cpx: Now supports SLC/MLI, DEM, DIFF/GEO and ISP offset/interferogram parameter files, both FCOMPLEX and SCOMPLEX data. Parallelized using OpenMP. Interpolation method is now 5th degree B-spline. Moved from ISP to DIFF.

ionosphere_mitigation, ionosphere_mitigation_S1: Scripts can now also be used without access to the LAT package.

DISP

vis.py*: If the backend selected using option -k is not available, the vis*.py scripts will now try to select another available backend.

get_data_values: New program to extract data values along profiles, in polygon regions, or from indicated positions. Replaces the programs *takethat* and *takethat_dem_par*.

update_par: New program to update a parameter file to its latest format.

LAT

-

IPTA

dis_ipta, gras8_ipta, gras24_ipta: Increased the image window size to 1000x480, zoom window to 160x16. Increased gnuplot window size to 800x600. Increased the size of the plot character to "2", plot character color set to blue for better visibility.

ras_data_pt: Now plots points even when the background is not available in an area, due to limitations of the DEM extent or DEM inaccuracy. The point data are drawn with a constant intensity.

atm_mod_pt: Now calculates phase model 1 (range dependent phase), for all points in the list Previously the model was only calculated for points marked as valid in the mask.

atm_mod_pt: Models 2 and 3 added to atm_mod_pt, which are equivalent to the models available in *atm_mod2* in the DIFF package.

sp_stat, mk_sp_all: Added az_ovr command line option to support point scatterer candidates selection for SLC data that have been oversampled in range and azimuth (previous version supported only data oversampled in range).

mb: Now only opens a limited number of files simultaneously. Parallelized using OpenMP.

expand_data_pt, lin_comb_pt, cat_pt: Now also support data type POSITION (x,y DOUBLE). This modification is used to support working with position data, e,g, in the combined IPTA processing of PALSAR-2 ScanSAR data (see the related new demo example).

mb: Now also supports the case where the DIFF_tab only contains filenames corresponding to unmasked pairs in the ITAB.

Python wrapper

-

Matlab wrapper

-

All packages

makefile_static_macOS: Modified to work with the new directory structure introduced by Apple for system libraries and include files. Now defining SYS_LIB, SYS_INCLUDE, and SDK_VERSION environment variables.