Release Notes GAMMA Software, 20191209

Urs Wegmüller, Charles Werner, Andreas Wiesmann, Othmar Frey,
Christophe Magnard, Oliver Cartus
Gamma Remote Sensing AG
Worbstrasse 225, CH-3073 Gümligen
http://www.gamma-rs.ch
9-Dec-2019

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, OSX, and Windows

The Gamma software has been compiled and tested on Linux (different distributions), Apple MacOS Mojave (10.14.6) and Catalina (10.15.1), and Windows 10 (64-bit, should also function on Windows 7,8). 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 18.04 LTS 64-bit Linux and is tested extensively with this distribution. Hence it is highly recommended to run the software on this distribution.

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 Mojave (10.14.6) and Catalina (10.15.1). You will need to install libraries such as GDAL using MacPorts.

Announcement: MacOS Mojave (10.14.6) will no longer be supported after the mid 2020 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 7, 8, 10 version of the Gamma software is 64-bits and multi-threaded. The software has been compiled under Win10 and is expected to run on Win7, and 8. The build uses the MINGW64 GCC 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_20190606 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.

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_map	Opens the reference manual web page for g*c_map*.

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
```

Hardware Recommendations

Using multi-core processors (4 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) and a PSI (IPTA) training are planned for spring 2020. See also our web-site under http://www.gamma-rs.ch/courses/training-courses.html.

Significant Changes in the Gamma Software Modules since the Mid 2019 Release

Replacing gc_map by gc_map2

Why?

The calculation of terrain-geocoding lookup tables and DEM derived data products using *gc_map* is part of most processing pipelines. In 2017, *gc_map2* was introduced as an alternative to *gc_map*, and had in particular an improved algorithm for calculating the layover-and-shadow map. The layover-and-shadow maps produced by *gc_map* were suboptimal for acquisitions not performed on a North-South or South-North heading, such as in spaceborne acquisitions at high latitudes and airborne acquisitions. This shortcoming was addressed in *gc_map2*, however the program required noticeably more time and computing resources than *gc_map*. Recently, we noticed some additional shortcomings in *gc_map* products, such as slightly shifted incidence angle maps (fractions of pixels), also impacting the maps derived from the incidence angle map such as the simulated SAR backscatter image. Values for these products were also missing at the edges.

Updating gc_map2

We decided to address the issues described above by completing and speeding-up *gc_map2*. Furthermore, the outputs that were not yet available (*sim_sar*, *u*, *v*, *psi* and *pix*) were added and the computations were optimized in several ways, such as better using calculations made for previous pixels in the current pixel calculation, or scaling down the data to avoid redundant calculations. The result was a speed-up factor of up to ten times, without noticeable quality decrease, yielding processing times approaching those of *gc_map*. As a consequence of the modification of the parameters on the command line of gc_map2 scripts using *gc_map2* should be checked and updated, if necessary.

Renaming the old gc_map to gc_map1, new emulating script

Since *gc_map* is used in so many pipelines and processing scripts, a replacing script with the same name was written to emulate the old *gc_map* usage while calling *gc_map2*. Hence scripts and pipelines that call *gc_map* will still work as expected. It is nevertheless recommended to directly call *gc_map2* and adapt the command line to its slightly different usage. The old *gc_map* program was renamed to *gc_map1* and is hence still available.

The *OFF_par* input is not supported by *gc_map2*. In case an *OFF_par* input is used (which is not recommended), the script will call the old *gc_map1* program. The *ls_mode* options 1 (linear interpolation across these regions) and 3 (nn-thinned) are not available, and the script will replace them automatically by the option 2 (actual value).

Comparison between gc_map1 and gc_map2

A Sentinel-1 acquisition over Antarctica is used to illustrate the improvements brought by *gc_map2*. We used the Reference Elevation Model of Antarctica for this study. In Fig. 1, we notice that large areas were detected as layover using *gc_map1*. On the actual image and on the layover-shadow map produced using *gc_map2*, the layover areas are much smaller or non-existent at all. Some shadow areas were also wrongly detected using the old version, some of these being actually detected as layover using the new version.

In addition to producing incidence angle or simulated backscatter maps up to the edges, the correction of the incidence angle calculation now ensures much higher consistency between the simulated backscatter images produced using *gc_map2* (*sim_sar*) and *pixel_area* (*pix_gamma0*). This can be confirmed by an offset analysis, as shown in Table 1 where the *sim_sar* image produced using *gc_map1* yields a noticeably different and less accurate result.

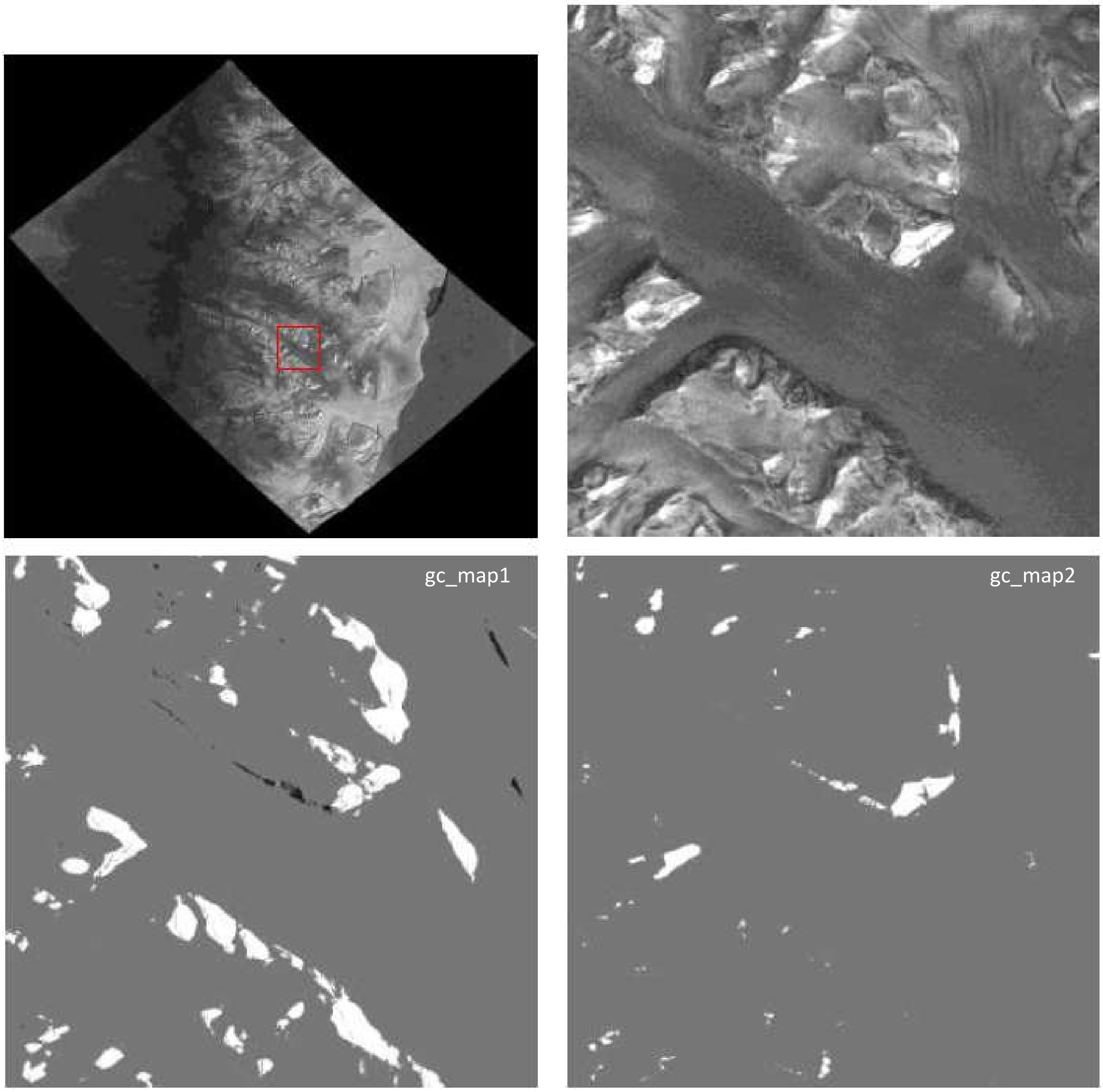

Fig. 1 Top-left: Geocoded Sentinel-1 image in Antarctica with area of interest marked by a red square. Top-right: Zoom of the area of interest. Bottom-left: Layover-shadow map using *gc_map1* program. Bottom-right: layover-shadow map using *gc_map2* program. Layover is shown in white and shadow in black.

For the offset analysis, the simulated images produced by *gc_map1* and *gc_map2* were transformed back to radar geometry using the *geocode* program. Note that the DEM had to be edited to fill some void areas and that its absolute accuracy may not be exceptional.

Table 1. Results of an offset analysis using *gc_map1* and *gc_map2*. Units are in pixels. Slant range pixel spacing: 9.78 m, azimuth pixel spacing: 20 m.

	gc_map1	*gc_map2*
offset MLI - *pix_gamma0* (rg, az)	(1.33, 0.24)	(1.40, 0.06)
offset MLI - *sim_sar* (rg, az)	(1.47, 2.51)	(1.37, 0.14)
std. dev. MLI - *pix_gamma0* (rg, az)	(0.68, 0.54)	(0.74, 0.42)
std. dev. MLI - *sim_sar* (rg, az)	(1.00, 2.46)	(0.65, 0.49)

Noise artifacts at image borders of Sentinel-1 GRD products

Sentinel-1 GRD products from before 2018 contain noise artifacts at the image borders. These include very low non-zero values at near and far range, and ramps with border noise in azimuth direction at the beginning of the first swath and the end of the last swath. This issue is well described in [1] and several methods are used to clean these borders.

Note that new products from 2018 (Sentinel-1 IPF version >= 2.90) do not include these artifacts anymore.

A new *edge_flag* option was added to *par_S1_GRD* program. It provides two methods for cleaning the edges.

- The first method is a coarse method. It analyzes a range line at the azimuth center of the data and sets a minimum and maximum range distance using a threshold hence cleaning most artifacts at near and far range.
- The second method extracts the image edges using a Canny edge detection filter [2]. These edges are analyzed: only vertical and horizontal edges are kept; among these, the most intense and longest edges are used to set the new image margins. Data outside the margins are set to *no-data*.

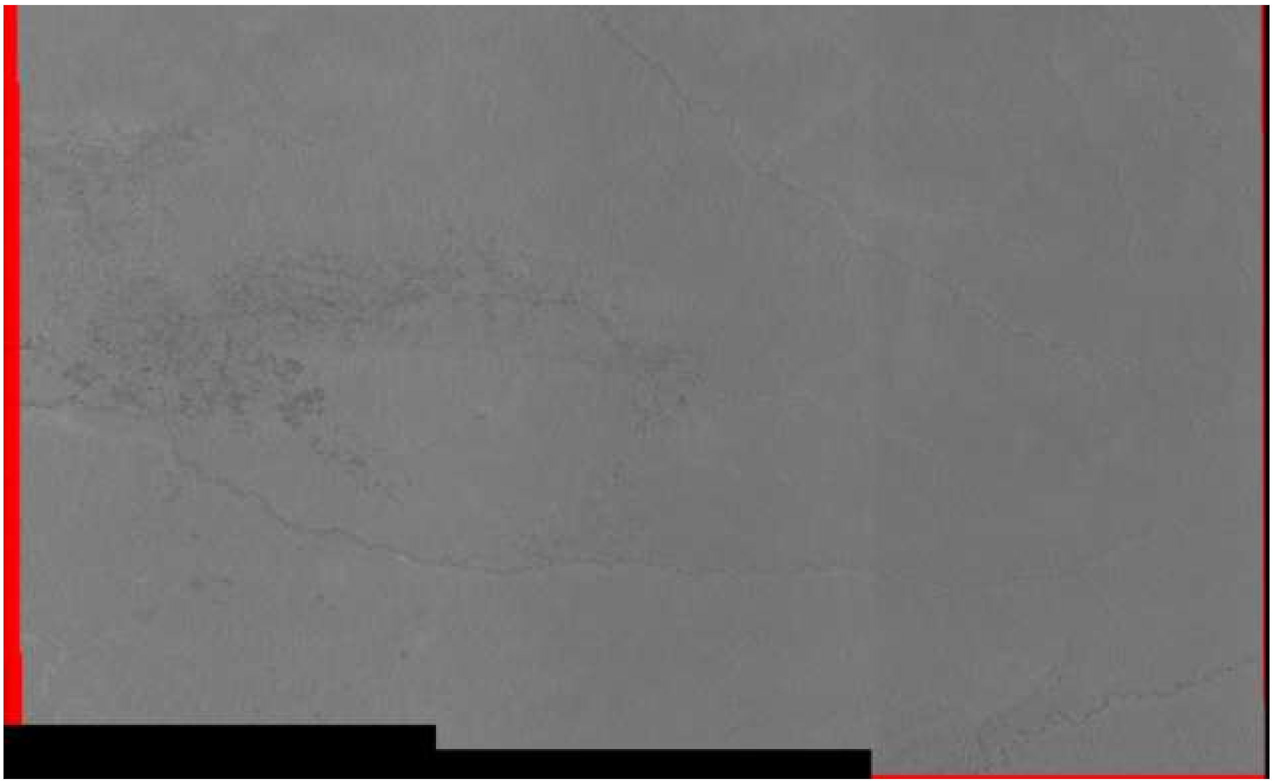

Fig.2 Sentinel-1 GRD image acquired over the Amazon rainforest. The red areas show the non-zero pixels that were detected as noise artifact using the second method and replaced by *no-data* values. Black areas were already filled with *no-data* values in the original product.

[1] I. Ali, S. Cao, V. Naeimi, C. Paulik and W. Wagner, "Methods to Remove the Border Noise From Sentinel-1 Synthetic Aperture Radar Data: Implications and Importance For Time-Series Analysis," in *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 11, no. 3, pp. 777-786, March 2018.

[2] J. Canny, "A Computational Approach to Edge Detection," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 6, pp. 679-698, Nov. 1986.

edge_detection

The *edge_detection* program permits detecting edges in images using a Canny edge detector [2]. In addition, it includes an optional linear regression-based extraction of line segments. Float and raster images are supported. RGB images are internally transformed into a YCbCr color space, and the edge detection is performed on the luminance channel. The output is an edge map in float type (see example in Fig. 3). When line segments are extracted, a text file describing these segments can be written as output.

An option permits performing a sqrt or log operation on the image before detecting the edges: it is recommended to perform one of these operations on MLI images, to reduce their dynamic range. Filtering MLI images prior to the edge detection can considerably improve its results, e.g. using the bm3d program.

The Canny edge detection includes following steps:

- Gaussian filtering (smoothing) to reduce noise in the image.
- Calculation of the gradient magnitude and direction.
- Non-maximum suppression of the gradient magnitude using the gradient direction.
- Hysteresis thresholding: all pixels above a high threshold are kept; pixels above a second, lower threshold that can be connected to those above the high threshold are also kept; the other pixels are discarded.

The linear regression-based extraction of line segments works as follows:

- The process starts from a pixel detected as an edge, defining the seed of a line segment.
- The neighborhood of the last pixel added to the line segment is scanned, and a valid pixel is added to the line segment as a candidate.
- It is kept in the line segment if:
    - o The standard deviation of the linear regression computed on the last n pixels of the segment is below a threshold.
    - o The distance between the pixel candidate and the linear regression computed on the last n pixels is below a threshold.
    - o The angle between the vector formed by the two last pixels of the line segment and the vector formed by the last pixel and the candidate pixel must not be larger than 90°, to make sure that the line segment detection goes forwards and not backwards.

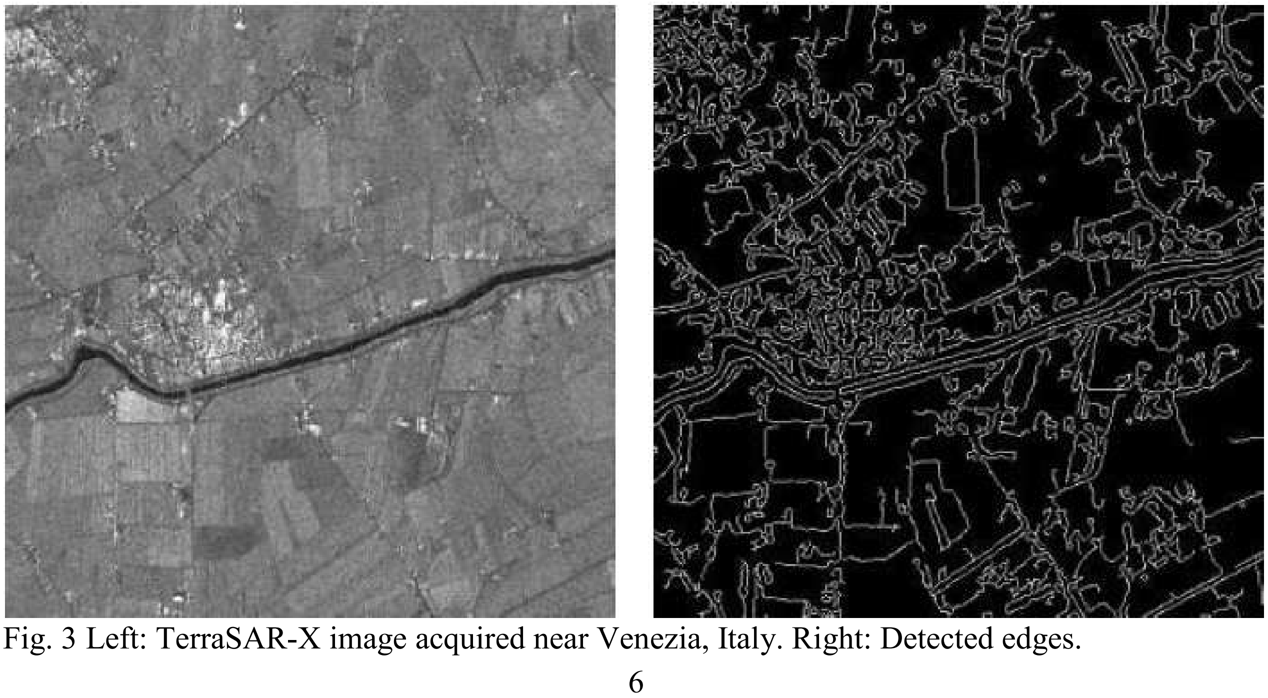
Fig. 3 Left: TerraSAR-X image acquired near Venezia, Italy. Right: Detected edges.

ScanSAR_burst_overlap

ScanSAR_burst_overlap extracts and mosaics overlapping parts of ScanSAR / TOPS burst data.

The data of the two bursts available for the overlap areas have significantly different Doppler centroids (or skew angles). Investigating the data of such overlap areas between neighbor bursts is useful:

- to investigate directional scattering effects (see Figure 4)
- to identify and estimate ionospheric effects in interferometric products (see Figure 5)
- to identify and estimated along track displacements (see Figure 5)

The program includes three modes. The first two modes yield two mosaics containing the burst overlap areas respectively from the earlier and later bursts; in the first mode, the non-overlapping areas are set to 0 while in the second they are included in both mosaics. The third mode yields burst SLC files containing the overlap areas respectively from the earlier and later bursts for each swath.

ScanSAR data may have multiple overlapping bursts. In such cases, it is possible to compare the overlapping areas not only between consecutive bursts but also between bursts and their "second", "third", etc., neighbors.

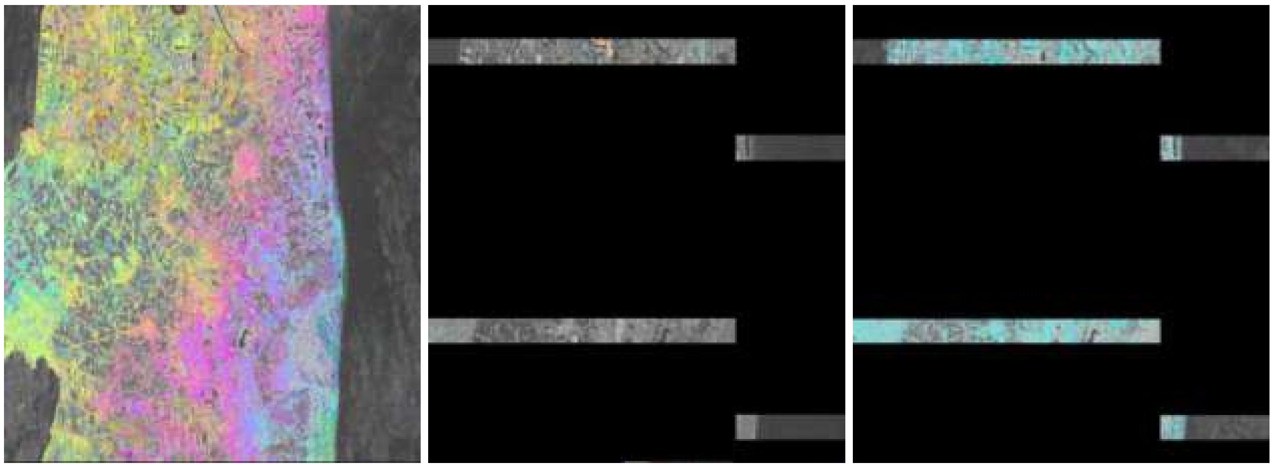

Fig. 4 S1 differential interferogram over the Netherlands (left), forward to backward look direction intensity ratio in a Hue-Intensity-Saturation display (center, red and blue areas indicate fields with strong directional scattering, and double difference interferometric phase (right, phase difference between forward and backward looking interferometric phase).

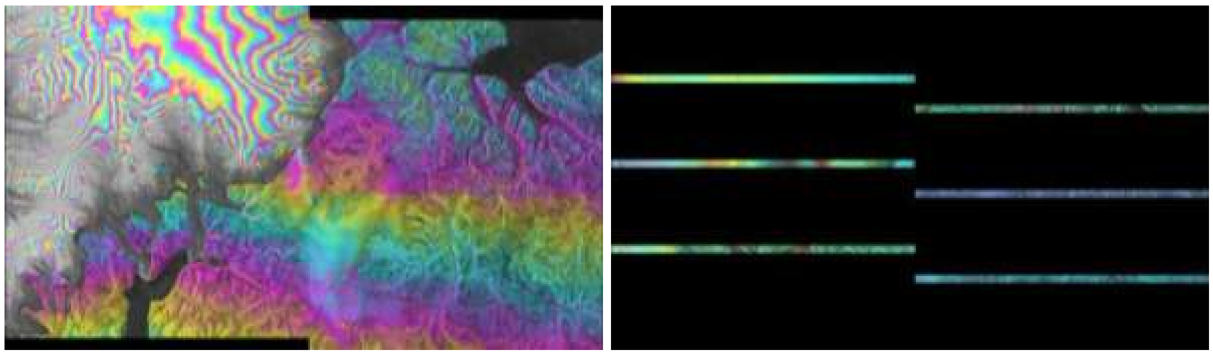

Fig. 5 S1 differential interferogram over Devon Ice Cap (left), and corresponding double difference interferometric phase of burst overlap areas (right). Colors different from blue indicate significant non-zero phase relating to ionospheric effects and along track ice motion.

SLC_intf_geo

SLC_intf_geo reads the two co-registered, single-look complex SAR images SLC-1 and SLC-2, that have been terrain geocoded and creates a multi-look interferogram and 2 co-registered intensity images. The complex (normalized) interferometric coherence is defined by:

$$\gamma = \frac{\left|\sum_i x_i y_i^*\right|}{\sqrt{\sum_i x_i x_i^* \sum_i y_i y_i^*}}$$

where x and y are the single look complex values of SLC-1 and SLC-2, and * stands for conjugate complex, i.e, (a + jb)* = (a - jb). The multi-look interferogram pixel is estimated by (coherent) averaging of the single-look values.

The argument of the complex interferogram corresponds to the interferometric phase. The magnitude of the complex interferogram corresponds to the interferometric correlation derived from the number of interferometric looks used in the multi-looking.

In order to create a differential interferogram, the simulated phase in slant-range geometry must be resampled to the geometry of the interferogram using *geocode_back* combined with a lookup table that resamples the simulated phase to map geometry.

The HTML-based documentation for *SLC_intf_geo* includes an example showing the calculation of the deformation caused by the Hector Mine earthquake from a pair of ERS acquisitions. Fig. 6 shows the geocoded interferogram including the topographic phase.

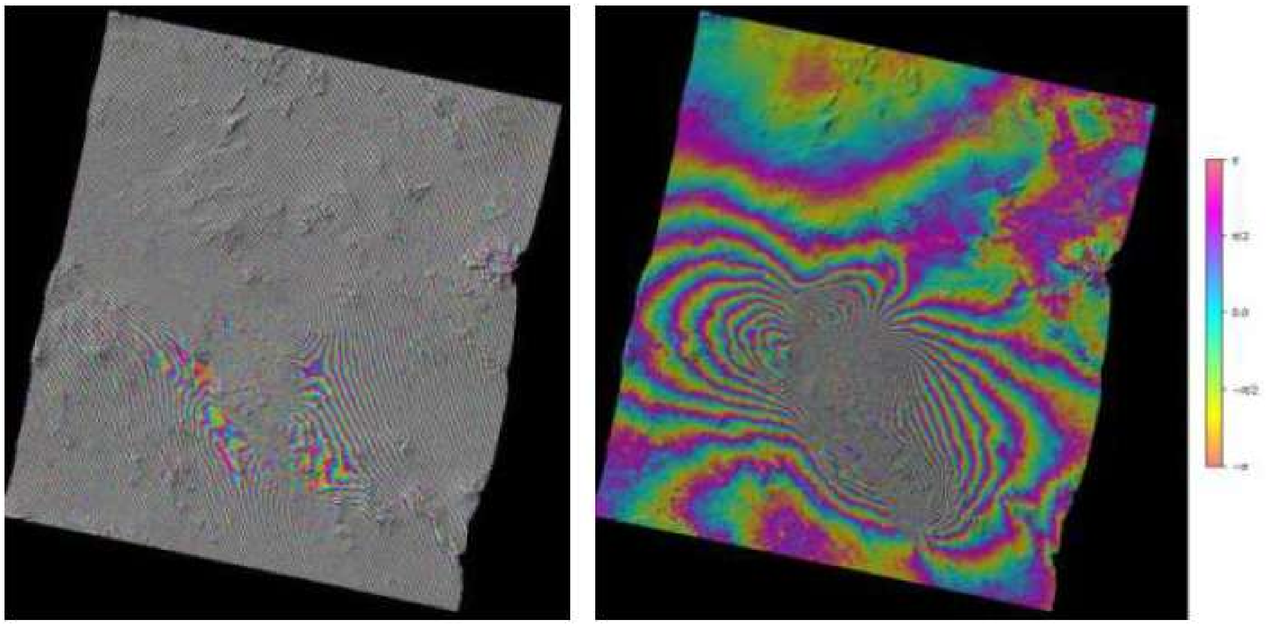

Fig. 6 Left: Geocoded interferogram showing the deformation caused by the Hector Mine earthquake produced using *SLC_intf_geo*. The interferogram includes the topographic phase. Right: Same interferogram after removal of topographic phase.

Python wrapper

The Gamma Software Python wrapper py_gamma permits a smooth usage of the Gamma Software tools and data formats within Python scripts as well as within a Python Interactive Development Environment (IDE) such as Spyder or PyCharm.

Using the wrapper:

- Binary images, point lists and data, parameter files, tab files, can be easily read, inspected, and written.
- Gamma Software program calls become Python function calls where variables can be used as function arguments, and system outputs can be stored in variables or written to log files.
- Usage within an interactive Python environment permits function name search and automatic completion.
- The documentation for each function can be easily accessed.

The Python wrapper user guide py_gamma_user_guide.pdf is available in the main directory of the Gamma Software installation. Information is also available from the main menu of the HTML-based Gamma Software documentation.

Program list

The main menu of the Gamma Software documentation contains a list of all programs available in the distribution (Gamma_documentation_contents_sidebar.html).

- Each program has its own short description (i.e. the usage) and a link to the detailed documentation.
- All programs can be easily searched in the search box. The search will also work when entering whichever parts of the program name, or text from the usage.
- Simply type *gamma_doc* in your terminal and check this feature!

The program list is generated automatically using a Python script. When needed, e.g. when Gamma Software packages are purchased separately, the program list can be regenerated using the Python script program_list.py found in the main folder of the Gamma Software distribution: in your terminal, go to the main directory of your Gamma Software installation and run the following command:

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

ASNARO-2 (Advanced Satellite with New system ARchitecture for Observation-2)

ASNARO-2 is a Japanese X-band SAR Earth imaging mission developed by the NEC Corporation and USEF (Institute for Unmanned Space Experiment Free Flyer), funded by NEDO (New Energy and Industrial Technology Development Organization), a Department of METI (Ministry of Economy, Trade and Industry) of the Government of Japan. The ASNARO-2 minisatellite was launched on 17 Jan. 2018. On 25 Sep. 2018: JEOSS (Japan EO Satellite Service) announced the start of commercial sales of ASNARO-2 imagery products. ASNOARO-2 supports stripmap mode (up to 2m resolution), spotlight mode (up to 1m resolution) and ScanSAR mode. Both SLC and detected data products are available.

The GAMMA programs *par_ASNARO2* and p*ar_ASNARO2_geo* are used to read the ASNARO-2 data. Example images are shown in Figures 7 and 8. For access to ASNARO data please see the JEOSS web site, http://jeoss.co.jp .

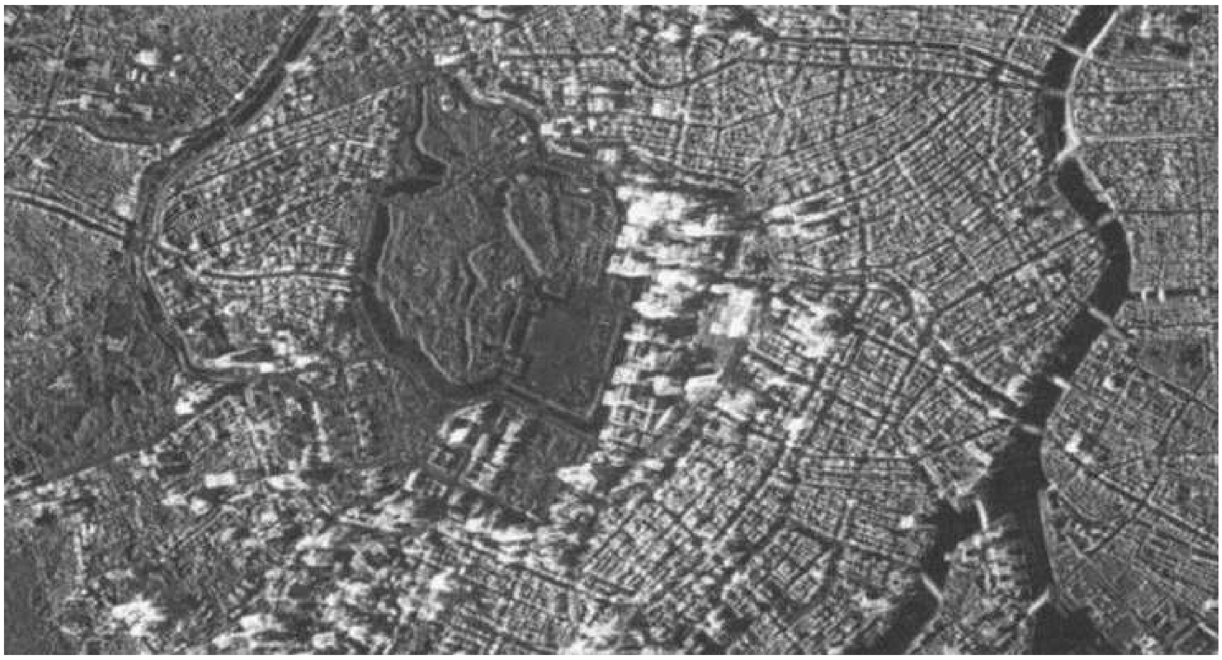
Fig. 7 Geocoded spotlight mode ASNARO-2 X-band SAR backscatter over a section Tokyo.

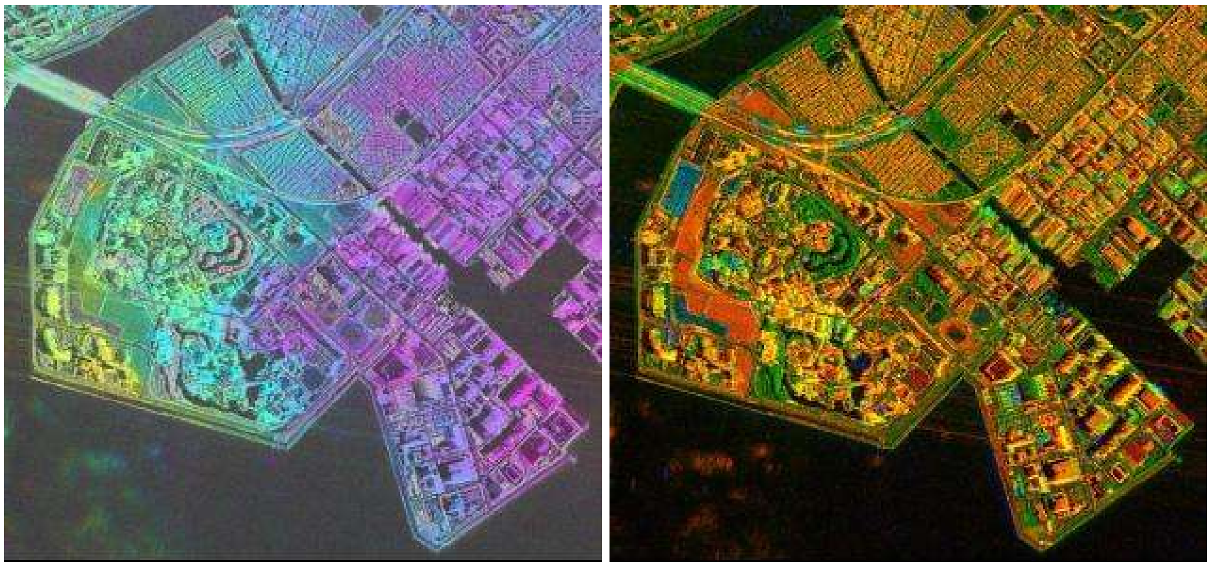
Fig. 8 ASNARO-2 X-band differential interferogram (left) and RGB composite of coherence, backscatter and backscatter change (right) over the Disneyland Tokyo area.

ICEYE X-band SAR small satellite constellation

Up to fall 2019 the ICEYE X-band SAR small satellite X1 to X5 have been launched by ICEYE Oy (www.iceye.com). Individual ICEYE SAR satellites form a constellation that enables frequent imaging revisit rates on a global scale. ICEYE's SAR satellite constellation provides different angle imaging multiple times a day for specified area of interest. Each satellite unit continues to introduce individual improvements on the system.

In the mid 2019 release the *par_ICEYE_SLC* program to read ICEYE SLC data was added. Now, in the Dec. 2019 release the program *par_ICEYE_GRD* is added, supporting the reading of detected images. : Added new program for generating SLC parameter and image files for ICEYE Small satellite X-band SAR by ICEYE Oy (www.iceye.com).

NovaSAR-1 S-band SAR

NovaSAR-1, launched on 16 Sep. 2018, is a joint technology demonstration initiative of SSTL (Surrey Satellite Technology Ltd.), UK, and Airbus DS (former EADS Astrium Ltd, Stevenage, UK), funded by the UK Government via the UKSA (UK Space Agency).

NovaSAR-1 provides medium resolution (6-30 m) imagery to support applications as flood monitoring, agricultural crop assessment, forest monitoring, land use mapping, disaster management, and maritime applications (e.g. ship detection, oil spill monitoring).

Data access to NovaSAR is through SSTL (Surrey Satellite Technology Ltd.). CSIRO has purchased a 10 per cent share of time on NovaSAR-1 (see also https://research.csiro.au/cceo/novasar/data/.

The GAMMA programs *par_NovaSAR_SLC* and *par_NovaSAR_GRD* were added to support the reading NovaSAR S-band SAR SLC and detected data products.

RCM, Radarsat Constellation Mission

The RADARSAT Constellation Mission (RCM) is Canada's new generation of Earth observation satellites. Launched on June 12, 2019, the three identical satellites, each with a C-band SAR, are operated in the same orbit. Spotlight, Stripmap and ScanSAR modes are supported. For information on RCM it is referred to the related page of the Canadian Space Agency: https://www.asc-csa.gc.ca/eng/satellites/radarsat/

The GAMMA programs *par_RCM_geo*, *par_RCM_GRC*, *par_RCM_GRD*, *par_RCM_SLC*, *par_RCM_SLC_ScanSAR* support reading the different RCM data products.

Gamma Software Demo examples

In this period again some Gamma Software Demo examples were added/modified. Gamma Software Users with a valid license or evaluation license can download the Gamma Software Demo examples here:

account: http://www.gamma-rs.ch/userdata/Gamma_Software_demo/README_Gamma_Software_demo.html
user: gamma_user
password: KLM_891

New demo example:	Contents
Gamma_demo_ASNARO2_Uluru.tar.gz	Demo example on the use of ASNARO2 X-band SAR data and the generation of a high-resolution DEM using multiple ASNARO2 interferometric pairs and the SRTM 3" DEM (see Figure 9).
Gamma_demo_DEM.tar.gz	This demo shows how to import DEMs into the Gamma Software. It includes examples for the widely used SRTM DEM, the newer 90m TDX DEM and a freely available DEM from New Zealand.
Gamma_demo_S1_coherence_estimation.tar.gz	An alternative coherence estimation path for S1 IWS data that estimates the coherence in the burst MLI geometry is demonstrated. The advantage of this approach is that it is not affected by phase jumps at burst or sub-swath interfaces (see Figure 10).

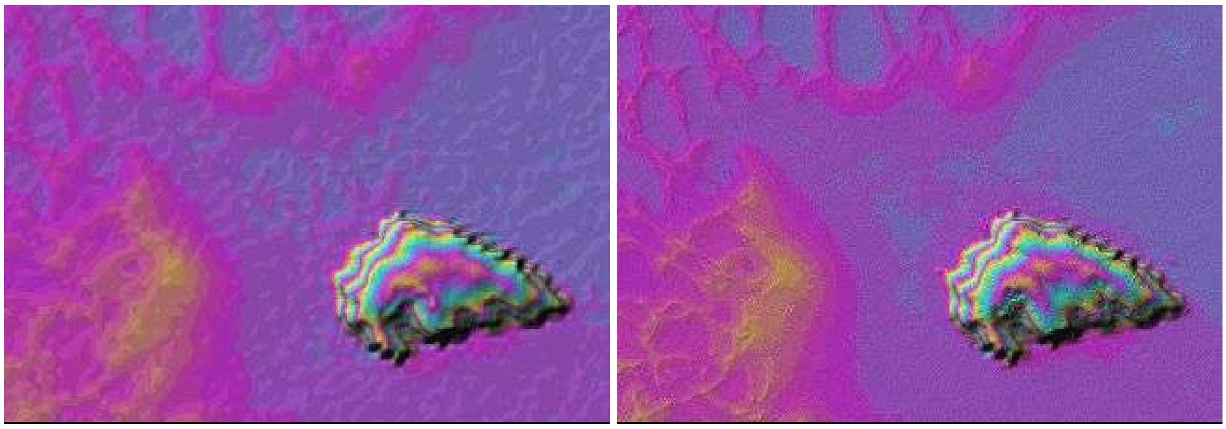

Fig. 9 High resolution DEM (right) over Uluru, Australia, generated using ASNARO2-X InSAR and the 3" SRTM DEM (left).

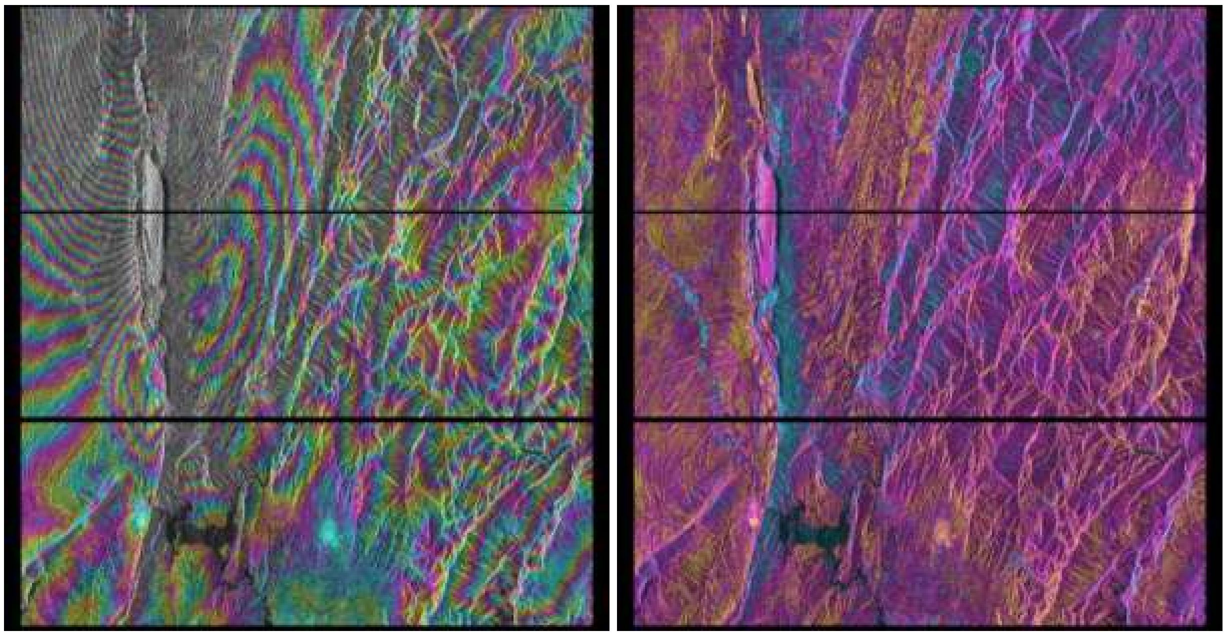

Fig. 10 Differential interferogram (left) and coherence estimate (right) estimated in the burst MLI geometry to avoid possible effects on the coherence estimate from phase jumps at the burst interface.

The following examples were updated to match the latest Gamma Software version:

Updated demo example:	Contents
Gamma_demo_S1_TOPS_Stripmap_InSAR.tar.gz	Demonstration of the generation of a Sentinel-1 IWS – to – stripmap mode cross-interferogram.
Gamma_demo_SCH_and_gc_map2.tar.gz	Geocoding demo for orbits clearly different from N-S direction using either SCH coordinates or the program gc_map2.
IPTA_demo_S1_Athens.tar.gz	IPTA processing example using S1 data over a small section to the West of Athens. The site includes small areas with fast, potentially non-uniform, motion (related to the compaction of landfills). It demonstrates: - IPTA processing for S1 data. - Combined use of single-pixel and multi-look phases. - Use of a multi-reference stack to map fast non-uniform motion and to optimize the spatial coverage achieved. - Alternative approaches to separate phase related to deformation and atmospheric path delay. Data consists of stack of co-registered SLC sections.
py_gamma_demo.tar.gz	Demonstration how the Gamma Python wrapper supports running the GAMMA Software within Python scripts as well as within a Python Interactive Development Environment (IDE) such as Spyder3.
S1_DevonIceCap_demo.tar.gz	Sentinel-1A offset tracking and split-beam interferometry (co-registration, differential interferogram, split-beam interferogram, offset tracking). Data consist of 2 sub-swaths. The example also shows the identification and effects of ionospheric effects on S1 TOPS data.
S1_Greenland_tracking_demo.tar.gz	Sentinel-1A offset tracking (co-registration, differential interferogram offset tracking, post-processing of offset field, geocoding of result). Data consist of 2 sub-swaths with 2 bursts each. The sequence includes an initial offset estimation followed by a second offset estimation using offset_pwr_tracking2 and post-processing of the results.
S1_Magdeburg_multitemp_demo.tar.gz	Demonstration of multi-temporal filtering of Sentinel-1 data. Processing starts from 24 co-registered SLC mosaics at VV and VH Pol.
S1_Mexico_coreg_demo.tar.gz	Sentinel-1A co-registration example (reading of SLC data, selection of bursts corresponding to the bursts present in the reference “burst SLC”, co-registration, calculation of differential interferogram, deramping, and split-beam interferogram generation).
S1_Mexico_INSAR_demo.tar.gz	Sentinel-1A DInSAR (co-registration, differential interferogram) for a pair over Mexico showing significant deformation.
S1_RFI_filtering.tar.gz	Demonstration of Radio Frequency Interference (RFI) filtering of a Sentinel-1 IWS SLC.
S1A_S1B_DINSAR_ItalyEarthquake_demo.tar.gz	Sentinel-1A - Sentinel-1B DInSAR (co-registration, differential interferogram, unwrapping) for co-seismic pair.
TDX_demo_Etna.tar.gz	Generation of a DEM from a pair of TanDEM-X resampled slc data with support of the SRTM DEM for geocoding and phase unwrapping.

MSP

-

ISP

init_offset: Updated to use similar method as in offset estimation programs.

par_NovaSAR_SLC: New program for generating SLC parameter and image files for NovaSAR SLC data.

par_NovaSAR_GRD: New program for generating MLI and GRD image and parameter files for NovaSAR GRD and SCD data.

par_ASNARO2: New program for generating SLC parameter and image files for ASNARO-2 Spotlight, Stripmap and ScanSAR level 1.1 data. ASNARO2 is a commercial Japanese X-band Satellite.

par_ASNARO2_geo: New script for generating DEM parameter and image files for ASNARO-2 geocoded and georeferenced level 1.5 data in GeoTIFF format.

par_S1_SLC: Updated to support the case where burst data and sensing date of the first burst fall before and after midnight. Now also best aligns the Doppler and azimuth FM rate polynomial times to the burst times when the provided number of polynomials is larger than the number of bursts.

ScanSAR_burst_overlap: New program for extracting and mosaicing overlapping parts of ScanSAR / TOPS burst data.

multi_look2: Calculate multi-look intensity image with independently specified decimation factors and averaging window dimensions. Also supports the optional application of a Kaiser window weighting in addition to uniform rectangular weighting.

SLC_intf2: Added program to calculate interferogram, coherence map, and intensity images using independently specified decimation factors and averaging window dimensions. Also supports the optional application of a Kaiser window weighting in addition to uniform rectangular weighting.

par_S1_GRD: Interpolation from ground range to slant range geometry is now performed using a bicubic B-spline interpolation.

par_S1_GRD: Data processed before Jan-2018 with Sentinel-1 IPF version < 2.90 show low, non-zero values at the edges. A new option was implemented for cleaning the edges. It extracts and analyzes the image edges using a Canny filter. Only vertical and horizontal edges are kept; among these, the most intense and longest edges are used to set the new image margins. Data outside the margins are set to 0.0.

par_S1_SLC, *par_S1_GRD*: The calibration and noise look-up tables are now interpolated separately in range and azimuth directions, to solve the issue of non-constant range samples in the LUTs. Errors in the noise look-up tables (negative or low values) are now detected and replaced by extrapolated values.

multi_look_MLI: A new option (flag) was added for including or excluding pixels without the full number of looks with valid data at the edges of the multi-looked image.

par_ICEYE_SLC: Adapted for some small format changes in ICEYE SLC Metadata.

par_ICEYE_GRD: New program for generating MLI and GRD image and parameter files for ICEYE GRD data.

par_RCM_GRC, par_RCM_GRD, par_RCM_SLC, par_RCM_SLC_ScanSAR: Added new programs for generating Gamma Software image and parameter files for RCM (Radarsat Constellation Mission) GRC (ground-rang complex), GRD (ground-range detected), SLC and ScanSAR data. For the ScanSAR data the same parameter files as for Sentinel-1 are used (SLC files per sub-swath, SLC_par files, “TOPS_par” files, and an SLC_tab file listing all these files of an acquisition).

par_RCM_geo: New program for generating DEM parameter and image files from an RCM (Radarsat Constellation Mission) GCD (geocoded detected) product.

RCM_ORB_filt.py: Added new script for filtering state vectors from early RCM products (necessary to avoid anomalies in the simulated interferometric phase).

DIFF&GEO

Stacking: Stacking has no limit anymore in the number of interferograms it can handle.

init_offsetm: Updated to use similar method as in offset estimation programs.

create_dem_par: Now includes a non-interactive mode.

dem_import: When using an existing DEM parameter file containing the extent of the output file, if the input file is a GeoTIFF/GDAL-supported file, *dem_import* will only read the useful area of the input file. Furthermore, the output DEM in binary format is now optional.

srtm2dem, *utm2dem.pl*, *vrt2dem*, *vrt2dem_latlon*: The *srtm2dem, utm2dem.pl, vrt2dem, vrt2dem_latlon* Perl scripts no longer internally rely on third-party programs (such as gdalwarp, gdal_translate) but use only Gamma Software programs (in particular *dem_import).*

MLI_interp_lt, lk_vec_lt, sarpix_coord, sarpix_coord_list: Corrected the calculation of the azimuth image time when using the multi-look azimuth line time parameter in the MLI parameter file.

gc_map2: The program should now run significantly faster. As in gc_map, gc_map2 now also includes the [sim_sar], [u], [v], [psi], and [pix] options. Masking now has an additional option: "masking shadow and values outside swath". A new [az_dec] option was added for speeding up layover-and-shadow map calculation. The default range oversampling and azimuth decimation factors are now automatically calculated for optimum quality/processing time ratio.

gc_map → *gc_map1:* Considering that *gc_map2* provides a better result than *gc_map* and that the processing time is now similar between the two programs it is our clear recommendation to always use *gc_map2*. To make most users follow this recommendation we added a script named *gc_map* to call *gc_map2*. The old program *gc_map* has been renamed to *gc_map1* and is considered deprecated. Hence using *gc_map* or *gc_map2* (recommended) will both run the newer program *gc_map2*. But the old program *gc_map* is still available as *gc_map1*.

geocode_back: A new option (flag) was added for extrapolating (or not) the coverage up to 0.5 pixels outside the extent of the input data.

DISP

visras.py: Updated to support GeoTIFF files, displays map/geographical coordinates.

DISP/cmaps, mpl_gallery.py mpl_cmaps.py: Updated colormaps to support Matplotlib 3: Added the following colormaps: *wistia.cm binary.cm cividis.cm gist_gray.cm gist_yarg.cm, tab10.cm tab20.cm tab20b.cm tab20c.cm turbo.cm twilight.cm twilight_shifted.cm* and added programs *mpl_gallery.py* and *mpl_cmaps.py* to display swatches and create the text-based color maps used by programs in the gamma software.

DISP/cmaps cmocean, colorcet, turbo: Added directories *cmocean*, and *colorcet* to the *cmaps* directory that contain colormaps and swatches of colormaps in the *colorcet* and *cmocean* collection. Images of the colormaps with swatches are available, and referenced in the documentation:
cmaps/cmocean.cmocean_gallery.png
cmaps/colorcet/colorcet_swatches.png

Added the *turbo.cm* colormap in the *cmap* directory to use as a possible replacement for *jet.cm*. The turbo colormap is available only in text format (turbo.cm). Colormaps in the different packages can called named colormaps and can be referenced using the name alone (without the .cm extension). Otherwise all colormaps can be called using the text format filename with the .cm extension (rmg.cm, hls.cm...). Text format colormap names are required for all compiled programs that permit specification of the colormap, including *rasdt_cmap*, *rasdt_cmap_pt*, *and ras_data_pt*.

LAT

MLI_ovr: The resampling method was updated better handled the data edges.

edge_detection: New program for detecting edges in an image using a Canny edge detector and extracting line segments.

IPTA

triangle (libraries triangle.h): Triangle library was modified to support large datasets in *mcf_pt*, *fspf_unw_pt*, and *mcf* (ISP). Also involved minor changes in *pt2d*, *pt2data*, and *ras_triangle*.

xpt_slc, *def_mod_pt*: Initialize variables to 0.0 that were not initialized.

pwr_stat: Updated to also support MLI data (intensity, FLOAT type) as input.

fspf_pt, fspf_unw_pt: Increased the extent of the filtered array to avoid discarding points at the right and bottom edges of the data.

Python wrapper

py_gamma.py: Now uses *collections.deque* library instead of *queue.Queue* for handling stdout and stderr messages. This change makes the displaying of Gamma Software programs outputs on the console considerably faster.