Release Notes GAMMA Software, 20200709

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-Jul-2020

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. There is also a version available for Ubuntu 20.04 LTS.

Announcement: Ubuntu 18.04 LTS will no longer be supported after the mid-2021 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 Mojave (10.14.6) and Catalina (10.15.5). You will need to install libraries such as GDAL using MacPorts. The build uses the MINGW64 GCC 9 compiler.

Announcement: The present upgrade is the last upgrade for MacOS Mojave (10.14.6). MacOS Catalina (10.15) will no longer be supported after the December 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 10 version of the Gamma software is compiled with 64-bit support and multi-threaded. The software is expected to also run on Windows 7, and 8. The build uses the MINGW64 GCC 9 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.

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
```

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 fall 2020 and/or spring 2021. See also our web-site under http://www.gamma-rs.ch/courses/training-courses.html.

Significant Changes in the Gamma Software Modules since the End of 2019 Release

ICEYE X-band update (DInSAR)

In the mean time we were able to test ICEYE differential interferometry. Results from an 18-day repeat-interval pair acquired by ICEYE-X2 in Spotlight mode are shown in Figure 1. To get approximately square pixels we applied a multi-looking with 3 range and 4 azimuth looks. The SRTM 3” DEM was used for the geocoding and to calculate the topographic phase. The state vectors were found of good quality (confirmed by small geocoding offset refinement and a reasonably flat differential interferogram).

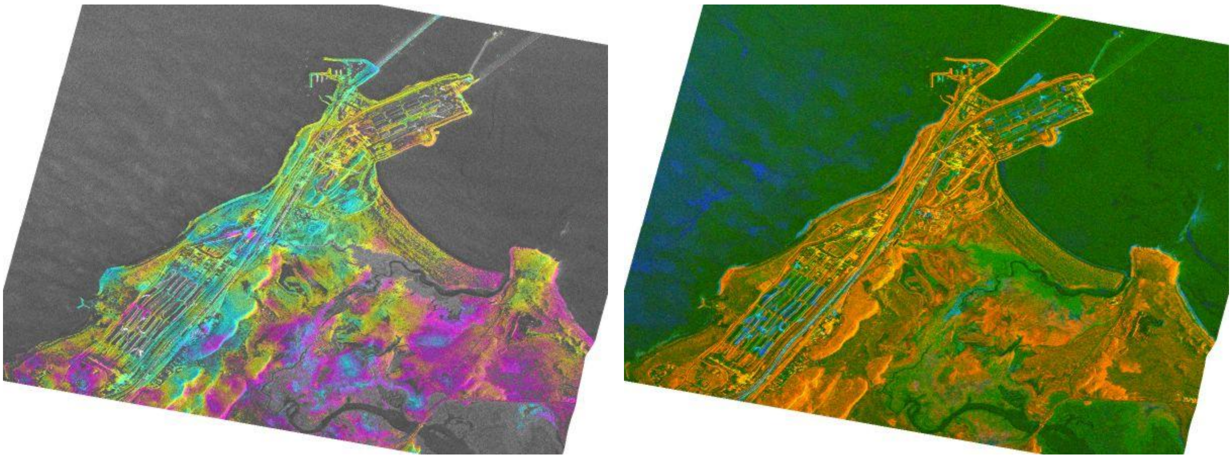


Fig. 1 ICEYE-X2 Spotlight Mode differential interferogram 20200405-20200423 over Wickham, Australia (left, one color cycle corresponds to one phase cycle), and related RGB composite of the coherence (using a linear gray scale), the backscatter of the reference (using a dB gray scale) and the backscatter change (absolute value of ratio, using a dB gray scale) (right). Yellow/orange areas have high coherence and low backscatter change between the first and second date.

SAOCOM 1 (Satélite Argentino de Observación COn Microondas)

SAOCOM 1A and 1B are two L-band SAR satellites of Argentina's space agency CONAE. SAOCOM 1A was launched in October 2018 while the launch of SAOCOM 1B is now planned for the end of July 2020. SAOCOM 1 satellites are operated jointly with the Italian X-band COSMO-SkyMed constellation. SAOCOM 1 operates in stripmap and TOPS mode, and can provide fully polarimetric data. Stripmap data can be used for interferometric processing; currently TOPS data are not synchronized and therefore not intended to be used for interferometry. SLC and geocoded data products are available.

The GAMMA programs *par_SAOCOM_SLC* and *par_SAOCOM_geo* are used to read SAOCOM 1 data. Example images are shown in Fig. 2. For access to SAOCOM test data please see the CONAE web site: https://catalogos.conae.gov.ar/catalogo/catalogosatsaocomadel.html

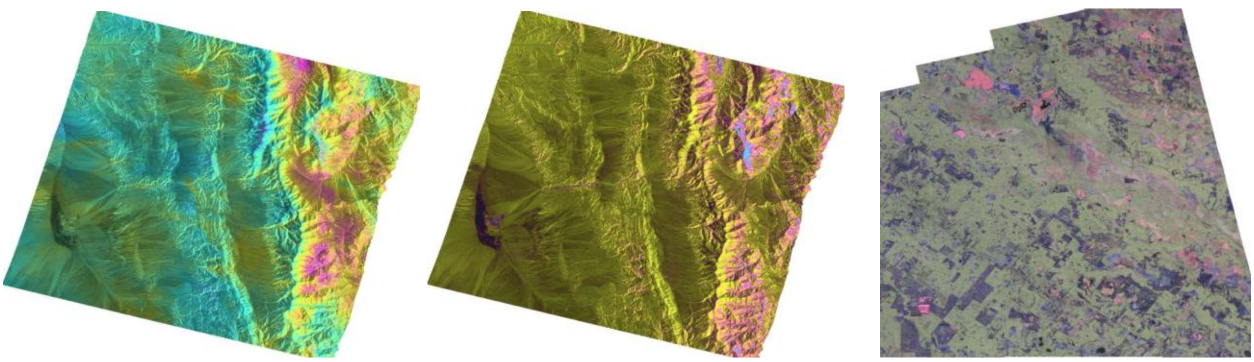


Fig. 2 SAOCOM 1A L-band stripmap differential interferogram (left), related coherence (center) and Pauli decomposition using SAOCOM L-band polarimetric TOPS mode data (right).

RCM (Radarsat Constellation Mission)

The GAMMA programs for reading RCM data that were introduced in the previous release were tested using actual data (see Fig. 3) and updated. In some RCM ScanSAR modes, the data include large overlaps between consecutive bursts so that every pixel in a mosaic may be picked from two or more bursts (except at the beginning and the end of the subswaths); using the program *ScanSAR_burst_to_mosaic*, a mosaic can be generated using averaging for the overlap areas. In that program, new options were added to set the minimum and maximum number of overlapping bursts.

For access to RCM test data please see the Canadian Space Agency FTP site:
ftp://ftp.asc-csa.gc.ca/users/OpenData_DonneesOuvertes/pub/RCM/

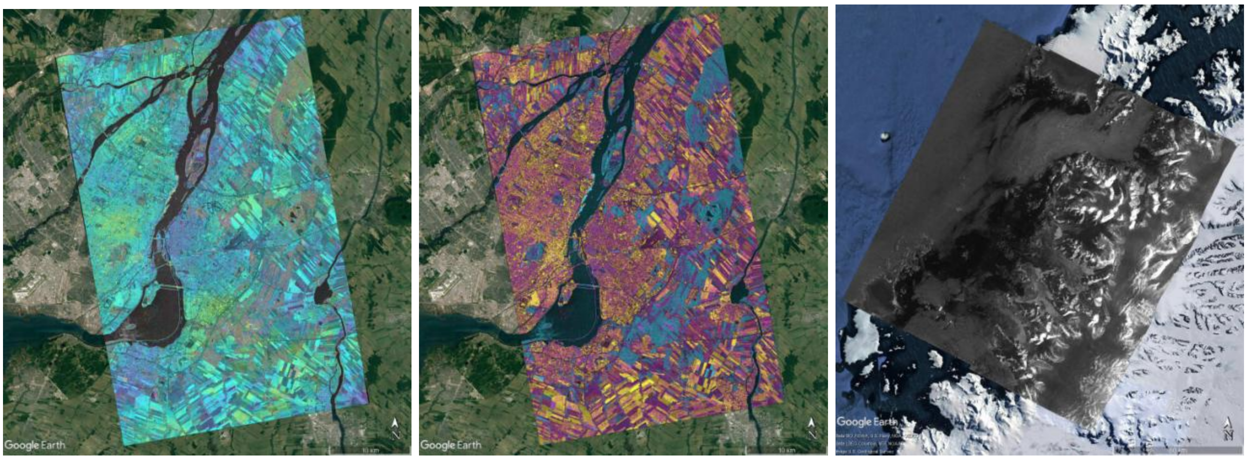
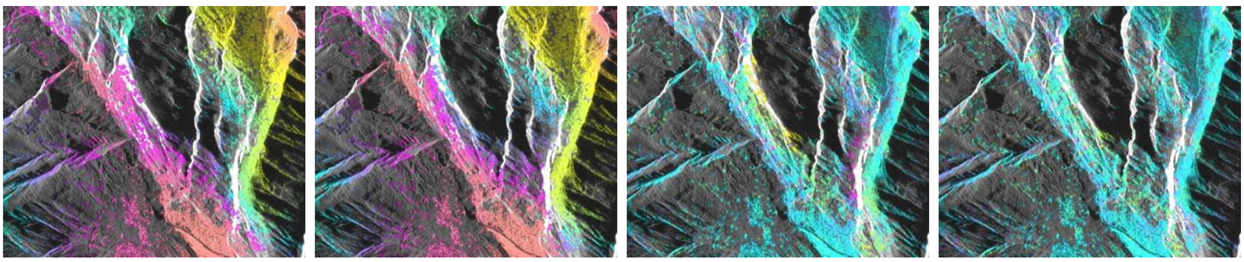


Fig. 3 RCM stripmap differential interferogram (left) and related coherence (center) over Montreal, Canada, and a ScanSAR mosaic (right) over the Antarctic peninsula.

Filtering, interpolating and unwrapping point data in map geometry

In mountainous areas, radar coordinates place scatterers that are actually widely separated in altitude in very close proximity in terms of slant-range. Concurrently, atmospheric phase distortions typically have a strong dependency with altitude. The combination of these two effects impairs spatial filtering, spatial interpolation and phase unwrapping. For data acquired over mountainous areas, significant improvements are observed when performing those operations in a map geometry (see spatial filtering results in Fig. 4), because the distances between points are closer to reality. Their use results in a more robust and reliable processing chain, and is therefore strongly recommended.



a) filtering in SAR geometry
b) filtering in map geometry
c) filtering in SAR geometry, difference with unfiltered data
d) filtering in map geometry, difference with unfiltered data

Fig. 4 For a small subset of the point data from the Aletsch example, results of the filtering of the atmospheric phase for one layer are shown when performed in SAR geometry (a) and in map geometry (b), as well as the respective difference with the unfiltered data (c, d). The color scale is cyclical over 4π. The results are shown here in SAR slant range geometry.

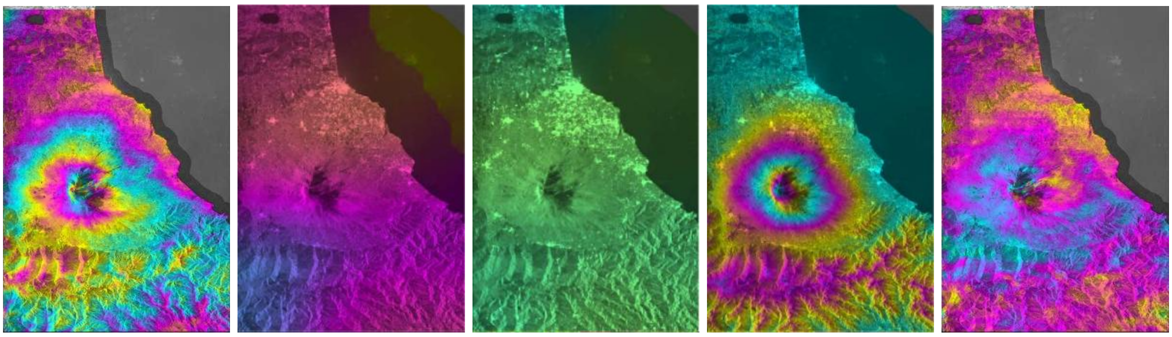
The GAMMA programs *spf_pt*, *fspf_pt*, *expand_data_pt*, *expand_data_inpaint_pt*, and *mcf_pt* now support both radar and map geometries. A new version of *multi_def_pt* called *multi_def_geo_pt* was also added and uses patches defined in map coordinates.

The Sentinel-1 IPTA demo example over the Aletsch area demonstrates these new functionalities.

Spatially adaptive atmospheric path delay estimation

Before this upgrade the height dependent atmospheric path delay in an interferogram was mainly estimated using *atm_mod* (and *atm_mod_pt* for files in IPTA vector data format). The model used a single set of parameters (phase intercept and slope of the linear height dependence) for the entire scene. Now atm_mod_2d was added to do the same with spatially varying model parameters. With *atm_mod_2d* the model parameters are estimated based on the unwrapped differential interferometric phase and the DEM height (both in the same geometry) using a spatial window. The estimation window is typically quite large (e.g. 40km) to generate model parameters that vary spatially only slowly. As an intermediate step the model parameters can be post-processed (rejecting outliers, applying filtering and interpolation etc.). Then the model parameters are used to calculate the height dependent path delay phase using *atm_sim_2d*. Figures from the related new demo example are shown in Fig. 5.

It is also possible to estimate the model parameters from a band-pass filtered differential interferogram using the corresponding band-pass filtered DEM heights. In the band-pass filtered differential interferogram very large-scale phase variations and overall phase slopes are no longer present and overall, the phases (and height variations) are smaller and therefore unwrapping is simpler.



(a) (b) (c) (d) (e)

Fig. 5 Height dependent atmospheric path delay estimation for the PALSAR differential interferogram 20080616-20080916 over the Etna Volcano, Italy. (a) shows the differential interferogram, (b) the constant term a0, (c) the slope term a1, (d) the path delay calculated based on the slope term only and (e) the corrected differential interferogram obtained by subtracting (d) from (a).

ScanSAR and TOPS co-registration update

New Python scripts supporting co-registration of both ScanSAR and TOPS data are replacing the CSH scripts that used to support co-registration of Sentinel-1 TOPS data. The co-registration algorithm is the same as before, but was reimplemented to work for any TOPS or ScanSAR system and to be more user-friendly.

Main features of the new Python scripts:

- Bursts from the secondary scene are automatically adapted to match the reference, i.e. bursts in excess are removed and empty burst are added where missing.
- Spectral diversity method converges faster.

- A maximum number of iterations can be specified for each refinement method, in most cases 1 iteration for each step should be sufficient.
- Options have a --*keyword value* pattern, simplifying the usage.
- Several new options for improved supervision of the co-registration steps.
- Improved error handling.
- Improved handling of temporary files.
- Optimized for faster processing.

The table below summarizes the changes:

New script	Replacement for	Functionality
ScanSAR_coreg.py	*S1_coreg_TOPS* *S1_coreg_TOPS_lt1* *S1_coreg_TOPS_no_refinement* *ScanSAR_coreg* *ScanSAR_coreg_no_refinement*	Co-register a ScanSAR or TOPS mode burst SLC to a reference burst SLC.
ScanSAR_coreg_pol.py	*S1_coreg_TOPS_dual_pol*	Co-register polarimetric ScanSAR or TOPS mode burst SLCs to reference burst SLCs.
ScanSAR_coreg_check.py		Check and adapt a ScanSAR or TOPS burst SLC relative to a reference burst SLC.
ScanSAR_coreg_overlap,py	*S1_coreg_overlap*	Determine an azimuth co-registration offset in ScanSAR or TOPS data based on the burst overlap

Note that calling the old syntax still work as before, however the CSH scripts are now simple wrappers that call the new Python scripts. Also note that the Python wrapper *py_gamma.py* is used by the Python scripts: its location must be found by Python and the required libraries must be properly installed. This may require adding the following line to the *.bashrc* file:

```
export PYTHONPATH=.:$GAMMA_HOME:$PYTHONPATH
```

For more information, refer to *py_gamma* instructions available in the software documentation.

InSAR programs for map geometry

When focusing data with the Time Domain Back Projection (TDBP) processor an SLC in map geometry is generated. To support multi-looking, interferogram generation, and offset map estimation in map geometry a range of programs have been added.

Program	Functionality
multi_look_geo, multi_look_geo2	Generation of a multi-look intensity image based on an SLC in map geometry. In *multi_look_geo2* different multi-looking and sampling decimation factors can be selected.
SLC_intf_geo, SLC_intf_geo2	Generation of an interferogram based on an SLC in map geometry. In *SLC_intf_geo2* different multi-looking and sampling decimation factors can be selected.
offset_pwr_geo, offset_pwr_tracking_geo	Generation of an offset map based on an SLC in map geometry. The two programs differ in the way the location of the offset estimates is indicated.

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_DEM.tar.gz	Demo showing how to import DEMs into the Gamma Software. An example using the Copernicus DEM was added to the demo.
Gamma_demo_ICEYE.tar.gz	Demo example on the handling of ICEYE X-band SAR data for different acquisition modes and product levels.
Gamma_demo_RCM.tar.gz	Demo showing how to read Radarsat Constellation Mission (RCM) data for different acquisition modes and processing levels. (Fig. 3)
Gamma_demo_SAOCOM.tar.gz	Demo example on the handling of SAOCOM L-band SAR data for different acquisition modes and product levels and SAOCOM L-band InSAR and polarimetry. (Fig. 2)
PALSAR_Etna_atm.tar.gz	Demo on the estimation of a height dependent atmospheric path delay correction using spatially varying model parameters (programs *atm_mod_2d*, *atm_sim_2d*) (Fig. 5).
S1_Yibal_InSAR_demo.tar.gz	Demo introducing SAR interferometry with Sentinel-1 data. Starts from the download, extraction and burst selection of Sentinel-1 data and includes the steps leading to the generation of a differential interferogram and a coherence map. The co-registration is performed using the new *ScanSAR_coreg.py* script. (Fig. 6)
IPTA_demo_Aletsch_from_orig.tar.gz IPTA_demo_Aletsch_from_rslc.tar.gz IPTA_demo_Aletsch_all_single_look_test.tar.gz	S1 IPTA demo example over the Aletsch area in the Swiss Alps. The demo shows processing approaches using a multi-reference stack with short time intervals suited to map relatively fast and non-uniform deformation using single pixel and multi-looked phases. Alternatives starting from the original zip files of the co-registered rslc are provided, as well as a documented test using all single look phases as candidates (Fig. 7).

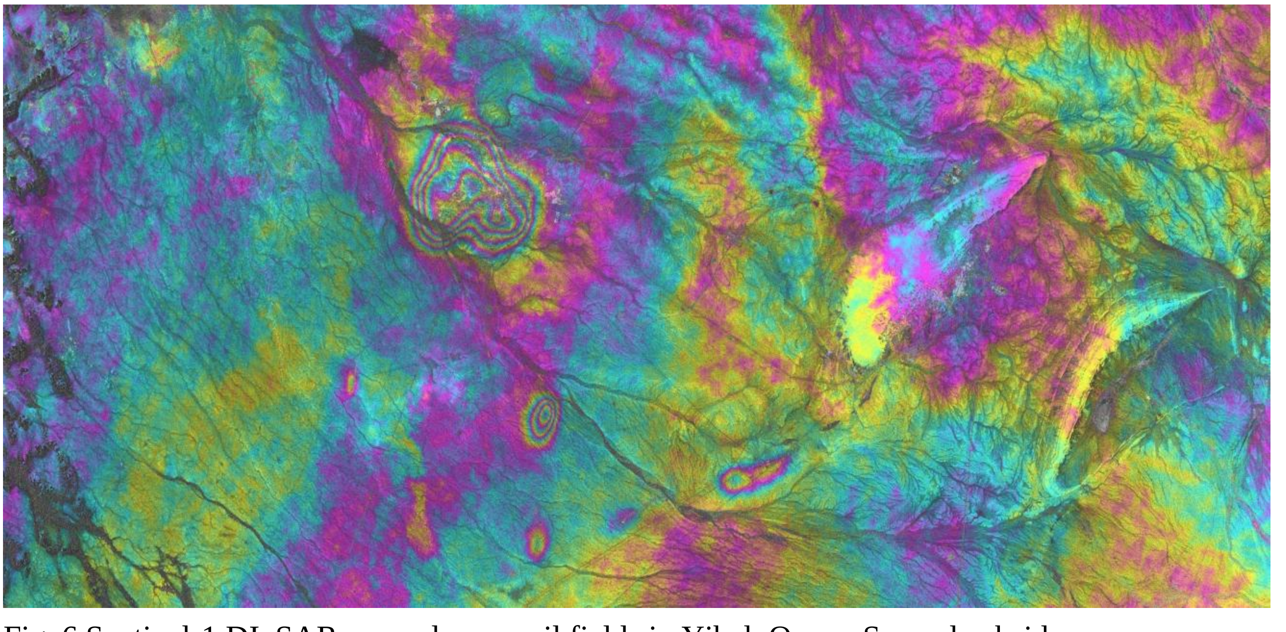
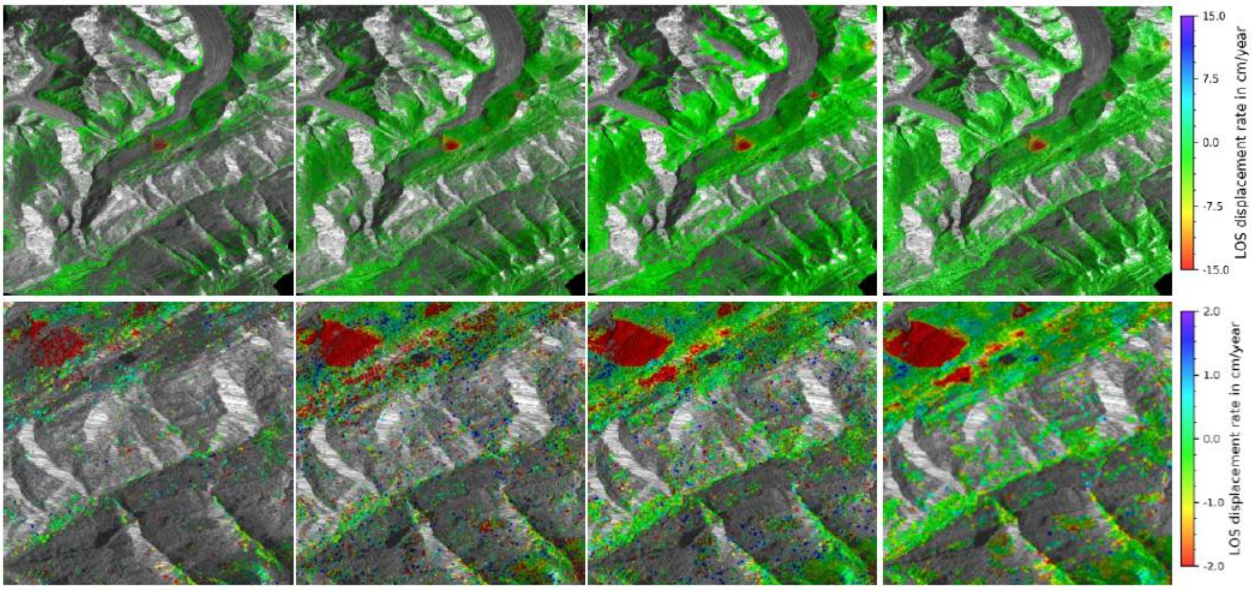


Fig. 6 Sentinel-1 DInSAR example over oil fields in Yibal, Oman. Several subsidence areas can be observed on the produced differential interferogram (2-years temporal baseline).



a) *Single-look phase* elements of initial solution. b) *Multi-look phase* elements of initial solution. c) Solution using all *single-look phase* elements without spatial filtering of phase time series. d) Solution using all *single-look phase* elements with spatial filtering of phase time series.

Fig. 7 Linear deformation rate results for different processing approaches shown in the top row for a larger section using a color scale between -15cm/year and +15cm/year and in the bottom row for a smaller section using a color scale between -2cm/year and +2cm/year.

MSP

-

ISP

par_S1_GRD, *par_S1_SLC*: Updated. The noise calculation now also considers the noise azimuth vectors (available in the noise XML file since S1 processor version IPF 2.90). The calibration and noise LUT interpolation is now more robust in the case of missing data.

adf2: New program for adaptive interferogram filtering (similar to *adf*) with an adaptive Goldstein/Werner type spectral filter. Unlike *adf* the filter parameters are not just adaptive to the power spectrum, but also to the local coherence estimate (with further reduced filtering for low coherence areas).

RCM_ORB_filt.py: Updated. *RCM_ORB_filt.py* now checks if the state vector interval is large. In that case, state vectors are supposed to be accurate and the program doesn't perform the state vector filtering except if the new [force] flag is activated.

par_SAOCOM_SLC: New program for generating SLC parameter and image files for SAOCOM stripmap and TOPS mode SLC data.

ScanSAR_burst_to_mosaic: Updated. New options [min_ovr] and [max_ovr] for setting the minimum and maximum number of overlapping bursts were added.

ave_cpx: The program *ave_cpx* was moved from the LAT to the ISP. It is used for stacking SLC images (in particular SLC data of the Gamma Portable Radar Interferometer (GPRI) acquired with short time intervals.

ScanSAR_burst_overlap: New option for writing polygons encompassing the overlap areas in SLC or MLI mosaics.

offset_pwr_tracking_polygons: New program to calculate polygon vertices of patches used for offset estimation. These polygons can be used in the MLI geometry to mask the lookup table used to resample SLCs. The LAT program poly_mask can be used to mask the lookup table. Resampling only the areas of the SLCs covered by the patches can speed up iterative coregistration. The generated polygons can be visualized superposed to a rasterfile using the LAT program *drawthat*.

par_S1_SLC: Updated. Now saving all state vectors available in the metadata rather than those that are 15 sec before the data start and 15 sec after the end of the data.

SLC_intf2: Updated to calculate the interferogram, correlation map, and MLI images using the same weighting function. Different weighting functions that can be selected are rectangular, Kaiser, and circular Gaussian windows. Added the capability to subtract simulated interferometric phase for calculation of a differential interferograms. The user can independently select the range and azimuth decimation factors and the smoothing parameters on the command line. Oversampling of the SLC data is only performed when required or specified.

DIFF&GEO

coord_to_sarpix: Add output of the azimuth time of the pixel in HH:MM:SS.sssss UTC

map_section: Added option to set the width and number of lines in the new lookup table to be specified on the command line. When specified, these values override values determined from the corner coordinates.

ScanSAR_coreg_check.py: New script to check and adapt a ScanSAR or TOPS burst SLC relative to a reference burst SLC. Empty bursts are added where required.

ScanSAR_coreg_overlap.py: New script to determine an azimuth coregistration offset in ScanSAR or TOPS data using spectral diversity method, based on phase of burst overlap double-difference interferograms. For Sentinel-1 *ScanSAR_coreg_overlap.py* replaces *S1_coreg_overlap*.

ScanSAR_coreg.py: New script to coregister a ScanSAR or TOPS mode burst SLC to a reference burst SLC by iterative application of intensity matching and "spectral diversity" methods. Sentinel-1 *ScanSAR_coreg.py* replaces *S1_coreg_TOPS.* Also replaces *ScanSAR_coreg_no_refinement* and *S1_coreg_TOPS_no_refinement* using the appropriate options.

par_SAOCOM_geo: New script for generating DEM parameter and image files for SAOCOM geocoded data in GeoTIFF format (GEC and GTC / level 1C and 1D data).

multi_look_geo: Added capability to perform multi-look processing on FLOAT format data.

dem_import: An error when using a global geoid was corrected. This error was causing up to 0.5 pixel offsets in the geoid in the longitude direction and may have caused some segmentation faults in some systems.

atm_mod_2d, *atm_sim_2d*: New programs to estimate and simulate height dependent atmospheric phase trends using either a linear or exponential height dependence. The image is divided into overlapping patches and an estimate of the phase parameters is estimated for each patch. The phase is simulated using a DEM and the interpolated values of the model parameters. We also added the capability to use SVD fit to estimate parameters in the atmosphere model.

lk_vec_lt: Improved documentation in cmd line help and html help text.

ScanSAR_coreg_pol.py: New script to coregister polarimetric ScanSAR or TOPS mode burst SLCs to reference burst SLCs.

S1_coreg_overlap, S1_coreg_TOPS, S1_coreg_TOPS_no_refinement, S1_coreg_TOPS_lt1, S1_coreg_TOPS_dual_pol: Now only interfaces, calling *ScanSAR_coreg.py ScanSAR_coreg_overlap.py*, and *ScanSAR_coreg_pol.py* for the effective processing. We recommend to use directly the new programs to also benefit from the additional options available.

ls_map_mask: New program to generate a mask rasterfile from a layover and shadow map.

multi_look_geo2: Added new program that performs multi-look intensity image calculation for FCOMPLEX and SCOMPLEX images in DEM geometry that separates the decimation and window size parameters. Different weighting functions applicable are rectangular, Kaiser, and circular Gaussian windows. The user can specify if the data will be oversampled by a factor of 2 prior to detection.

SLC_intf_geo2: Added new program that performs interferogram image calculation for coregistered FCOMPLEX and SCOMPLEX SLC images in DEM geometry that separates the decimation and window size parameters. The averaging window can be specified as either rectangular, kaiser, or circular gaussian. If simulated topographic and orbital phase are provided on the command line, this will be subtracted to permit calculation of a differential interferogram. The program calculates the interferogram, correlation map and MLI images using the same weighting function. Oversampling of the SLC occurs only if necessary or specified by the user on the command line.

DISP

create_array: Updated to support additional output file types: UNSIGNED CHAR and SUN/BMP/TIFF 8-bit grayscale raster image.

cpx_math: Added option to calculate conjugate of input data. Added multiplication by the conjugate of a second scene: d1*conj(d2). Updated documentation for the different modes for normalization by average of data in the reference region.

vis_colorbar.py: Added options to plot minor ticks and add a label to the colorbar.

vismph_pwr.py: Added colorbar title, and minor ticks to the scale. Added display of the background intensity image in areas with no phase data.

visdt_pwr.py, *vispwr.py*, *viscpx.py*: Added colorbar label, and minor ticks to the scale.

dispwr, dis2pwr: Added calculation and display (as part of the screen output) of the average image intensities in dB.

visbyte.py, *vismph_pwr.py*, *visdt_pwr.py*, *vispwr.py*, *viscpx.py*: New option -q for quantizing output file to 8-bit instead of 24-bit when using option -u.

dis2ras: Now supports simultaneous display of 24- and 8-bit/pixel raster images (previously, both images were required to have the same number of bits/pixel).

LAT

poly_mask: New program to mask data using polygonal image regions. Supports different data formats, allowing for example to mask lookup table (FCOMPLEX) or rasterfiles.

IPTA

IPTA_users_guide.pdf: The IPTA users guide has been substantially updated.

disp_plot_pt, *vu_disp*: Added [zero_flag] option for interpreting 0.0 values either as no data values or as valid values.

mcf_pt: Minimum Cost Flow phase unwrapping for point data stack can now be performed in MAP coordinates for improved reliability in case of data acquired over mountainous areas and steep topography. Various weighting models are now available. Points with identical coordinates are all unwrapped individually.

ts_rate_pt: Added program to calculate a moving estimate of deformation rate from a deformation time-series using least-squares linear fit for each point. The width of the time window can be specified, or the maximum number of time-series values in the window.

fspf_pt, *spf_pt*: Now also support point data in map coordinates. Filtering in map coordinates can be very helpful for data acquired over mountainous areas / areas with steep topography.

expand_data_pt, *expand_data_inpaint_pt*: Now also support point data in map coordinates. Resampling in map coordinates can be very helpful for data acquired over mountainous areas / areas with steep topography.

multi_def_geo_pt: New version of *multi_def_pt* that performs the patch selection in map coordinates instead of slant-range/azimuth coordinates.

phase_sim_orb_pt: New option for selecting the simulated phase model: standard model (unflattened, default), or height + deformation phase, relative to ellipsoid (flattened).

Python wrapper

-