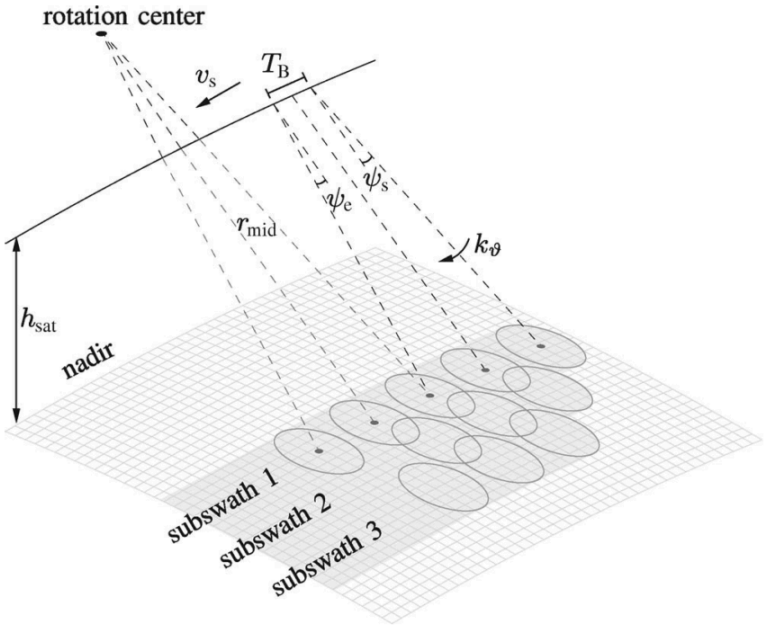
# Overview of S1A TOPS processing with ISCE – topsApp.py

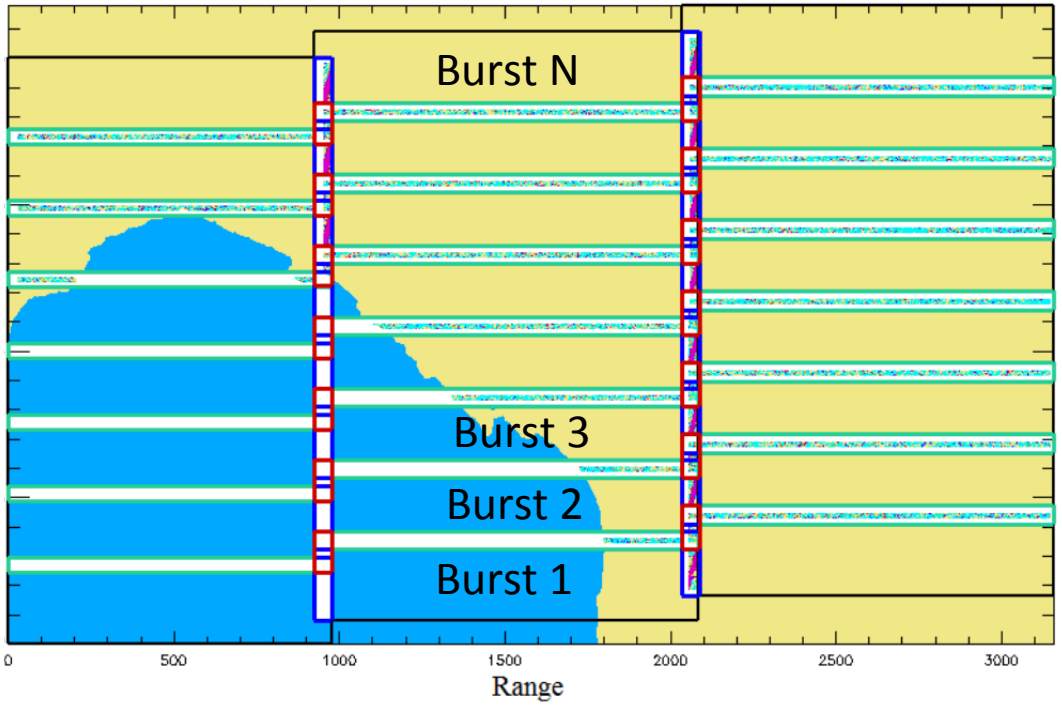
ISCE team
JPL/Caltech

# Sentinel-1A

## TOPS




[Prats-Iraola et al, 2012]




Swath1 Swath2 Swath3

[Sakar et al, 2015]

# What can topsApp.py currently do?

- Single swath processing of S1A data
  - Multiple slice stitching (SAFE directories)
  - Cropping and processing small region of interest
- DEM-assisted coregistration
- Fine azimuth coregistration with Enhanced Spectral Diversity
- Phase corrections when SLCs are processed with different versions of IPF processor
- Merging of bursts in a swath
- Standard post processing – Filtering, unwrapping, geocoding etc.

# topsApp.py

- topsApp.py is a prototype for many new features in ISCE.
- First productized  workflow. All metadata is stored in XML files and is human readable.
- First workflow using dem-assisted coregistration.
- First workflow that uses spectral diversity

# Design choices – user familiarity

- To allow users to familiarize themselves with the new features, currently
  - Only single swath processing is supported
  - A lot of intermediate files are generated to let users explore ESD and new geometric coregistration features. These may be turned off by default in future releases.
  - These files take up disk space. Another motivation for providing only single swath processing for now.

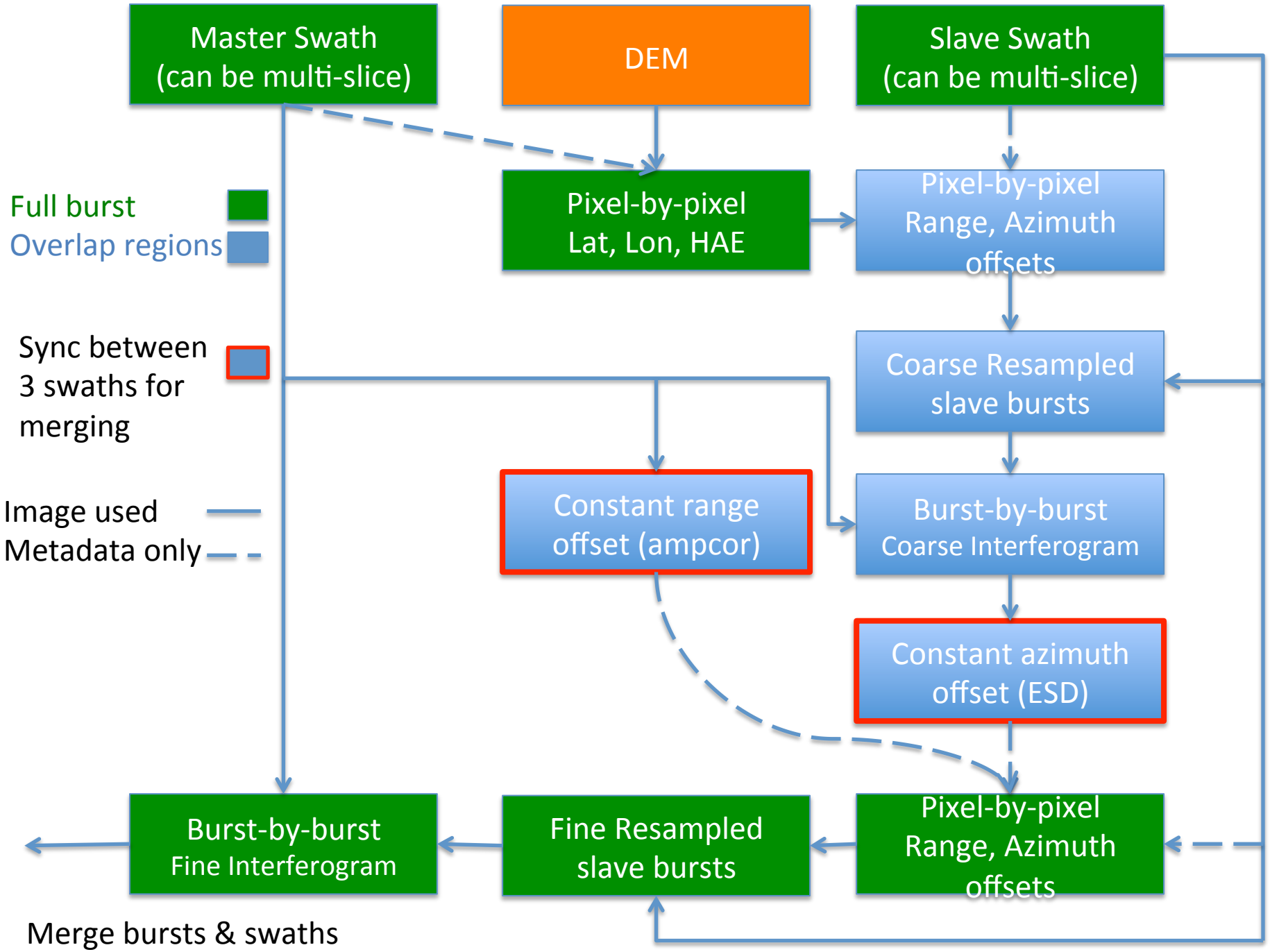
# Detailed but incomplete wishlist for S1A TOPS processing

| Feature | Status | Comment |
|---|---|---|
| Multi-slice stitching | Implemented | Users can provide multiple SAFE directories at input for stitching |
| Cropping with region of interest | Implemented | User can provide region of interest as a lat/lon bounding box |
| Dem-assisted coregistration | Implemented | Generalized for both zero doppler and native doppler geometries. Parallelized with openMP |
| ESD-based fine azimuth coregistration | Implemented | Estimates fine offsets using burst overlap interferograms |
| Merging of bursts | Implemented | Currently for a single swath |
| Single-swath processing | Implemented | topsApp.py processes single swath |
| Multi-swath processing | Planned | Will be included at a future date |
| EAP corrections | Implemented | https://sentinel.esa.int/documents/247904/1653440/Sentinel-1-IPF_EAP_Phase_correction |

# Wishlist continued ….

| Feature | Status | Comment |
|---|---|---|
| Standard geometry layers | Implemented | Lat/Lon/Hgt/LOS (stripmap-like) |
| Steered LOS vectors | Planned | Overlaps are viewed from different LOS. Currently LOS vectors correspond to zero doppler geometry. |
| Standard post processing | Implemented | Filtering, Unwrapping, Geocoding etc |
| PASTA-like SLC corrections | Not planned | Rodriguez-Cassola, Marc, et al. "Doppler-related distortions in TOPS SAR images." Geoscience and Remote Sensing, IEEE Transactions on 53.1 (2015): 25-35. |
| Reduced disk space usage | Planned | Virtual cropping and mosaicking etc to reduce disk space usage |
| Polarimetric processing | Planned | Not a high priority |
| Optimized ESD | Implemented | Only overlap interferograms are created for ESD. Cuts processing time by 90% for generating coarse interferogram. |

# Flow chart of implemented workflow

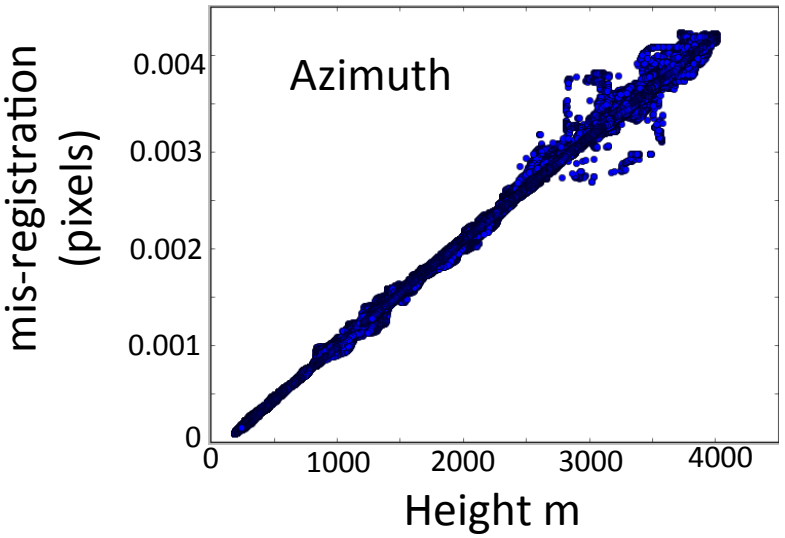

Master Swath
(can be multi-slice)
DEM
Slave Swath
(can be multi-slice)
Full burst
Overlap regions
Pixel-by-pixel
Lat, Lon, HAE
Pixel-by-pixel
Range, Azimuth
offsets
Sync between
3 swaths for
merging
Coarse Resampled
slave bursts
Image used
Metadata only
Constant range
offset (ampcor)
Burst-by-burst
Coarse Interferogram
Constant azimuth
offset (ESD)
Burst-by-burst
Fine Interferogram
Fine Resampled
slave bursts
Pixel-by-pixel
Range, Azimuth
offsets
Merge bursts & swaths

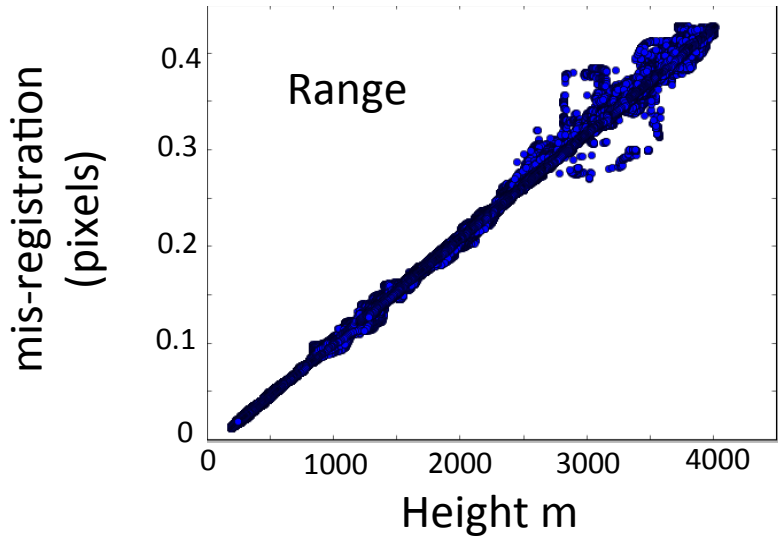
# Impact of Dem-assisted coregistration

- Sansosti, Eugenio, et al. "Geometrical SAR image registration." IEEE Transactions on Geoscience and Remote Sensing 44.10 (2006): 2861.
- Performs better than traditional polynomial based coregistration over wider range of baselines.
- Available in ISCE since May 2015 release.
- For TOPS mode, even azimuth offsets are sensitive to topography (see next slide)

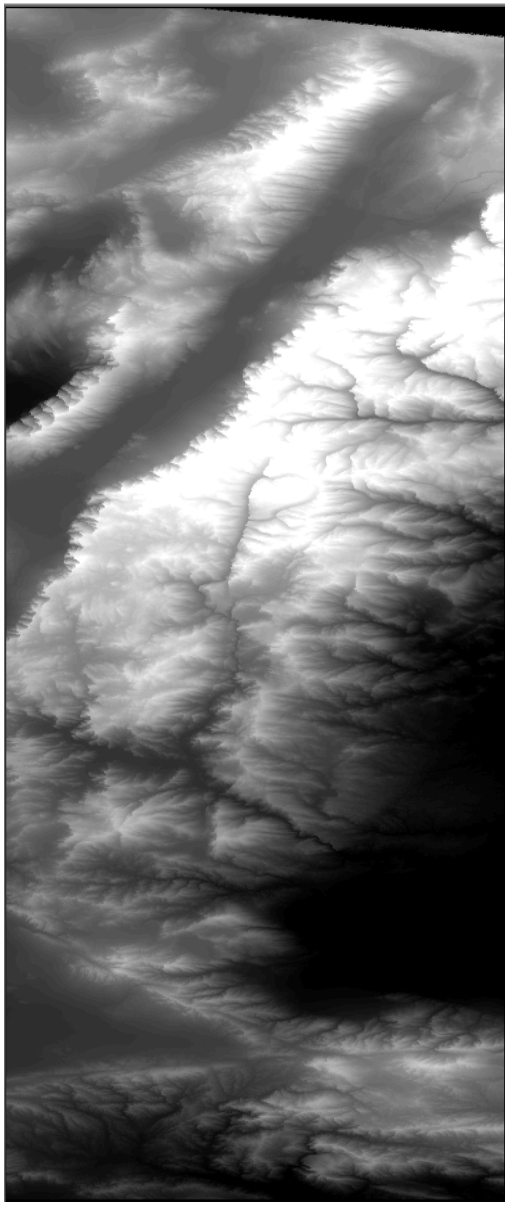
# Coregistration with DEM+orbits vs Polynomials

**Impact of DEM on coregistration**
**Bperp = ~ 100 m**



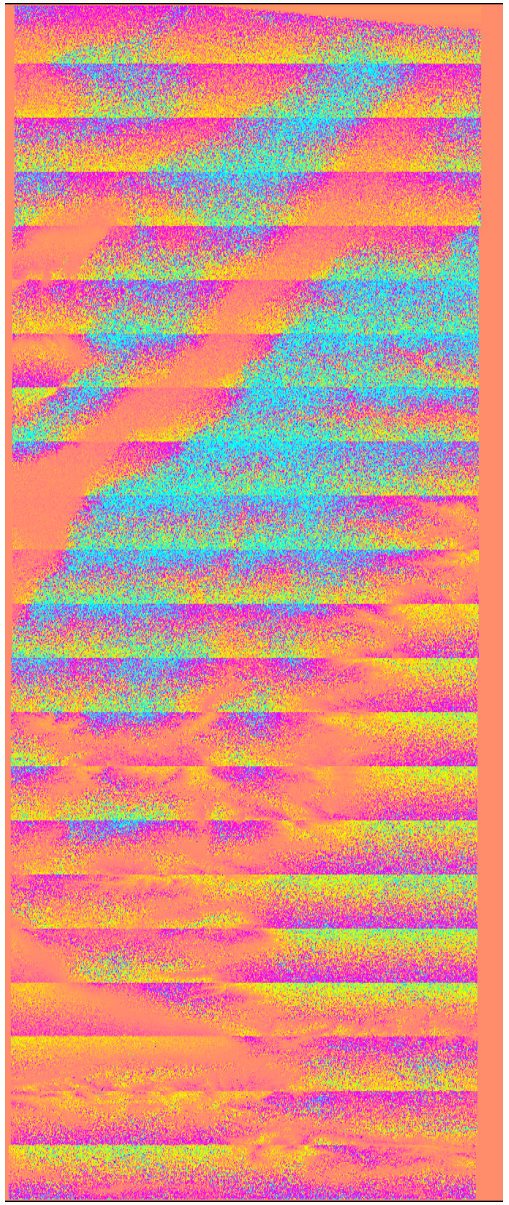




DEM



Impact on phase

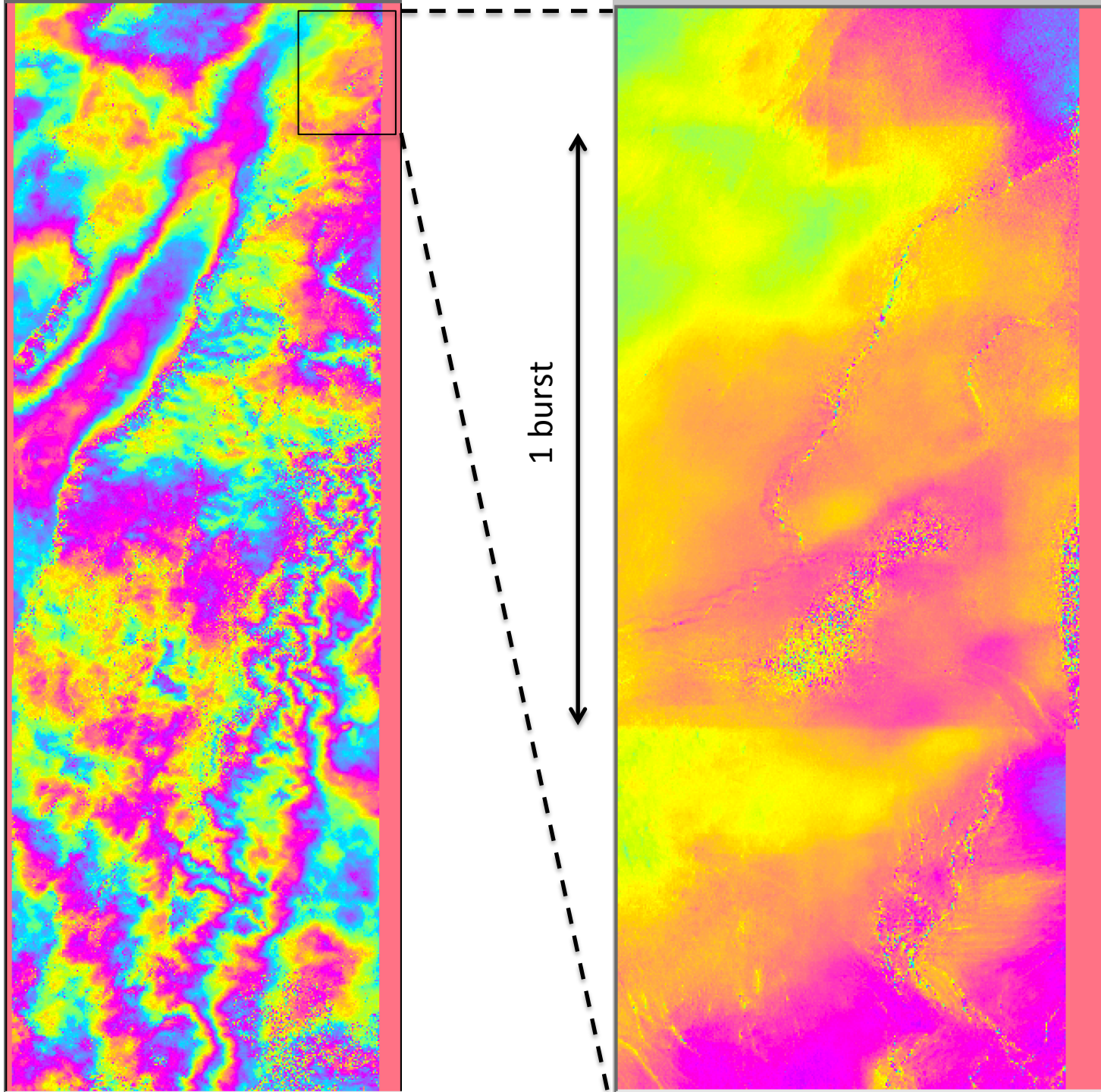
**-0.5 to 0.5 radians**



Coregistration with geometry (orbit + DEM).

Coarse interferogram with no fine azimuth registration.

Burst discontinuties are clear in most cases.

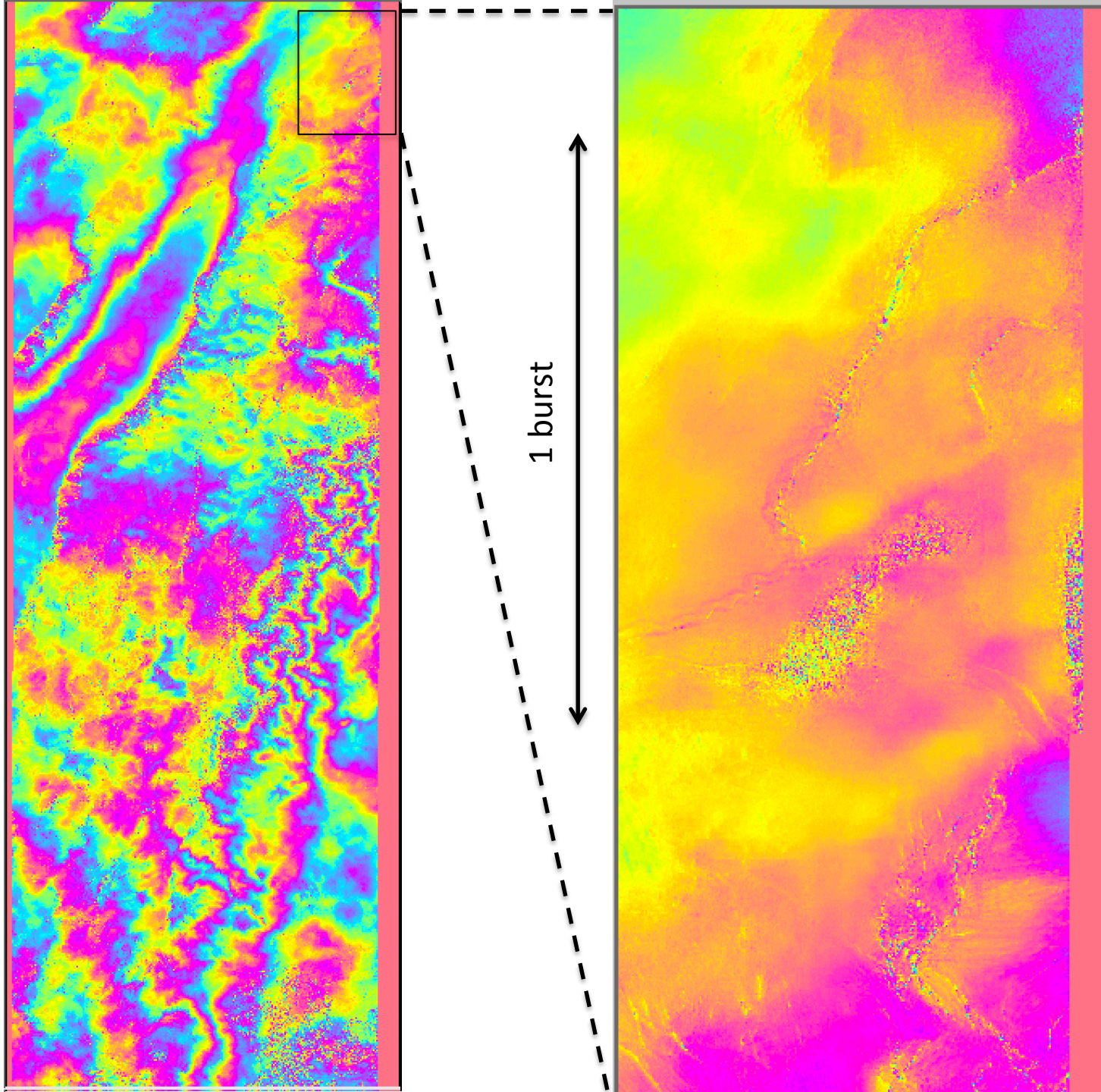


# Impact of Fine azimuth coregistration

- Prats-Iraola, Pau, et al. "TOPS interferometry with TerraSAR-X." Geoscience and Remote Sensing, IEEE Transactions on 50.8 (2012): 3179-3188.
- Offset estimated using overlap interferograms between bursts.
- Currently, multiple temporary products are output to help users get familiar with the ESD process. These may not be output by default in the future versions.

Coregistration with geometry (orbit + DEM).
+
Constant Azimuth misreg = 0.003 pixel

Fine azimuth shift estimated using Enhanced Spectral Diversity

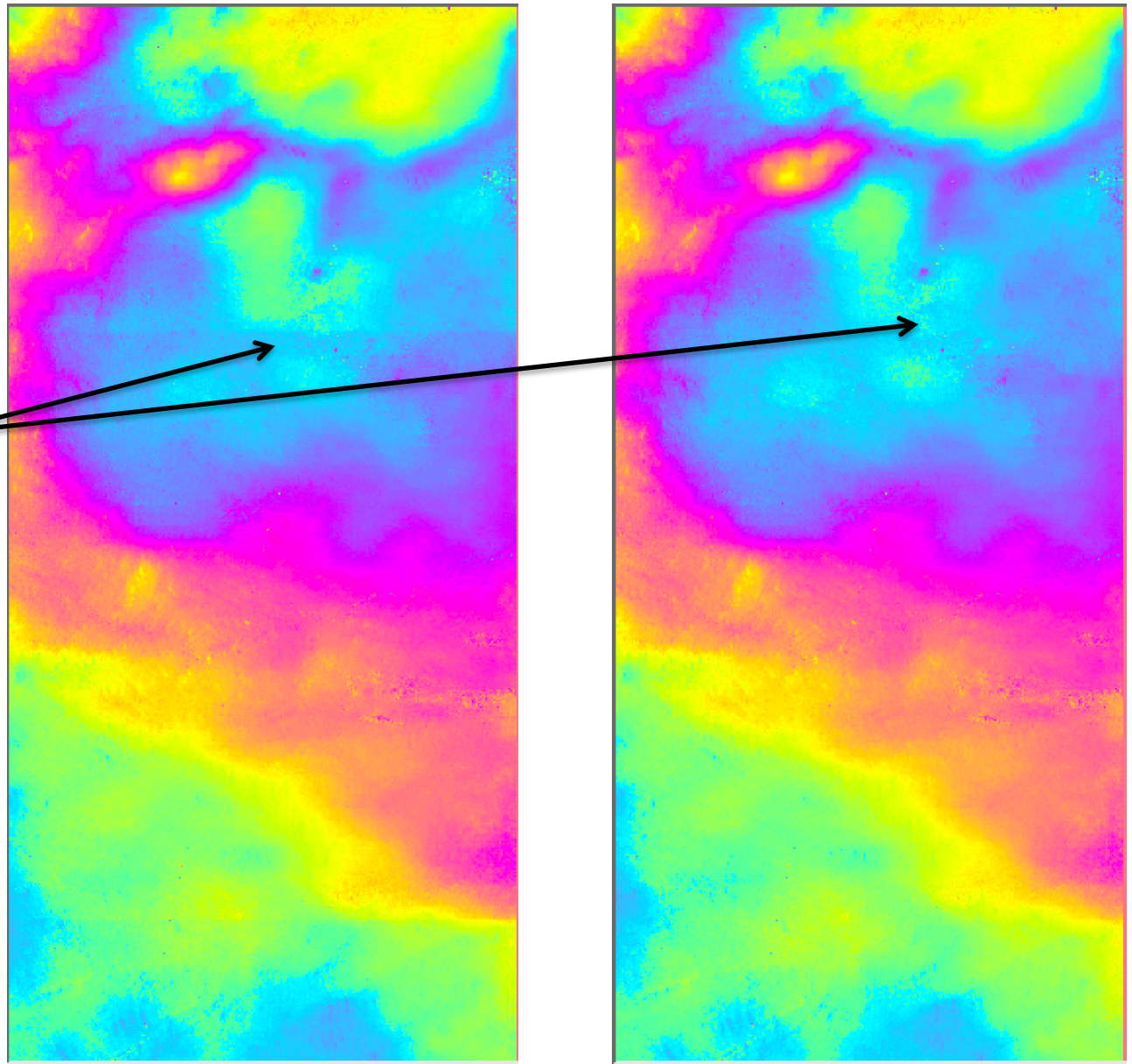


Another example with much smaller misregistration.

The burst continuity is still visible when ESD is not applied.

Geometrical coregistration

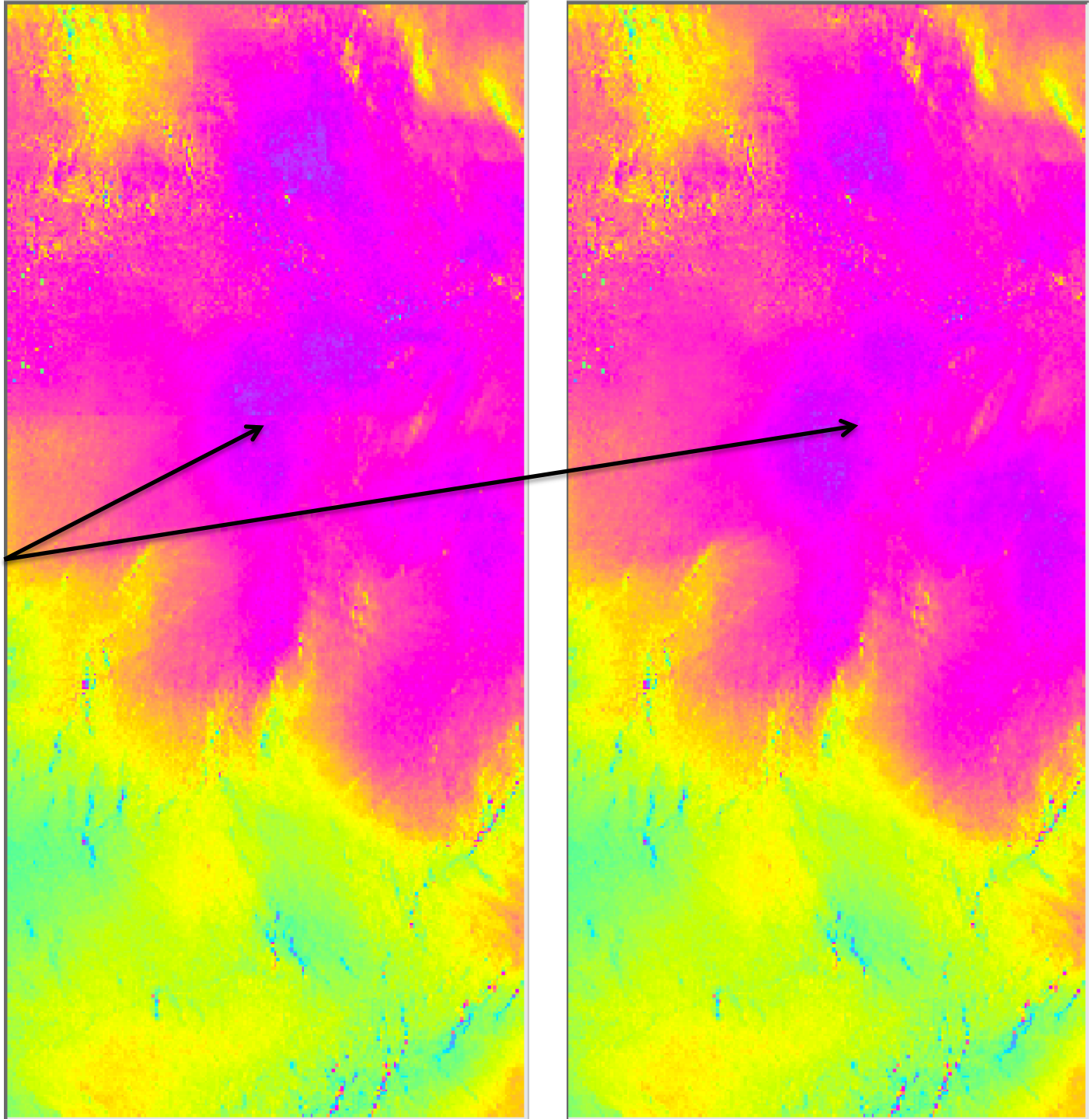
Geometrical coregistration + constant azimuth misregistration



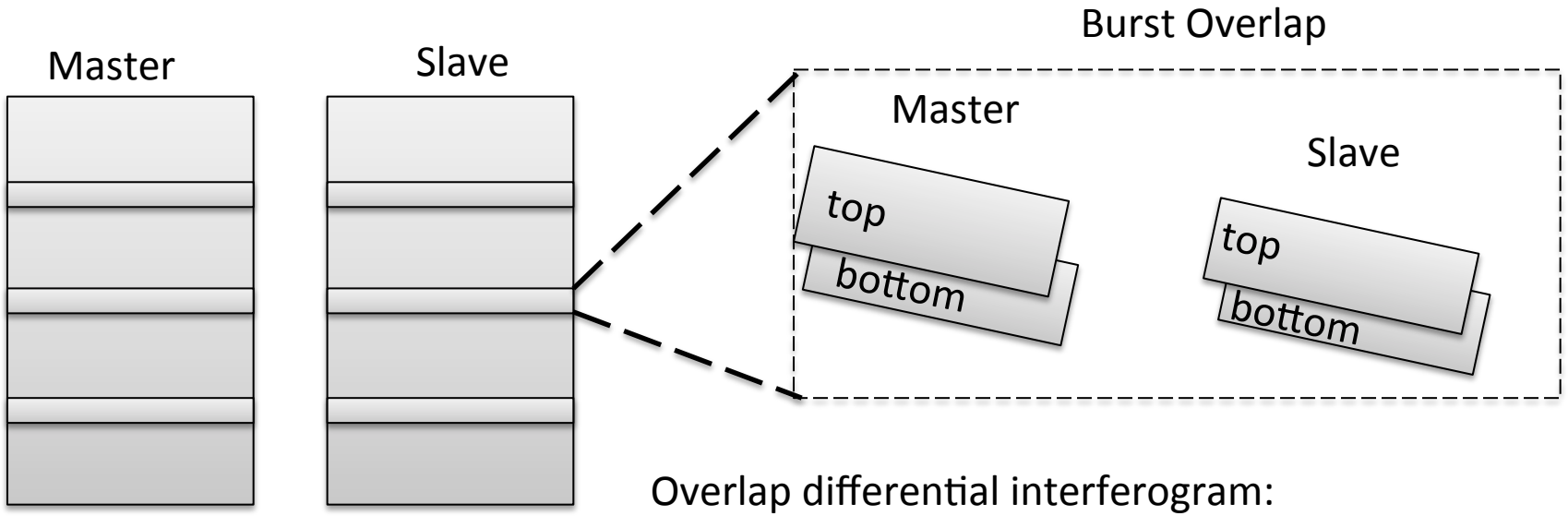
Example 3:
With even smaller misregistration. At first pass discontinuities are not clear.

Using ESD still helps improve the quality of the interferogram.

Recommendation: ESD correction is always needed for S1A interferograms.

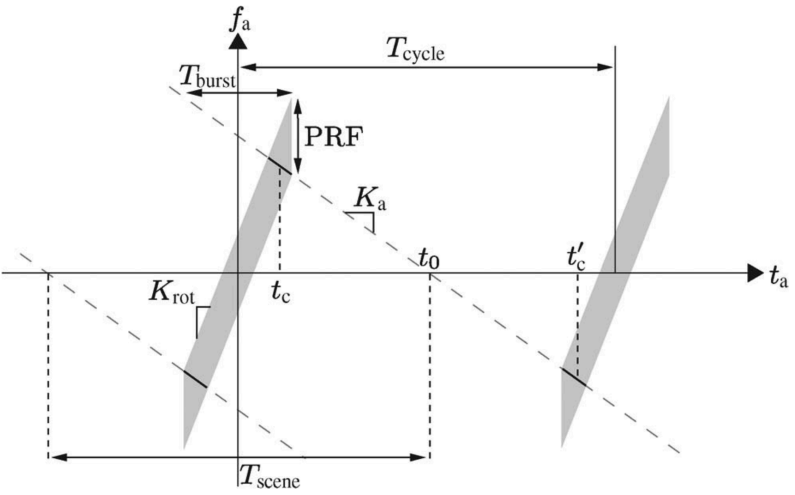


# Enhanced Spectral Diversity (fine azimuth misregistration)




Overlap differential interferogram:

$$sd = (M_{top} \;.\; S_{top}^{*}).(M_{bot} \;.\; S_{bot}^{*})^{*}$$

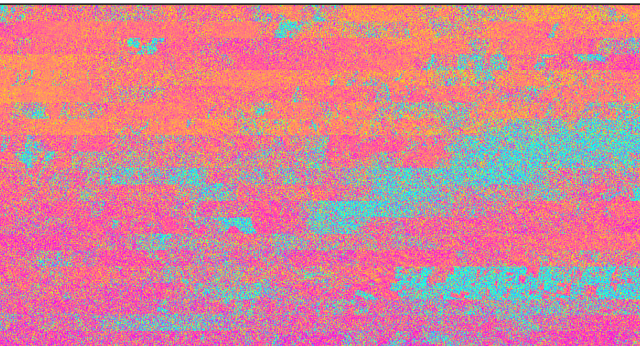



$$\phi_{sd} = 2\pi \Delta f \, \Delta t$$

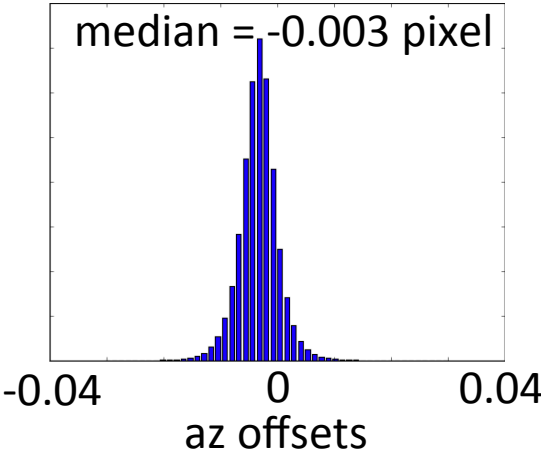
$\Delta f$ : Spectral separation of the two sublooks (can be computed from geometry)

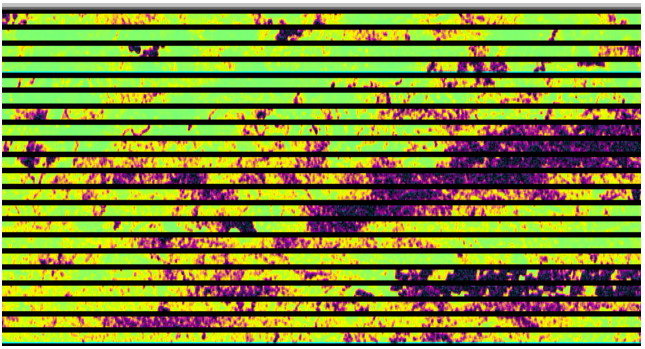
$\Delta t$ : Azimuth misregistration (in seconds)

# Enhanced Spectral Diversity (fine azimuth misregistration)



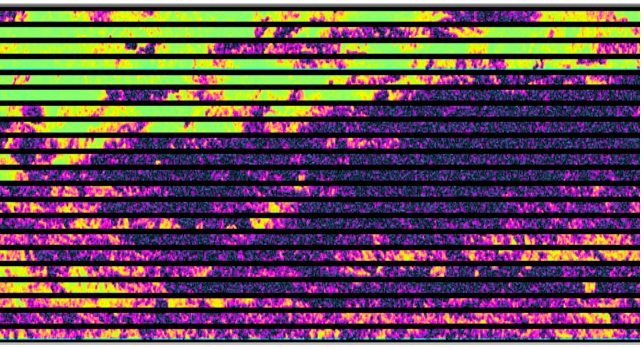
21 burst overlap interferograms



21 burst overlap coherence

**20150706-20150730 (24 days)**



21 burst overlap coherence

**20141226 – 20150401 (96 days)**

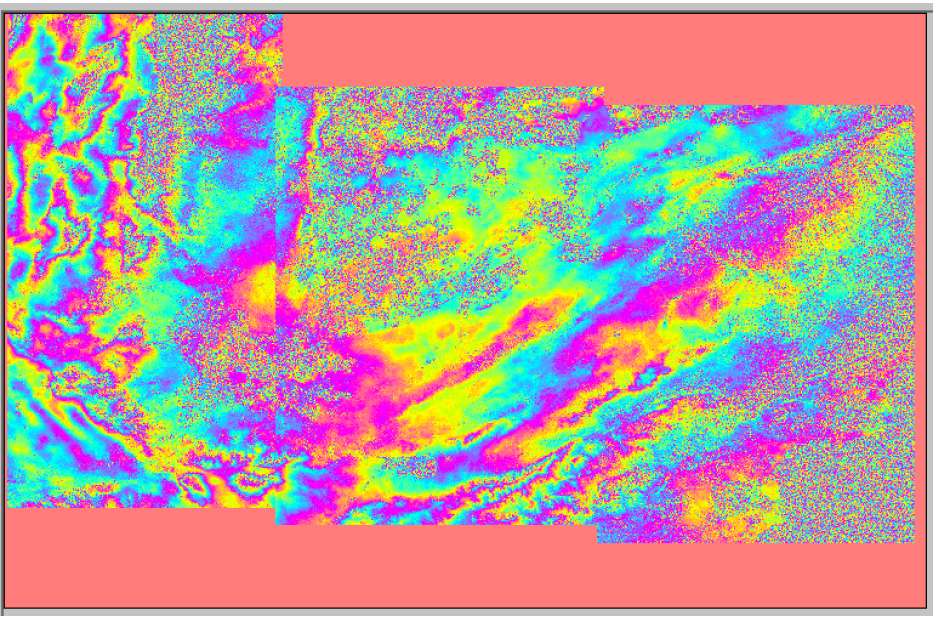
Low coherence between SLCs with long temporal separation can pose a problem for accurate azimuth coregistration.

Can be solved using stack processing approach.

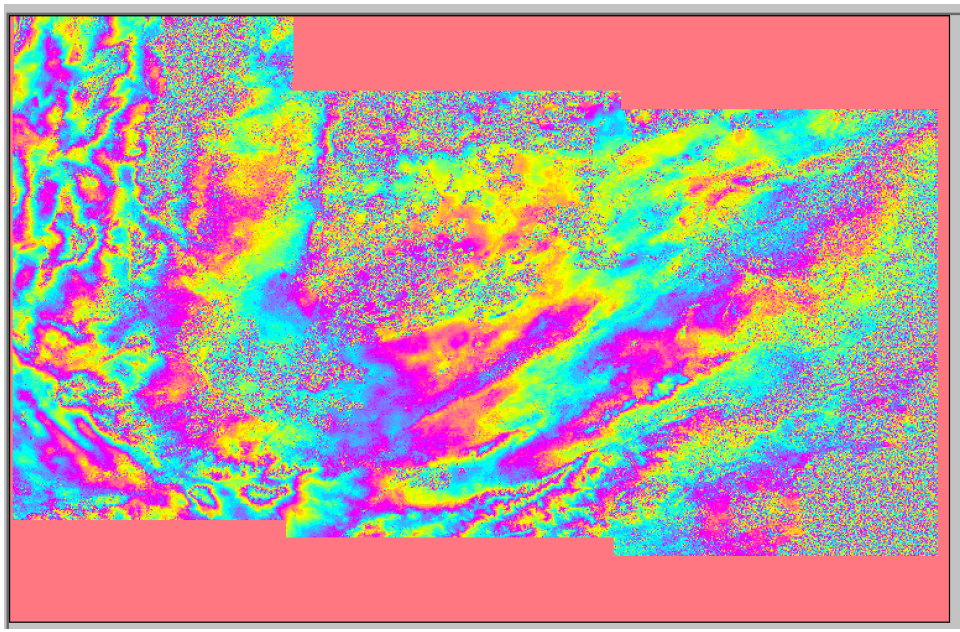
# Impact of Elevation Antenna Pattern Phase Corrections

# Elevation Antenna Pattern

Without correction



After EAP correction



Phase discontinuities between swaths.

Only observed in pairs constructed from SLCs processed with different IPF versions

# topsApp.py - inputs

- Example input xml templates are included with the tarball
- See examples/input_files/topsApp.xml
- Examples/input_files/ master_TOPS_SENTINEL1A.xml
  - Remember to change output directory name for each input file to avoid rewriting the master with the slave

# topsApp.py - Steps

- 1: Preprocess
  - Unpacks data from SAFE directories and populates folders for master and slave dates
  - Name of master and slave folders is controlled using user-defined input files (e.g, topsApp.xml)
  - Each directory includes bursts as SLC files
  - Metadata corresponding to given unpacked directory  is stored in “folder”.xml file

# topsApp.py - steps

topsApp.py –help –steps

The step names are chosen from the following list:

['startup', 'preprocess', 'computeBaselines', 'verifyDEM', 'topo']

['subsetoverlaps', 'coarseoffsets', 'coarseresamp', 'overlapifg', 'prepesd']

['esd', 'rangecoreg', 'fineoffsets', 'fineresamp', 'burstifg']

['mergebursts', 'filter', 'unwrap', 'geocode']

# topsApp.py - Steps

- 2: computeBaselines
  - Identifies common bursts between the master and slave products and saves indices.
  - Estimates rough baselines. Note these estimated baselines are never used for actual processing.

# topsApp.py - steps

- 3: verifyDEM
  - Just like insarApp.py
  - Checks if user defined DEM exists
  - If not, it is downloaded
  - Conversion from EGM96 to WGS84

# topsApp.py - steps

- 4: topo
  - Compute pixel-by-pixel lat, lon, hgt, LOS for every pixel in the master product bursts
  - Performed for common bursts only
  - Products stored in separate folder (default: geom_master)
  - Products stored separately for each burst
  - Note: Master timing errors are negligible for S1A. Hence no master timing correction step has been implemented.

# topsApp.py - steps

- 5: subsetoverlaps
  - Extract burst overlap region outputs for topo into a separate sub-folder (default: geom_master/overlaps)
  - Metadata for top and bottom bursts stored in separate xml files - master_top.xml and master_bottom.xml

# topsApp.py - steps

- 6: coarseoffsets
  - Estimate offset fields to resample slave bursts only for the burst overlap regions
  - Outputs saved in separate folder (default: coarse_offsets/overlaps)

- 7: coarseresamp
  - Create coregistered Slave SLCs for overlap regions
  - Outputs stored in separate folder (default: coarse_coreg/overlaps)

# topsApp.py - steps

- 8: overlapifg
  - Create burst overlap interferograms and take looks
  - Product created in separate folder (default: coarse_interferogram)

- 9: prepesd
  - Create cross interferograms for ESD
  - Creates separate folder (default: ESD)

# topsApp.py - steps

- 10: esd
  - Estimate azimuth misregistration using ESD
  - Can control coherence threshold etc via user input file

- 11: rangecoreg
  - Estimate range misregistration using amplitude correlation of overlap regions only

# topsApp.py - steps

- 12: fineoffsets
  - Refine slave metadata with range and azimuth misregistration numbers
  - Estimate fine offset fields for full bursts using outputs of topo and refined metadata
  - Outputs saved to a separate folder (default: fine_offsets)

# topsApp.py - steps

- 13: fineresamp
  - Resample slave bursts using fine offset fields
  - Output to separate folder (default: fine_coreg)

- 14: burstifg
  - Create burst-by-burst interferograms by cross multiplication
  - Output to separate folder (default: fine_interferogram)
  - Metadata file written (default: fine_interferogram.xml)

# topsApp.py - steps

- 15: mergebursts
  - The burst-by-burst products are merged to look like a stripmap product
  - Interferogram, lat, lon, z, los
  - Output to separate folder (default: merged)
  - All processing after this occur in merged folder

- 16: filter
  - Just like insarApp.py (for merged products)

# topsApp.py - steps

- 17: unwrap
  - Just like insarApp.py (for merged products)

- 18: geocode
  - Just like insarApp.py (for merged products)

# What is needed to merge swaths?

- For adventurous users
- By design, the implemented workflow preserves phase
- For seamless processing, two parameters need to be constant for all 3 swaths
  - At the end of step “rangecoreg”
  - Use commonvalues for “slavetimingcorrection” and “slaverangecorrection” for all 3 swaths
  - Modify PICKLE/rangecoreg.xml

# Merging …

- We use orbits + DEM for coregistration
  - No issue of using different polynomials for different swaths and introducing phase discontinuities
- By design, S1A swaths are offset by integer number of pixels in azimuth and range
  - One can use any standard GIS software (e.g, GDAL) to mosaic these 3 swaths seamlessly
  - This functionality will be introduced in later versions
  - Using existing features like virtual crop, mosaic etc from GDAL will make handling of TOPS datasets really easy and also reduce need to create temporary intermediate filess

# How can I use metadata in my python code?

- For developers and users who want to build on topsApp
- Classes of importance – from isceobj.Sensor.TOPS import BurstSLC, TOPSSwathSLCProduct

- Load product into interactive python terminal

```
import isce
from iscesys.Component.ProductManager  import ProductManager as PM
pm = PM()
pm.configure()
obj = pm.loadProduct(xmlname) #Example: fine_interferogram.xml
print(len(obj.bursts), obj.bursts[0].sensingStart)
```