Lab Exercise II: SAR Interferometry – Generation of Digital Elevation Models

Contact: xiao.zhu@dlr.de
Date: 2014-05-08

Synthetic aperture radar interferometry is widely used for the generation of digital elevation models (DEM). The topographic information is derived by exploiting the phase differences of the coherent radar signal from 2 complex-valued SAR images (defined as Master and Slave image), which are acquired from slightly different orbit positions. Three essential steps are the main components of the standard InSAR processing (Fig. 1):

1) coregistration of the two complex images
2) interferogram generation and coherence estimation
3) phase unwrapping

The exercise will be split into three parts according to the above mentioned components.

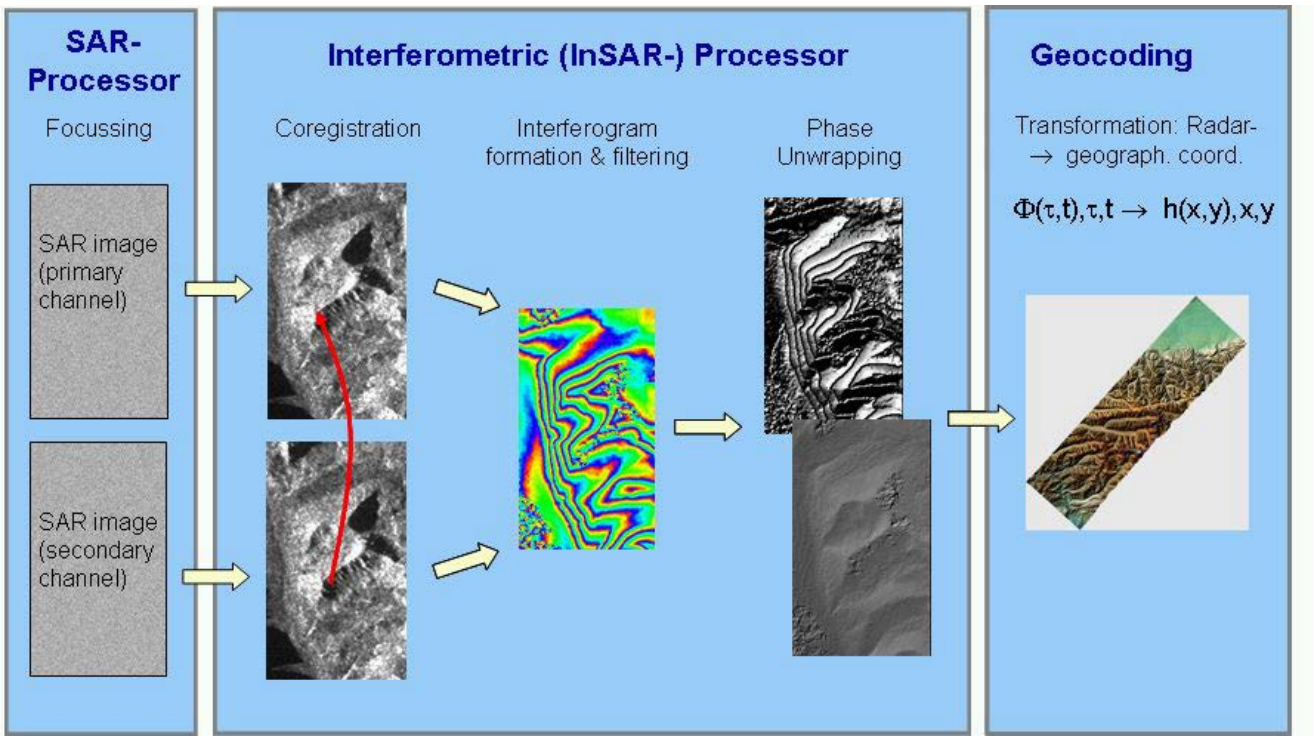

Figure 1: interferometric processing

Task 3: Phase Unwrapping (PhU)

In an interferogram the phase value is not known absolutely, but it is a 2π-modulus of the unknown absolute phase. In order to estimate terrain heights and generate a DEM, the fringes have to be unwrapped. To convert the measured phase of the interferogram to an absolute unwrapped phase the appropriate number of cycles has to be added for every pixel. PhU of incoherent areas does not perform as desired due to variable phase noise.

Working steps:

1. Create a function file for unwrapping **one vector** (e.g., one line of a interferogram)
   - try to avoid loops in the function to increase speed of unwrapping
   - input: phase (1xn vector), initial cycles to add to **all** elements (needed for 2D PhU)
   - output: unwrapped phase (1xn vector), number of cycles added **for each** pixel (1xn vector)
   - use Itoh's method for 1-D Phase Unwrapping:
     - Compute the phase differences: $D(i) = \psi(i+1) - \psi(i)$ for $i = 1, \ldots, N-1$
     - Detect the positive & negative 2π-phase jumps (threshold for gradient: $\pm\pi$)
     - calculate the total number of cycles to add for each pixel for PhU
     - add initial cycles to this vector (given by input parameter)
     - initialize the first unwrapped value using the first element of the input phase vector: $\varphi(1) = \psi(1)$
     - unwrap the measured phase based on the calculated gradients and total number of cycles (add multiples of $\pm 2\pi$)

2. apply PhU to one line of interferogram;
   - plot the wrapped phase, the unwrapped phase and the limits of $\pm\pi$ (horizontal lines)
   - in a second figure plot the cycles that have to be added for each pixel of the line

3. 2-D Phase Unwrapping
   - Select a suitable area from the phase image, e.g. [1140:2730, 240:800] (Volcano)
   - unwrap the first column to get the initial cycles for each line
   - unwrap the whole dataset along **each row** (range direction) using the function created for working step 1; remember to make use of the initial cycles obtained by the previous step
   - repeat 2D PhU, but this time unwrap **column-wise**; compare results !

4. Calculation of residues
   - remember to use the wrapped phases of the initial interferogram
   - create a function file for calculating the residues
     - try to implement a fast algorithm (residues per line without a loop but loop for all lines of interferogram)
     - use quadruples of pixels for the integral of gradients
     - remember to wrap each phase difference between pixels before calculating the integral
     - output: residue matrix (values: 0 = no residue, 1 = positive res., -1 = negative res.)
   - plot the residue map; interpret the influence of noise on phase unwrapping based on the residue maps

Provided data:

cplx_coherence.mat:	coherence & interferometric phase (results of task 2)
Part.mat:	area of the volcano

Hints:

- the filtered interferogram computed from task 2 can be used for phase unwrapping
- heights in [m] can be obtained by using conversion factor of 167.89 m/cycle
- a plot of the locations of the residues (use find function) in the interferogram and/or unwrapped phase image may help for an interpretation of results

Useful Matlab commands

- cumsum — cumulative sum of elements in vector
- line([x1, x2], [y1, y2]) — plot line from (x1, y1) to (x2, y2)
- ginput — get pixel coordinates by manual selection
- angle — get phase of complex data
- abs — get amplitude of complex data
- A_part = A(:,1) — store first column of matrix A in variable A_part
- B_part = A(1,:) — store first row of matrix A in variable B_part
- mesh — 3D plot (meshgrid)

  find — find indices of nonzero elements