Delft Object-oriented Radar Interferometric Software
User’s manual and technical documentation
Preface

This document describes the Doris Software for Interferometric SAR processing. It is compliant with Doris v3.16 (Document Revision : 1.55). At the moment this manual contains technical documentation besides the information that is required for running the software. We try to be as complete as possible, but some chapters may be very brief in the description. Please report any incompleteness in the documentation or the source code.

The latest information on the Doris software can always be found on the internet: http://enterprise.lr.tudelft.nl/doris/.

Doris is freely available to the scientific community. The conditions of use for the Doris software are as follows.

1. Doris is a scientific-purpose software and cannot be commercialized, nor can parts or products of it be commercialized. Parties interested in using Doris or its products for any commercial purposes are requested to contact Prof. Dr. Ramon Hanssen of DEOS (r.f.hanssen@tudelft.nl)

2. Our version of the software is the only official one. Please do not distribute the Doris software to third parties, instead refer to the Doris home page. This in order to guarantee uniformity in the distribution of updates and information.

3. Delft University of Technology is not responsible for any damage of any kind caused by errors in the software or in the documentation.

4. Users are very welcome to extend the capabilities of the Doris software by implementing new algorithms or improving the existing ones. It is intended that if new software is developed based on Doris, that this also is made available for free to the other users (through us).

5. We would appreciate if any addition or modification of the software would be announced first to us, so that it can be included in the official (next) version of the software.

6. Publications that contain results produced by the Doris software should contain an acknowledgment. (For example: The interferometric processing was performed using the freely available Doris software package developed by the Delft Institute for Earth-Oriented Space Research (DEOS), Delft University of Technology. or include a reference to: Bert Kampes and Stefania Usai. "Doris: The Delft Object-oriented Radar Interferometric software." In: proceedings ITC 2nd ORS symposium, August 1999. (cdrom)).

This document is typeset in \LaTeX2e.
Contents

Preface 3

1 Introduction 1
   1.1 Overview of the InSAR Processing ............................................ 1
       1.1.1 Processing order ......................................................... 2
   1.2 General considerations and conventions .................................... 5
       1.2.1 Inputfile ................................................................. 7
       1.2.2 Outputfiles .............................................................. 8
   1.3 Outline of this document ..................................................... 10

2 General Cards 13
   2.1 General Input Cards .......................................................... 13
   2.2 Example General Input Cards ............................................... 16

3 M_READFILES 19
   3.1 Input Cards ................................................................. 19
   3.2 Output Description ........................................................ 20
   3.3 Implementation ............................................................. 21
       3.3.1 Changes for X86 platforms ........................................... 22

4 M_PORBITS 23
   4.1 Input Cards ................................................................. 23
   4.2 Output Description ........................................................ 24
   4.3 Implementation ............................................................. 25

5 M_CROP 27
   5.1 Input Cards ................................................................. 27
   5.2 Output Description ........................................................ 28

6 M_OVS 29
   6.1 Input Cards ................................................................. 29
   6.2 Output Description ........................................................ 30
   6.3 Algorithm ................................................................. 30
Chapter 1 Introduction

This user’s manual guides you through the radar interferometric processing with the Doris software. A user will be able to process ERS1/2, ENVISAT, JERS, RADARSAT and ALOS data with the aid of this manual. The data must be in Single Look Complex format (SLC). Doris is not a SAR processor, i.e., you cannot process/focus RAW radar data. The manual also contains implementation specifications which can be helpful to a programmer for further development of Doris, or to understand WARNING and ERROR messages.

1.1 Overview of the InSAR Processing

A high-level description of the InSAR processing is shown in Figure 1.1 where a division in four blocks has been made. Block I depicts the preprocessing of the raw (radar and orbit) data to another format.

We won’t be concerned with the raw orbit data, but it is included in the flow chart for completeness. The Delft precise orbits are used for ERS1/2, obtained via the getorb package (see, e.g., [Scharroo and Visser, 1998]). The second block consists of the co-registration where the slave image is aligned with the master image, and of the computation of the reference phase of the ellipsoid. In block III the interferometric products (complex phase image and coherence map) are computed. Finally in block IV the endproducts (e.g., a DEM or a deformation map) are computed.

The processing steps that are identified to be implemented in the Doris software are listed in the table below. Note that the bold numbers are included in Doris manual since v3.16 (Document Revision : 1.55), and that UNWRAP is NOT directly implemented in Doris.

To run each step, these names must be used as arguments for the PROCESS card, as explained in Chapter 2. A specific algorithm (module, method) can be selected with the cards that are special to this step, see the Chapters 3 to 29.

<table>
<thead>
<tr>
<th>#</th>
<th>PROCESS (chapter)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>M_READFILES (3)</td>
<td>Read the processing parameters from the SLC files (null/leader/volume and data) for the master image.</td>
</tr>
<tr>
<td>02</td>
<td>M_PORBITS (4)</td>
<td>Retrieve the precise Delft orbital data records with the getorb package.</td>
</tr>
<tr>
<td>03</td>
<td>M_CROP (5)</td>
<td>Write the SLC data from paf format to disk in the 'raw' (pixel interleaved 2b/2b complex short integer) format.</td>
</tr>
<tr>
<td>06</td>
<td>S_READFILES (7)</td>
<td>See M.*</td>
</tr>
<tr>
<td>07</td>
<td>S_PORBITS (8)</td>
<td>See M.*</td>
</tr>
<tr>
<td>08</td>
<td>S_CROP (9)</td>
<td>See M.*</td>
</tr>
<tr>
<td>11</td>
<td>M_FILTAZI (13)</td>
<td>Spectral filter for master image in azimuth (line) direction.</td>
</tr>
<tr>
<td>12</td>
<td>S_FILTAZI (14)</td>
<td>Spectral filter for slave image in azimuth (line) direction.</td>
</tr>
<tr>
<td>21</td>
<td>COARSEORB (11)</td>
<td>Compute the translation between master and slave with the orbits (precision 30 pixels).</td>
</tr>
<tr>
<td>22</td>
<td>COARSECORR (12)</td>
<td>Compute the translation between master and slave on pixel level by correlation technique.</td>
</tr>
<tr>
<td>23</td>
<td>FINE (15)</td>
<td>Compute translation vectors over the total image on sub-pixel level.</td>
</tr>
<tr>
<td>24</td>
<td>COREGPM (16)</td>
<td>Compute the actual transformation (2d-polynomial) model for the alignment of the slave on the master image.</td>
</tr>
<tr>
<td>25</td>
<td>RESAMPLE (17)</td>
<td>Resample the slave image according to the transformation model of the coregistration.</td>
</tr>
<tr>
<td>31</td>
<td>FILTRANGE (18)</td>
<td>Spectral filter for master and slave image in range (pixel) direction.</td>
</tr>
<tr>
<td>41</td>
<td>INTERFERO (19)</td>
<td>Compute the (complex) interferogram.</td>
</tr>
<tr>
<td>42</td>
<td>COMPREFPHA (20)</td>
<td>Compute the reference phase of the ellipsoid to be subtracted from the interferogram (polynomial).</td>
</tr>
<tr>
<td>43</td>
<td>SUBTRREFPHA (21)</td>
<td>Subtract the reference phase of the ellipsoid from the interferogram.</td>
</tr>
<tr>
<td>44</td>
<td>COMPREFDEM (23)</td>
<td>Compute the reference phase of a DEM to be subtracted from the interferogram.</td>
</tr>
<tr>
<td>45</td>
<td>SUBTRREFDEM (24)</td>
<td>Subtract the reference phase of the DEM from the interferogram.</td>
</tr>
<tr>
<td>46</td>
<td>COHERENCE (22)</td>
<td>Compute the (complex) coherence map.</td>
</tr>
<tr>
<td>47</td>
<td>FILTPHASE (25)</td>
<td>Filter the the interferogram.</td>
</tr>
<tr>
<td>48</td>
<td>UNWRAP (26)</td>
<td>Unwrap the interferogram.</td>
</tr>
<tr>
<td>51</td>
<td>SLANT2H (28)</td>
<td>Compute the heights of the pixels in the radar coded system.</td>
</tr>
<tr>
<td>52</td>
<td>DINSAR (27)</td>
<td>3/4 pass differential interferometry.</td>
</tr>
<tr>
<td>53</td>
<td>GEOCODE (29)</td>
<td>Geocode the pixels (convert pixels from the radar coordinate system to a earth fixed reference system.).</td>
</tr>
</tbody>
</table>

### 1.1.1 Processing order

The processing order is not restricted, but, in general, the output of a step is the input for the next. The following flow charts (Figures 1.2, 1.3, and 1.4) show how we view the processing order.

The order in which the modules are called in the main switch in the source code is (generated by the grep command):

```
(grep "// ===" processor.c)
```
Figure 1.1 Coarse flow of the interferometric processing of SAR images.
Figure 1.2 Processing order block II of InSAR processing.
1.2 General considerations and conventions

In this section some important definitions are described that are used in the Doris software. This will clarify the terminology used. The general set up of the input file and output files is
After compilation with the Makefile, the executable is named: "doris". In this document therefore this name is used to refer to the executable. The command line options are:

- **doris -ver**
  
  return version number.

- **doris -h**
  
  return help (system call to shell script named helpdoris).

- **doris [file]**
  
  run, use input in "file" (default: "inputoptionsfile").

It is advised also to compile an executable "doris.debug" (with the Makefile). This version is somewhat slower and more verbose, but it can be used if something seems to go wrong with the normal executable, and it is not clear what. See also Annex ???. We advice to use the utility scripts to generate input files and to run the processor.

Conventions are:

- We use the term **lines** to refer to the azimuth direction (slow time), and **pixels** for the range direction (fast time). (A **pixel** might also refer to an element, which will be clear from the context.) In the source code we frequently use the term azimuth **buffers** and range **blocks**.

- Our convention is to use **first lines**, **second pixels**, e.g., for the order of input arguments. The line direction (azimuth) corresponds to the vertical (y). The pixel direction (range) corresponds to the horizontal (x). Note that other software may use x before y.

- The first line (pixel) of an image is indexed as 1 (this may be a bit unusual). (In the
software, the first index of an image (in a matrix) is equal to 0. To index a matrix, use \text{MAT}(y,x), i.e., as in linear algebra, Matlab, etc.)

- The name, format, and dimensions of the current master/slave/interferogram are stored in information structs that are filled by reading the corresponding \text{result files}. The files do not have a header.

- Generally all coordinates are in the master (radar) coordinate system. The first and last line are given, as well as multilook factors for both directions. If for example an interferogram is multilooked at the generation with a certain factor and later on again at the subtraction of the reference phase, then the first line is still the same, see Figure 1.2.

- In our view, the \text{output files} can only contain the results of one algorithm per step. So it is not possible to re-run a step (for example with another algorithm) without deleting the previous result. New users may be confused by this approach, but Doris should give an appropriate error message if it is attempted to run a step twice.

- Temporary files are created during the processing. Their names always start with "scratch". If such files are not removed by Doris, for example after an error in the processing, they can be safely removed by a \text{rm} command.

In the logfile additional information is written that is not sent to standard out or the \text{result files}, such as statistical information on a least squares estimate. (For example, we always inspect the correlation value for step \text{COARSE_CORN}, and the error between model and observations for reference phase computations in the log file.)

1.2.1 \textbf{Inputfile}

There is one ascii input file containing cards and zero or more parameters that controls the processor. A card is the first word on a line in the input file. The card and parameter(s) are delimited by blanks or tabs. There are mandatory cards (such as STOP at the end) and optional cards (because there are defaults for example for a filename for the output, or cards like \text{COMMENT}.) In this document the mandatory \text{CARDS} are in sans serif bold face, the optional \text{CARDS} are in normal style. Default parameters are underlined, and [ optional ] parameters are between square brackets.

The \text{input file} consists of a header and a tail. In the header, the general cards are placed (see
Chapter 2), and in the tail the cards specific to a certain step are placed (described in the other chapters).

The order of the cards is not restricted (except the STOP card), though we advice to group them by processing step. Blank lines are allowed in the *input file*, but the line counter will not function properly in that case (which does not affect the processing in any way). We advice to place a comment on otherwise empty lines.

If (accidently) a certain card is used more than ones, then a WARNING is generated and the latter one is ignored (this behavior is not guaranteed, not true for PROCESS cards).

The case of cards and parameters is not restricted. We advice to use UPPER case (except for comment or c) for cards and lower case for parameters.

Text after the last expected parameter is ignored. Be careful with putting comments in like this if the number of parameters may be varied for a certain card. We normally do not make one big *input file*, but we use several small ones, for a group of processing steps. See also the run file in Annex ??.

The examples in the next chapters will make this more clear. All keywords are described in this manual, and also in the interactive helpdoris script, and in the run script. It can occur that some keywords are not mentioned in the latter two. To obtain all possible cards, give the command:

```
  grep keyword readinput.c | grep else
```

### 1.2.2 Outputfiles

There are three ascii *output files*: one for (results of) processing steps specific to the master, one for the slave, and one for the rest of the processing (the ‘products’). These files are referred to as master, slave and product *result file* (parameter files). For example, the wavelength of the sensor and the filename of the master image can be found in the *master result file* (and for the slave parameters in the *slave result file*), while coregistration parameters, which aren’t unique to a particular image, can be found in the *products result file*. These *output files* serve as input for Doris for running later steps. (Of course, a step also can generate binary data output. This is described in the following chapters.)

The *result files* all consist of a header and a tail, which grows with the processing.

In the header some general information and an overview of the processing is given with *process control flags*. (These flags do not imply a certain order.) By convention, each processing step can be run only once (0 or 1 in the *process control flag*), to avoid confusion on the correct/latest results are. (This implies that a result section in the tail has to be deleted, and the *process control flag* reset to 0, before running a step a second time.)

In the (growing) tail the results of the processing is stored. The *result files* are read again and the read parameters are used in the further processing. (In order to *trick* Doris to use other parameters then the ones that result from a previous processing step, simply edit the *result file*, e.g, in order to coregister complex interferograms.)

For the master *output file* the header with the information and the *process control flags*
Looks like:

```
MASTER RESULTFILE:  master.res

Created by:
InSAR Processor:  Doris (Delft o-o Radar Interferometric Software)
FFTW library:  used
VECLIB library:  not used
LAPACK library:  not used
Compiled at:  Sep 6 2007 17:26:52
By GNU gcc:  4.1.4
File creation at:  Sat Jan 5 19:08:21 2008

--------------------------------------------------
| Delft institute of Earth-oriented Space research |
| Delft University of Technology |
| http://enterprise.lr.tudelft.nl/doris/ |
--------------------------------------------------
```

Start_process_control
readfiles:  0
precise_orbits:  0
crop:  0
filt_azi:  0
filt_range:  0
NOT_USED:  0
End_process_control

The last flag (NOT_USED) is reserved for future use. In the slave result file the following line (and processing step) is extra:

```
resample:  0
```

The products result file is build up in the same manner. The process control flags in the header of the products result file are:

```
[SKIP] [SKIP]
Start_process_control
coarse_orbits:  0
coarse_correl:  0
fine_coreg:  0
comp_coregpm:  0
interfero:  0
coherence:  0
comp_refphase:  0
subtr_refphase:  0
comp_refdem:  0
subtr_refdem:  0
filtphase:  0
unwrap:  0
slant2h:  0
geocoding:  0
dinsar:  0
```
The latter flag is reserved for future use. As already mentioned, after the process control flags the results of a (successfully ran) processing step are appended.

A section of the tail always starts with some lines like

```
 **************************************************************
 * _Start_coarse_orbits:
 **************************************************************
```

After which the results for this processing step follow. A section always ends with a statement shown below. (This \texttt{End\_step: NORMAL} statement is important because the status of the process flag in the header is updated with it.)

```
 **************************************************************
 * End_coarse_orbits: _NORMAL
 **************************************************************
```

Note that not all steps that are in the process control flags actually have to be implemented in Doris for the moment. (Unwrapping, extra flags)

Since only one result section is allowed for every processing step, it is not possible to re-run a certain step without editing the \texttt{result file}. The process control flag in the header has to be reset, and the total section in the tail (from \texttt{Start\_step} to \texttt{END\_NORMAL}) has to be deleted, or commented out.

If the section is not deleted, Doris will likely exit, but if not, the further processing may be affected, because wrong values may be used (i.e., read from the \texttt{result file}).

It is of course possible to change the results (parameter values, for example correlation value for an estimated offset) in the \texttt{result files}, so that the altered value is used in the further processing. However, if you change the strings describing the output Doris will likely protest (i.e., hang or exit).

### 1.3 Outline of this document

In Chapter 2 the general purpose cards are described. All further chapters describe a certain processing step. The Chapter name is equal to the argument of the \texttt{PROCESS} card that should be given in the \texttt{input file} to switch on the processing of that step.

The step is introduced at the beginning of the chapter. In the first section the input cards are described and also an example \texttt{input file} is given. In the second section the output is described, as well with an example. In most chapters there is also a third section describing the implementation.

Cards that are mandatory are written in boldface sans serif. Optional cards are in normal sans serif.\text{\texttt{NOT\_USED2}: 0
\text{End\textunderscore process\textunderscore control}
[SKIP] [SKIP]
serif font. Parameters are notated in italic, and defaults are underlined. Optional parameters are in between square brackets.

The definitions used in the software are described in Annex B. Here, amongst others, the baseline definition, the file formats used, and normalization of polynomials are described.

In Annex ?? the installation of Doris is described. It also contains a small trouble shoot section.

Annex A shortly describe (third party) packages that should be installed for a complete version of the Doris software. Also some utilities we have developed are described.

The matrix class which comes with the Doris software is described in Annex C. This matrix class can be freely used in other non-commercial programs.

Finally, Annex D describes how to add a module to the Doris software. Extention of the software is encouraged.
Chapter 2

General Cards

This chapter deals with input cards that are not specific for a certain processing step, the general input cards. For example, such a card could specify whether you like to do batch processing or interactive processing. These cards are best placed at the start of the input file.

They do not generate any specific output. An example of the (header of the) input file is given in section 2.2.

2.1 General Input Cards

\[ ... \]
After this card everything up to a newline is ignored. A space after \ is not required.

# [ ... ]
After this card everything up to a newline is ignored. A space after # is not required.

C [ ... ]
After this card everything up to a newline is ignored. A space after this card is not required.

COMMENT [ ... ]
After this card everything up to a newline is ignored. A space after this card is not required.

SCREEN DEBUG | INFO | PROGRESS | WARNING | ERROR
This card controls the level of standard output. It is recommended to start with this card, since it is in effect only after it is read.

BEEP OFF | WARNING | ERROR | PROGRESS | ON
This card controls the level of beeping.

BATCH [ ON | OFF ]
Specifies to run the processor in non-interactive mode. If this card is omitted then the processing is done in interactive mode, asking to press a key before each step. BATCH can be specified without arguments, which means non-interactive processing. (If there is an ONLYPROCESS card present, this forces BATCH ON).

OVERWRITE [ ON | OFF ]
Specifies whether or not to overwrite existing files. If this card is omitted files are not overwritten; if no parameter is given it defaults to ON (do overwrite).

PREVIEW [ OFF | ON | XV ]
Specifies whether or not to generate SUNraster preview files with the help of the utility program cpxfiddle. (download and install separately from Doris website.) Default is OFF since this program may not be installed. If ON, shell scripts are created in the working directory which create the SUNraster file if run. If XV is given, also the command is given to view the generated file with xv.

LISTINPUT [ ON | OFF ]
Specifies if the input file has to be copied to the logfile. If this card is omitted then input is not copied. if no parameter is given it defaults to ON (do copy).

MEMORY 20
With this card the user can indicate the maximum amount of memory to be used by the processor (in Megabytes). It is advised setting this lower than the maximum available amount, because it may be somewhat inaccurate (up to a factor 2, particularly due to temporary copies created by the copy constructor). A lot of routines actually try to use a minimum of memory, even if this card is set to a large value.

PROCESS M_READFILES | M_READFILES | M_PORBITS |
M_CROP | M_OVS | M_FILTAZI | FILTRANGE |
S_READFILES | S_PORBITS | S_CROP | S_OVS |
S_FILTAZI | COARSEORB | COARSECORR |
FINE | COREGPM | RESAMPLE | INTERFERO |
COMPREFPHA | SUBTRREFPHA |
COMPREFDEM | SUBTRREFDEM |
COHERENCE | FILTPhase | DINSAR |
UNWRAP | SLANT2H | GEOCODE

With this card the processing steps that have to be processed can be switched on. More than one PROCESS card can be specified in the input file. An ONLYPROCESS card overrides possible PROCESS cards. Description of these steps can be found in the introduction, chapter 1, and in the following chapters. At least one PROCESS or ONLYPROCESS card is mandatory.

ONLYPROCESS same arguments as PROCESS card
With this card a processing step that has to be processed can be switched on. Overrides possible PROCESS cards. This card also automatically switches: BATCH ON. At least one PROCESS or an ONLYPROCESS card is mandatory.

LOGFILE

Output filename for the logfile.

M_RESFILE

Output filename for the master result file.

S_RESFILE

Output filename for the slave result file.

I_RESFILE

Output filename for the products result file.

ORB_INTERP

Orbit interpolation method. Defaults to a polynomial of degree numberofdatapoints-1, but smaller than degree 5 (order 5). Optionally, the DEGREE can be given (i=numberofdatapoints-1). The x,y,z are independently interpolated, the velocities are estimated from the position (though envisat gives you the velocities too, which could better be interpolated I guess.) If method SPLINE is selected, natural cubic splines are used. This may be inaccurate if there are only a few orbit datapoints. (default interpolation, not approximation for polyfit is used to go smoothly through the datapoints since the points do probably not contain noise since they are already the result of an orbit propagator somewhere. It is not advised really to use a DEGREE smaller than the maximum possible, except if it gets too large to avoid oscillations.)

DUMPBASELINE

Dump the baseline parameters for a grid of 0 lines by 0 pixels as INFO to stdout. The baseline is only evaluated after the orbits are known. The perpendicular baseline to the reference ellipsoid is also computed as a 2D polynomial of degree 1. And also theta as function of azimuth line and range (though it hardly varies over azimuth).

HEIGHT

Average terrain height above WGS84. This can be used in future to correct the unwrapping for the integration constant. Now it is only used if GEO card is used for cropping.

TIEPOINT

coordinates of a point in lat lon hei in WGS84. For now, only informational. The point is converted to pixel/line coordinates, and the interferometric phase is computed, etc.

M_RG_T_ERROR

[ 0.0 ]
range timing error for master. One-way in seconds. Use this card for example to calibrate the geo-referencing using a corner reflector with known coordinates. Can also be used to "shift" the DEM with respect to the interferogram in step COMPREFDEM. A shift of one (non-multilooked or oversampled) pixel corresponds to a one-way timing error of \(1/(2*RSR)\). For ers this is approximately \(M_{RG,T}\text{ERROR}=0.00000002637\) seconds. By multiplication of the signal velocity speed of light \((3e8)\) this amount in seconds can be converted to the slant-range resolution (i.e., pixel posting) of 7.9 meter.

\[
M_{AZ,T}\text{ERROR}\quad[0.0]
\]

azimuth timing error for master. Use this card to account for timing errors in azimuth direction. Card can be used to shift a DEM in azimuth direction. Note that such a shift may indicate incorrectly estimated Doppler.

\[
S_{RG,T}\text{ERROR}\quad[0.0]
\]

range timing error for slave. See \(M_{RG,T}\text{ERROR}\). Since the geometry of the interferogram is related to the master this card has not a large effect.

\[
S_{AZ,T}\text{ERROR}\quad[0.0]
\]

azimuth timing error for slave. See \(M_{AZ,T}\text{ERROR}\). Since the geometry of the interferogram is related to the master this card has not a large effect.

STOP

After this card the input file is no longer interpreted. This card is mandatory.

2.2 Example General Input Cards

```
c ************************************************** ***************
c * Doris \ inputfile generated by: run at: Nov 27, 2000 (Monday) *
c ************************************************** ***************
c *
c * Filename: Inputfiles/input.s_initial
nc * Author: Doris User
nc * Master: 23185
nc * Slave: 03512
nc * Baseline: 170 m
nc * Remarks: Test: s2h routine (exact)
c *
c ************************************************** ***************
c comments __general options__
c SCREEN debug
// level of output to standard out
```
MEMORY 150 // MB
OVERWRITE
// overwrite existing files
BATCH // non-interactive
  c LISTINPUT OFF
// prevents copy of this file to log
  c
PROCESS m_readfiles // read parameters
PROCESS m_porbits // obtain precise orbits
PROCESS m_crop // crop data to internal format
  c //
  c //
  comment ___the general io files___ //
  c //
LOGFILE log.out // log file
M_RESFILE master.out // parameter file
S_RESFILE slave.out // parameter file
I_RESFILE interferogram.out // parameter file
  c //

[SKIP][SKIP]
...
... more cards specific to step specified by (ONLY)PROCESS cards,
... see next chapters for details on these cards.
...
[SKIP][SKIP]
STOP

Note that "//" is not a delimiter for comments, text after the last expected parameter is simply ignored.
In this chapter the processing of step M_READFILES is described. It can be selected by a PROCESS M_READFILES line in the input file. This is the first step if the ERS1/2 SLC images are processed.

The SLC leader, volume and (header of the) data file are read, and relevant parameters are written to the master result file specified by the general card M_RESFILE. These parameters are used in the further processing. Currently, ERS1/2 SLC and ENVISAT SLC files can be read. If the output of this step is mimicked, Doris can be tricked to process the other steps. The sole purpose of this step is to create result file where relevant parameters are stored (PRF, wavelength, etc.), also see the example in the next section.

### 3.1 Input Cards

M_IN_METHOD  

ERS | ASAR (ENVISAT) | RSAT (RADARSAT) | ATLANTIS | JERS  
Method selector to read ERS header or ENVISAT, or RADARSAT, note that both master and slave need to have same sensor in principle. JERS simply uses ERS programs, ATLANTIS (sar processor) uses the ceos reader for RSAT, and will write this in the Product Type Specifier field. RSAT must be tested, problems may be orbit data. In later steps, the Product field is read, and the CROP step uses the appropriate function automatically (Envisat, ERS/JERS, RSAT/ATLANTIS).

M_IN_NULL  

filename  
The filename of the SLC null file. This may be a dummy name since it is not used. Not used for method ASAR (ENVISAT).

M_IN_VOL  

filename  
The filename of the SLC volume file. Not used for method ASAR (ENVISAT).

M_IN_LEA  

filename  
The filename of the SLC leader file.
Not used for method ASAR (ENVISAT).

**M_IN_DAT**

*filename*

The filename of the SLC data file. This is the only file required for method ASAR (ENVISAT).

An example of the input cards for this step is given below. This example can be inserted in the general cards described in Section 2.2.

```plaintext
M_IN_METHOD ERS
M_IN_VOL /cdrom/scene1/vdf_dat.001 // name of volume file
M_IN_LEA /cdrom/scene1/lea_01.001 // name of leader file
M_IN_NULL dummy // name of null file
M_IN_DAT /cdrom/scene1/dat_01.001 // name of data file
```

### 3.2 Output Description

The **process control flag** at the start of the **master result file** is switched to 1 at successful exit.

```plaintext
readfiles: 1
```

Example of output of this step (in **master result file**). (The positioning data of the platform from the leader file has been deleted in this example. This happens automatically after step M_PORBITS (getting the precise orbits)). This output is appended to the **master result file**.

```plaintext
***************************************************
* _Start_readfiles: *
***************************************************
Volume file: /cdrom/scene1/vdf_dat.001
Volume_ID: 1
Volume_identifier: 0004093800014027
Volume_set_identifier: 19950830 9491991
(Check) Number of records in ref. file: 26558
Product type specifier: PRODUCT:ERS-1.SAR.SLC
Location and date/time of product creation: IPAF 24-07-1998
Scene identification: ORBIT 21567 DATE 30-08-95
Scene location: FRAME 2781 LAT:40.94 LON:14.03
Leader file: /cdrom/scene1/lea_01.001
Scene_centre_latitude: 40.9380000
Scene_centre_longitude: 14.0270000
Radar_wavelength (m): 0.0566660
First_pixel_azimuth_time (UTC): 30-AUG-1995 09:49:20.453
Pulse_Repetition_Frequency (actual, Hz): 1679.9020000
Total_azimuth_band_width (Hz): 1378.0000000
Weighting_azimuth: HAMMING
```
Xtrack_f_DC_constant (Hz, early edge): 437.9780000
Xtrack_f_DC_linear (Hz/s, early edge): 7154.0000000
Xtrack_f_DC_quadratic (Hz/s/s, early edge): -380000000.00000
Range_time_to_first_pixel (2way) (ms): 5.5458330
Range_sampling_rate (leaderfile, MHz): 18.9624680
Total_range_band_width (MHz): 15.5500000
Weighting_range: HAMMING
Datafile: /cdrom/scene1/dat_01.001
Number_of_lines_original: 26183
Number_of_pixels_original: 4900
******************************************************************************
* End_readfiles: _NORMAL
******************************************************************************

Note that Product specifies "ASAR" for ASAR, which is used later.

A number of these lines is only for your information. The lines after Leader file: are used for
the further processing (except Weighting identifiers). These strings may not be altered. We
encountered a problem once when there was a blank in the UTC time, instead of a zero, but
this should be fixed.

The logfile shows more details. Also some information is echoed to the screen (as "INFO: .."),
such as the Doppler centroid frequency, evaluated at some ranges, the corners of the images
in latitude longitude, etc.

Defaults for parameters (slcimage.cc; Doris can still crash if not correct, e.g., the approximate
coordinates of the scene):

wavelength = 0.0566660.; // [m] default ERS2
t_range1 = 5.5458330/2.0e3; // [s] one way, default ERS2
prf = 1679.902.; // [Hz] default ERS2
abw = 1378.0; // [Hz] default ERS2
rsr2x = 18.9624680*2.0e6; // [Hz] default ERS2
rbw = 15.550e6; // [Hz] default ERS2

### 3.3 Implementation

The SLC file is opened in binary mode. The file pointer is set to the right position in bytes to
read a certain word (number). The SLC format is described, e.g., on the ESA website.

Then the word is read in the character array and terminated by the null character. This is
repeated for all words we want to read and for all files. A short example of the calls is shown
below.

```cpp
char c16physid[17];
volumefile.open(readfiles_arg.volfile, ios::trunc
  | ios::nocreate | ios::binary);
assert(volumefile, readfiles_arg.volfile, __FILE__, __LINE__);
volumefile.seekg(44, ios::beg);
```
```c
// physical logical volume ID
volumefile.read((char *)&c16physid, sizea16);
c16physid[16] = '\0';
```

A nicer way might be to define a struct and to read this in one go. Also see the software that esa provides on their website (c).

### 3.3.1 Changes for X86 platforms

Version 2.4 onward can be (easily) compiled on Linux systems. For little endian machines like (intel) PC’s this means the byte order is different. Since the record length in the leader, volume, and data file are stored as B4 (4bytes unsigned integers), we had to use the function ntohl, see the manual pages. routines readvolume, readleader, readdat.

Also the SLC data itself in the datafile is stored as 2x 2B short signed integer byte data. (real part, imaginary part, real, imaginary, real, imag, ...). We transform the data that is read from the data file with the function htons if _X86PROCESSOR_ is defined (see the Makefile or source code) for more information.
In this chapter the step M_PORBITS is described. This step should be run after READFILES, because in that step the azimuth time is written to the master result file from the SLC leader file.

We use the DEOS fortran program getorb for obtaining the precise orbits. (This program has to be installed separately, see http://www.deos.tudelft.nl/ers/precorbs/ or [Scharroo and Visser, 1998].) This step actually is only a system call to getorb, and converts the output to a 4 column table: secofday, x, y, z.

It requires the Orbital Data Records (ODR files) to be in an archive directory. The arclist file (which should be downloaded together with the ODR archives) has to be present in this directory. The ODR files have to be untarred and unzipped as from version 2.5.

This step introduces a section in the result file where the ephemerides are placed, and it deletes the ephemerides from the SLC leader file, obtained by the processing step M_READFILES (if there was such a section). The ephemerides (x, y, z) span the time 4 seconds before the first line and 4 seconds after the last line by default. The time is, and should be, in seconds of day. The time interval is 1 second by default. Natural cubic splines are used for interpolation, and the boundary conditions may affect the interpolation if only a few datapoints are used, e.g., 5 points with a time interval of 30 seconds. We nowadays use a time interval of 30 second, and approximately 21 points. This implies that only spline (degree 3 piecewise polynomial) is used for the whole image, which gives better results for, e.g., reference phase computation. The interpolation errors in Doris are probably due to interpolation of interpolated values of getorb, which output format is in 3 digits.

If you want to use other ephemerides you can simply insert them in the result file in the format described in section 4.2. You will have to correct the number of POINTS in the result file. Note that the orbit system is WGS84 (only).

### 4.1 Input Cards

**M_ORBDIR** 
*directory name*
the tar archive directory name for the Delft Orbital Data Records.
M_ORB_INTERVAL 1
Time in seconds between ephemerides.

M_ORB_EXTRATIME 3
Time in seconds before first and after last line to output ephemerides.
Since interpolation is done with natural cubic splines, it is advised to have at least 3 extra data points before the first and after the last line.
To use a single polynomial of degree 3 for interpolation of the orbit for the full scene, select a time interval of 20 seconds, and, for example, extra time of 200 seconds.

M_ORB_DUMP \( \text{delta_t} \)
Write interpolated orbit to ascii output file "masterorbit.dat".
With delta_t seconds interval between ephemerides. Time interval between \( t_0 \) and \( t_N \) of the precise ephemerides. output is: \( t,x,y,z,xdot,ydot,zdot,xddot,yddot,zddot \). If compiled with \_DEBUG defined, then also the matrices for spline interpolation are dumped.

Example of the cards for this step:

```
c
comment ___PORBITS___
c
M_ORBDIR /data/delftorbits/ERS1/
M_ORB_INTERVAL 1
M_ORB_EXTRATIME 6
c M_ORB_DUMP 0.1
```

4.2 Output Description

If a normal termination of this step, then the process flag at the start of the result file is switched to 1:

```
precise_orbits: 1
```

The output of this step is written in the section: precise_datapoints. This section looks like the following. It is important that all lines are present following NUMBER_OF_DATAPoints: 23.

```
*************************************************** ****************
*_Start_precise_orbits:**********************************************
*************************************************** ****************
t(s) X(m) Y(m) Z(m)
NUMBER_OF_DATAPoints: 23
35360.000000 5161849.442 1645908.227 4678710.927```
The time is in seconds of day, which can be computed as: fractional \( \text{day} \) \( \times \) 60*60*24 or hours*60*60 + min*60 + sec.

If card M\_ORB\_DUMP then an ascii file "masterorbit.dat" is written with the computed t, x, y, z, xdot, ydot, zdot, xddot, yddot, zddot.

### 4.3 Implementation

Based on the UTC time of the image and the PRF and number of lines, basically the program getorb is called through a UNIX system call. Functions from the standard library ctime (or time.h) are used for the time conversions. This call is echoed to the screen as DEBUG. The commands can be executed stand-alone as well.

```
% getodr 950727094923,950727094945,1 /data/orbits/ERS2.ARCs > dummyout
%
%(...read in ODR file name from dummyout (ODR.422))
% untar /data/orbits/ERS1.ARCs ODR.422
getorb 950727094923,950727094945,1 /data/delftorbits/ERS1/ \\
> scratchorbit
```

The ephemerides are first written to a dummy file (named "scratchorbit") and later placed in the result file (without the velocity output, only t,x,y,z). It has been noted that sometimes this scratch file is not automatically removed. This file can be safely removed by hand.
Natural cubic splines are used for the interpolation so it may be wise to have a short time-interval and some data before the first and after the last line.

If there is a section with the ephemerides of the SLC leaderfile in the master result file, then this section is removed.

In the routine splineinterpol, file utilities.c, where the coefficients are computed, \_\_NATURALSPLINE\_\_ is defined. This sets the boundary condition to use zero second derivative at the borders. Otherwise, the first derivative is set to a specified value. This does not seem to make a big difference.
In this chapter the processing of step M_CROP is described. This step normally is the second one that is run, after READFILES. It requires the SLC data file on disk or cdrom. For ENVISAT, a utility is called that does the work.

In this step the SLC datafile is put on disk in a raw (pixel interleaved, 2x2byte signed (short) integer complex) format. (The reason for this step is that we normally work with the SLC images on cdrom, and that we want to have the files on disk to perform operation requiring both the images. It also serves as a common format for different input.)

A few checks are performed regarding the number of lines, which is written in the header of the SLC data file as well as in the leader file. The image is read/written line by line, no data conversion takes place (though a cutout can be made). If you are working on a little endian platform (X86 PC) then the data is converted from big endian (which is the CEOS format).

### 5.1 Input Cards

- **M_IDCROP**
  
  master
  
  Identification of this step. this is not used in the further processing.

- **M_CROP_IN**
  
  filename
  
  Filename of the SLC data file.

- **M_CROP_OUT**
  
  master.raw
  
  Filename of the raw data output file.

- **M_DBOW**
  
  linelow linehi pixelow pixelhi
  
  Master database output window. You can make a cutout of the image with this card. Hi values larger than the size of the image are reset to the maximum. If card omitted it defaults to total image. line/pixel 1 refers to the the first line/pixel.

- **M_DBOW_GEO**
  
  lat_0 lon_0 height width
Master database output window. Alternative to and overrides normal DBOW card. You can make a cutout of the image with this card. Latitude of the center pixel of the desired crop, longitude (in decimal degrees, WGS84 system of orbit), then height, width in pixels. For approximately square areas, heights should be a factor 5 of width for ERS.

Example input cards for this step:

c
c
comment ___CROP___
c
M_IDCROP master // identifier
M_CROP_IN /cdrom/SCENE1/DAT_01.001
M_CROP_OUT Output/21066.raw
M_DBOW 1 5000 1 1000 // linelow hi pixellow/hi

5.2 Output Description

The process control flag at the start of the result file is switched to 1 at successful exit.

crop: 1

The output section in the result file will resemble the following.

********************
*_Start_crop: master
Data_output_file: Output/21066.raw
Data_output_format: complex_short
First_line (w.r.t. original_master): 1
Last_line (w.r.t. original_master): 5000
First_pixel (w.r.t. original_master): 1
Last_pixel (w.r.t. original_master): 1000
********************
*_End_crop: _NORMAL

If the SLC data is already on disk, for example because the SAR processing was done, this section will have to be simulated. (As well as the result from READFILES.) The format complex_real4 is available.

Note that the byte order must be the same as the (host) platform order. This means that if data is copied from big endian platforms, they have to be swapped. Use for example a dd command like:

dd if=/cdrom/file.slc of=file.slc conv=swab

or use gmt (-Zh for short, -Zf for float, see man pages):

xyz2grd /cdrom/file.slc -Zh -Sfile.slc
Chapter 6

M_OVS

In this chapter the processing of step M_OVS is described. This step can be run optionally to oversample the cropped data, and is done after M_CROP.

Range oversampling has been implemented by Raffaele Nutricato, who uses an oversampling factor of 4 in range, for advanced processing in the multi-temporal analysis.

For PS type processing, factor two in both directions seems reasonable. This avoid aliasing in the spectrum of the interferogram, which implies you can interpolate correctly in the interferogram.

6.1 Input Cards

M_OVS_OUT master_ovs.raw
Filename of the oversampled data.

M_OVS_FORMAT ci2
Output file format.

M_OVS_FACT_RNG 1
Oversampling factor of output image in range (pixels).

M_OVS_FACT_AZI 1
Oversampling factor of output image in azimuth (lines).

M_OVS_KERNELSIZE 16
Kernel size (sinc) used for oversampling.

Example input cards for this step:

```
c
comment ___OVS___
c
M_OVS_OUT Outdata/master_ovs.raw // output filename
M_OVS_OUT_FORMAT ci2     // output format
```
6.2 Output Description

The process control flag at the start of the result file is switched to 1 at successful exit.

```
oversample: 1
```

The output section in the result file will resemble the following.

```
*************************************************** ****************
* _Start_oversample : slave
*************************************************** ****************
Data_output_file: Outdata/slave_ovs.raw
Data_output_format: complex_short
First_line (w.r.t. original_image): 101
Last_line (w.r.t. original_image): 133
First_pixel (w.r.t. original_image): 991
Last_pixel (w.r.t. original_image): 1023
Multilookfactor_azimuth_direction: 1
Multilookfactor_range_direction: 0.25
Number of lines (multilooked): 33
Number of pixels (multilooked): 132
First_line (w.r.t. ovs_image): 101
Last_line (w.r.t. ovs_image): 133
First_pixel (w.r.t. ovs_image): 3961
Last_pixel (w.r.t. ovs_image): 4092
*************************************************** ****************
* End_oversample: _NORMAL
*************************************************** ****************
```

6.3 Algorithm

Based on description by Raffaele Nutricato who provided this code: In the code, look for:

```
// ___RaffaeleNutricato START MODIFICATION SECTION 1
```

As I explained in the previous e-mail range oversampling is obtained as convolution of the zero filled signal with a truncated sinc. The loading of the image is performed line by line and consequently the oversampling is performed line by line too.

In particular given an input signal:
```
Input signal: xxxx
```
and an oversampling ratio of let’s say 3
```
I first generate a zero-filled copy of the input signal:
zero-filled Input signal: x00x00x00x
```
then I convolve the zero-filled Input signal with the interpolation
kernel obtaining the output signal:
Output signal: x++x++x++x++
where +'s are the new samples.

Bert Kampes implemented the azimuth oversampling. In azimuth a 6 point raised cosine kernel is used. The kernel is normalized. I also normalized the range kernel (typically 16 point sinc).
In this chapter the processing of step S_READFILES is described. It is the same as step M_READFILES but then for the slave image. See chapter 3 (M_READFILES) for more information on this step.

### 7.1 Input Cards

<table>
<thead>
<tr>
<th>Card</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>S_IN_NULL filename</td>
<td>The filename of the SLC null file. This may be a dummy name since it is not used.</td>
</tr>
<tr>
<td>S_IN_VOL filename</td>
<td>The filename of the SLC volume file.</td>
</tr>
<tr>
<td>S_IN_LEA filename</td>
<td>The filename of the SLC leader file.</td>
</tr>
<tr>
<td>S_IN_DAT filename</td>
<td>The filename of the SLC data file.</td>
</tr>
</tbody>
</table>
In this chapter the processing of step S_PORBITS is described. It is actually the same as step M_PORBITS, but then for the slave image. See chapter 4 (M_PORBITS) for more detailed information on this step.

8.1 Input Cards

\textbf{S\_ORBDIR} \hspace{1cm} \textit{directory name}
\begin{itemize}
  \item the tar archive directory name for the Delft Orbital Data Records.
\end{itemize}

\textbf{S\_ORB\_INTERVAL} \hspace{1cm} 1
\begin{itemize}
  \item Time interval between data points. Card M\_ORB\_INTERVAL has the same effect. It is not possible to have a different S\_ORB\_INTERVAL.
\end{itemize}

\textbf{S\_ORB\_EXTRATIME} \hspace{1cm} 3
\begin{itemize}
  \item Time before first line (and after last) for extra datapoints data points. Card M\_ORB\_EXTRATIME has the same effect. It is not possible to have a different S\_ORB\_EXTRATIME.
\end{itemize}

\textbf{S\_ORB\_DUMP} \hspace{1cm} \textit{dt}
\begin{itemize}
  \item Dump interpolated t,x,y,z to ascii file slaveorbit.dat.
\end{itemize}
In this chapter the processing of step S_CROP is described. It is the same as step M_CROP but then for the slave image. See chapter 5 (M_CROP) for more information on this step.

### 9.1 Input Cards

- **S_IDCROP**  
  *slave*  
  Identification of this step. This is not used in the further processing.

- **S_CROP_IN**  
  *filename*  
  Filename of the SLC data file.

- **S_CROP_OUT**  
  *slave.raw*  
  Filename of the raw data output file.

- **S_DBOW**  
  *linelow linehi pixellow pixelhi*  
  Slave database output window. You can make a cutout of the image with this card. If card omitted it defaults to total image. line/pixel 1 refers to the the first line/pixel.
In this chapter the processing of step S_OVS is described. It is the same as step M_OVS but then for the slave image. See chapter 6 (M_OVS) for more information on this step.

10.1 Input Cards

S_OVS_OUT  
filename of the oversampled data.

S_OVS_FORMAT  ci2
output file format.

S_OVS_FACT_RNG  1
oversampling factor of output image in range (pixels).

S_OVS_FACT_AZI  1
oversampling factor of output image in azimuth (lines).

S_OVS_KERNELSIZE  16
Kernel size (sinc) used for oversampling.
Chapter 11

COARSEORB

This chapter describes the processing step COARSEORB. In this step the coregistration based on the orbits of slave and master is computed with an accuracy of about 30 pixels (precise orbits). This is a fast way to get the coarse offsets. Before the FINE coregistration however the step COARSECORR has to be run, in order to get the coarse offsets within a few pixels. (FINE requires the initial estimated offset within a few pixels.)

The offset is defined in such a way that for a point P in the master with coordinates \( P_m(\text{line, pixel}) \) and the same point in the slave image with (slave system) coordinates \( P_s(\text{line, pixel}) \) it holds:

\[
P_s(l, p) = P_m(l, p) + \text{offset}(l, p)
\]  

11.1 Input Cards

There are no input cards for this step. (i.e. the parameters/orbits are read from the master and slave result file.)

11.2 Output Description

Since this normally is the first step that is not specific to master nor slave, a products result file is created. The process control flag at the start of this file is switched to 1 at successful exit.

\[
\text{coarse_orbits: 1}
\]

Example of output of this step.

```
*************************************************** ****************
* _Start_coarse_coregistration_based_on_orbits
*************************************************** ****************
Some info for pixel: 3037, 590 (not used):
Bperp [m]: 36.1
```
In the logfile some extra information is given, such as the number of iterations. The baseline parameters are not used, but given here to make it possible to write scripts that grep these values if a lot of interferograms are processed of the same scene.

### 11.3 Implementation

The algorithm described in Annex B is used for the conversion between (line,pixel) coordinates to the corresponding point P on an ellipsoid. (The Doppler, range and ellipsoid equation.) This step consists of three steps basically.

1. For the center (line,pixel) of the master image, compute the position \((x,y,z)\) in system of the orbits of the point \(P\) on an ellipsoid.
2. Based on the Doppler equation, compute the position of the slave satellite, corresponding to the point \(P\) on an ellipsoid, and compute the (line,pixel) coordinates in the slave system.
3. The difference (slave-master) between the (line,pixel) coordinates is defined as the offset.
In this chapter the processing of step COARSECORR is described. The offset in line (azimuth) and pixel (range) direction between master and slave is computed with an accuracy of about 1 pixel (1 offset for whole image).

The magnitude images are used; correlation is computed in the space or spectral domain.

At a number of positions (geometrically distributed or at positions read from an input file) in the image the correlation between master and slave is computed for different offsets. The offset with the highest correlation is the estimate for that position. The (approximate) offset between the two images is set to the offset that most occurred over the positions, so the one that is most likely. (Sometimes an estimated offset is totally unreliable, for example for a position in a sea, but the correlation is not very small. The estimated correlation at a position is likely to be biased. Therefore it would not be wise to use the offset between the two images based on the highest correlation values only, but we use this ‘consistency test’ instead.)

12.1 Input Cards

**CC_METHOD** *magfft | magspace*
Method selector for this step. Either perform the correlation computation on the magnitude images in the space or in the spectral domain.

**CC_IN_POS** *filename*
Input filename for ascii file with positions in original master system to place windows for correlation computations.

**CC_NWIN** *11*
Number of windows to be distributed over the total image to estimate the offset. Should be at least 5 or so because the most consistent estimate is selected. This card is ignored if CC_IN_POS is set. Only 1 large window could be used, e.g., of size 1024x1024.

**CC_WINSIZE** *64 64*
Size of the window in lines pixels. For method in space domain it defaults to 64 64. For method in space domain it is converted to odd numbers if necessary.

CC_ACC \[32 8\]
ONLY for method in space domain. Accuracy to search within for maximum correlation. For fft method it automatically equals half of the CC_WINSIZE.

CC_INITOFF \[0 0 | \text{"orbit"}\]
Initial offset for coarse co-registration. if the word "orbit" then the estimate of the step COARSEORB are read from the products result file and used. If there are 2 numbers then these are used.

Example input cards for this step:

| c | comment ___COARSE CORR (COREGISTRATION)___ |
| c |
| CC_METHOD | magfft | // default |
| CC_METHOD | magspace | // (no veclib) |
| CC_ACC | 30 30 | // (only for magspace) |
| CC_NWIN | 21 | // number of windows |
| CC_WINSIZE | 1024 512 | // size of windows |
| CC_INITOFF | orbit | // use result of orbits |
| c CC_INITOFF | 0 0 | // for initial offset |
| c CC_INITOFF | 0 0 | // use this if no precise orbits |

### 12.2 Output Description

At successful exit, the process control flag is switched to 1 in the products result file. If this file does not exist, it is created (I_RESFILE card):

| coarse_correl: 1 |

The output in the (products) result file resembles:

```
******************************************************************************
* _Start_coarse_correlation
******************************************************************************
Estimated translation slave w.r.t. master:
Coarse_correlation_translation_lines: 241
Coarse_correlation_translation_pixels: 3
******************************************************************************
* End_coarse_correlation: _NORMAL
******************************************************************************
```

In the logfile the estimated offset is given for all windows.
12.3 Implementation

12.3.1 Method magspace

The implementation in the space domain requires an odd window size, which is automatically forced (not strictly necessary, but this made the implementation a bit easier because the center of the shifting window is defined at a pixel.) For each location the (zero meaned) slave magnitude window is shifted over the (zero mean) master window, and the correlation is computed (see equation B.24) by computing all pointwise products and dividing by the norms of the particular windows.

12.3.2 Method magfft

The implementation in the frequency domain is more or less the same as in the space domain. We only use FFT's to compute the products for the correlation (see equation B.24) in an efficient way, due to the fact that a convolution in the space domain corresponds to a multiplication in the frequency domain. Input are the zero mean magnitude images.

The cross products are obtained by computing the pointwise product of the zeropadded master x conj(slave). A block function is used to compute the norms. Note that the correlation window (the overlap) does not have a constant size with this method, but varies between winsizeL/P and .5winsizeL/P.
In this chapter the processing of step M_FILTAZI is described. This optional step filters the spectrum of the master in azimuth direction. The part of the spectrum that does not overlap with the spectrum of the slave is filtered out. This non overlap is due to the selection of a Doppler centroid frequency in the SAR processing, which normally is not equal for master and slave image.

This step can in general best be performed after the COARSE coregistration and before the FINE. (The coarse offset in pixel direction is used to evaluate the polynomial for the Doppler Centroid frequency.) The FINE steps can benefit a lot from this filtering (TODO add plots).

By processing the RAW data to SLC at the mean Doppler centroid frequency this step can be avoided in the InSAR processing chain. (For ESA SLC images this cannot be done obviously.)

Normally the step S_FILTAZI is performed at the same time. (requires a PROCESS S_FILTAZI card in the input file, see chapter 14.) However, we kept this two seperate steps to be able to only filter the slave images in a large stack (all slaves coregistered on the same master image). This has the advantage that for each interferogram of the stack not a large file is created for the master. The disadvantage of not filtering the master of course is that a small part of the spectrum of the master is not shared with the slave spectrum, yielding loss of coherence in the interferogram.

### 13.1 Input Cards

- **AF_BLOCKSIZE 1024**
  Length of fft per buffer in azimuth direction. In general, the larger the better.

- **AF_OVERLAP \(AF\_BLOCKSIZE/8\)**
  Half of the overlap between consecutive buffers in azimuth direction. Partially the same data is used to estimate the spectrum, which might have certain advantages. However it has not been studied yet if taking an overlap is required. Setting this card to 0 is fastest.

- **AF_HAMMING 0.75**
The weighting of the spectrum in azimuth direction. The filtered output spectrum is first de-weighted with the specified hamming filter, then re-weighted with a (newly centered) one. If this parameter is set to 1, no weighting is performed. For more information see 18.3.3.

AF_OUT_MASTER  \textit{master.afiltered}

Output file name for the master image.

AF_OUT_SLAVE  \textit{slave.afiltered}

Output file name for the slave image.

AF_OUT_FORMAT  \textit{cr4}

Format of outut data. Either complex real4 (cr4) or complex shorts (ci2).

An example of the \textbf{input file} (save general cards):

\begin{verbatim}
PROCESS m_filtazi
PROCESS s_filtazi
  c  //
  c  //
  comment ___AZIMUTH FILTERING___  //
  c  //
  c  AF_METHOD
AF_BLOCKSIZE 1024  // fftlength each column
AF_OVERLAP 64    // hbs
AF_HAMMING 0.75
AF_OUT_MASTER Outdata/1393.azifilt
AF_OUT_SLAVE  Outdata/21066.azifilt
AF_OUT_FORMAT ci2
\end{verbatim}

13.2 Output Description

In the process control array, the switch for azimuth filtering is turned on:

\begin{verbatim}
  filt_azi: 1
\end{verbatim}

In the \textbf{master result file} a section is added with the new file name:

\begin{verbatim}
******************************************************************************
*_Start_filt_azi:******************************************************************************
Input_file:  Outdata/1393.raw
Data_output_file:  Outdata/1393.azifilt
Data_output_format:  complex_real4
First_line (w.r.t. original_master):  1
Last_line (w.r.t. original_master):  3500
First_pixel (w.r.t. original_master):  1
Last_pixel (w.r.t. original_master):  500
******************************************************************************
\end{verbatim}
Figure 13.1 Azimuth filtering for a master (left) and slave (right) SLC image (frame 2781, orbit 1393 (master, ERS2, 27-JUL-1995) and orbit 21066 (slave, ERS1, 26-JUL-1995)). The Doppler centroid frequency for the master is $f_{DCm} = 117$ Hz (constant for all columns), for the slave $f_{DCs} = 425$ Hz, (obtained from the result file (read from SLC leader)). The mean Doppler centroid equals $f_{DC} = 271$ Hz. (Doppler centroid are indicated by dashed magenta lines, x axis are frequencies from [-PRF/2:PRF/2].) The azimuth spectrum was weighted with a Hamming window ($\alpha = 0.75$). (Pictures on first row, original spectra for range column 101, red dashed line is a 51 point moving average). The filtering (middle row) first de-weights by ‘inverse’ Hamming, centered at the image Doppler centroid, and bandlimited to the total azimuth bandwidth ($ABW = 1378$ Hz). Next a new Hamming filter is applied, centered at the mean Doppler centroid, and bandlimited to $ABW - 2 \parallel f_{DCm} - f_{DC} \parallel = 1070$ Hz. Obviously, the filter for the slave is the inverse of that of the master. The resulting spectra are shown in the bottom row. The frequencies that did not overlap are filtered out, yielding a better coherence between master and slave image. The spectrum and filters depicted here are FFT shifted for clarity.

A file (mph) is created for the master, with filtered spectrum. Figure 13.1 demonstrates the filter for 2 images:

13.3 Implementation

For each buffer of AF_BLOCKSIZE lines and width pixels do (taking care of AF_OVERLAP)

- Take 1DFFT in azimuth direction (over the columns).
- if the Doppler centroid frequencies do not vary per column, use the same filter for all columns, else compute the correct filter foreach column and use that. (First coarse coreg, align, then evaluate FDC polynomial.)
- Take inverse 1DFFT in azimuth direction (over the columns), yielding the output.

The azimuth spectrum is also weighted for the antenna pattern,
\[ \frac{\sin((f_a - f_{DC})/f_{Dop})}{((f_a - f_{DC})/f_{Dop})^2} \pi \] (13.1)

Where:

\( f_{Dop} = 1505 \text{ Hz} \), the Doppler bandwidth, see [Geudtner, 1996].

We did not de-weight (and re-weight) the spectrum for this. This might be visible in figure 13.1 as a slightly asymmetric spectrum, for master and slave. We are not convinced that this re-weighting can be performed, without changing the signal. However we believe that possible errors are small.
In this chapter the processing of step S_FILTAZI is described. Normally the step S_FILTAZI is performed at the same time as M_FILTAZI. (PROCESS M_FILTAZI card in input file.) However, we kept this two separate steps to be able to only filter the slave images in a large stack (all slaves coregistered on the same master image). This has the advantage that for each interferogram of the stack not a large file is created for the master. The disadvantage of not filtering the master of course is that a small part of the spectrum of the master is not shared with the slave spectrum, yielding coherence loss of the interferogram.

Further information on the input/output of this step can be found in Chapter 13 (M_FILTAZI).
In this chapter the processing of step FINE is described. The offset vectors to align the slave image to the master are computed with sub pixel accuracy for a number of locations in the master. Over the total image, for a large number of windows (e.g., 500, distributed by Doris or from a file with locations in the master coordinate system), the offset between master and slave is estimated by computing the correlation of the magnitude images for shifts at pixel level. Next, in a local neighborhood of the maximum (correlation at pixel level) these correlations are harmonically oversampled (interpolated, requires FFT) to find the maximum at sub pixel level. These offsets are then written to the (products) result file. The offset is computed in the spectral or in the space domain (which is implemented to avoid the use of FFT, but that is required later anyway, and to provide a check of the method in the spectral domain, which should be faster). The correlation is computed on the magnitude images. Though we believe this to be a good method, we would like to investigate first oversampling the images itself, and directly computing the correlation for a small number of shifts (assuming initial offsets are known within a few pixels), as we suspect that there may be an error introduced due to aliasing with the method that is implemented. (This method will be named ‘oversample’.)

The actual computation of the transformation model (2d polynomial) is done by the step COREGPM (computation of coregistration parameters). See also [Samson, 1996].

15.1 Input Cards

FC_METHOD 

Select method for the computation. Compute cross-correlation based on magnitude images either in the space or the spectral domain. Magnitude patches are zero-meaned. Method magfft is fast, but patch size varies depending on shift. Magspace keeps constant patch-size and shifts it over the master, but is slower. Computations are done in space domain. Method oversample is best, theoretically, avoids aliasing of spectrum when magnitude is computed (using FFTs).

FC_NWIN 400

The number of windows to be distributed over the total image. If points are read from file (FC_IN_POS), then this card is ignored.
Figure 15.1  Plot produced by the command 'plotoffsets interferogram.out 11 6000 21 1000 0.6 Outdata/1393.raw' (keycard FC_PLOT 0.6 BG). The magnitude is plotted in the background. Correlation is indicated by the size of the circles, estimates with a correlation below 0.6 are filtered out.

FC_IN_POS  
file name
A ascii file with (integer line pixel pairs) coordinates in the original master coordinate system with locations where the windows should be placed. After the last coordinate there should NOT be a EOL (enter) (though Doris should ignore this). The coordinates should be within the current overlap of master and slave.

FC_WINSIZE  32 32
The size of the correlation window. Recommended is 64 64

FC_ACC  4 4
The search accuracy for the maximum correlation. Advised is 8 8. (total search area is from -Acc to +Acc). For FFT methods this must be a power of 2. In the logfile after step COARSECORR the variation of the initial offsets w.r.t. the estimated values can be seen. If this variation is larger than 1 (1 is normal for ERS1/2 SLC images) then one should select a bigger window and a larger search accuracy.

FC_INITOFF  0 0 | COARSECORR
The initial offset between master and slave. "COARSECORR" indicates that the results of the step COARSECORR are used.

FC_OSFACCTOR  16
The oversampling factor for the harmonic interpolation of the correlation. Recommended is 32 to co-register the images within a tenth of a pixel.
Call gmt script plotoffset to plot results and to view with gv. (An example of a plot is given above.) This script gets the section with estimated fine offsets from the interferogram result file. The argument threshold filters out estimates with a correlation below this threshold. The second argument (BG or NOBG) selects a call to cpxfiddle to generate a magnitude background, while BG does call cpxfiddle. See the script plotoffsets and the c program cpxfiddle for more information. cpxfiddle can be downloaded from Doris internet pages. The command is echoed to stdout as INFO, which can be repeated outside Doris. Before running the step COREGPM to estimate a transformation model, it is very convenient to view a number of offset vectors above a correlation threshold to select the appropriate value for the card CPM_THRESHOLD. Actually, a background call is made to the script ‘plotoffsets’ (something like: ‘plotoffsets interferogram.out 11 6000 21 1000 0.6’). This command can be given from the prompt as well, for different values of the threshold.

With a command like:

```
awk 'BEGIN{ for (i=100;i<25200;i=i+500) 
{ for (j=750;j<5400;j=j+200) 
{ printf "%i %i \n",i,j} } exit}'
```

the file for FC_IN_POS can be easily generated for a grid of locations.

Example input cards for this step:

```
c  
c  comment ___FINE COREGISTRATION___  
c  
FC_METHOD oversample  //  
c FC_METHOD magfft  //  
c FC_METHOD magspace  //  
FC_NWIN 101  // number of windows  
FC_WINSIZE 64 64  // size of windows  
FC_ACC 8 8  // search window, 2^n  
FC_INITOFF coarsecorr  // use result of coarse to compute first  
FC_OSFACOR 32  // oversampling factor
```

15.2 **Output Description**

The process control flag at the start of the products result file is switched to 1 at successful exit.

```
fine_coreg: 1
```

Example of output of this step (products result file).

```
 *************************************************** ****************
```
In the logfile addition information is given.

15.3 Implementation

The current names for the master and slave image are read from the result files, crop section. Here also the dimensions of the files are read. This can be checked with the debug version of Doris. Doris can be tricked to coregister other complex files, e.g., complex interferograms for 4 pass differential interferometry, by substitution the right parameters in that section.

15.3.1 magspace

The computations are similar to the COARSECORR magspace method.

15.3.2 oversample

See source code.

15.3.3 magfft

The correlation is computed at pixel level, similar to step COARSECORR. These computations are described in that chapter. (That step still has to be performed because the FINE step requires accurate initial estimates of offsets. The AccL and AccP cards define the size of the searchwindow (2*AccL x 2*AccP) around the initial offsets to interpolate a maximum.)

The oversampling is done as follows:

1. Transformation to spectral domain of searchwindow with correlation values at pixel level.
2. Padd with zeros, half the last term.
3. Inverse transform.
4. Find maximum in space domain, this corresponds to estimated offsetvector.

Note that this way of computing is exactly the same if you first interpolate the signal and compute all correlations and find the maximum, or that you first compute at pixel level and interpolate the correlation values.
This chapter describes the processing step COREGPM is described (coregistration parameters, computation of a polynomial that models the alignment of slave on master).

Based on the estimated offsets computed in step FINE, a 2d-polynomial model of certain degree of the coregistration is computed. A least squares solution is used, solution by cholesky decomposition of the normal matrix. Data may be excluded a priori by setting a threshold value for the correlation. Data can also be excluded by editing the products result file and artificially decrease the correlation for a certain offset window.

After the computations, the residuals between the estimated model and the 'observed' offsets are plotted with the csh-script *plotcpm*. These plots are useful to iteratively come to a good transformation model (changing CPM_THRESHOLD or CPM_DEGREE for each iteration, or identify and remove some estimated offsets (the 'observations', blunders) by setting their correlation to 0.000001 in the output section of the FINE processing step.

Also the observations itself and some statistics are plotted (w-tests, a large value indicates an unreliable estimate).

The script can be adapted to your own wishes, it simply calls GMT (see [Wessel and Smith, 1998]) based on the ascii data file CPM_DATA.

This step is important, since the interferogram is sensitive to misalignments of slave on master. Therefore, we always took a very cautious approach. However, that meant running this step, editing the result file, running again, etc. which got quite cumbersome. To reduce the manual effort, we introduced a card CPM_MAXITER that performs a number of iterations automatically. It also should remove no more windows than necessary for a good fit. I have experimented with values like 20 for this card. (having say 600 windows after step FINE). If you want to approach that after each computation you want to have full control what to do, simply set this card to 0.

The first run of coregpm for the area of Fig. 19.1 is shown in Figures 16.1, 16.2, and 16.3. We have used a polynomial of degree 1, and a threshold of 0.4 here.

The second run of coregpm is shown in Figures 16.4, 16.5, and 16.6. In the products result file the outliers are artificially set to 0 correlation (thus being below the threshold), to exclude them from the least squares estimation. After this run we continued with the resampling.
Figure 16.1 Plot produced by 'plotcpm' for the first run. The estimated offsets are plotted here (normalized), together with a (90 degrees rotated) w test as ellipses.

Figure 16.2 Plot produced by 'plotcpm' for the first run. The absolute error (estimated offsets minus observed offsets) are plotted for azimuth direction.
Figure 16.3  Plot produced by 'plotcpm' for the first run. The absolute error (estimated offsets minus observed offsets) are plotted for the range direction.

Figure 16.4  Plot produced by 'plotcpm' for the second run. The estimated offsets are plotted here (normalized), together with a (90 degrees rotated) w test as ellipses.
Azimuth_direction

Figure 16.5 Plot produced by ‘plotcpm’ for the second run. The absolute error (estimated offsets minus observed offsets) are plotted for azimuth direction.

Range_direction

Figure 16.6 Plot produced by ‘plotcpm’ for the second run. The absolute error (estimated offsets minus observed offsets) are plotted for the range direction.
Figure 16.7 Plot produced by ‘plotcpm’. The magnitude is plotted in the background.

Degree $d=1$ is enough to account for most effects in normal images. The 2d-polynomial has the form:

$$f(x, y) = \sum_{i=0}^{d} \sum_{j=0}^{i} \alpha_{i-j,j} x^{i-j} y^{j}$$

(16.1)

Perhaps one might do the resampling with a lower quality polynomial, and thereafter do the fine coregistration (initial offsets 0,0) and this step (yielding new coefficients). The polynomial coefficient can then be added (?) to form a new model, with which the slave can again be resampled. This has not been tested.

### 16.1 Input Cards

**CPM_THRESHOLD  0.4**
Threshold for correlation value to use estimated offset of step **FINE** in estimation of polynomial coefficients. This depends on the size of the window during FINE. Estimated coherence using small windows are more biased towards 1.0, so a higher threshold is better. For window size 64 64 a threshold 0.2 seems OK. The plotoffsets script can be used from the prompt to figure out a good threshold value.

**CPM DEGREE  1**
Degree of 2d-polynomial. See annex for definition of degree. Degree 2 is advised.
CPM_DUMP OFF | ON
Dump computed model to files in float format. Filename for azimuth model is offsetazi_#l,##p.r4 (where number of lines,pixels are substituted). Filename for range is similar. Content of file is evaluated model in master system. via INFO the dimensions are also echoed.

CPM_PLOT NOBG | BG
Call gmt script plotcpm to plot results and to view with gv. (An example of the plots is given above.) The argument NOBG prevents a call to cpxfiddle to generate a magnitude background, while BG does call cpxfiddle. See the script plotcpm and the c program cpxfiddle for more information. cpxfiddle can be downloaded from Doris internet pages. The command is echoed to stdout as INFO, which can be repeated outside Doris.

CPM_WEIGHT BAMLER | NONE | LINEAR | QUADRATIC
Experimental card. Weight estimated offsets (observations) based on correlation in least squares solution. weighting option Bamler was added in v3.16 and made the default (recommended). The theoretical precision of shift estimation using coherent patches is the basis of this weighting option.

CPM_MAXITER 10
Number of outlier to remove automatically based on outlier test. The least squares adjustment is repeated, until all tests are accepted, or the max. number of iterations is reached.

CPM_K_ALPHA 1.97
Critical value of outlier detection. A higher value accepts more outliers. This value can be found as the sqrt of normal distribution. if you want a level of significance for the outlier test of 0.05, then look the value up under a half sided test.

Example input cards for this step:

```
c
comment ___COMPUTE COREGISTRATION PARAMETERS___

c
CPM_THRESHOLD 0.4
CPM_DEGREE 2
CPM_WEIGHT linear // none
CPM_WEIGHT quadratic // none
CPM_MAXITER 20
CPM_PLOT NOBG
```
16.2 Output Description

The plots are made if `CPM_NOPL T` is not set. The plotcpm uses a file `CPM_DATA` which is created in the working directory containing the data to be plotted.

The **process control flag** at the start of the **products result file** is switched to 1 at successful exit.

```
comp_coreg: 1
```

Example of output of this step (**products result file**).

```
***************************************************
* Start_coregpm:
***************************************************
Degree_cpm: 1
Estimated_coefficientsL:
  2.41088165 e+02 0 0
  -1.48768713 e-05 1 0
  -1.75315145 e-05 0 1
Estimated_coefficientsP:
  3.11544442 e+00 0 0
  7.39316101 e-06 1 0
  1.91161205 e-04 0 1
***************************************************
* End_coregpm: _NORMAL
***************************************************
```

In the logfile some additional statistical information is written. The standard deviation of the estimates and the residuals after the least squares adjustment.

An ascii file `CPM_Data` is created with some information for the plotcpm. An example is shown below:

```
File: CPM_Data
This file contains information on the least squares estimation of the coregistration parameters.
This info is used in the plotting scripts.
There are 10 columns containing:
Window number, position L, position P,
  offsetL (observation), offsetP (observation), correlation,
  estimated errorL, errorP, w-test statistics for L, P.
win posL posP offL offP corr eL eP wtstL wtstP
---------------------------------------------------
0  268  30   -241.06   -3.19   0.42   0.08   0.19  81.23  200.79
1  268  369  -241.00  -3.31   0.55   0.01   0.24  15.73  249.91
3  268 1048  -241.00  -3.38   0.46   0.01   0.16  11.35  170.22
5  268 1726  -242.38  -3.31   0.47  1.39  0.05 1443.57  53.66
8  268 2744  -240.69  -3.56   0.71   0.02   0.02 323.20  19.65
[SKIP][SKIP]
492 14974 1260  -241.12  -3.38   0.41   0.14   0.18 150.86  184.45
```
16.3 Implementation

The observation equations are given by the polynomial model \((y = A \cdot x)\):

\[
\begin{bmatrix}
  y_1 \\
  y_2 \\
  \vdots \\
  y_N
\end{bmatrix} = \begin{bmatrix}
  1 & l_1 & p_1 & l_1^2 & \cdots & p_1^d \\
  1 & l_2 & p_2 & l_2^2 & \cdots & p_2^d \\
  \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
  1 & l_N & p_N & l_N^2 & \cdots & p_N^d
\end{bmatrix} \begin{bmatrix}
  \alpha_{l=0,p=0} \\
  \alpha_{10} \\
  \alpha_{01} \\
  \alpha_{20} \\
  \vdots \\
  \alpha_{0d}
\end{bmatrix}
\]

(16.2)

Where:

- \(y\) contains the observed offsets in a certain direction.
- \(l_i\) denotes the location (line number) of the observed offsets in a certain direction.
- \(p_i\) denotes the location (pixel number) of the observed offsets in a certain direction.
- \(\alpha_{lp}\) denotes the unknown coefficients of the polynomial.

The data is rescaled (to the interval \([-2, 2\]), see Annex B) so the normal matrix is rescaled. Otherwise there could occur very high values for, e.g., \(l^d = 25000\). The least squares parameter solution is given by:

\[
A^TQ_y^{-1}y = A^TQ_y^{-1}Ax = Nx
\]

(16.3)

Where:

- \(Q_y^{-1}\) is the (diagonal) covariance matrix of the observations. This matrix can be equal to identity or to the correlation values in version 1. (CPM_WEIGHT card).

The coefficients are estimated by factorization of the matrix \(N\).

The inverse of matrix \(N\) is also computed to check the solution (stability) and to compute some statistics.

A check number is given \(\max(\text{abs}(NN^{-1} - I))\) that gives a hint on the stability of the solution.
In this chapter the resampling, or interpolation, of the slave image on the master grid is described. The slave image is resampled (reconstruction of original signal from the samples by correlation with interpolation kernels in space domain) accordingly to the transformation model from the step `COREG_PM`. This model states with sub-pixel accuracy which points of the slave correspond to the master grid.

**Note:** This step may be fairly time consuming.

The evaluation of the co-registration polynomial could be done faster, not point by point but in a matrix. One could change the code to optimize this.

The spectrum in azimuth can be centered at zero before resampling, and shifted back to its original Doppler centroid frequency afterwards. This is required since the spectrum of the kernel function is centered at zero. See also [Geudtner, 1996]. This shifting has been implemented in release 3.0, but not in prior versions.

The polynomial described by the strings in the *slave result file* (Xtrack.f,DC,constant, etc.) is used. If the spectrum should be shifted, use the card `RS_SHIFTAZI`.

New in v3.4 is that the azimuth kernel is shifted to the Doppler centroid, not as before the dataspectrum to zero and back. This is made default.

To assess the quality of the resampling, the resampled slave image can again be coregistered (step FINE) onto the master. This should yield offset vectors that are normally distributed with zero mean. The slave image could also be resampled in steps, first resampling it by a first degree model, then again with a higher degree model.

### 17.1 Input Cards

<table>
<thead>
<tr>
<th>RS_METHOD</th>
<th>RECT</th>
<th>TRI</th>
<th>CC4P</th>
<th>CC6P</th>
<th>TS6P</th>
<th>TS8P</th>
<th>TS16P</th>
<th>KNAB6</th>
<th>KNAB8</th>
<th>KNAB10</th>
<th>KNAB16</th>
<th>RC6P</th>
<th>RC12P</th>
</tr>
</thead>
</table>
Select kernel for interpolation. A simple step function (nearest neighbor),
or a linear interpolation (tri), or cubic convolution kernel (4 or 6 point),
or a knab sampling window using a default data oversampling factor, or
a Raised Cosine (best) kernel (6 or 12 points), or a truncated sinc (6, 8
or 16 point) can be used.

RS_OUT_FILE _s_resampled.raw_
Output filename of resampled slave, cannot be equal to the input file
name.

RS_OUT_FORMAT ___CR4 | CI2___
Output format of resampled slave, complex_real4 or complex_short (same
as SLC input format, this causes an error of maximum about 1 percent (?)).

RS_DBOW _________linelo linehi pixello pixelhi________
Data base output window (in master coordinate system) for slave to make
a cutout. If card is omitted it defaults to the overlap between master
and slave (and corrected for half the kernel size where no interpolation
is possible). For stacking of interferograms on top of each other, use the
coordinates of the master after cropping. In this way all interferograms
are automatically aligned. If the slave image is smaller than the window,
the pixels are set to 0.

RS_SHIFTAZI _______ [ ON | OFF ]________
If ON, then it is accounted for non centered azimuth spectrum of the
data. The azimuth interpolation kernel is shifted to the Doppler center
frequency before resampling. If the fDC is small, users could switch this
card to OFF.

Example input cards for this step:

c
c
comment ___RESAMPLING SLAVE___
c
RS_METHOD cc4p
RS_METHOD cc6p
c RS_METHOD ts6p
c RS_METHOD ts8p
c RS_METHOD ts16p
RS_OUT_FILE Output/01393. resampled
RS_OUT_FORMAT ci2
RS_DBOW 1001 2105 501 700

17.2 Output Description

The process control flag at the start of the slave result file is switched to 1 at successful
exit.
Example of output of this step (in *slave result file*).

```
resample: 1

Example of output of this step (in slave result file).

************************************************************
*_Start_resample
************************************************************

Data_output_file: Output/01393.resampled
Data_output_format: complex_short
Interpolation kernel: 6 point cubic convolution.
First_line (w.r.t. original_master): 1001
Last_line (w.r.t. original_master): 2105
First_pixel (w.r.t. original_master): 501
Last_pixel (w.r.t. original_master): 700

************************************************************
*_End_resample:_NORMAL
************************************************************
```

Note that the line and pixel numbers are given in the master coordinate system, because the slave is interpolated to that grid now.

### 17.3 Implementation

The overlap between slave and master is computed as indicated in figure 17.1. This is not a good way if the angle between master and slave is large, but for ERS1/2 this is not the case.

Interpolation is done with a kernel function such as, e.g., a truncated sinc function. First a look-up table is computed for the selected interpolation kernel. This table evaluates the kernel every 1/INTERVAL = 0.05 positions, which should be accurate enough.

Interpolation is independent for azimuth and range direction. For all points in the overlap between master and slave, the co-registration polynomial is evaluated. This is done per point at the moment, while it is more efficient to evaluate a grid. We plan to do some tests in future to see if this can speed up this step. Then the correct window from the slave is put in a matrix and the correct interpolation function is fetched from the lookup table (for both directions). Multiplication of the window with the interpolators yields the interpolated value. Effectively this means the real and imaginary part are interpolated independently.

The slave image is put in a buffer before processing. There is an overlap with the last buffer of FORSURE=4 pixels. In case of memory trouble (segmentation faults, or the matrixclass says out of bounds (if compiled with debug option)) you may want to try to increase this value. It seems this occurs sometimes when the co-registration polynomial has a high degree (3 or larger). I could not find a bug in the code explaining these errors.

If the total image does not fit in the memory, processing is done in buffers.

The azimuth spectrum of the complex SLC data is not centered around zero in general. The location of the peak in the spectrum is given by a polynomial in the header file.
A simple derivation shows how the kernel should be shifted. Suppose we have a signal:

\[ s(x) = \exp(i \cdot 2 \pi x \cdot \frac{f_{dc}}{prf}), \]

and we want to interpolate this signal at \( xi = 5.1 \) with a triangular kernel \( k(x_0) \), such that the interpolated signal

\[ s_i(xi) = \text{sum}(k(x_0) \cdot s(x_1)). \]

This means with \( x_0 = [-0.1, 0.9] \) and \( x_1 = [5, 6] \) (as implemented in Doris), the kernel is formed as

\[ k(x_0) = \text{triangle}(x_0) = [0.9, 0.1] \]
and we shift this with the MINUS sign by multiplication with 
\[ t(x_0) = \exp(-i \cdot 2\pi \cdot x_0 \cdot \frac{fdc}{prf}). \]

Then the interpolated value at 5.1 equals:

\[
s_i(5.1) = k(-0.1) \cdot t(-0.1) \cdot s(5) + k(0.9) \cdot t(0.9) \cdot s(6) = 0.9 \cdot \exp(-i \cdot 2\pi \cdot 0.1 \cdot \frac{fdc}{prf}) \cdot \exp(i \cdot 2\pi \cdot 5 \cdot \frac{fdc}{prf}) + 0.1 \cdot \exp(-i \cdot 2\pi \cdot 0.9 \cdot \frac{fdc}{prf}) \cdot \exp(i \cdot 2\pi \cdot 6 \cdot \frac{fdc}{prf}) = \exp(i \cdot 2\pi \cdot \frac{f}{prf}) = s(5.1); // perfect interpolation!
\]

On the contrary, when we use the PLUS, it follows that

\[
s_i(5.1) = 0.9 \cdot \exp(i \cdot 2\pi \cdot -0.1 \cdot \frac{fdc}{prf}) \cdot \exp(i \cdot 2\pi \cdot 5 \cdot \frac{fdc}{prf}) + 0.1 \cdot \exp(i \cdot 2\pi \cdot 0.9 \cdot \frac{fdc}{prf}) \cdot \exp(i \cdot 2\pi \cdot 6 \cdot \frac{fdc}{prf}) \neq s(5.1); // wrong sign used!
\]

See also [Hanssen and Bamler, 1999].

17.3.1 Output formats

Computations are done in complex float. Casting these values to complex short format introduces an error. Of course, the main advantage is a factor 2 reduction in the size of the output file.

Now an example follows for the error in amplitude and phase for a complex value of about (100,100). If the actual interpolated complex value equals (100.5,100.5), then the error in the magnitude approximately is

\[
e_m \approx 100(\sqrt{100^2 + 100^2} - \sqrt{100.5^2 + 100.5^2})/\sqrt{100.5^2 + 100.5^2} = 0.5\% \quad (17.1)
\]

If the actual value did equal (100.5,100.0), then the error in the phase is approximately

\[
e_p \approx 100(\arctan(100/100) - \arctan(100.5/100))/\arctan(100.5/100) = 0.3\% \quad (17.2)
\]

These are worst case scenarios. If the complex value is larger, then the relative error decreases. Note that the maximum for a signed short integer is \(2^{15} = 32768\).

17.3.2 Interpolation Kernels

In this section the available kernels are defined. See also [Hanssen and Bamler, 1999]. The KNAB interpolation kernel is described in a IEEE letter of 2003. The Raised Cosine interpolation kernel is described in an article in J. Of electromagnetic waves, 2005. Cho et al.
\[ sinc(x) = \frac{\sin \pi x}{\pi x} \quad (17.3) \]

\[ rect(x) = \begin{cases} 
0 & |x| > 0.5 \\
0.5 & |x| = 0.5 \\
1 & |x| < 0.5
\end{cases} \quad (17.4) \]

\[ i(x) = tri(x) = \begin{cases} 
0 & |x| > 1 \\
1 - |x| & |x| < 1
\end{cases} \quad (17.5) \]

\( \alpha = -1 \)

\[ i(x) = \begin{cases} 
(\alpha + 2)|x|^3 - (\alpha + 3)|x|^2 + 1 & 0 \leq |x| < 1 \\
\alpha|x|^3 - 5\alpha|x|^2 + 8\alpha|x| - 4\alpha & 1 \leq |x| < 2 \\
0 & 2 \leq |x|
\end{cases} \quad (17.6) \]

\( \alpha = -0.5; \beta = 0.5 \):

\[ i(x) = \begin{cases} 
(\alpha - \beta + 2)|x|^3 - (\alpha - \beta + 3)|x|^2 + 1 & 0 \leq |x| < 1 \\
\alpha|x|^3 - (5\alpha - \beta)|x|^2 + (8\alpha - 3\beta)|x| - (4\alpha - 2\beta) & 1 \leq |x| < 2 \\
\beta|x|^3 - (8\beta)|x|^2 + (21\beta)|x| - (18\beta) & 2 \leq |x| < 3 \\
0 & 3 \leq |x|
\end{cases} \quad (17.7) \]

\( L = 6, 8, 16 \)

\[ i(x) = sinc(x)rect\left(\frac{x}{L}\right) \quad (17.8) \]
In this chapter the processing of step FILTRANGE is described. This optional step filters the spectra in range direction of master and slave to reduce noise in the interferogram. The noise reduction results from filtering out non overlapping parts of the spectrum. This spectral non overlap in range between master and slave is caused by a slightly different viewing angle of both sensors. The longer the perpendicular baseline, the smaller the overlapping part. Eventually a baseline of about 1100 m results in no overlap at all (the critical baseline for ERS). (Assuming no local terrain slope.) A reduction of typically 10-20% in the number of residues can be achieved.

Method porbits filters based on the orbits (perpendicular baseline) for a constant (given) terrain slope. Perform this step after coarse coregistration, since the approximate overlap is used to filter both images. The output images are cropped to this overlap. To filter ‘on the save side’, i.e., not to filter out too much, use a negative terrain slope of, e.g., 10 degrees.

This step is not recommended, except perhaps to improve the coregistration polynomial for long baseline pairs. After the resampling the range filtering then could be repeated on the original data with the adaptive algorithm.

Method adaptive should be performed after the resampling of the slave to the master grid, because the fringe frequency is estimated from the interferogram (that is temporary computed). It is performed simulataneous for the master and slave image.

### 18.1 Input Cards

<table>
<thead>
<tr>
<th>Option</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RF_METHOD</td>
<td>adaptive</td>
<td>Method selector for range filtering. Either adaptive (recommended) or based on the precise orbits.</td>
</tr>
<tr>
<td>RF_METHOD</td>
<td>porbits</td>
<td>Method selector for range filtering. Either adaptive (recommended) or based on the precise orbits.</td>
</tr>
<tr>
<td>RF_FFTLENGTH</td>
<td>64</td>
<td>For method porbits and adaptive. For method porbits: length of block in range direction, 512 or 1024 (default for this method) advised. For method adaptive: Length of window for adaptive method. A peak is estimated for parts of this length.</td>
</tr>
<tr>
<td>RF_OVERLAP</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>
For method adaptive. Overlap between input buffers in range direction.

**RF_HAMMING** 0.75
For method porbits and adaptive. Weight for hamming filter (1 is rect).

**RF_SLOPE** 0
For method porbits. Terrain slope in degrees. Positive slope is towards radar. A slope equal to the viewing angle implies total filtering.

**RF_NLMEAN** 15
For method adaptive. Take (walking) mean over RF_NLMEAN lines to reduce noise for peak estimation. Has to be odd. Compare with periodogram.

**RF_THRESHOLD** 5
For method adaptive. Threshold on SNR of peak estimation to perform range filtering.

**RF_OVERSAMPLE** 2
For method adaptive. Oversample master and slave with this factor before computing the complex interferogram for peak estimation. This factor has to be a power of 2. 2 is default to be able to estimate the peak for frequency shifts larger than half the bandwidth. A factor of 4 for example might give a better estimate, since the interval between shifts that can be estimated is in that case halved (fixed FFTLENGTH).

**RF_WEIGHTCORR** [ON | OFF]
For method adaptive. In peak estimation, weight values to bias higher frequencies. The reason for this card is that the low frequencies are (for small OVERSAMPLE factors) aliased after interferogram generation. The deweighting is done by a dividing by a triangle function (convolution of 2 rect functions, the shape of the range spectrum). Effect of this card may be neglectable.

**RF_OUT_MASTER** *master.rfilter*
Output data file name of master.

**RF_OUT_SLAVE** *slave.rfilter*
Output data file name of slave.

**RF_OUT_FORMAT** *cr4 | ci2*
Output data format for master and slave file.

Example input:

```
c
  comment ___ ADAPTIVE RANGE FILTERING ___
```
18.2 Output Description

The **process control flags** in the **result files** for the master and the slave are switched on:

```
filt_range: 1
```

In the filtrange section the filename and format is described after the filtering. Master and slave are cut out so they exactly fit on each other.

```
*
*_Start_filt_range:
***********************************************
Method: adaptive
Data_output_file: Outdata/23070.rfilter.3
Data_output_format: complex_real4
First_line (w.r.t. original_master): 201
Last_line (w.r.t. original_master): 6000
First_pixel (w.r.t. original_master): 100
Last_pixel (w.r.t. original_master): 3000
***********************************************
*_End_filt_range:_NORMAL
***********************************************
```

Figure 18.1 shows the improvement in the correlation for method porbits. A histogram is made of the correlation values of the FINE coregistration with and without range filtering. The area was relatively flat and the perpendicular baseline was approximately 175 meters.

18.3 Implementation

18.3.1 Method: porbits

The frequency shift $\Delta f$ between the master and slave range data spectra equals

$$
\Delta f = -\frac{c}{\lambda r_1 \tan(\theta - \alpha)} = -\frac{c}{\lambda \Delta \theta \tan(\theta - \alpha)} \approx -\frac{c}{\lambda \Delta \theta} (18.1)
$$
Figure 18.1  Frequency histograms of the FINE coregistration correlation values. At 301 locations of a image the fine coregistration was performed, and a histogram has been made of the correlation values. The bin width equals 0.05. It can be clearly seen that the histogram of the filtered data is located to the right, i.e., is better.

Where:
\[
\Delta \theta = \theta_1 - \theta_2, \quad \alpha \text{ is the local terrain slope w.r.t. the ellipsoid, } c \text{ is the speed of light, } \theta \text{ is the local incidence angle }, \lambda \text{ is the radar wavelength, } r_1 \text{ is the slant range ground to master. }
\]

The approximation is used in Doris. Of course, the sign of \( B_\perp \), or \( \Delta \theta \) is important to filter the correct side of the spectra. Note that

\[
\alpha \rightarrow 23^\circ \iff \Delta f \rightarrow \infty \quad (18.2)
\]

The local incidence angle is computed with the dot product of vectors P, and P-M. See also [Gatelli et al., 1994].

The algorithm in Doris works as

- While there is a line in the overlap, get next line for master and slave.
- Get block of FFT_LENGTH pixels.
- Compute viewing angle, perpendicular baseline, delta theta for middle pixel of block.
- Compute frequency shift by equation 18.1, and compose filter of rect and hamming.
- Filter master and slave.
- Write block back (for last block only partially).
18.3.2 Method: adaptive

After the resampling of the slave on the master grid is performed this algorithm can be used. The local fringe frequency is estimated using peak analysis of the power of the spectrum of the complex interferogram. The resampling is required since the local fringe frequency is estimated from the interferogram. This fringe frequency is directly related to the spectral shift in range direction. (Note this shift is not a shift, but different frequencies are mapped on places with this shift...) The algorithm generally works as follows.

- Take part of master and slave (e.g., 500 lines by 128 range pixels.)
- Oversample master and slave and generate complex interferogram.
- Take FFT over range for all lines of complex interferogram.
- Take power. If requested, weight this powerspectrum with auto-convolution of 2 rect functions with appropriate bandwidth. (Actually, perhaps the spectrum should also be weighted with autoconvolution of Hamming, but since I am not sure that this has a big impact on real data this is not done.)
- Take moving average over the lines of the power FFT’s for noise suppression (kind of periodogram). (This was better implemented as a convolution with a block function (e.g., 9 x 128)?)
- Estimate peak per line in oversampled/averaged powerspectrum of complex interferogram. Estimate $SNR = \frac{fftlength \cdot peak}{rest}$.
- This peak is directly related to overlap of spectra for this part of this line. (See also Fig. 18.2.) $\Delta f_r = f_{fringe}$.
- If SNR is above threshold (input of user, e.g., 3), remove appropriate parts of spectra of master/slave. Optionally compute inverse hamming window and new hamming window, and rect window to filter one side of master spectrum, and other side of slave spectrum. (See also Fig. 18.3.) Note that the filter is mirrored (matlab fliplr) for master/slave. The SNR of the peak of a random spectrum (sea) probably is a little larger than 1, so threshold of 3 may not be large enough.
- Do inverse FFT for filtered master, slave, which yields the filtered image in the space domain.
- Take next part of master and slave (e.g., 500 lines by next 128 range pixels) until all lines are filtered.

In practice this is done in blocks. These blocks are overlapping in lines (because the averaging over lines means one cannot filter all lines in the block), and not in range. Parameters that can be adjusted are the FFT length, the moving average mean, the SNR threshold.

The fftlength should be large enough to yield a good estimate of the local fringe frequency, and small enough to contain a constant slope of the terrain. The total number of fringes in range direction can be easily estimated using the perpendicular baseline.

It is probably a good idea to add a card so an overlap in range between blocks can be used. This avoids ‘edge’ effects, and increases the filtering of terrain near, e.g., a lake (since the SNR for peak detection will be higher for a number of blocks towards the noise). This is not implemented yet.
See also [Gatelli et al., 1994], [Geudtner, 1996], [Curlander and McDonough, 1991]. See also our matlab toolbox.

Figure 18.2  Peak estimation in spectral domain of (oversampled) complex interferogram. Non FFTshifted.

Figure 18.3  Spectral filtering windows (inverse hamming, boxcar (rect), and new hamming. Note these are FFTshifted.
Figure 18.4  Detail of interferogram with and without rangefiltering. (fftlength=128, nlmean=15, snrthreshold=5). The perpendicular baseline is about 200 m for this interferogram. The fringes are clearly sharper after the filtering. The number of residues for the interferogram was reduced by 20%. Subtraction of both interferograms yielded a random phase, so no structural effect of range filter implementation is suspected.
18.3.3 Hamming filter

The Hamming filter that optionally is used to de-weight and re-weight the spectrum of master and slave has the form:

$$ W(f_r) = \left[ \alpha + (1 - \alpha) \cos(2\pi \frac{f_r}{f_s}) \right] \text{rect} \left( \frac{f_r}{B_r} \right). $$

(18.3)

Where $f_r$ is the frequency axis (-fs/2:df:fs-df, df=fs/N). $f_s$ is the range sampling rate (18.96MHz), and $B_r$ is the bandwidth in range (15.55MHz). $\alpha$ is a parameter controlling the amount of weighting.

$$ \text{rect}(x) = \begin{cases} 1, & \|x\| < 0.5 \\ 0, & \text{otherwise} \end{cases} $$

(18.4)

Note: rect not periodic.
Chapter 19

INTERFERO

In this chapter the processing of step INTERFERO is described. In this step the following is computed.

The (complex) interferogram is computed, with or without subtraction of the reference phase. The reference phase is subtracted if there is a 2d-polynomial in the products result file (result of step FLATEARTH). It is not subtracted if this is not in the result file or if the number of coefficients is set to 0.

The complex interferogram minus reference phase is defined as:

\[ I = M \cdot S^* \cdot R^* \]  

(19.1)

Where:
\{ \}^* denotes the complex conjugated;
\cdot denotes a pointwise multiplication;
\( I \) is the complex interferogram;
\( M \) is the complex master image;
\( S \) is the complex (resampled) slave image;
\( R \) is the complex (amplitude \( \equiv 1 \)) reference phase.

The phase image (of the complex interferogram minus reference phase) is defined as:

\[ \phi = \arctan_2(I_{\text{imag}}, I_{\text{real}}) \]  

(19.2)

Where:
\arctan_2 is the four quadrant arc tangent;
\( \phi \) is the phase image;
\( I \) is the complex interferogram;

This is identical to

\[ \phi = \phi_M - \phi_S - \phi_R \]  

(19.3)
Multilooking can be performed to reduce noise. Usually a ratio of \(\text{(line:pixel)} = 5:1\) between the factors is chosen to obtain more or less square pixels \((20\times20\text{m}^2\text{ for factors 5 and 1})\). (The resolution decreases of course if multilooking is applied.)

19.0.4 NEW

Note that for the optimal results, i.e. to avoid an aliasing of spectras, the images first have to be oversampled by a factor two before multiplication for an optimal result. See sections on \text{'OVERSAMPLE'} card.

For a more modular approach in the new method it is not advisable to subtract the reference phase in this step. If you do want to subtract the reference phase here, then make sure you first run Doris to run \text{comprefpha}, and then make a second run for \text{step interfero}.

After generation of the complex interferogram, the reference phase can be computed by the new module \text{comprefpha} and subtracted by the new module \text{subtrrefpha}. (Also a reference height model can be computed and subtracted in future modules.)

Figure 19.1 shows an example of a complex interferogram. Only the phase is shown here. We processed orbits 21066 and 1393 of frame 2781 (Italy), acquired at 26th and 27th July 1995 respectively (ERS1,2 Tandem mission). The parallel baseline is about 35 meters, which implies a height ambiguity of about 270 meters. Clearly a large trend caused by the ‘flat earth’ is present, but also some topographic features can be seen. In the frame the elevation ranges from zero to 1400 meters. (The original SLC images were cut out to 20000 lines by 4000 pixels. The interferogram is multiooked by factors 10 in azimuth and 2 in range, which yields a dimension of 1475 lines by 1997 pixels.)
19.1 Input Cards

INT_OUT_CINT filename
filename of output datafile for complex interferogram (of step interferogram). one of INT_OUT.* is mandatory.

INT_OUT_INT filename
filename of output datafile for (real) interferogram (of step interferogram). one of INT_OUT.* is mandatory.

INT_MULTILOOK 5 1
multilookfactor, if no multilooking is desired, set this to "1 1". If the reference phase is not subtracted in this step, be careful not to multilook too much in this step. In step subtrrefpha again a multilook card is present (where one can multilook by factors 2 2 for example).

Example input section:

```
c
  comment ___product generation___
c
  INT_OUT_INT  Output/int.raw    // optional
  INT_OUT_CINT  Output/cint.raw  // optional
  INT_OUT_FE   Output/flatearth.raw // optional
  INT_MULTILOOK 10 2 // line, pixel
```

19.2 Output Description

At successful exit, the process control flag is switched on:

```
interfero: 1
```

The output looks like (in the products result file):

```
*************************************************** ********************
* _Start_interfero_*
*************************************************** ********************
Data_output_file: Output/cint.raw
Data_output_format: complex_real4
First_line (w.r.t. original_master): 1001
Last_line (w.r.t. original_master): 2105
First_pixel (w.r.t. original_master): 501
Last_pixel (w.r.t. original_master): 700
Multilookfactor_azimuth_direction: 10
Multilookfactor_range_direction: 2
Number of lines (multilooked): 110
Number of pixels (multilooked): 100
```
The (complex) output data file can be viewed with, e.g., in Matlab with a following script:

```matlab
fid = fopen('Output/cint.raw','r');
cint = (fread(fid,[100 220],'float32')).';
fclose(fid);
realpart = cint(:,1:2:size(cint,1));
cplxpart = cint(:,2:2:size(cint,1));
cint = realpart + i*cplxpart;
phase = angle(cint);
imagesc(phase);
colorbar;
```

Output may include the matrix with the reference phase. For flat earth correction this normally resembles a plane, and is not very useful output.

### 19.3 Implementation

The following is computed (in buffers).

1. Read in (buffer of) complex master and slave image (M and S).
2. If there is a reference phase R (real values), evaluate it at the master grid (buffer), and then subtract it (in a complex way) from slave S and store it (in S). (matlab like notation, pointwise multiplication notated by `.*`)

   \[ S = S.* (\cos R + i \sin R) \]  

   \[ (19.4) \]

3. Compute complex interferogram and store it in M.

   \[ M = M.* \text{conj}(S) (\equiv M.* \text{conj}(S).* \text{conj}(R)) \]  

   \[ (19.5) \]

4. Multilook complex interferogram if requested. Do not divide (scale) multilooked interferogram.
5. Write this buffer to disk and start next one.

See Figure 19.2 for an explanation of the use of buffers in the implementation. In this example, the number of lines equals 17. The number of pixels equals P. Suppose multilook factor in line direction mL=3 and in pixel direction mP=3. Furthermore, after computing the available memory, the number of lines of one buffer (BL) is maximum 7. BL is set to a multiple of mL, BL=6.

Now the number of fully filled buffers NB = 17/6 = 2. The number of lines left in the last buffer equals 17. We can compute something in this last buffer only for the first three lines, so the last buffer is 3 lines long.
The first buffer is read (master and slave image), and computations are performed. These results are written to disk. Then next buffer, etc. If buffers==3 then resize the matrices for the computations.

The number of lines of the total result are $L/mL$ and $P/mP$ (floored).
In this chapter the processing of step COMPREFPHA is described. This step can be performed as soon as the precise orbits are known. The recommended approach is to compute this only after the computation of the interferogram, and then use the step SUBTRREFPHA to subtract it. This step is not required if method "exact" is used in step SUBTRREFPHA.

The flatearth correction (the phase caused by the reference surface (WGS84 for now)) is computed in this step. For a certain (line,pixel) in the master image the corresponding coordinates \((x,y,z)\) of the master and slave satellite and the point \(P\) on the reference ellipsoid are computed, utilizing the set of equations (Doppler, range and ellipsoid equation), see annex B. Then the parallel baseline and the phase is computed.

The parallel baseline \(B_\parallel\) is defined as \((M/S\) are positions of master/slave, \(P\) is the position of the point on the reference surface):

\[
B_\parallel = d(M, P) - d(S, P) \quad (20.1)
\]

The phase of a pixel in the master image is defined as:

\[
\phi = -\frac{4\pi}{\lambda} d(M, P) \quad (20.2)
\]

The reference phase for this pixel is defined as:

\[
\phi = -\frac{4\pi}{\lambda} B_\parallel \quad (20.3)
\]

The reference phase is computed in a number of points distributed over the total image in this way, after which a 2d-polynomial is estimated (least squares) fitting these 'observations'. (So a plane can be fitted by setting the degree to 1.)

A 2d-polynomial is defined as:

\[
f(x, y) = \sum_{i=0}^{d} \sum_{j=0}^{i} \alpha_{i-j} x^{i-j} y^{j} \quad (20.4)
\]
Thus the order of the coefficients (line, pixel) is (degree d):

- d=0: $A_{00}$ (1)
- d=1: $A_{10}, A_{01}$ (2, 3)
- d=2: $A_{20}, A_{11}, A_{02}$ (4, 5, 6)
- d=3: $A_{30}, A_{21}, A_{12}, A_{03}$ (7, 8, 9, 10)

Thus the number of coefficients (unknowns) equals:

$$\frac{1}{2}((d + 1)^2 + d + 1)$$  \hspace{1cm} (20.5)

A polynomial of degree 5 normally is sufficient to model the reference phase for a full scene. A lower degree might be selected for smaller images, which also should increase the stability of the normal matrix. We would expect the higher order terms to be small because the polynomial describes a smooth, long wave body (ellipsoid). To force the polynomial to be smooth one might consider always using a polynomial of degree 2 or 3.

See also [Schwäbisch, 1995] or [Geudtner and Schwäbisch, 1996].

### 20.1 Input Cards

**FE_METHOD**

*porbits*

Method selector for this step. Currently there is only one method.

**FE_DEGREE**

5

degree of 2d polynomial.

**FE_NPOINTS**

501

Number of points to compute reference phase for least squares estimation.

**FE_IN_POS**

*filename*

Ascii file with positions (master coord. system) in it where to compute the reference phase, and then to model it by a polynomial. Card FE_NPOINTS is ignored if this card is specified. This card can be used, e.g., if it is desired to have the points on a grid, including the edges. Possibly one might even select points outside the grid (though not smaller than 0), in order to avoid excessive fluctuations at the edge if a higher degree polynomial is used.

**FE_OUT_FILE**

*filename*

Card will be added in future for optional output of reference phase.
One can use an awk like to make a grid:

```bash
awk 'BEGIN { for (i =100;i<25200;i=i+500) \
       { for (j =750;j<5400; j=j +200) \n         { printf "%i %i \n",i,j}} \
       exit }' > positions.in
```

and a card in the **input file**:

```plaintext
FE_IN_POS positions.in
```

This step used to be named "FlatEarth". This explains the prepended FE's, instead of some abbreviation for reference phase.

**Example input section:**

```plaintext
c
c
comment ___ COMPREFPHA ___
c
FE_METHOD porbits
FE_DEGREE 3
FE_NPOINTS 201
```

**20.2 Output Description**

At successful exit the **process control flag** is switched on:

```plaintext
comp_refpha: 1
```

The output (in the **products result file**) looks like:

```
*************************************************** ****************
* _Start_comprefpha
*************************************************** ****************
Degree_flat:  5
Estimated_coefficients_flatearth:
  5.17144173e+03 0 0
  4.03705656e-03 1 0
  2.1736976e-01 0 1
  2.05452064e-06 2 0
  2.15880157e-07 1 1
  1.27934869e-05 0 2
 -1.80499980e-10 3 0
 -1.64339133e-11 2 1
 -8.28354289e-11 1 2
 -6.04376860e-10 0 3
 -1.64239900e-13 4 0
  2.25037286e-15 3 1
  1.4824391e-14 2 2
  5.52622526e-14 1 3
```
In the log file, some statistics are given for the errors (observation minus estimated value). These errors should have a maximum of 0.1 phase cycle. The can be plotted with GMT or someother package to evaluate the difference between the polynomial and the ‘observations’.

Here also the standard deviation per estimated coefficients is given. This std. seems to be too large, but an error in the computations could not be found.

The polynomial can not easily be visualized at the moment. (It is normalized, not evaluated in this step.) In the old method of the interfero step (see Chapter 19), there was a card to output the reference phase polynomial because it was evaluated there anyway. It seems logical to add a card for outputting the (wrapped or unwrapped) reference phase in the step subtrrefpha (Chapter 21), which likely will be added in one of the cumming releases.

<table>
<thead>
<tr>
<th>Value</th>
<th>Error</th>
<th>Index</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.74773307e-12</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>1.11724810e-17</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>2.74597475e-20</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>-2.64743817e-18</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>4.10973605e-18</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>-3.09581184e-17</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>-6.94891272e-16</td>
<td>0</td>
<td>5</td>
</tr>
</tbody>
</table>

*************************************************** ****************
* End_comrefpha: _NORMAL
*************************************************** ****************
In this chapter the processing of step SUBTRREFPHA is described.

This step requires the steps INTERFERO and COMPREFPHA for obvious reasons. In this step the reference phase of a mathematical body (ellipsoid) is subtracted from the complex interferogram. This is done by complex multiplication with the conjugated, written symbolically as follows:

\[ I = I \cdot (\cos R_\phi - i \sin R_\phi) \]  

(21.1)

Where \( I \) is the complex interferogram, \( \cdot \) denotes pointwise multiplication, and \( R_\phi \) is the reference phase for a certain point.

### 21.1 Input Cards

**SRP_METHOD** *polynomial | exact*

Method selector for subtraction of reference phase. "polynomial" evaluates the polynomial computed in step COMPREFPHA, "exact" computes the reference phase explicitly for each pixel, and subtracts it. Computations are done by evaluation system of 3 equations foreach pixel.

**SRP_OUT_CINT** *cint.minrefpha.raw*

Filename of output datafile for complex interferogram (of step subtrrefpha).

**SRP_MULTILOOK** *1 1*

Multilook factors in line (azimuth) and pixel (range) direction.

**SRP_DUMPREFPHA** [ OFF | ON ]


This card specifies to dump the reference phase as a complex real4 file, containing the evaluated reference phase polynomial just as it would have been subtracted from the complex interferogram (multilooked). The amplitude should be equal to one by definition. **WARNING:** the reference phase is not subtracted, only dumped, if this card is specified. If you want to study the reference phase for different ellipsoids, compile different versions of Doris, changing the parameters in the file refsystem, and use these executables to generate the reference phase.

**SRP_OUT_REFPHA**  
*refphase.raw*

Name of **output file** reference phase. Only if **SRP_DUMPREFPHA**.

Example input section for dumping the reference phase:

```
c c ___ step subtrrefpha ___
c SRP_METHOD exact
c SRP_METHOD polynomial
c SRP_OUT_CINT Outdata/cint.minrefpha.raw
SRP_MULTILOOK 4 4
SRP_DUMPREFPHA
SRP_OUT_REFPHA Outdata/refpha.raw
```

## 21.2 Output Description

At successful exit the **process control flag** is switched on:

```
subtrrefpha: 1
```

The output (in the **products result file**) looks like:

```
************************************************************************************
*_Start_subtrrefpha:
************************************************************************************
Data_output_file: Outdata/cint.minrefpha.raw
Data_output_format: complex_real4
First_line (w.r.t. original_master): 245
Last_line (w.r.t. original_master): 14964
First_pixel (w.r.t. original_master): 7
Last_pixel (w.r.t. original_master): 3998
Multilookfactor_azimuth_direction: 40
Multilookfactor_range_direction: 8
Number of lines (multilooked): 368
Number of pixels (multilooked): 499
************************************************************************************
*_End_subtr_refphase: _NORMAL
************************************************************************************
```

We noticed that if the precise orbits are not long enough (not enough time before and after first/last line), this results in a wrong reference phase for obvious reasons. Interpolation near
the end of the data points is not very good with cubic splines. This can be solved by using more orbital data points after the last line of the scene (see cards for step M_porbits).

Figure 21.1 shows the result of subtracting the reference phase polynomial from the interferogram (Figure 19.1). (The same scene as described in section 19.0.4, again multilooked, now by factors 4 and 4, resulting in total multilooking of 40 and 8, which agrees on the terrain with a resolution of about 160 meters square.)

This step can be mis-used to correct for residual orbital fringes (if you know what you are doing).

To do this, first count the number of fringes you want to remove from the interferogram. Then, edit the products result file and create a section for the step COMPREFPHA. In this output section, simply define a polynomial that describes for example a linear trend in range of, say, 2.5 fringes. Then, run this step, and doris will not know that it is not the reference phase polynomial that is subtracted from the interferogram, but an additional correction polynomial that has been inserted by hand.
In this chapter the processing of step COHERENCE is described.

In this step the following is computed. The (complex) coherence image is computed, with or without subtraction of the reference phase. The reference phase is subtracted if there is a 2d-polynomial in the products result file (result of step FLATEARTH). It is not subtracted if this is not in the result file or if the number of coefficients is set to 0.

The complex coherence image between two images is defined as:

$$\gamma_c = \frac{E\{M \cdot S^*\}}{\sqrt{E\{M \cdot M^*\} \cdot E\{S \cdot S^*\}}} \tag{22.1}$$

Where:
- $E\{\cdot\}$ is the expectation;
- $*$ is the complex conjugated;
- $\gamma_c$ is the complex coherence;
- $M$ is the complex master image;
- $S$ is the complex slave image (possibly minus (complex) reference phase: $S = S \cdot R^*$).

The coherence is defined by $|\gamma_c|$ and its estimator as:

$$\hat{\gamma} = \left| \frac{1}{N} \sum_{i=0}^{N} M_i S_i^* \right| \sqrt{\frac{1}{N} \sum_{i=0}^{N} M_i M_i^* \frac{1}{N} \sum_{i=0}^{N} S_i S_i^*} \tag{22.2}$$

Multilooking can be performed to reduce noise. Usually a ratio of (line:pixel) = 5:1 between the factors is chosen to obtain more or less square pixels (20x20m$^2$ for factors 5 and 1). (The resolution decreases of course if multilooking is applied.)

Data buffering is applied for the memory considerations.

In the new coherence computation method, we will assume a complex interferogram is already computed. (and a reference phase has been subtracted.) This product can then be used to compute the coherence. How one can use a different multilook factor and coherence estimation window size in that case is not clear yet.
22.1 Input Cards

COH_OUT_CCOH  filename
filename of output datafile for complex coherence image (of step coherence). one of COH_OUT_CCOH and *_COH is mandatory.

COH_OUT_COH  filename
filename of output datafile for (real) coherence image (of step coherence). one of COH_OUT_CCOH and *_COH is mandatory.

COH_MULTILOOK  10 2
multilookfactor, if no multilooking is required, set this to "1 1".

COH_WINSIZE  10 2
window size of shifting window for coherence estimation.

Example input section for this step.

```
c
comment ___product generation___
c
 c  COH_OUT_CCOH  Output/ccoah.raw // complex image
 c  COH_OUT_COH  Output/coah.raw    // real
 COH_MULTILOOK  10 2
 COH_WINSIZE  10 2
```

22.2 Output Description

At successful exit the process control flag is switched on:

coherence: 1

Example output section for this step (in products result file).

```
***************************************************************
*_Start_coherence*******************************************************************************
***************************************************************
Data_output_file:  Output/coah.raw
Data_output_format:  real4
First_line (w.r.t. original_master):  1001
Last_line (w.r.t. original_master):  2105
First_pixel (w.r.t. original_master):  501
Last_pixel (w.r.t. original_master):  700
Multilookfactor_azimuth_direction:  10
Multilookfactor_range_direction:  2
Number of lines (multilooked):  110
Number of pixels (multilooked):  100
```
22.3 Implementation

The images are read in in buffers for memory considerations. First complex interferogram is computed as in INTERFERO, and the norms of the master and slave images are computed.

Then a shifting window of size COH_WINSIZE is used to estimate the complex coherence (see Annex B).

The coherence is computed with a function of the matrix class. This function returns only the lines of the input which can be computed due to the edge of the estimator window. Then this is multilooked (requires number of lines to be a multiple of the multilook factor).

Therefore, the buffer should contain an overlap with the previous one.
In this chapter the processing of step COMPREFDEM is described. A DEM is radarcoded at the grid of the interferogram. In step subtrrefdem (chapter 24) it can be subtracted from the complex interferogram.

This step requires a DEM. It can best be performed after the interferogram generation. Currently the USGS gtopo30 DEM’s are used by default. This is a global DEM with relatively low precision and gridspacing (30 seconds, approximately 1 km at equator). There are 33 tiles covering the globe, in total requiring about 3GB of disk space. (For more information, see for example http://edcwww.cr.usgs.gov/landdaac/gtopo30/gtopo30.html or Google for more information). The DEM has to have the byte order of your platform. That means that if you work on a PC, the files from gtopo should be byteswapped in principle. However, I added an option to have this done automatically by Doris.

The input DEM has to be in an equiangular grid. The format is short signed integers, real4 floats, or real8 doubles (meters). The DEM should be in WGS84 system (same as orbit ephemerides). The UpperLeft pixel of the DEM matrix on file should be the most North, most West pixel, i.e., the pixel with largest latitude (between -90,90) and smallest longitude (between -180,180 (?)).

NOTE: this step can use a lot of memory. decrease CRD_DENSE if processed crashes due to “out of memory”. (Or change in the code the buffersize in function radarcodedem. Try to simply increase the value of MEMORY card, and check with tools such as top, yamm, what the actual RAM usage is.)

Internally the input DEM is interpolated by means of a bilinear interpolator, so the resolution is approximately equal to that of the interferogram. Problems can arise if a lot of interpolating points are required to come to this (e.g., if the DEM spacing is approx. 1 km, and the resolution of the interferogram is 20 meters, then 50 points are interpolated based on 2 points). A solution is to reduce the resolution of the interferogram by using larger multilook factors, but the input DEM could also be interpolated by other software first to obtain a ‘higher’ resolution. In Matlab or GMT one could use cubic splines for the interpolation for example.
23.1 Input Cards

CRD_METHOD  NEAREST | TRILINEAR
The DEM is oversampled and radarcoded to (li,pi,phi_i) irregular triplets in radar geometry; The goal is to get phi at the regular grid (l,p) of the interferogram. NEAREST simply takes the closest (li,pi,phi_i), and none if there is none within half the resolution. TRILINEAR interpolates linearly using three nearby values.

CRD_IN_FORMAT  I2 | I2_BIGENDIAN | R4 | R8
format of input DEM on file (signed short for gtopo30, or real4, or real8; input matrix is raw binary data w/o header, endianness of host platform is assumed, except for I2_BIGENDIAN).

CRD_IN_DEM  filename
filename of input DEM (gtopo30). File is assumed to be stored in a raster. Major row order from North to South, line-by-line. See also internet links at Doris home page for available DEMs.

CRD_IN_SIZE  6000 4800
Number of rows and columns of input DEM file. Default is set to tile w020n90.DEM.

CRD_IN_DELTA  0.00833333333333333333
[ deltalon ]
Grid spacing of input DEM in decimal degrees, latitude longitude. Default is equal gridspacing, default set to tile w020n90.DEM.

CRD_IN_UL  89.995833333333333333 -19.995833333333333333
Coordinates of UL (upperleft) corner in decimal degrees, latitude [-90, 90] longitude [-180, 180]. Default is set to tile w020n90.DEM. It is interpreted as max-latitude, min-longitude in source.

CRD_IN_NODATA  -9999
Identifier to ignore data in input DEM with this value. Default is set to tile w020n90.DEM.

CRD_INCLUDE_FE  OFF | ON
If this card is switched on, the reference phase of the DEM is computed, including the 'flat earth' term. Otherwise, the phase is computed with respect to the ellipsoid, yielding only topographic phase.

CRD_DENSE  0.2
Oversamplings factor to make the DEM more dense, after the minimum factors have been computed. For method NEAREST dense should likely be 2, which costs lots of memory. For method TRILINEAR a value 0.1 may be OK, depending on resolution of input DEM. Much faster and low memory.

CRD_OUT_DEM  filename
Request optional debug output to float file of input DEM per buffer, cut to the interferogram window. Info on these files is written as DEBUG.

**CRD_OUT_DEMI**  
 filename
Request optional debug output to float file of interpolated input DEM, cut to the interferogram window. Info on these files is written as DEBUG.

**CRD_OUT_FILE**  
 refdem.raw
filename of output radarcoded DEM.

Most of these parameters can be found in the .HDR file of gtopo30 DEM's. Example input section:

```plaintext
c c ___ step comprefdem ___
c CRD_METHOD gtopo30
 c CRD_IN_FORMAT sshort
 CRD_IN_DEM  / home / user /d2/ dem / gtopo30 / w020n90 / W020N90.DEM
 CRD_IN_SIZE  6000 4800  // rows cols
 CRD_IN_DELTA  0.00833333333333333333  // same value for lat/lon
 CRD_IN_UL  89.99583333333333333333  -19.99583333333333333333
 CRD_IN_NODATA  -9999  
 CRD_INCLUDE_FE OFF  // phase w.r.t. ellipsoid
 CRD_DENSE  4  // oversample extra
 c CRD_OUT_DEM Outdata/DEM.raw  // request output
 c CRD_OUT_DEMI Outdata/DEMi.raw  // request output
 CRD_OUT_FILE Outdata/refdem.raw  //
```

## 23.2 Output Description

At successful exit the **process control flag** is switched on:

```plaintext
comp_refdem: 1
```

The output (in the **products result file**) looks like:

```plaintext
************ Start_comprefdem ************
Method: dem gtopo30
DEM source file: /home/user/d2/dem/gtopo30/w020n90/W020N90.DEM
Data_output_file: Outdata/refdem.raw
Data_output_format: real4
First_line (w.r.t. original_master): 245
Last_line (w.r.t. original_master): 14964
First_pixel (w.r.t. original_master): 7
Last_pixel (w.r.t. original_master): 3998
Multilookfactor_azimuth_direction: 40
Multilookfactor_range_direction: 8
Number of lines (multilooked): 368
```
Figure 23.1 Interferogram of radarcoded DEM for area described in section 19.0.4.

Number of pixels (multilooked): 499

The output in the logfile is more verbose, specifying the results of the intermediate steps. Also go over the standard out in case of problems, with the SCREEN set to DEBUG level.

Figure 23.1 shows an example of a real valued phase map (interferogram) for a radarcoded DEM (the Data_output_file). (This applies to the same scene as described in Section 19.0.4.)

23.3 Note on using SRTM C-band data

This file details how I experimented (first try) with C band SRTM data for Bam area with Doris step COMPRFDEM. These data have a resolution of 3 arc seconds (90 meter at equator) and can be ftp-ed for most areas worldwide. Main problem seems that files need to be stitched together and that height system is not WGS84 ellipsoid (unsolved).

http://www2.jpl.nasa.gov/srtm/cbanddataprodcts.html
ftp://e0mss21u.ecs.nasa.gov/srtm/Africa/

File: Quickstart.txt:
- Heights are in meters referenced to the WGS84 geoid.
- Data voids are assigned the value −32768.
- SRTM–3 files contain 1201 lines and 1201 samples.
- The rows at the north and south edges as well as the columns at the east and west edges of each cell overlap and are identical to the edge rows and columns in the adjacent cell.

Since I do not know the offset between GEOID and ellipsoid here I ignore this and assume it is a constant for my area. For real–world
application use, e.g., Matlab and spherical harmonic global geoid model.

SRTM tiles are lower left corner in tiles of 1 deg; ftp all relevant tiles for this area:

N28E057.hgt  N29E057.hgt  N30E057.hgt  QuickStart.txt
N28E058.hgt  N29E058.hgt  N30E058.hgt
N28E059.hgt  N29E059.hgt  N30E059.hgt  SRTM_Topoby.txt

Center of pixel is height value [m]. I assume they are stored row-by-row, with most North row first.

spacing is 3×1deg/60/60 = 0.00083333333333333 [deg]

user@fringe2:\$ grep corner output.

: Coordinates of corner interferogram: 7818, 1869 = 29.3128, 58.7186
: Coordinates of corner interferogram: 22817, 1869 = 28.7763, 58.5887
: Coordinates of corner interferogram: 7818, 4868 = 29.4189, 58.1333
: Coordinates of corner interferogram: 22817, 4868 = 28.8823, 58.0068

Thus, required are N28E058.hgt and N29E058.hgt and these need to be stitched. Doris expects DEM to be line-by-line, first line most North. DO:

Check whether tiles are indeed overlapping:

1.1) first line of most south tile N28E058.hgt

cpxfiddle -w 1201 -f i2 -oascii -q normal -L1 -P10 N28E058.hgt
   2322 2315 2323 2349 2377 2403 2451 2450 2415 2388

1.2) last line of most south tile N28E058.hgt

cpxfiddle -w 1201 -f i2 -oascii -q normal -l1201 -P10 N28E058.hgt
   out: 476 476 474 474 477 479 474 478 475 473

2.1) first line of most north tile N29E058.hgt

cpxfiddle -w 1201 -f i2 -oascii -q normal -L1 -P10 N29E058.hgt
   608 602 598 596 594 588 584 582 580 578

2.2) last line of most north tile N29E058.hgt

cpxfiddle -w 1201 -f i2 -oascii -q normal -l1201 -P10 N29E058.hgt
   out: 2317 2311 2324 2358 2384 2405 2444 2443 2405 2380

NOTE: it seems these lines are not the same in both tiles! Probably caused by 1 arc to 3 arc sec conversion per tile. But still, have to remove it the double data it seems the first line of most south tile is most north, and almost equal to the last line of most north tile.

3) QL image of DEM

cpxfiddle -w 1201 -f i2 -osunraster -q normal -c gray N28E058.hgt > q.ras
   xv -gamma 0.2 q.ras

4) stitch two tiles NS to a new file using cpxfiddle (or use dd)
Finally, for COMPREFDEM parameters to be used with Doris:

```
CRD_IN_DEM      dem_srtm.hgt
CRD_IN_UL       30.0 58.0 // N,E (stitched file)
CRD_IN_SIZE     2401 1201 // rows cols (2 tiles stitched file)
CRD_IN_DELTA    0.0008333333333333333 //
CRD_IN_NODATA   0 // ignore these
CRD_DENSE       1 // first test
MEMORY          500 // 500MB (I have 32GB RAM available)
```

STITCHING TWO TILES IN LONGITUDE DIRECTION is a bit more difficult from the unix prompt. I used an awk to first stitch the most northern two files N28E057.hgt and N28E058.hgt, then almost the same to stitch the most south tiles, and then a cat to create the 4 tiles (simply paste this at prompt): Doris expects DEM to be line–by–line, first line most North: (Note: below the same is achieved using Matlab)

```
Note listing below assuming csh:

rm -f q.hgt

set prg = "cpxfiddle -w1201 -f i2 -q normal -o short -L1200 N29E058.hgt > q.hgt
rm q.hgt

ls -l dem_srtm.hgt
-rw-r--r-- 1 user group 5767202 Mar 8 16:53 dem_srtm.hgt

this is expected, since 2*1201+2401=5767202 (1201 width, 2401 lines)
```

```
# copy line for line 1200 pixels of file N29E057 and N29E058:
# this removes the double column at right of first file (and 2nd)

echo 1 | awk '{for(i=1;i<=1200;i++)
{printf "$prg$prg","i,i,"N29E057.hgt",i,i,"N29E058.hgt") }}' \|
sh >& /dev/null

mv q.hgt north_two_tiles.hgt

# and copy 1201 lines of the second N28E057 and N28E058:

echo 1 | awk '{for(i=1;i<=1200;i++)
{printf "$prg$prg","i,i,"N28E057.hgt",i,i,"N28E058.hgt") }}' \|
sh >& /dev/null

# and cat the two together, North stitched tiles on top of South tiles:
cat north_two_tiles.hgt q.hgt > dem_srtm.hgt
rm -f q.hgt n_two_tiles.hgt

# visually inspect if it worked (yes, it did, but note the NaNs
# make it scale badly..):
cpxfiddle -w2400 -f i2 -q normal -osunraster -c gray dem_srtm.hgt > q.ras
xv -gamma 0.2 q.ras

# should be 2400 lines by 2400 pixels, UL=30.0N 57.0E
ls -l dem_srtm.hgt
-rw-r--r-- 1 user group 11520000 Mar 8 17:48 dem_srtm.hgt
exactly: 2*2400*2400=11520000B
```
Doris expects a DEM with the same or better resolution as the interferogram. Therefore, use GMT, Matlab, etc. to interpolate the SRTM DEM before running Doris. I did the following (Note that you can also use Matlab to easily stitch the tiles of course):

```matlab
demN1 = freadbk('N29E057.hgt',1201,'short');
demS1 = freadbk('N28E057.hgt',1201,'short');
demN2 = freadbk('N29E058.hgt',1201,'short');
demS2 = freadbk('N28E058.hgt',1201,'short');
qN1 = find(demN1 < -9999);
qS1 = find(demS1 < -9999);
qN2 = find(demN2 < -9999);
qS2 = find(demS2 < -9999);
demN1(qN1) = 0;
demS1(qS1) = 0;
demN2(qN2) = 0;
demS2(qS2) = 0;
figure(1);
imagesc(demN1,[0,3500])
figure(2);
imagesc(demS1,[0,3500])
figure(3);
imagesc(demN2,[0,3500])
figure(4);
imagesc(demS2,[0,3500])
dem = [[demN1(1:1200,1:1200),demN2(1:1200,1:1200)]; ...
     [demS1(1:1200,1:1200),demS2(1:1200,1:1200)]];
fwritebk(dem,'dem_stitched.hgt','short');%// same as with cpxfiddle
% // factor 4 denser (seems MEM problem, so do each tile separately.)
zi = interp2(dem,2,'linear');% factor 4
fwritebk(zi,'dem_interpolated.hgt','short');%// use this in doris
```

% Interpolation using GMT

```matlab
% CRD_INDEM dem_interpolated.hgt //
% CRD_IN_UL 30.0 57.0 // N,E (4 tile stitched file)
% CRD_IN_SIZE 9597 9597 // rows cols (stitched file)
% CRD_IN_DELTA 0.0002083333333333333 0.0002083333333333333
% CRD_IN_NODATA -32768 // ignore these
% CRD_DENSE 1 // first test
% MEMORY 500 // 500MB (I have 32GB RAM)
```
I have done some preliminary test with GMT. Basically, read the DEM into a grd file, perform the interpolation, and write it back to unsigned short format (−Zh):

```
xyz2grd N28E057.hgt --dem.grd --l1 --R0/1200/0/1200 --Zh
grdsample dem.grd --demi.grd --l0.5
grd2xyz demi.grd --Zh > N28E057_interp.hgt
```

# info:
grdinfo dem.grd
grdinfo demi.grd

% CRD_IN_DEM = N28E057_interp.hgt
% CRD_IN_UL = 29.0 57.0 // N.E (1 tile GMT interpolated)
% CRD_IN_SIZE = 2401 2401 // rows cols (2*1201−1)
% CRD_IN_DELTA = .000416666666666666 .0004166666666666666666666666
% CRD_IN_NODATA = −32768 // ignore these
% CRD_DENSE = 1 // first test
% MEMORY = 500 // 500MB (I have 32GB RAM)

* Remarks on CPU time  
*  
GTOPO30 data (DEM~1km spacing),

--- (1) ---------------------------------------------
MULTILOOK total: 25x25 (100x100m)
Interferogram: 600x600 pixels, ~60km^2
CRD_DENSE: 2
MEMORY: 500MB (allocated seems much less?)
computation in 1 buffer of 372 lines + 1 buffer of 208 lines
CPU time: 1 minute 38 seconds.

--- (2) ---------------------------------------------
MULTILOOK total: 15x15 (60x60m)
Interferogram: 1000x1000 pixels, ~60km^2
CRD_DENSE: 2
MEMORY: 500MB (allocated seems much less ~110MB?)
computation in 6 buffer of 144 lines + 1 buffer of 138 lines
CPU time: 10 minutes 16 seconds.

time factor 6 caused by number of pixels factor 2.8, quadratic?

--- (3) ---------------------------------------------
MULTILOOK total: 15x15 (60x60m)
Interferogram: 1000x1000 pixels, ~60km^2
CRD_DENSE: 1
MEMORY: 500MB (allocated seems required ~170MB?)
BEEP: error
using srtm dem of above, 4 tiles stitched file:
computation in 1 buffer of 1000 lines
CPU time: 2 minutes 1 second (~3600 no nearest neighbors (due to NaNs?).

--- (4) ---------------------------------------------
MULTILOOK total: 15x15 (60x60m)
Interferogram: 1000x1000 pixels, ~60km^2
CRD_DENSE: 2
MEMORY: 500MB (allocated seems required ~390MB?)
BEEP: error
23.4 Implementation

The algorithm that is used basically transforms the DEM to radar coordinates (radarcoding), and then interpolates these to integer coordinates by nearest neighbor.

This is done in buffers with sufficient overlap between patches. First the DEM is bilinearly interpolated to a denser resolution.

New method trilinear interpolates also linearly in radar geometry domain using three nearest points. This can be easily to more advanced interpolators using the approach in code.

For all buffers do:

1. Read part of DEM
2. Compute interval between DEM points. Compute interval between pixels in the interferogram. 3. Oversample the DEM in such a way that after radarcoding the grid is dense enough that every pixel is expected to have a nearest neighbor (depends on layover). The ratio between the intervals of step 2 is computed, and that is multiplied by CRD_DENSE to ensure a dense enough grid. Oversampling is done by bilinear splines in the space domain.
4. Get the nearest neighbors, discard others, warn if there is no nearest.
5. Compute the reference phase with respect to a mathematical body (refpha), or including the 'flat earth' (CRD_INCLUDE_FE ON).
In this chapter the processing of step SUBTRREFDEM is described.

This step requires the steps INTERFERO and COMPREFDEM for obvious reasons. In this step the reference phase of a digital elevation model is subtracted from the complex interferogram. This is done by complex multiplication with the conjugated, as explained in step subtrrefpha (chapter 21).

24.1 Input Cards

SRD_OUT_CINT  cint.minrefdem.raw
Filename of output complex interferogram.

SRD_OFFSET  0 0
Offset to be applied in azimuth line, range pixel direction. The synthetic phase image outputed by COMPREFDEM is shifted by the specified offset before subtraction. A positive shift indicates a shift of the synthetic phase image to the right (range), up (azimuth).

Example input section:

```plaintext
c ... step subtrrefdem ...
c SRD_METHOD  gtopo30 // this card will be added in future
c SRD_METHOD  gtopo30corr // this card will be added in future
SRD_OUT_CINT  Outdata/cint.minrefdem.raw
SRD_OFFSET  1 -2
```

24.2 Output Description

At successful exit the process control flag is switched on:

```plaintext
subtr_refdem :  1
```
Figure 24.1 Phase image of complex interferogram. The phase of the reference DEM is subtracted ('flat earth' was already subtracted in subtrrefpha), leaving residual topographic, atmospheric, and erroneous fringes.

The output (in the products result file) looks like:

```
*_Start_subtrrefdem*

Method: gtopo30
Additional_azimuth_shift: 1
Additional_range_shift: -2
Data_output_file: Outdata/cint_minrefdem.raw
Data_output_format: complex_real4
First_line (w.r.t. original_master): 245
Last_line (w.r.t. original_master): 14964
First_pixel (w.r.t. original_master): 7
Last_pixel (w.r.t. original_master): 3998
Multilookfactor_azimuth_direction: 40
Multilookfactor_range_direction: 8
Number of lines (multilooked): 368
Number of pixels (multilooked): 499

*_End_subtrrefpha: NORMAL*_
```

Figure 24.1 shows the result of subtracting the reference phase of the radar coded DEM from the interferogram (See Figure 21.1). It can be seen the number of topographic fringes is reduced, though there still remain some residual effects. A next version of Doris will include an option to first correlate the radar coded DEM with the interferogram, to find an additional offset. The radar coded reference DEM is then shifted on (multilooked) pixel level before subtraction.
This chapter describes the processing of step FILTPHASE. This step can be optionally used to filter the (latest) complex interferogram, in order to reduce noise, e.g., for visualization or aiding the phase unwrapping. It is probably best run after step SUBTRREFPHA. A lot of warnings can be generated if an image containing a lot of zeros is processed. These warnings can be ignored.

The method goldstein is described in [Goldstein and Werner, 1998]. Basically the fringes become sharper after filtering because the peak in the spectrum (caused by the fringes) is given a higher relative weight. Method spatialconv simply is a spatial convolution with a certain kernel function, e.g., a 3 point moving average. Method spectral is a multiplication of the spectrum with the kernel specified in an input file (e.g. a spectral low pass filter (LPF)).

25.1 Input Cards

PF_METHOD goldstein
Select goldstein method ("goldstein"), or spatial convolution ("spatial-conv") with cards PF_KERNEL and PF_IN KERNEL2D, or spectral filter ("spectral") with cards PF_IN KERNEL2D, PF_BLOCKSIZE and PF_OVERLAP. For more info see implementation section.

PF_OUT_FILE cint.alpha.filtered
Output filename for complex real4 file with filtered phase (goldstein filter) where alpha is substituted. For method spatialconv, default is "cint.filtered"

PF_IN_FILE filename numlines
Optional filename of complex real4 interferogram (mph) file to be filtered instead of the default (which is obtained by reading the 'products' result files). Also specify the number of lines in this file as second argument. This card is included to be able to filter files without having to create dummy result files to 'trick' Doris. For now, the interferogram to be filtered must be complex real4.

PF_ALPHA 0.2
This card is for method goldstein only. Alpha parameter for filtering. This parameter must be between 0 (no filtering) and 1 (most filtering). The card PFKERNEL influences this value, since a higher smoothing, relative decreases the peak, and thus the effect of alpha.

PF_OVERLAP 3
This card is for method goldstein and spectral only. Half of the size of the overlap between consecutive blocks and buffers, so that partially the same data is used for filtering. The total overlap should be smaller than PF_BLOCKSIZE. If this parameter is set to BLOCKSIZE/2-1 (the maximum value for this parameter) then each output pixel is filtered based on the spectrum that is centered around it. This is probably the best, but may be time consuming.

PF_BLOCKSIZE 32
This card is for method goldstein and spectral only. Size of the blocks that are filtered. This must be a power of 2. It should be large enough so that the spectrum can be estimated, and small enough that it contains a peak frequency (1 trend in phase). (32 is recommended.)

PF_KERNEL 3 1/3 1/3 1/3
This card is for method goldstein and spectralconv only. 1D Kernel function to perform convolution. First the number of elements in the kernel is given, then the values. The kernel is always normalized to 1 by dividing the kernel by the sum of the absolute values of the kernel.
For method goldstein: defaults to kernel [1 2 3 2 1]. This kernel is used to smooth the amplitude of the spectrum of the complex interferogram.
The spectrum is later scaled by the smoothed spectrum to the power alpha.
For method spectralconv: Default is a 3 point moving average [1 1 1] convolution. The real and imaginary part is averaged separately this way.
For more info see implementation section. The output matrix has a zero valued edge of size floor(kernel/2).

PF_IN_KERNEL2D filename
This card is for method spectralconv and spectral only. Name of ascii input file to specify a 2D spatial kernel function. This file must start with a 1 line header containing numlines, numcols, scalingfactor. The next numlines lines contain the filter. numlines and numcols must be odd (centered) for method spectralconv. For method spectral they may be even; the zero frequency is located at position kernelsize/2-1 (starting at 0). The values of the kernel are multiplied by the scale factor. The kernel is not normalized in any other way. The output matrix has a zero valued edge of size floor(kernel/2).

Example of the cards for this step:

c

c
comment ___PHASEFILT___
A simple example of PF\_IN\_KERNEL2D in an ascii **input file** (use this example with method *spatialconv*)

```
5  5  0.05
 0  1  1  1  1
−1  0  1  1  1
−1 −1  0  1  1
−1 −1 −1  0  1
−1 −1 −1 −1  0
```

A second example of PF\_IN\_KERNEL2D ascii **input file** One could use cards: PF\_BLOCKSIZE 32, and PF\_OVERLAP 4, PF\_METHOD spatial

```
15 15 1.0
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

The idea is to have a template directory with ascii kernels in it, that then can be used in doris. I do not have much experience with this.

One could generate the different filters with Matlab. For spectral method, one may want to filter with an Hamming window.

If you have matlab, paste the following to your terminal (uses Matlab) to generate the ascii file
Moreover, if you want to offer your ascii kernels for standard distribution of doris, please email it to us.

matlab << _EOFHD > /dev/null
SIZE = 32;
filterfile = 'filter.hamming';
f = (standing(hamming(SIZE))*ones(1,SIZE)) .* ... 
    (ones(SIZE,1)*lying(hamming(SIZE)));
fid = fopen(filterfile , 'w');
fprintf(fid , '%i %i 1.0\n' , SIZE , SIZE);
for ii=1:SIZE
    fprintf(fid , '%4.2f ' , f (ii , :));
    fprintf(fid , '\n');
end
exit;
_EOFHD

And in Doris use the cards (using filter in spectral domain in this case):

PF METHOD spectral
PF_IN_KERNEL2D filter.hamming
PF_BLOCKSIZE 32
PF_OVERLAP 4

## 25.2 Output Description

The process control flag for this step is switched on in the products result file:

```
filtphase: 1
```

And in the same result file a section will be added like (except if PF_IN_FILE is specified):

```
************************************************************************************************************
_*Start_filtphase:
************************************************************************************************************
Method: goldstein: size , alpha , overlap : 32 0.5 4
Input_file: Outdata/cint.srp.raw
Data_output_file: cint.0.5.gf
Data_output_format: complex_real4
First_line (w.r.t. original_master): 1073
Last_line (w.r.t. original_master): 4302
First_pixel (w.r.t. original_master): 148
Last_pixel (w.r.t. original_master): 985
Multilookfactor_azimuth_direction: 10
Multilookfactor_range_direction: 2
Number of lines (multilooked): 323
Number of pixels (multilooked): 419
************************************************************************************************************
_*End_filtphase: NORMAL
************************************************************************************************************
```
Figure 25.2  Phase of unfiltered complex interferogram.

Figure 25.4  Phase of filtered complex interferogram. (Method: spatialconv.) A spatial convolution with a kernel [1 4 9 4 1] was used. Clearly a lot of detail is lost.
Figure 25.6  Phase of filtered complex interferogram. (Method: spectral.) A pointwise multiplication in the spectral domain by a 32 point hamming filter was used, a blocksize of 32, and an overlap of 4.

Figure 25.8  Phase of filtered complex interferogram. (Method: goldstein.) Parameters used are alpha=0.5, smooth=3, overlap=4. This filter seems to preserve most detail.
25.3 Implementation

25.3.1 spatialconv

The complex interferogram is convoluted with a kernel by FFT’s. The card PF_KERNEL specifies the 1D kernel. The 2D kernel is computed as: PF_KERNEL^T PF_KERNEL, e.g. for a 3 point moving average 1D kernel

\[ \frac{1}{3} [1 1 1] \tag{25.1} \]

This becomes

\[ \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \tag{25.2} \]

The blocksize for the convolution is chosen as high as possible. A 2D kernel can be specified in an input file. Only odd sized kernels can be used, but simply add a zero to an odd kernel.

If a real4 matrix containing phase should be convoluted by a certain kernel, first convert this real4 to a complex real4 matrix. Do this either by computing the phase for complex numbers with amplitude 1, or by setting the real part of the file to the phase and the imaginary part to 1 (arbitrary).

25.3.2 spectral

This method is implemented the same as goldstein method w.r.t. the overlap, blocksize etc. Algorithm per block (SIZE,SIZE) is to perform a 2D FFT of the block, and then to multiply pointwise with the kernel, which is padded with zeros. The kernel is centered around zero frequency.

25.3.3 goldstein

The algorithm is implemented as:

- Read in buffer \( B_i \) of PF_BLOCKSIZE lines (overlap).
- Get block \( B = B_{ij} \) as input block, see Fig. 25.9.
- \( B = \text{fft2d}(B) \) (obtain complex spectrum)
- \( A = \text{abs}(B) \) (magnitude of spectrum)
- \( S = \text{smooth}(A) \) (convolution with kernel)
- \( S = S / \max(S) \) (S between 0 and 1)
- \( \alpha \cdot (S) \) (weight complex spectrum)
- \( B = \text{ifft2d}(B) \) (result in space domain)
- If all blocks of buffer done, write to disk.
Figure 25.9 Buffering of complex interferogram in blocks for phase filtering.

For a block \([\text{pixlo:pixhi}], \text{ e.g., } [0:15]\), the output equals for an overlap \(=3\), \([\text{pixlo}+\text{overlap:pixhi}-\text{overlap}], \ [3:12]\). The number of output equals \(\text{size-2overlap} = \text{pixhi-pixlo+1-2overlap} = 10\).
In this chapter the processing of step UNWRAP is described. This step is currently not implemented within the Doris software. To obtain the unwrapped interferogram, you should use another software, for example one of the routines of [Ghiglia and Pritt, 1998] which can be obtained by ftp at ftp.wiley.com/public/sci_tech_med/phase_unwrapping. These software should not be considered public domain, you ought to buy the book. The slant to height conversion and geocoding can only be done with an unwrapped interferogram.

Recently "snaphu" of Curtis Cheng was put in the public domain. It is recommended you install this software as standalone executable, and continue with Doris for geocoding afterwards. METHOD snaphu can be used from within Doris. But experience has to be gained how this software best performs.

Sometimes the coherence as computed by doris seems to contain NaNs (not-a-number). snaphu does not expect this and exits when this happens. In Matlab the created coherence file can be easily corrected with, e.g.,

```matlab
q=freadbk('9192_6687.coh',2577,'float32');
idxx=isnan(q);
idx=where(idxx==1);
q(idx)=0.0001;
fwritebk(q,'coh_no_nan','float32');
```

If you use a standalone application to unwrap the interferogram, you might have to mimic the output as described below, so Doris can obtain the current filename and dimensions for the unwrapped interferogram from the interferogram result file.

### 26.1 Input Cards

For the snaphu program, please also refer to their website and read the man page. Doris is a wrapper for a system call to the executable snaphu. Therefore, a program called snaphu should be executable and in your path. An input file for snaphu is created in the current directory. You can rerun snaphu with a changed inputfile from the prompt if required (but keep same output file name, format as it is in the result file for the interferogram.) We assume to unwrap the complex interferogram, mph format always for snaphu.
UW_METHOD     SNAPHU | TREEFRAMON
Select method for unwrapping. For general users, if they have installed
the Snaphu program, this will make a system call. Other methods are
not available for public domain.

UW_OUT_FILE    uint.hgt
Output filename of unwrapped interferogram.

UW_OUT_FORMAT  HGT | REAL4
Output format of unwrapped interferogram.

UW_SNAPHU_MODE TOPO | DEFO | SMOOTH | NOSTATCOSTS
Output format of unwrapped interferogram. Snaphu options -t, -d, -s
respectively. Refer to snaphu manual for more information.

UW_SNAPHU_COH  filename
use specified file for correlation values. This file must be registered and
have the same number of looks. Snaphu option -c. Refer to snaphu
manual for more information.

UW_SNAPHU_LOG  filename
Output log file name for snaphu option -l. Refer to snaphu manual for
more information.

UW_SNAPHU_INIT MST | MCF
Output log file name for snaphu option -l. Refer to snaphu manual for
more information.

UW_SNAPHU_VERBOSE ON | OFF
Snaphu option -v. Refer to snaphu manual for more information.

26.2 Output Description

At successful exit the process control flag is switched:

| Unwrap: 1 |

The section for the unwrapping in the result file for the interferogram looks like the following
for method TREEF (file name and format are used later):

```
/* Start unwrap:
*******************************************************************************/
Data_output_file: Outdata/uint.raw
Data_output_format: real4
Data_output_file_regions: Outdata/regions.raw
Data_output_format: short int (2B)
First_line (w.r.t. original_master): 1001
```
If the 'Data output format:' is 'real4', then the output is assumed to be real4 unwrapped phase values. If the unwrapping was not successful, these pixels are set to -999. and ignored for slantrange to height conversion and differential insar.

The 'Data output format:' of the unwrapped interferogram also can be 'hgt', a band interleaved format (amplitude, phase) for SNAPHU. For more details on the definition if unwrapping went wrong, see Annex B.
This chapter describes the processing step DINSAR, which stands for (3 or 4 pass) differential interferometry.

Three pass differential interferometry is described in [Zebker et al., 1994]. It is a method to remove the topographic induced phase from an interferogram containing topography, deformation, and atmosphere. This module thus can also be used to study atmospheric effects in interferograms, if no deformation is expected.

This step can be performed if an unwrapped topography interferogram (topo pair) and a complex deformation interferogram (defo pair) are present (with a common master, perform 2 separate runs of Doris to achieve this). The interferograms have to be corrected for the phase of the 'flatearth' (see step SUBTRREFPHA), and sampled on the same grid (see step RESAMPLE). The files must have the same multilook factors and the same dimensions (i.e. overlap exactly). The perpendicular baseline of the topo-pair should be larger than that of the defo-pair, to prevent that noise is blown up, but this cannot always be controlled of course.

This step is performed in the defo-pair processing tree. First create a directory to run the topo-pair processing until an unwrapped interferogram is obtained (keep the master, slave, and products result files). Then perform the defo-pair processing. After interferogram generation and 'flatearth' subtraction, start this step (DINSAR), specifying the location of the result files of the topo-pair processing with input cards. For 3 pass, use a common master. For 4 pass, coregister the complex interferogram on the complex interferogram of the defo-pair, and then unwrap, or first coregister master and slave on the master of the deformation pair.

To geocode the differential phase values, geocode the topo-interferogram and use the latitude/longitude matrices for the differential grid.

27.1 Input Cards

<table>
<thead>
<tr>
<th>DI_OUT_FILE</th>
<th>differentialinterf.raw</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output filename for complex real4 file with differential phase (in slant range system).</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DI_IN_TOPOMASTER</th>
<th>same as master result file card</th>
</tr>
</thead>
</table>

Specify this card if 4 pass differential interferometry is required. Do not use this card for 3 pass, or use the same name as the master result file for the defo pair (see chapter 2). Filename of the master of the topo-pair. To obtain the orbit and other parameters for the topo master.

**DI_IN_TOPOSLAVE**  
*result file name*  
Filename of the slave of the topo-pair. To obtain the orbit and other parameters for the primary slave.

**DI_IN_TOPOINT**  
*result file name*  
Filename of the interferogram result file of the topo-pair. To obtain the name and dimensions of the unwrapped (topography) interferogram.

**DI_OUT_SCALED**  
*filename*  
Filename for optional (debug) output of a real4 file with scaled (with ratio of perpendicular baselines) unwrapped topo interferogram.

Example of the cards for this step:

```plaintext
c
c  comment       ___ DINSAR 3 PASS ___
c
DI_OUT_FILE     ./Outdata/difg.raw
DC  DI_IN_TOPOMASTER /data/project/topo/master.res // if 4 pass method
DI_IN_TOPOSLAVE /data/project/topo/slave.res
DI_IN_TOPOINT   /data/project/topo/products.res
DC  DI_OUT_SCALED  ./Outdata/scaled.raw      // debug
```

### 27.2 Output Description

In the defo processing *result file* for the products, the *process control flag* for dinsar is switched on.

```plaintext
dinsar: 1
```

A complex real4 (‘mph’) file is created with the wrapped differential phase. (The amplitude is the same as that of the original ‘deformation’ interferogram). A complex value (0,0) indicates unwrapping was not ok.

If the the debug version of Doris is used (compiled with _DEBUG) then ascii matrices are dumped for Linenumber, Pixelnumber, Bperptopo, Bperpdefo, and Ratio.

Figures 27.1, 27.2, and 27.3 give example of output.

### 27.3 Implementation

See also [Zebker et al., 1994]. See figure 27.4
Figure 27.1  Phase of complex ‘topography’ interferogram (flat earth corrected, cropped). The area is dead sea Israel. Temporal baseline is 1 day (tandem). The perpendicular baseline is approximately 105 meters. This interferogram has been coregistered on the defo pair (27.2) by tricking Doris.

Figure 27.2  Phase of complex ‘deformation’ interferogram (flat earth corrected, cropped). The area is dead sea Israel. Temporal baseline is 28 months day. The perpendicular baseline is approximately -30 meters.
Figure 27.3  Phase of differential complex interferogram (result of step DINSAR, cropped). The area is dead sea Israel. The topography is removed from the original interferogram (Figure 27.2) by scaling the topography interferogram (Figure 27.1). The perpendicular baseline is approximately 30 meters.

Simple equations for topo-pair (no deformation, no atmosphere, no other errors, $r_1 \parallel r_2$)

$$\theta = \theta_0 + \delta \theta$$  \hspace{1cm} (27.1)

$$B_{\parallel} = r_1 - r_2$$  \hspace{1cm} (27.2)

$$B_{\perp} = B \cos(\theta - \alpha) = B \cos(\alpha - \theta)$$  \hspace{1cm} (27.3)

$$B_{\parallel} = B \sin(\theta - \alpha) = -B \sin(\alpha - \theta)$$  \hspace{1cm} (27.4)

The baseline components, for points on the reference ellipsoid ($h = 0$) are

$$[B_{\perp}]_{h=0} = B_{\perp 0} = B \cos(\theta_0 - \alpha)$$  \hspace{1cm} (27.5)

$$[B_{\parallel}]_{h=0} = B_{\parallel 0} = B \sin(\theta_0 - \alpha)$$  \hspace{1cm} (27.6)

The ‘true’ phase of the interferogram is

$$\phi = -\frac{4\pi}{\lambda} B_{\parallel}$$  \hspace{1cm} (27.7)

And corrected for the phase of the reference body

$$\phi = -\frac{4\pi}{\lambda} (B_{\parallel} - B_{\parallel 0})$$  \hspace{1cm} (27.8)

For the defo-pair (1,3), denoted with a prime, similar equations follow. Deformation in the line of sight (range), that occurred in between the acquisitions, is denoted by $\Delta r$

$$\Delta r = -\frac{\lambda}{4\pi} \phi \Delta r$$  \hspace{1cm} (27.9)
A positive $\Delta r$ implies deformation in the $B_\parallel$ direction (away from the sensor, i.e., subsidence). The phase of this interferogram is
\[
\phi' = -\frac{4\pi}{\lambda} (r_1 - (r_3 + \Delta r)) = -\frac{4\pi}{\lambda} (B_\parallel' + \Delta r) \tag{27.10}
\]
Combining the expressions for the interferometric phase for the topo-pair (27.7) and defo-pair (27.10) yields:
\[
\phi' = \phi \frac{B_\parallel'}{B_\parallel} + \frac{4\pi}{\lambda} \Delta r \tag{27.11}
\]
The problem here is that the 'true' parallel baselines are unknown.

The (actually wrapped) phase of the deformation interferogram, corrected for reference phase, is defined as:
\[
\phi' = -\frac{4\pi}{\lambda} \left[ \frac{B_\parallel'}{B_\parallel} - B_\parallel 0' + \Delta r \right] \tag{27.12}
\]
where $B_\parallel' = B_\parallel 0' - \alpha'$. Using the approximation for small $\delta\theta$ (which is about $1^\circ$ or 0.0175 rad for terrain height differences of 5 km)
\[
\sin(\beta + \delta\theta) = \sin \beta \cos \delta\theta + \cos \beta \sin \delta\theta \approx \sin \beta + \delta\theta \cos \beta \tag{27.13}
\]
it follows from equation 27.13 that the 'flat earth' corrected phase equals
\[
\phi' = -\frac{4\pi}{\lambda} \left[ B_\parallel' \left( \sin \beta_\parallel' + \delta\theta \cos \beta_\parallel' \right) - B_\parallel' \sin \beta_\parallel' + \Delta r \right] \tag{27.14}
\]
The corrected phase for the topo pair equals $\phi = -\frac{4\pi}{\lambda} \delta\theta B_{\perp 0}$ (using the same approximation), and combining this with 27.15 yields (for the 'flat earth' corrected phases)
\[
\phi' = \phi \frac{\delta\theta B_{\perp 0}'}{\delta\theta B_{\perp 0}} = \phi \frac{B_{\perp 0}'}{B_{\perp 0}} - \frac{4\pi}{\lambda} \Delta r \tag{27.15}
\]
or
\[
\Delta r = -\frac{\lambda}{4\pi} [\phi' - \phi \frac{B_{\perp 0}'}{B_{\perp 0}}] \tag{27.16}
\]
or for the phase $\phi_{\Delta r}$ caused by the deformation $\Delta r$
\[
\phi_{\Delta r} = \phi' - \frac{B_{\perp 0}'}{B_{\perp 0}} \phi \tag{27.17}
\]
This important equation shows how to obtain offset vectors from 3 SLC images, i.e., by scaling the (reference phase corrected) unwrapped phase of the topo-pair by the ratio of the perpendicular baselines (to points on reference body), and subtracting this from the phase of the defo-pair. This can thus be performed without the 'true' values for $\theta$ are required.
Figure 27.4 Geometric configuration for 3-pass differential insar. The orbits go ‘into’ the paper. All angles are defined counterclockwise. The terrain element $P$ corresponding to the radar coordinate $(l,p)$ is located at a height $h$ above the ellipsoid. The perpendicular baseline required for this method is the one for points $P$ located on the reference ellipsoid ($h = 0$). $\delta \theta$, the change in $\theta$ since $P$ is on a height $h$, due to a 5 km height difference, is approximately $1^\circ$. 
27.3.1 Algorithm

Input is the unwrapped topo-interferogram (corrected for 'flat earth'). Format is hgt, or real4. Not unwrapped thus indicated by NaN==999. (real4) or if amplitude==0 (hgt). Defo-interferogram is wrapped (complex real4, mph) (specified in interferogram result file).

1. Obtain orbit for topo-slave (result file). Obtain filename/dimension of unwrapped interferogram (result file).

2. Read in matrices, appropriate size/format etc. per line. Check if they exactly overlap.

3. Compute $B_\perp$ and $B'_\perp$ on a small grid (20x10 points over the image).

4. Model the ratio of the perpendicular baselines by a 2D polynomial of degree 1. ($r(l,p) = a00 + a10l + a01p$) Give statistics for max. error due to modelling. (It seems the ratio hardly changes over the image for ERS1/2).

5. Compute wrapped deformation phase (phase corrected for topography) with formula 27.17, using the modeled ratio $r_{ij}$. (Actually compute it complex: $c_{ij} = \cos(r_{ij} \cdot \phi) - i \sin(r_{ij} \cdot \phi)$.)

6. Set not unwrapped regions to (0,0).

7. Write output file (complex real4, mph format). (If a problem with unwrapping occurred, write (0,0).) If requested, also write the scaled unwrapped interferogram.
In this chapter the processing of step SLANT2H is described. In this step in principle the heights in the radar coded system are computed. However with the exact method, the geocoding can be done in the same step.

The results of the three implemented methods are different, so a comparison has been made.

Processing is in buffers for all methods, while it is possible just to do it line by line. In case a polynomial has to be evaluated (rodriguez method) it is more efficient to have a buffer.

### 28.1 Input Cards

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>S2H_METHOD</td>
<td><em>ambiguity</em></td>
</tr>
<tr>
<td></td>
<td>Method selector. <em>ambiguity</em> geocodes as well, uses height ambiguity to compute the height. <em>schwabisch</em> method uses polynomials to compare the phase with the reference phase. <em>rodriguez</em> method uses geometry, contains an approximation, it is not clear how to compute a certain parameter.</td>
</tr>
<tr>
<td>S2H.OUT.HEI</td>
<td><em>hei.raw</em></td>
</tr>
<tr>
<td>S2H.OUT.PHI</td>
<td><em>phi.raw</em></td>
</tr>
<tr>
<td>S2H.OUT.LAM</td>
<td><em>lam.raw</em></td>
</tr>
<tr>
<td>S2H_NPOINTS</td>
<td>200</td>
</tr>
<tr>
<td>S2H_DEGREE1D</td>
<td>2</td>
</tr>
</tbody>
</table>
Only for schwabisch method, the degree of the 1d polynomial to fit reference phase through at every location.

\[ S2H_{\text{NHEIGHTS}} = s2h\_degree1d+1 \]

Only for schwabisch method, the number of heights to evaluate the reference phase. Minimum is default.

\[ S2H_{\text{DEGREE2D}} = 5 \]

Only for schwabisch method, the degree of the 2d polynomial to fit 1d coefficients as function of location.

Example input:

```
comment --- SLANT 2 HEIGHT CONVERSION ---
S2H_METHOD schwabisch
S2H_NPOINTS 500
S2H_DEGREE1D 2
S2H_NHEIGHTS 3
S2H_DEGREE2D 5
S2H_OUT_HEI Outdata/hei.schw

S2H_METHOD ambiguity
S2H_OUT_HEI Outdata/hei.ambi
S2H_OUT_LAM Outdata/lam.ambi
S2H_OUT_PHI Outdata/phi.ambi

S2H_METHOD rodriguez
S2H_OUT_HEI Outdata/hei.rodr
```

### 28.2 Output Description

The **process control flag** is switched at successful exit:

```
slant2height : 1
```

An example of the output in the **products result file**:

```
************************************************************
< _Start_slant2h
************************************************************
Method: schwabisch
Data_output_file: Outdata/hei.schwabisch
Data_output_format: real4
First_line (w.r.t. original_master): 1001
Last_line (w.r.t. original_master): 2105
First_pixel (w.r.t. original_master): 501
Last_pixel (w.r.t. original_master): 700
Multilookfactorazimuth_direction: 10
Multilookfactorrangefirection: 2
Ellipsoid (name,a,b): WGS84 6.37814e+06 6.35675e+06
```
In the output files the height is stored. (number of lines etc. multilooked unwrapped interferogram.) etc.

### 28.3 Implementation

#### 28.3.1 Method ambiguity

This method yields first the heights of the (line,pixel) and the position $P(x,y,z)$. In this manner the geocoding can be done in the same step. by converting $P(x,y,z)$ (known $h$) to phi,lambda. If there is a trend in the height this has to be removed first, e.g. by using tiepoints. This means the computed phi and lambda matrices are not correct anymore.

With the baseline defined as in Annex B the following equations hold.

\begin{align}
B_\parallel &= r_1 - r_2 \\
B_\parallel &= B \sin(\theta - \alpha) \\
B_\perp &= B \cos(\theta - \alpha)
\end{align}  

(28.1) (28.2) (28.3)

![Geometric configuration for slant to height conversion.](image)

Figure 28.1  Geometric configuration for slant to height conversion.
Note the sign, must be minus.

\[ \phi_i = -\frac{4\pi}{\lambda} r_1 + \phi_{obj} \] (28.4)

Content of unwrapped interferogram (never mind the \(-\phi_R\)).

\[ \phi = \phi_1 - \phi_2 - \phi_R = -\frac{4\pi}{\lambda} B_\parallel - \phi_R \] (28.5)

\[ \frac{d\phi}{d\theta} = -\frac{4\pi}{\lambda} \frac{dB_\parallel}{d\theta} = -\frac{4\pi}{\lambda} B_\perp \] (28.6)

Geometric equation:

\[ h = H - r_1 \cos \theta \] (28.7)

\[ \frac{dh}{d\theta} = r_1 \sin \theta \] (28.8)

Height ambiguity:

\[ \frac{dh}{d\phi} = \frac{dh}{d\theta} \frac{d\theta}{d\phi} = \frac{\lambda}{4\pi} \frac{r_1 \sin \theta}{B_\perp} = -\frac{\lambda}{4\pi} \frac{r_1 \sin \theta}{B_h \cos \theta + B_v \sin \theta} \] (28.9)

Final equation to convert phase to height:

\[ h = -\frac{\lambda}{4\pi} \frac{r_1 \sin \theta}{B_h \cos \theta + B_v \sin \theta} \phi \] (28.10)

The procedure to compute the height is as follows (note that computation is skipped if unwrapping went wrong, indicated by NaN (not a number) in the unwrapped interferogram.)

1. for all lines
   
   (a) .i1. compute \( B_h \) and \( B_v \) (by computing \( h \) for middle pixel, then compute baseline).
   
   (b) .i2. \( h_{current} = 0 \)
   
   (c) for all pixels
      
      i. .j1. if value unwrapped phase equals NaN \( \equiv -999 \) then goto next pixel
      
      ii. .j2. compute \( \theta \) (corresponding to \( h \))
      
      iii. .j3. \( h_{last} = h_{current} \), compute \( h_{current} \) with formula
      
      iv. if \((h_{last} - h_{current}) > k\) then goto .j2.

28.3.2 Method rodriguez

Use same definitions as exact method, see also Annex B. See also [Rodriguez and Martin, 1992]. This method uses the geometry to compute \( \sin(\theta - \alpha) \). There are two errors in it for now. First \( H \) is not computed exact. Second the baseline parameters are not computed exact per line. \( P \) is evaluated at reference surface and \( S \) according to that position. This means that if the orbits are not parallel the point \( S \) is not computed correctly, which introduces errors in the baseline computation. The co-registration model is better used for that.

Known:

\( r_1 \) range to \( M, P \)
position M
B baseline
ξ baseline orientation with regard to (equals our def. of alpha)

Compute:
theta (angle state, look: with formulas exact (no iterations)
H height sat above some surface ?? how to compute this exact??

The following we derived for our baseline definition: (beta = angle (2-1,P-1) counterclockwise:)
\[
\beta = \angle(2 - 1, P - 1)
\] (28.11)
\[
\cos \beta = \cos\left(\frac{1}{2} \pi + (\alpha - \theta)\right)
\] (28.12)
\[
= \sin(\theta - \alpha)
\] (28.13)
\[
(r_1 - B\parallel)^2 = r_1^2 + B^2 - 2r_1B \cos \beta
\] (28.14)
\[
\sin(\theta - \alpha) = \frac{(r_1 - B\parallel)^2 - r_1^2 - B^2}{-2Br_1}
\] (28.15)
\[
B\parallel = -\frac{\lambda}{4\pi} \phi
\] (28.16)

So theta can be solved for exact with these formulas. Note:
\[
\theta - \alpha = \arcsin \left(\frac{(r_1 - B\parallel)^2 - r_1^2 - B^2}{-2Br_1}\right) = x
\] (28.17)
\[
\theta = \arcsin x + \alpha \quad \vee \quad \theta = \pi - \arcsin x + \alpha
\] (28.18)

I did not find an efficient way to always use the right expression yet. Now I use the fact that theta is about 20 degrees, but it should be possible to find out quadrants.

Compute H from known: theta, position Master (rho1), r1 In triangle (1,P,0) three Start with cosine law for line across theta = p
\[
p^2 = \rho_1^2 + r_1^2 - 2\rho_1 r_1 \cos \theta
\] (28.19)

Then compute cosine of angle mu across r1 in same triangle
\[
r_1^2 = \rho_1^2 + p^2 - 2\rho_1 p \cos \mu
\] (28.20)

Unclear how to compute H exact. for now use approximation. (set radius of earth at location of satellite equal to radius at location of P) compute satellite height by Bowring’s method (xyz2ell) then radius R of earth at phi,lambda to satellite:
\[
R = \rho_1 - H_{sat}
\] (28.21)

Approximate H in this way
\[
H = \rho_1 - R \cos \mu
\] (28.22)
Compute error of this approximation (preliminary study):

This will cause a bias and some trend in the height. Because there likely already is a trend due to orbit errors, this is not as bad as it might seem. By using tie points a good height may be computed. For now we did not implement a routine that uses tie points.

New way of computing H: (NOT implemented, to difficult)

1. compute in new system \((x,y)\) coordinates of \(1(0,\rho_1), P(\ldots),\ldots\)
2. ellips equation in the same system, rotated over co-latitude
3. snipunt \(P,\text{ellips} := R\)
4. \(H = \rho_1-R_q\)

Problems with this method: how do you know orientation of theta? rotation of ellips to new system.

A few more notes:

\[
B_\parallel = -\frac{\lambda}{4\pi}(\phi + \phi_R) \tag{28.23}
\]

So the reference phase has to be added again in order to compute \(B_{\parallel}\). (otherwise \(B_{\parallel}\) is 0.001 m or so.)

Processing:

1. per line compute \(B,\alpha\);
2. per pixel
   - (a) \(\phi\) to \(B_{\parallel}\)
   - (b) \(r\) known
   - (c) compute \(\theta\) (exact?)
   - (d) compute \(p\)
   - (e) compute \(\mu\)
   - (f) compute \(H\)
   - (g) compute \(h\)

The idea is to compute \(H\) from the position of \(M\). (\(H\) can be computed by (reference needed) as shown.) And then to find \(\theta = f(B,\phi, r)\). And then find \(h\) with the first equation. This method is implemented to check our exact method, the results are very different.

In this method the point \(P\) does not have to be computed. (though in order to compute baseline components we will compute a point \(P\) on height \(h\) (evaluate, iteratively) once for every line.)

A better way might be to use the co-registration model to compute the point \(S\).
28.3.3 Method schwabisch

This method is described in [Schwäbisch, 1995]. It is a fast method that yields the radar coded heights. It is based on the idea to first compute the reference phase at a number of heights and then to compare the actual phase from the interferogram with these values to determine the height.

A problem is that the interferogram does not contain the reference phase anymore, so that has to be added to the estimated phi.

It uses a number of steps which are described below.

1. Compute reference phase at a NL locations (line,pixel) at NH (=3) heights (0, 2000 and 4000m).
   for h=0, 2000, 4000:
   - ellips.a=wg84.a+h, ellips.b=wg84.b+h
   - compute position of master satellite, corresponding point P on ellips and position of
     slave satellite, see annex B.
   - \( B_\parallel = r_1 - r_2 \)
   - \( \phi_R = -\frac{4\pi}{\lambda} B_\parallel \)
   - store the values and locations and heights.

   Note that the reference phase defined like this typically is something like 5000 (rad) even
   for h=0. Therefor the reference phase for h=0 is set to 0, (because in the unwrapped
   interferogram the reference phase is removed, if the phase of the unwrapped interferogram
   is 0, then this should yield a height of 0) and the reference phase for height h is set
   to refphaseh - refphase0. (This makes the computations as done later a little stupid, to
   estimate coefficients which are by definition equal to 0.) (An other possibility is not to do
   this here, but later when the functions are evaluated to add the reference phase to each
   pixel. I have tested this and the results are identical (+- .4 m))

2. Compute for each location a polynomial (1d degree 1dD (= NH-1)) to describe the height
   as a function of reference phase at these points.
   For each location it holds (\( \phi_i \) for height i):
   \[
   h = 0 = \alpha_0 + \alpha_1 \phi_0 + \alpha_2 \phi_0^2 \\
   h = 2000 = \alpha_0 + \alpha_1 \phi_1 + \alpha_2 \phi_1^2 \\
   h = 4000 = \alpha_0 + \alpha_1 \phi_2 + \alpha_2 \phi_2^2 
   \]
   So it is easy to solve (exact) for \( \alpha_i \) per location.

3. Compute (1dD+1=NH) polynomials to describe the coefficients of the previous step as a
   function of location. (now, for a random location, the coefficients of height as a function
   of the reference phase can be computed.) For example for \( \alpha_0 \) computed at NL locations
   (l,p) a 2d polynomial can be used:
   \[
   \alpha_{0lp} = \beta_{00} + \beta_{10} l + \beta_{01} p + \beta_{20} l^2 + \ldots \\
   = \sum_{i=0}^{d} \sum_{j=0}^{i} \beta_{i-j,j} l^{i-j} p^j 
   \]
   (28.27)
   A linear system can be easily set up and solved (least squares) by cholesky factorization. A
   rescaling needs to be applied to avoid instability. The system can be solved simultaneously
for all alphas, because the normal matrix (and factorization) remains the same, but somehow our cholesky routine introduced an error (which is probably caused by using fortran in c) so we just solve 3 separate times with the same factored normal matrix.

4. Evaluate for all points at (line,pixel) with ok unwrapped phase the 2d polynomial to obtain the coefficients of the height(\(\phi\)) function. Then evaluate the 1d polynomial to obtain the height.

For all \((l,p)\) with ok unwrapped phase:

- Compute the alphas.
  - For betas appropriate to alpha0:
    \[
    \alpha_0 = \sum_{i=0}^{d} \sum_{j=0}^{i} \beta_{i-j,j} l^{i-j} p^j
    \] (28.28)
  - For betas appropriate to alpha1:
    \[
    \alpha_1 = \sum_{i=0}^{d} \sum_{j=0}^{i} \beta_{i-j,j} l^{i-j} p^j
    \] (28.29)

- repeat computing alphas until you have them all (1dD+1).
- Compute the height.
  \[
  h = \sum_{i=0}^{1dD} \alpha_i \phi^i
  \] (28.30)

### 28.4 Comparison of the methods

Here a simple test is described that was performed to see the differences between the methods. A unwrapped interferogram was obtained of the Veluwe (Holland) by processing the bottom half of the Tandem images 3512 (ERS2) and 23185 (ERS1). The interferogram was multilooked by 40x8, resulting in 367 lines and 610 pixels. There were about 3 fringes.

Baseline:
\[
B = 185 m, \alpha = 3^\circ \\
B_{||} = -62, B_\perp = -173 \\
B_h = -184, B_v = -2
\]

The processing was done with a debugger version of the Doris software, so the cpu times are not really representative for the performance of Doris. Table 28.1 shows some processing parameters.

<table>
<thead>
<tr>
<th>Method</th>
<th>cpu options</th>
<th>remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ambiguity</td>
<td>60</td>
<td>-</td>
</tr>
<tr>
<td>Schwabisch</td>
<td>12</td>
<td>1000 pnts, 1d=2, 2d=5 -</td>
</tr>
<tr>
<td>Rodriguez</td>
<td>4</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 28.1  Processing with the methods
Figure 28.2  Comparison methods in azimuth direction for line 250.

Figure 28.3  Comparison methods in range direction for pixel 110.

Figure 28.2 and 28.3 show plots for these three methods. Figure 28.4 shows a comparison between schwabisch and ambiguity method.

It can be seen there is a trend between schwabisch and ambiguity. And there is an offset with rodriguez.

Schwabisch method is always higher then ambiguity, suggests error in computation of baseline parameters?

Some other tests also showed that the method Schwabisch, as implemented in Doris, seems to yield a (more or less) scaled version of the height of the ambiguity method. (The schwabisch being higher).

After rescaling (with a factor 0.86) of the heights obtained by schwabisch method to the level of the ambiguity method, the differences between both methods were only a few meters.
Figure 28.4  Comparison ambiguity, schwabisch methods for total image.
In this chapter the processing of step GEOCODE is described. In this step the radar coded heights are converted to geocoded coordinates (i.e., to a known reference system).

Input is the height file from the step SLANT2H. Output are two files containing the latitude (phi) and longitude (lambda) corresponding to the height file.

These files can be further processed with programs cpxfiddle, proj, and GMT to create a DEM in a regular grid in any projection desired (UTM for example). See the bin directory and the shell scripts there for examples how to do this. Likely, for each specific application, these scripts are best copied and adapted to your needs. The interpolated matrices from GMT are in grd format that can be handled by Matlab, etc.

### 29.1 Input Cards

- GEO_OUT_PHI geo_phi.raw
  
  Output file name for latitude.

- GEO_OUT_LAM geo_lam.raw
  
  Output file name for longitude.

Example input:

```plaintext
cc
  comment    --- GEOCODING ---
cc
GEO_OUT_LAM  Outdata/lam.raw
GEO_OUT_PHI  Outdata/phi.raw
```

If you want to obtain the latitude/longitude of the pixels in an interferogram that was created, but you do not have a DEM in radar coordinates available, you will have to create one. This means that you will have to edit the **products result file** and create a SLANT2H section, see Chapter 28 for a description of this section in the **products result file**. The section will look something like:
(Also set the pcf to 1 on top of the *products result file.*) If you have an external DEM you can compute the required file (Outdata/dummy_height.raw) using step COMPREFDEEM. The dimensions and multilooking can be copied from the interferogram section.

If your area is flat, you may want to use a dummy file filled with zeros. You can create such a file with appropriate dimensions using Matlab for example. Alternatively, the much faster Unix way would be along these lines. First compute the height (number of lines) of the dummy file:

```
echo "((2105−1001+1)/10)" | bc -l 110.5
```

then the width (number of pixels):

```
echo "((700−501+1)/2)" | bc -l 100
```

(i.e., the file should be 110 lines by 100 pixels of 4 byte). Now create the file using dd:

```
dd if=/dev/zero of=Outdata/dummy_height.raw count=110 bs=400
```

### 29.2 Output Description

The *process control flag* is switched at successful exit:

```
geocoding: 1
```

An example of the output in the *products result file*:

```
*********************
*_Start_geocode_
*********************
Data_output_file_hei (slant2h): Outdata/hei.ambi
Data_output_file_phi: Outdata/phi.raw
Data_output_file_lamda: Outdata/lambda.raw
Data_output_format: real4
First_line (w.r.t. original_master): 1001
Last_line (w.r.t. original_master): 2105
*********************
```
29.3 Implementation

Known are the heights of each pixel in the master (line,pixel) system. The point P(x,y,z) corresponding to a (line,pixel) is computed with the 3 equations (see Annex B) in such a way that it lies on an ellipsoid of height h above the reference ellipsoid. When these coordinates are known, the equations of Bowring are used to transform them to an ellipsoid system (φ, λ, h). The semimajor axis is denoted by a, and the semiminor axis is denoted by b. The squared first eccentricity by:

\[ e^2 = \frac{a^2 - b^2}{a^2} \]  

(29.1)

The squared second eccentricity by

\[ e^2 = 1 - e^2 = \frac{a^2 - b^2}{b^2} \]  

(29.2)

\[ r = \sqrt{x^2 + y^2} \]  

(29.3)

\[ \nu = \arctan_2(z \cdot a), (r \cdot b)) \]  

(29.4)

\[ \sin 3 = \sin^3 \nu \]  

(29.5)

\[ \cos 3 = \cos^3 \nu \]  

(29.6)

\[ \phi = \arctan_2((z + e^2 \cdot b \cdot \sin 3), (r - e^2 \cdot a \cdot \cos 3)) \]  

(29.7)

\[ \lambda = \arctan_2(y, x) \]  

(29.8)

\[ N = \frac{a}{\sqrt{1 - e^2 \sin^2 \phi}} \]  

(29.9)

\[ h = \frac{r}{\cos \phi} - N \]  

(29.10)


In this annex the additional software that aids running Doris is described. This additional software is not required to run Doris, but it is highly recommended to use. We choose to only use freely available packages.

A.1 Packages

The program getorb for automatic retrieval of the Delft precise orbits for ERS1/2. See our web pages for a link.

The package gv (recommended) or ghostview to view postscript files, generated by plotcpm and plotoffsets, and cpx2ps.

The GMT (generic mapping tools). Highly recommended. We generally generate postscripts for visualizing the phase and amplitude with this program. See also cpxfiddle and cpx2ps.

A.2 Tools

We have developed some utilities for running Doris, and some display tools based on GMT.

A.2.1 Installation of SARtools

After downloading the gzipped, tarred archive of the SARtools, the installation is best done with a Makefile. I assume you are familiar with 'make' to compile code. If you are not, find someone who is. (See also annex ??).

1. Use a distinct directory to put the files, e.g.,

   mkdir /opt/doris_v3.16 (Document Revision : 1.55)
   cd /opt/Doris_v3.16 (Document Revision : 1.55)

2. Download the archived SARtools via the download area of our webpages at www.geo.tudelft.nl/doris/: SARtools.tar.gz
3. Expand compressed files: gzip -d SARtools.tar.gz
   This leaves a file **SARtools.tar**

4. Extract the files from the archive:
   tar -xvf SARtools.tar
   Now a subdirectory has been created, **SARtools**.

Now the compilation of the utilities can start.

1. Compile all utilities and install the executables:
   - cd SARtools (inspect/edit the Makefile, set INSTALLDIR)
   - make (this should compile the code)
   - make install (this installs in /usr/local/bin/)

2. Make sure the Installation directory is in your path. For tcsh users, it should be in your .cshrc file (startup file). Add it with a (csh) command like:
   _set path = ( /usr/local/bin/ $path )_

**A.2.2 run script**

This script can generate template **input files** and directories, and therefore generally speeds up the processing. If it is installed in a Bin directory (in your path), it can be run from several project directories for uniform processing. It uses the environment variables **EDITOR** and **PAGER** if set.

The basic idea is to start with

```
run -g
```

to generate the input, and then to edit (default with editor vi or EDITOR) the first **input file**.

```
run -e1
```

After you saved the file (located in directory Inputfiles), type:

```
run -s1
```

to process (call to doris software) the first **input file**. The output that goes to stdout can be viewed with

```
run -v1
```

(It is redirected to a file in directory Outinfo). To view the result files that are created by Doris, use

```
run -r1 (master);
run -r2 (slave);
run -r3 (products);
run -r4 (logfile)
```
This should be repeated for other steps. Of course this run file is only a helping hand, not the
solution to all your problems. Be careful! Template values aren’t always the best settings. Feel
free to improve it.

Usage:
run -s/-e/-v step [-f \inputfile -r file-id -d]
/-g [-M master -S slave -B baseline -R remark -A author]
/-h

For more help type run -h.

A.2.3 cp xfiddle

With this utility one can fiddle about with binary complex (cpx) files of all kinds of formats
(though only pixel interleaved). Make a cutout, multilook, scale, exponent, subsample, mirror,
etc. Now it also supports the generation of SUNraster files for visualization of the phase of
complex files (smaller temp files required). It is written in C++ using a template function.

Input is a complex file, for example the output of Doris, or SLC data. It should be pixel
interleaved, i.e., RE,IM, RE,IM, RE,IM, ... (Almost) all binary pixel interleaved formats are
supported.

Output is written to stdout channel (normally your screen) in ascii or binary float format. The
binary output should be redirected to a file or piped to a (GMT) program. Ascii output can
best be used to view a small cutout. Output option (-o) are normal (the file "as is"), the
magnitude, the phase, the real, or the imaginary part. Further options are making a cutout,
multilooking, subsampling, and/or mirroring in the vertical and horizontal plane.

Cpxfiddle does not handle band interleaved complex data, column major order files, nor
complex files. However cpxfiddle might be tricked.

See also cpxfiddle -h and the utility cpx2ps (A.2.4).

SYNOPSIS:

cpxfiddle -w width [-f informat] [-q output] [-o outformat]
[-S x/y] [-M x/y] [-m mirror] [-c file] [-V] [-h[elp]] [-] \inputfile

Dump content of complex binary file to stdout, either: as is, magnitude, phase, real or imaginary part.
Input files can be almost any complex file
though not (yet) band interleaved.
Output can be manipulated by:
multilooking, subsampling, mirroring, scaling, etc.
This program is useful for cropping and displaying, in combination
with e.g., GMT, ImageMagick, or xv.
Output format to stdout can be binary.
Careful! only pipe or redirect this.

A.2.4 cp x2ps

With this utility postscript files from complex data (and binary float data) can be generated,
such as the output of Doris and SLC files.
Various input formats are supported. Options are multilooking, mirroring, plotting the phase, magnitude, real, and imag part, etc. It has become pretty big.

It calls/requires **cpxfiddle** (see A.2.3), and **GMT** subprograms: grd2cpt, grdimage, psscale.

```
cpx2ps v2.1, FMR software, Bert Kampes, (c)1999–2000

PROGRAM: cpx2ps — produce various encapsulated postscript code from complex data files.

SYNOPSIS:

[...]
```

For more info, type: cpx2ps -h | more

### A.2.5 phasefilt.doris

Program to perform phase filtering from the prompt using Doris. Several methods can be used. For more info type: phasefilt.doris -h

```
PROGRAM: phasefilt.doris filter mph file using Doris.

SYNOPSIS:

[...]
```

### A.2.6 flapjack

Program to make integer linear combinations of interferograms.

```
PROGRAM: flapjack pixelwise complex integer multiplication of a float complex file. To be used to make linear combinations, ...

SEE ALSO: cpxmult...

USAGE: flapjack infil1 [factor==2]

EXAMPLE: flapjack cint.raw 3
```

### A.2.7 cpxmult

Program to subtract phase in 2 complex files.

```
PROGRAM: cpxmult subtracts or adds phase of two complex float files

USAGE: cpxmult infil1 infil2 [outfile [add==0]]
```
A.2.8 floatmult

This utility multiplies a (complex) float file by a scalar.

**PROGRAM:** floatmult pixelwise **float** multiplication of a (complex) **float** complex file.

To be used to scale **float** files, or magnitude of complex files.

**SEE ALSO:** cpxmult, flapjack, cpxfiddle...

**USAGE:** floatmult infile1 [factor==2.]

**EXAMPLE:** floatmult cint.raw cint2.raw subtract.raw

A.2.9 wrap

With this utility you can wrap your interferogram to arbitrary interval instead of [-pi, pi). Can be used for example to make fringes correspond to, e.g., 1 cm displacement.

**PROGRAM:** wrap wraps **float** binary file to interval [a,b)

**USAGE:** wrap infile [a b [ofile]]

**EXAMPLE:** wrap interferogram.raw -4pi 4pi interf4.raw

default \outputfile == infile.wrap
default interval [a b] == [-pi pi]

A.3 Completes for tcsh users

Complete commands are used in tcsh shell to complete commands by pressing the TAB key. These completes can be added to the ones you already have. Simply put them in your resource file, likely .cshrc or .tcshrc. Possibly in your configuration there is a file "complete.tcsh" that is sourced from the .cshrc. If you are not using tcsh, you cannot use them (?) (Find out by the commands "who am i" and "finger".)

```
complete doris  c/-""(c h q ver)""/ \ 
n/-h""(<search term>)""/ \ 
n/*/f:+{in,IN,doris}+/

complete cpx2ps  c/-""(w f q e s l L p P T F c z o G C g K S U V Z h m)""/ \ 
n/-f""(ci2 cr4 cr8 r4)""/ \ 
n/-q""(normal mag phase real imag)""/ \ 
n/-T""(<title >)""/ \ 
n/-c""(cool copper gebc ge hot jet green polar rainbow red2green relief topo sealand split wysiwyg)""/ \ 
```
```
complete cpxfiddle
   c/-"(w f q o e s l L p P S M m c V h)"
   n/-c"(<filename> gray jet hot cool bert)"
   n/-f"(cc1 cuc1 ci2 ci4 cr4 cr8)"
   n/-q"(normal mag phase real imag)"
   n/-m"(X XY Y)"
   n/-o"(ascii float sunraster uchar)"
```
Definitions

In this annex a number of definitions as used by Doris are described. In Section B.2 the baseline representations are described, while in Section B.3 the definition of the interferogram is described. Section B.4 describes the definition of the polynomials. And in Section B.5 the use of the pulse repetition frequency and range sampling rate are discussed. Section B.6 describes a system of 3 equations which is frequently used to compute the position on the ellipsoid for a certain line, pixel. The way the orbits are interpolated is described in Section B.7. Finally section B.8 gives some information on the formats of the images.

B.1 Constants

Constants used in the processing can be found in the source files constants.h and refsystems.h. The main parameters in constants.h are:

\[
\begin{array}{ll}
\text{const real8 } & \text{SOL} = 299792458.; \quad \text{// speed of light in m/s} \\
\text{const real8 } & \text{EPS} = 1e^{-13}; \quad \text{// small number} \\
\text{const int32 } & \text{NaN} = -999; \quad \text{// Not a Number} \\
\text{const real8 } & \text{PI} = 4.*\text{atan}(1.);
\end{array}
\]

The main parameters in refsystems.h are (actually only WGS is used for now):

\[
\begin{array}{ll}
\text{const real8 } & \text{WGS84}_A = 6378137.0; \quad \text{// semimajor axis wgs84} \\
\text{const real8 } & \text{WGS84}_B = 6356752.3; \quad \text{// semiminor axis wgs84} \\
\text{const real8 } & \text{GRS80}_A = 6378137.0; \quad \text{// semimajor axis grs80} \\
\text{const real8 } & \text{GRS80}_B = 6356752.3; \quad \text{// semiminor axis grs80} \\
\text{const real8 } & \text{BESSEL}_A = 6377397.155; \quad \text{// semimajor axis bessel} \\
\text{const real8 } & \text{BESSEL}_B = 6356078.963; \quad \text{// semiminor axis bessel} \\
\text{const real8 } & \text{RADIUS} = .5*(\text{WGS84}_A+\text{WGS84}_B); \quad \text{// for sphere pol2car}
\end{array}
\]
The basic configuration of InSAR is shown in figure B.1.

There are different representations for the baseline, see Figure B.2.

Conversions between baseline representations:

<table>
<thead>
<tr>
<th></th>
<th>([B_h, B_v])</th>
<th>([B, \alpha])</th>
<th>([B_{\perp}, B_{\parallel}])</th>
</tr>
</thead>
<tbody>
<tr>
<td>([B_h, B_v])</td>
<td>-</td>
<td>(B_h = B \cos \alpha)</td>
<td>(B_{\parallel} = B_{\perp} \cos \theta + B_h \sin \theta)</td>
</tr>
<tr>
<td></td>
<td>-</td>
<td>(B_v = B \sin \alpha)</td>
<td>(B_{\parallel} = B_{\perp} \sin \theta - B_h \cos \theta)</td>
</tr>
<tr>
<td>([B, \alpha])</td>
<td>(\alpha = \arctan(B_v/B_h))</td>
<td>-</td>
<td>(\alpha = \theta - \arctan(B_{\parallel}/B_{\perp}))</td>
</tr>
<tr>
<td></td>
<td>(B = \sqrt{B_h^2 + B_v^2})</td>
<td>-</td>
<td>(B = \sqrt{B_{\parallel}^2 + B_{\perp}^2})</td>
</tr>
<tr>
<td>([B_{\perp}, B_{\parallel}])</td>
<td>(B_{\parallel} = B_h \sin \theta - B_v \cos \theta)</td>
<td>(B_{\parallel} = B \sin(\theta - \alpha))</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>(B_{\perp} = B_h \cos \theta + B_v \sin \theta)</td>
<td>(B_{\parallel} = B \cos(\theta - \alpha))</td>
<td>-</td>
</tr>
</tbody>
</table>

Table B.1 Conversion between baseline representations (note that the four quadrant arctangent should be used).

The baseline parameters can be computed when the statevectors of the points M, S and P
(master, slave and point on surface) are known. (The distance between the points $x$ and $y$ is denoted by $d(x,y)$; and the sharp angle between two vectors $x$ and $y$ by $\angle(x,y)$.)

\[ B = d(M, S) \]  
\[ B_\parallel = d(M, P) - d(S, P) \]  

Now the perpendicular baseline has to be computed. The definition states that $B_\perp$ is positive if the slave satellite is to the right of the slant range line of the master. Which yields for a mountain an increasing phase from foot to summit? We had some trouble finding a simple expression to find out the correct sign but at the moment we do something like the following.

\[ B_\perp^2 = B^2 - B_\parallel^2 \]  
\[ \vec{r}_1 = \vec{M} - \vec{P} \]  
\[ \vec{r}_2 = \vec{S} - \vec{P} \]  
\[ \gamma_1 = \angle(\vec{P}, \vec{r}_1) \]  
\[ \gamma_2 = \angle(\vec{P}, \vec{r}_2) \]  
\[ \text{sign} = \begin{cases} -1, & \gamma_1 < \gamma_2 \\ 1, & \gamma_1 > \gamma_2 \end{cases} \]  
\[ B_\perp = \text{sign} \sqrt{B_\perp^2} \]  
\[ \theta = \angle(\vec{M}, \vec{r}_1) \]  
\[ \alpha = \theta - \arctan 2(B_\parallel, B_\perp) \]  
\[ B_h = B \cos(\alpha) \]  
\[ B_v = B \sin(\alpha) \]  

\[ \text{Figure B.2 Definition of the baseline parameters. (a) parallel/perpendicular; (b) horizontal/vertical; (c) length/orientation; Position 1 is the reference position. } B_\parallel > 0 \text{ when } R_1 > R_2, \text{ where } R_i \text{ is the corresponding slant range. The angle } \alpha \text{ is defined counter-clockwise from the reference satellite (1), starting from the horizontal at the side of the look direction.} \]
### B.3 Interferogram

The phase for a certain pixel in a single SLC image \( i \) is defined as:

\[
\phi_i \equiv \frac{4\pi}{\lambda} r_i
\]  

(B.14)

The complex interferogram minus reference phase is defined as:

\[
I = M \cdot S^* \cdot R^*
\]  

(B.15)

Where:
- \{ \}^* denotes the complex conjugated;
- \cdot denotes a pointwise multiplication;
- \( I \) is the complex interferogram;
- \( M \) is the complex master image;
- \( S \) is the complex (resampled) slave image;
- \( R \) is the complex (amplitude \( \equiv 1 \)) reference phase;

The phase image (of complex interferogram minus reference phase) is defined as:

\[
\phi = \arctan_2(I_{\text{imag}}, I_{\text{real}})
\]  

(B.16)

Where:
- \( \arctan_2 \) is the four quadrant arc tangent;
- \( \phi \) is the phase image;
- \( I \) is the complex interferogram;

Which is equal to (with an ambiguity of 2\( \pi \))

\[
\phi_I = \phi_M - \phi_S - \phi_R
\]  

(B.17)

The reference phase is defined as (where \( r_1 \) denotes the range from the master satellite to a point on the reference surface)

\[
\phi_R \equiv -\frac{4\pi}{\lambda} (r_1 - r_2) = -\frac{4\pi}{\lambda} B_{\parallel}
\]  

(B.18)

Which is the same as

\[
R = M_r \cdot S_r^*
\]  

(B.19)

Where \( M_r \) denotes the phase of a point situated on the reference surface. (Of course, in this definition the phase of the interferogram equals zero if there actually is no topography (and \( M = M_r \)).) The values of the (real valued) reference phase are stored in a 2d-polynomial of certain degree. The subtraction of the reference phase \( \phi \) is actually computed as:

\[
I = (M \cdot S^*) \cdot (\cos \phi - \sin \phi)
\]  

(B.20)

because the complex conjugated of the reference phase \( \phi \) equals:

\[
(ae^{i\phi})^* = a(\cos \phi + \sin \phi) a \equiv 1 \cos \phi - \sin \phi
\]  

(B.21)

The complex coherence between two images is defined as (see [Touzi et al., 1996]):

\[
\gamma_c = \frac{E\{M \cdot S^*\}}{\sqrt{E\{M \cdot M^*\} \cdot E\{S \cdot S^*\}}}
\]  

(B.22)
Where:
- $E\{.\}$ is the expectation;
- $\ast$ is the complex conjugated;
- $\gamma_c$ is the complex coherence;
- $M$ is the complex master image;
- $S$ is the complex slave image (possibly minus (complex) reference phase: $S = S \cdot R^\ast$).

The coherence is defined by $|\gamma_c|$, and its estimator as:

$$\hat{\gamma} = \left| \frac{1}{N} \sum_{i=0}^{N} M_i S_i^\ast \right| \sqrt{\frac{1}{N} \sum_{i=0}^{N} M_i M_i^\ast \frac{1}{N} \sum_{i=0}^{N} S_i S_i^\ast}$$

(B.23)

The correlation between two images is defined by (see [Bähr and Vögtle, 1991]):

$$\Gamma = \frac{\text{cov}(M, S)}{\sqrt{\text{var}(M)\text{var}(S)}} = \frac{E\{M \cdot S^\ast\} - E\{M\}E\{S^\ast\}}{\sqrt{(E\{M \cdot M^\ast\} - E\{M\}E\{M^\ast\}) \cdot (E\{S \cdot S^\ast\} - E\{S\}E\{S^\ast\})}}$$

(B.24)

Thus the mean is first subtracted in comparison to the coherence. The coherence is equal to the correlation only if $E\{M\} = E\{S\} = 0$.

A problem is that the estimator for the coherence and correlation is biased. For small window sizes its outcome is too high. This probably also causes the problems in the coarse coregistration, where the most likely offset is not selected based on its correlation value but on its consistency.

### B.4 Polynomials

A 1d-polynomial is defined as:

$$f(x) = \sum_{i=0}^{d} \alpha_i x^i$$

(B.25)

A 2d-polynomial is defined as:

$$f(x, y) = \sum_{i=0}^{d} \sum_{j=0}^{i} \alpha_{i-j,j} x^{i-j} y^j$$

(B.26)

Thus the order of the coefficients (line,pixel) is independent of degree $d$:
- $d=0$: $A_{00}$ (1)
- $d=1$: $A_{10}, A_{01}$ (2, 3)
- $d=2$: $A_{20}, A_{11}, A_{02}$ (4, 5, 6)
- $d=3$: $A_{30}, A_{21}, A_{12}, A_{03}$ (7, 8, 9, 10)

Thus the number of coefficients (unknowns in least squares estimation) equals for a 2d-polynomial of degree $d$:

$$\frac{1}{2}((d+1)^2 + d + 1)$$

(B.27)
And the degree of a polynomial with \( N \) coefficients is equal to:

\[
d = \frac{1}{2} \left( \text{int}32(\sqrt{1+8N}) - 1 \right) - 1
\]  
(B.28)

**B.4.1 Computation of coefficients**

Suppose we have 2d data \( f(l,p), l[1,25000] p[1,5000] \) and we want to estimate a 2d polynomial B.26 with these data. The system of equations looks like

\[
\begin{bmatrix}
  f(l_1,p_1) \\
  f(l_2,p_2) \\
  \vdots \\
  f(l_N,p_N)
\end{bmatrix} =
\begin{bmatrix}
  1 & l_1 & l_1^2 & l_1p_1 & \cdots & p_1^d \\
  1 & l_2 & l_2^2 & l_2p_2 & \cdots & p_2^d \\
  \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
  1 & l_N & l_N^2 & l_Np_N & \cdots & p_N^d
\end{bmatrix}
\begin{bmatrix}
  \alpha_{00} \\
  \alpha_{10} \\
  \alpha_{01} \\
  \alpha_{20} \\
  \alpha_{11} \\
  \alpha_{0d}
\end{bmatrix}
\]  
(B.29)

The convention used in the Doris software is that we first normalize the data to avoid numerical instabilities (see source utilities.[hc]). The maximum coordinates are that of the original master (stored in the result file of the master image, typically 25000 lines and 5000 range pixels). The coordinates are rescaled to the interval \([-2,2]\).

\[
l[a,b] \rightarrow l[−2,2] \leftrightarrow l \rightarrow 2\frac{l-a}{25(b-a)} - 2
\]  
(B.30)

(Another way, perhaps a better one (?), would be to make the data zeromean, unit standard deviation) The estimated coefficients thus correspond to the normalized data. For evaluation, the data has to be normalized by the same factors \( a,b \). Normally the information from the master.originalwindow.linelo etc. are used, e.g., for the coregistration and the reference surface polynomial. These numbers can be found in the master result file after the step readfiles at place number of lines original of datafile. A function normalize is called to do the normalization, so it is easy to change the implementation to a different normalization. It has been noticed that for higher order polynomials the normalization factor is very important to obtain a stable estimate.

**B.4.2 Evaluation of polynomials**

Evaluation of the polynomials should be done by normalizing the data as indicated above. Something like:

```c
const real8 minL = master.originalwindow.linelo;
const real8 maxL = master.originalwindow.linehi;
const real8 minL = master.originalwindow.pixlo;
const real8 maxL = master.originalwindow.pixhi;
matrixreal4(N,1) \_axis = linenumbers;
normalize(\_axis);
matrixreal4 f = polyval(\_axis, \_axis, \_axis, coefficients, [degree]);
```

**NOTE:** It is faster to evaluate a polynomial on a grid than point by point.
B.5 (SAR) System parameters

B.5.1 Azimuth

PRF

The actual pulse repetition frequency (PRF) is computed based on the data in the SLC leader file. However, the 'actual' value, as read from the leader file, is used (after private communications with ESA helpdesk). It is defined as:

\[ \text{PRF} = \frac{Nl - 1}{dt_a} \]  

(B.31)

Where:
- \( \text{PRF} \) is the pulse repetition frequency in Hz.
- \( Nl \) is the total number of lines (lastline - firstline).
- \( dt_a \) is the azimuth time of the last line minus the azimuth time of the first line, or the acquisition time of the image.

This equation, and the following, can be easily verified by substitution of the values for the first/last line/pixel.

Line number

The azimuth time of a certain line (number) \( t_{al} \) is computed as:

\[ t_{al} = t_{a1} + \frac{(l - 1) \text{PRF}}{\text{PRF}} \]  

(B.32)

Where:
- \( t_{a1} \) is the azimuth time to line 1 (first line).

And the line number \( l \), given a certain azimuth time \( t_a \), can be computed as:

\[ l = 1 + \text{PRF}(t_a - t_{a1}) \]  

(B.33)

Where:
- \( t_{a1} \) is the azimuth time to line 1 (first line).

Doppler centroid

The Doppler centroid frequency (azimuth) is computed as a second degree polynomial:

\[ f_{DC} = \alpha_0 + \alpha_1 \frac{p}{\text{RSR}} + \alpha_2 \left( \frac{p}{\text{RSR}} \right)^2 \]  

(B.34)

Where:
- \( p \) is the pixel number starting at 0.
- \( \alpha_i \) is read from the leader file.

In the master result file it are the variables (e.g.):
This frequency is used in the azimuth filtering, and in the resampling. It should also be used if the complex SLC data is harmonically oversampled (as is done in the range filtering routine), but we did not implement this yet. But, for \( || f_{DC} || < 150 \) Hz this should not have any effect (assuming PRF-ABW=300Hz).

Since for a signal \( f(t) \) with Fourier transform \( F(\omega) \)

\[
f(t) \xrightarrow{FT} F(\omega)
\]

\( e^{j\omega_0 t} f(t) \xrightarrow{FT} F(\omega - \omega_0) \) \hfill (B.36)

The azimuth spectrum of a SLC image processed on a certain Doppler frequency \( f_{DC} \) (spectrum shifted to this frequency) can be shifted back to zero by multiplication in the space domain by the term

\[ e^{-j2\pi \frac{f_{DC}}{PRF_{line}}} \] \hfill (B.38)

(See also any signals and systems book, or e.g., [Geudtner, 1996].) The spectrum can be shifted back to the original doppler centroid frequency by multiplication by (after e.g., interpolation):

\[ e^{j2\pi \frac{f_{DC}}{PRF_{line}'}} \] \hfill (B.39)

Proper care should be taken to get the correct line number in both situations.

### B.5.2 Range

#### RSR

The range sampling rate (RSR) is defined as:

\[
RSR = 0.001 \frac{Np - 1}{dt_r}
\] \hfill (B.40)

Where:

- \( RSR \) is the range sampling rate in MHz.
- \( Np \) is the number of (range) pixels.
- \( dt_r \) is the zero Doppler two-way time to the last pixel minus the range time to the first pixel in milliseconds.

#### Pixel number

The pixel number \( p \) \([1:Np]\), given a certain one-way range time \( t_a \), can be computed as:

\[
p = 1 + RSR \cdot 2(t_r - t_{r1})
\] \hfill (B.41)
The one-way range time for a given pixel (number) $p$ can be computed as:

$$t_r = t_{r1} + \frac{(p - 1)}{2RSR}$$  \hspace{1cm} (B.42)

Where:
- $RSR$ is in Hz.
- $t_{a1}$ is the range time to pixel 1 (first pixel) in seconds.

The range is of course equal to:

$$r = t_r \times c$$  \hspace{1cm} (B.43)

Where:
- $c$ is the speed of light (constants.h: 299792458. m/s).

### B.6 Doppler, range and ellipsoid equations

The following three equations are used regularly throughout the software to compute the point $P$ that corresponds to a certain line and pixel in the master or slave image (see also [Geudtner, 1996]). Precise orbits are necessary.

1. **Doppler:** The point $P$ at the surface lies perpendicular to the orbit due to zero Doppler processing (otherwise this equation has to be adapted with a slant angle).
2. **Range:** The geometrical distance to $P$ on the surface is equal to the speed of light times the range time.
3. **Ellipsoid:** Force the point to lie on an ellipsoid.

The equations for the point $P$ on the ellipsoid and the satellite $S$ in its orbit are (where $x$ denotes $(x,y,z)$):

$$dx = x - x_s$$  \hspace{1cm} (B.44)

$$E_1 : x_s \cdot dx = 0$$  \hspace{1cm} (B.45)

$$E_2 : dx \cdot dx - (v_{light} t_{range})^2 = 0$$  \hspace{1cm} (B.46)

$$E_3 : \frac{x^2}{a^2} + \frac{y^2}{a^2} + \frac{z^2}{b^2} - 1 = 0$$  \hspace{1cm} (B.47)

To compute the coordinates of a point $P$ on the ellipsoid, corresponding with line $l$ and pixel $p$ in the master image the following has to be done. First the position of the satellite has to be computed (assumed exact) based on the line number and PRF ($l$ to azimuth time to interpolated position), and the velocities for this time (by interpolation). Also the range time corresponding to the pixel number is computed (based on RSR, assumed exact).
Next the set of equations is used to solve for \( P(x, y, z) \). This is done iteratively by linearization, which requires the derivative of the equations to \( x \) and approximate values for the unknowns (the coordinates of the center \((\phi, \lambda)\) given in the SLC leader file, converted to \(xyz\) on a sphere).

\[
dx = x - x_s
\]

\[
\begin{bmatrix}
\frac{\delta E_1}{\delta x} & \frac{\delta E_1}{\delta y} & \frac{\delta E_1}{\delta z} \\
\frac{\delta E_2}{\delta x} & \frac{\delta E_2}{\delta y} & \frac{\delta E_2}{\delta z} \\
\frac{\delta E_3}{\delta x} & \frac{\delta E_3}{\delta y} & \frac{\delta E_3}{\delta z}
\end{bmatrix}
\begin{bmatrix}
dx & dy & dz \\
2 & 2 & 2 & dx & dy & dz \\
\frac{a^2}{x^2} & \frac{a^2}{y^2} & \frac{b^2}{z^2}
\end{bmatrix}
\]

Solving this exactly determined system of 3 equations yields the next solution \( dx_1 \) and the new values for the unknowns become \( dx_1 = dx_0 + dx_1 \) which are used to compute \( dy_1 \) and \( dA_1 \). The solution is updated until convergence (\( \Delta x < 1e-6 \) meters).

\[
dy_i = dA_i \cdot dx_i
\]

Where:
\( dy \) contains the observations (set of equations)
\( dx \) contains the unknowns (coordinates of \( P \)). \( dA \) contains the partials (evaluated for previous solution).

To solve for the azimuth time if the coordinates of a point on the ground is known, only the Doppler B.45 equation needs to be used. The derivative with respect to azimuth time of this equation equals

\[
\frac{\delta E_1}{\delta t_a} = \dot{x}_s \cdot dx - \ddot{x} \dot{x}
\]

The solution is equal to (use approximate solution \( t_{a0} \) to evaluate these expressions).

\[
t_{a1} = \frac{-E_1}{\delta E_1/\delta t_a}
\]

and

\[
t_{a1} = t_{a0} + t_{a1}
\]

The solution is updated until convergence (\( \Delta t < 1e-10 \) seconds).

The range time is then computed as in equation B.46

\[
t_r = \sqrt{\frac{(x - x_s)^2}{c}}
\]

### B.7 Orbit interpolation

We assume the precise orbits are given some time before the first and after the last azimuth line. Normally we use getorb to obtain satellite ephemerides with a time interval of 1 second (approximately 21 datapoints for a frame of typical 15 seconds).
Natural cubic splines are then used to interpolate the orbit. Because these splines do not behave very well at the edges we use some points before/after the first/last line. Note that the x, y, and z coordinate are interpolated independently.

The Delft precise orbits and the getorb package are used to obtain the points. Note that getorb also interpolates based on 30 second ephemerides.

We would like to test if setting the data interval to e.g. 30 second gives better results. A test can be easily performed for the computation and modeling of the reference phase. Assume that this phase can be accurately modeled by a 2d polynomial of degree 5. Now first let the precise orbit be given with a data interval of 1 second. Use step REFPHA to model the reference phase based on 501 points distributed over the scene. Next, let the precise orbit be given with a data interval of 30 seconds and again model the reference phase. In the log file some statistics on the error of the model w.r.t. the computed reference phase is given, which can be used to find out which orbit gives a better model. In both cases use at least 6 points before and 6 points after the last point in the frame.

The interpolation is done as follows, compare with numerical recipes in c (splint routine). First the piecewise polynomial coefficients are computed by solving a tridiagonal system and stored. For interpolation, the correct coefficients (interval) are read and the polynomial is evaluated.

Because we know that in our situation (with getorb) we always have ephemerides with a constant time interval we could speed up the computations. Also the fact that this interval equals 1 can be easily exploited. However, we decided not to exploit these features because we like to stay independent from a particular orbit format. (and these computations can be done fast anyhow.)

The velocity can be interpolated by the derivative of the piecewise polynomials: [see source code or numerical recipes].

The acceleration can be interpolated by the second derivative of the piecewise polynomials: [see source code or numerical recipes].

If less points are known (than the typically 21 of getorb) (one wants to use the SLC datapoints for a quick look analysis for example) then this kind of interpolation probably does not work very well. In future we will include an option to interpolate by a low degree polynomial which is estimated (least squares) from the datapoints. Getting the derivatives at any point is straightforward in this case.

As a satellite moves very smoothly, a polynomial of a lower degree might even be nearer to the 'true' orbit than a piecewise polynomial. In future we want to model the baseline (Bh, Bv) as a function of azimuth time by a first order polynomial. This probably is more efficient than computing the positions of the sensors each time the baseline is required. We do not know what the best way is to do this.

### B.8 Format of the products

Start at (azimuth) line 1, (range) pixel 1 (near range). Data is written line by line (major row order). We give the binary data a .raw extension.
The complex interferogram is written pixel interleaved (see B.2). Each complex pixel is written as 4 byte real, 4B imaginary part.

<table>
<thead>
<tr>
<th>1st line</th>
<th>2nd pixel</th>
<th>...</th>
<th>pixel P</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st pixel</td>
<td>Real,Imag</td>
<td>Real,Imag</td>
<td>...</td>
</tr>
<tr>
<td>2nd line</td>
<td>Real,Imag</td>
<td>Real,Imag</td>
<td>...</td>
</tr>
<tr>
<td>3rd line</td>
<td>Real,Imag</td>
<td>Real,Imag</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>line L</td>
<td>Real,Imag</td>
<td>Real,Imag</td>
<td>...</td>
</tr>
</tbody>
</table>

Table B.2 The way complex files (SLC data, resampled slave, complex interferogram) are stored on disk. Also referred to as mph format (magnitude phase). This is a major row order stored, pixel interleaved file with 2 (float 4B) canals (real,imag).

After unwrapping of the phase the result can no longer be stored as a complex value, because a complex number only can distinguish between phase values in the principal interval $\pm \pi$. Therefore a new format is used/will be used. Either the unwrapped phase is simply stored in a 4B float file, similar to table B.2 without the imaginary part (as are other files, like the phi, lambda, and height matrices after geocoding), or a hgt file is generated (see table B.3).

<table>
<thead>
<tr>
<th>1st line</th>
<th>2nd pixel</th>
<th>...</th>
<th>1st pixel</th>
<th>2nd pixel</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st pixel</td>
<td>Amplitude</td>
<td>Amplitude</td>
<td>...</td>
<td>Phase</td>
<td>Phase</td>
</tr>
<tr>
<td>2nd line</td>
<td>Amplitude</td>
<td>Amplitude</td>
<td>...</td>
<td>Phase</td>
<td>Phase</td>
</tr>
<tr>
<td>3rd line</td>
<td>Amplitude</td>
<td>Amplitude</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>line L</td>
<td>Amplitude</td>
<td>Amplitude</td>
<td>...</td>
<td>Phase</td>
<td>Phase</td>
</tr>
</tbody>
</table>

Table B.3 Format of a hgt (height) file. (unwrapped complex interferogram and others) Actually is a major row order, band interleaved data, with 2 float (4B) canals (amplitude phase).

Particularly after unwrapping the conventions we use for this file are as follows. The amplitude (equal to that of complex interferogram) and the unwrapped phase is stored for each pixel. If there is no unwrapped phase, the wrapped phase is stored, and the amplitude is set to 0 for that pixel. (I would prefer setting the phase to 0, and keeping the amplitude, but we selected this format to keep in line with other software.) The amplitude is stored, while it does not change after unwrapping, to keep all information in one file.

Computations are done in general in the original master system. (no matter if cut-out or multilooked.)

Time system (orbit) is in seconds of day. Ephemerides orbit system is more or less WGS84.

Matrix class:
A matrix is starts at 0,0 etc.

Offset:
Offset for a certain point is defined as: coordinate in slave system = coordinate in master system + offsets.
The template matrix class (called matrixbk) that is provided with the Doris software can be used for other applications as well. Please refer to the file matrixbk_test.c for an example how to use this class in your own programs. (I do not claim it is the best/fastest implementation ever, i just find it very useful, and the routines are checked and working fine.) The Makefile shows how to compile it. Also see Annex ??.

The data has to be linear in memory for the VECLIB library, thus this has been done and it used in some other functions to speed them up (see the constructor how this is done).

If the VECLIB library is not used, (slower) internal functions for FFT and matrix multiplication are used. If you like to use your own FFT, only the function four1d has to be changed since the 2d function call this one sequentially.

If the LAPACK library is not used, (slower) internal functions for cholesky are used.

The matrix class is used as a container class for (part of) the images. It has not been defined as a class ’radarimage’ because in that case it would have been difficult to perform operations on the images if they didn’t fit in the memory as a whole. However, it might be a good idea to define functions such as phasefilter for the matrix class. This would result in calls like:

```c
matrix < complex<real4> > BUFFERMASTER; // Container
for (i=0; i<NUMBUFFERS; ++i)
{
    // Read in phase image
    BUFFERMASTER. readfromfile ( filename , winodwsiz e , formatflag );
    // filter this buffer
    BUFFERMASTER. phasefilter ( parameters );
    // Write to \outputfile
    BUFFERMASTER. wri teto f ile ( filename2 , formatflag );
}
```

which seems very readable and maintainable (only the member function ”phasefilter” of the matrix class has to be changed if something has to be added).

If you do not have access to LAPACK, but you have a different library, we advice you to use that one. The Cholesky factorization as implemented internally has not been optimized in any way. The same holds for the VECLIB library, particularly the FFT routines. We have included the code for a Cooley-Turkey algorithm, but it handles the data quite slowly, compared to an
C.1 Matrix class functions

The functions in the matrix class are obtained by:

```
"/bin/ctags --c-types=f -x matrixbk.* | cut -c1-12,50-600
```

```cpp
allocate void matrix<Type>::allocate(uint numlines, uint numpixels) // allocate
checkindex void matrix<Type>::checkindex(uint line, uint pixel) const
clean void matrix<Type>::clean() // sets 2 zero
conj void matrix<Type>::conj()
conjugate matrix<Type> conj(const matrix<Type> &A)
calculate matrix<Real4> correlate(const matrix<Type> &A, matrix<Type> &M) Mats
dotxmatrix matrix<Type> diagxmat(const matrix<Type> &diag, const matrix<Type>&)
dotxdiv matrix<Type> dotxdiv(const matrix<Type> &A, const matrix<Type>&)
dotxmult matrix<Type> dotxmult(const matrix<Type> &A, const matrix<Type>&)
dumpasc void dumpasc(const char *file, const matrix<Type> &A)
fftshift friend void fftshift(matrix<Type>&A)
flipr void matrix<Type>::flipr()
flipud void matrix<Type>::flipud()
getcolumn matrix<Type> matrix<Type>::getcolumn(uint pixel) const
getdata matrix<Type> matrix<Type>::getdata(window win) const
getrow matrix<Type> matrix<Type>::getrow(uint line) const
ifftshift friend void ifftshift(matrix<Type> &A)
initialize void matrix<Type>::initialize(uint numlines, uint numpixels)
isvector bool matrix<Type>::isvector() const
lines uint matrix<Type>::lines() const // return number of lines
matxmat matrix<Type> matxmat(const matrix<Type> &A, const matrix<Type>&) const
matrix matrix<Type>::matrix() // constructor (0 arg)
matrix matrix<Type>::matrix(uint lines, uint pixels)
matrix matrix<Type>::matrix(const matrix<Type>&A)
matrix matrix<Type>::matrix(window win, const matrix<Type>&A)
max matrix<Type> max(const matrix<Type>&A)
mean real18 mean(const matrix<Type>&A)
min Type min(const matrix<Type>&A)
min_type min(const matrix<Type>&A, uint& line, uint& pixel)
multilook matrix<Type> multilook(const matrix<Type>&A, uint factorL, uint factorR)
mypow void matrix<Type>::mypow(Type s)
myswap friend void myswap(matrix<Type>&A, matrix<Type>&B)
operator != bool matrix<Type>::operator != (Type scalar) const
operator != bool matrix<Type>::operator != (const matrix<Type>&A) const
operator * matrix<Type> operator *(const matrix<Type>&A, const matrix<Type>&)
operator * matrix<Type> operator *(const matrix<Type>&A, Type scalar)
operator /= matrix<Type>& operator /= (Type scalar, const matrix<Type>&A) const
operator /= matrix<Type>& operator /= (const matrix<Type>&A) const
operator /= matrix<Type>& operator /= (const matrix<Type>&A, Type scalar)
operator += matrix<Type>& matrix<Type>::operator += (Type scalar)
operator += matrix<Type>& matrix<Type>::operator += (const matrix<Type>&A)
operator += matrix<Type>& matrix<Type>::operator += (const matrix<Type>&A, Type scalar)
operator += matrix<Type>& matrix<Type>::operator += (const matrix<Type>&A)
operator += matrix<Type>& matrix<Type>::operator += (Type scalar)
```
```cpp
operator -  matrix<Type>& matrix<Type>&::operator - (const matrix<Type>& A, Type scalar)
operator -= matrix<Type>& matrix<Type>::operator -= (const matrix<Type>& A)
operator - matrix<Type>& matrix<Type>::operator - (const matrix<Type>& A)
operator /  matrix<Type>& matrix<Type>::operator / (const matrix<Type>& A, Type scalar)
operator /  matrix<Type>& matrix<Type>::operator / (const matrix<Type>& A, Type scalar)
operator /= matrix<Type>& matrix<Type>::operator /= (const matrix<Type>& A)
operator /  matrix<Type>& matrix<Type>::operator / (const matrix<Type>& A, Type scalar)
operator /= matrix<Type>& matrix<Type>::operator /= (const matrix<Type>& A)
operator << friend ostream& operator << (ostream& file, const matrix<Type>& A)
operator =  matrix<Type>& matrix<Type>::operator = (const matrix<Type>& A)
operator instanceof friend ostream& operator instanceof (ostream& file, matrix<Type>& A)
operator []  Type* matrix<Type>::operator [] (uint line) const
operator()  Type& matrix<Type>::operator () (uint line, uint pixel) const
operator()  matrix<Type>& matrix<Type>::operator () (window win) const
operator()  matrix<Type>& matrix<Type>::operator () (uint l0, uint lN, uint p0, uint pN)
pixels   uint matrix<Type>::pixels() const // return number of pixels
readfile   friend void readfile(matrix<Type>& fileResult, const char* file,
resize    void matrix<Type>::resize(uint l1, uint p1)
setcolumn void matrix<Type>::setcolumn(uint pixel, const matrix<Type>& A)
setcolumn void matrix<Type>::setcolumn(uint pixel, Type scalar)
setdata    void matrix<Type>::setdata(Type w) // sets 2 w
setdata    void matrix<Type>::setdata(uint l1, uint p1, const matrix<Type>& A)
setdata    void matrix<Type>::setdata(uint l1, uint p1, const matrix<Type>& A)
setdata    void matrix<Type>::setdata(const matrix<Type>& A, window winA)
setrow     void matrix<Type>::setrow(uint line, const matrix<Type>& A)
setrow     void matrix<Type>::setrow(uint line, Type scalar)
showdata   void matrix<Type>::showdata() const // show all data in matrix
size       uint matrix<Type>::size() const // return nsize
sqrt       friend matrix<Type> sqrt (const matrix<Type>& A)
sum        matrix<Type>& matrix<Type>::sum (const matrix<Type>& A, int32 dim)
writefile  friend void writefile ()
wshift     friend void wshift (matrix<Type>& A, int32 n)
~matrix    matrix<Type>::~matrix()
```
In this annex a description is given how to add a module to the Doris software.

First get general idea of the structure of the software. (source2html?).

It is preferred to stay in same format as us.

General:

1. Read input cards and parameters for your module.
2. Add your module to the big selecting switch in main.
3. Implement your module, let in result file the output section end with same string as the other modules (END..NORMAL).
4. Documentation (author date description, for users and code developers)
5. Email the description and the total source to the owner of the mailinglist doris_users@tudelft.nl. And if approved, we will include your functions in the next version of Doris.

D.1 Formats

The example source code explains which rules for commenting I generally follow.

```cpp
/* ***************************************************************
 * ts16 *
 * truncated sinc 16 points *
 * input: *
 * - x-axis *
 * output: *
 * - y=f(x); function evaluated at x *
 * *
 * Bert Kampes, 16-Mar-1999 *
 ***************************************************************/
matrix<real4> ts16(
    const matrix<real4> &x)
```
```c
{  
    #ifdef __DEBUG  
    DEBUG("ts16.");  
    if (x.pixels() != 1)  
        ERROR("ts16: standing vectors only.");  
    #endif  
  
    matrix<real4> y(x.lines(),1);  
    for (register int32 i=0;i<y.lines();i++)  
        y(i,0) = sinc(x(i,0)) * rect(x(i,0)/16.);  
  
    return y;  
} // END ts16
```

- start routine a block with date/author/description/input/output;
- end routine met // END routinename;
- no block comments inside the function, only things like:
  
  // ====== Comment on block ======
  // _____ Comment on something smaller _____

Indenting is done with 2 spaces, with the curly braces as shown below.

```c
if (expression)
{
    action1;
    action2;
}
```

You can display information (depending on the value of the variable displevel) with the functions:

```c
DEBUG(char[ONE27]);
INFO(char[ONE27]);
PROGRESS(char[ONE27]);
WARNING(char[ONE27]);
ERROR(char[ONE27]);
```

## D.2 Adding a Step

Adding a new step is not intended to be necessary. The only thing that needs to be added are modules (methods) in pre-defined steps. However we will explain what you will have to change if you want to add a new step.

In file readinput.h:

1. you will have to add a const for the new step which is later stored in the process array;
2. also a struct has to be made to store the variables of this new step. (method selector, output file name, window sizes, etc.);
3. the prototype of the function readinput should be augmented with this new struct.

In the file readinput.c:

1. you will have to add a const for the new step which is later stored in the process array;
1. function readinput augment with new struct;
2. (only)process card, define new keyword for this step;
3. add reading of parameters into defined inputstruct by new keywords.

In file ioroutines.c: (only minor adding)

1. routine: doinitwrite: add new step
2. routine: initwrite: process control
3. routine: updateprocesscontrol: check for string
4. routine: checkprocesscontrol: check for string
5. routine: fillcheckprocesscontrol: check for string
6. routine: fillprocessed: check for string

In file processor.c:

1. add definition of new struct,
2. readinput augment with new struct,
3. add in big switch what to do if new step is requested.

General:

- document what you did, new keywords and arguments, new process control flag.
- how does the result file has to end? " * END_filtphase: _NORMAL"
- what strings in the result file are used later in the program?
- email to owner-doris_users@tudelft.nl.
FAQ (Frequently Asked Questions)

Here we stored some questions asked to the Doris email list with their answers. These are not formatted, the emails were just copied and pasted into this document. The list of emails is NOT in a particular order, roughly on time of question asked.

E.1 Installation Unix platforms

From Bert.Kampes@dlr.de Mon Mar 14 08:43:46 2005
Date: Wed, 7 Jan 2004 08:46:44 +0100 (CET)
From: Bert Kampes <Bert.Kampes@dlr.de>
To: songwei <qd_sw@sina.com>
Cc: Doris software list <doris_users@geo.tudelft.nl>,
doris_users@yahoogroups.com
Subject: Re: Installation Doris on PC Window?

+++ At 10:13, December 28th, songwei wrote:
<s: Dear Sir,
<s: I am a graduate student from China.I wonder whether
<s: the Doris v3.8 can be installed on PC Windows.And could you please
<s: tell me how to compile and install it on Window XP?

Dear Songwei,

Doris is unix oriented, therefore installation under windows is best done under a shell like Cygwin. In order to do that, simply install cygwin (including the development tools and tcsh) and then follow the instructions for compilation. I do not encourage compilation directly under windows, running it in a dos box or some kind of GUI, since i will not support this, but I have heard several people have tried/succeeded in doing so. I see no advantage whatsoever of this approach. Moreover, installing linux on a dedicated PC (for doris) may be your best option. Best regards,

Bert Kampes
Dear Severino,

The warning

WARNING.print("NOT USED ANYMORE. since v3.8");

you can ignore, it relates to a define (see code).
I think you should not get this message
when you compile v3.10 with the correct makefile.
Did you create a Makefile with the configure script?

The segmentation fault hopefully has to do with that
some array has zero length that Doris tries to access.
I guess the threshold you use for the coherence is too high,
i.e., you select zero windows for fitting a polynomial.

It seems you don’t use the plot scripts. I would recommend using them.
i.e., please use:
plotoffsets master_slave.res 1 25000 1 5000 0.4

where the second argument is the result file for the interferogram.
this will show you a plot with the selected windows. I normally use
plotoffsets master_slave.res 1 25000 1 5000 0.2
plotoffsets master_slave.res 1 25000 1 5000 0.3
plotoffsets master_slave.res 1 25000 1 5000 0.4
plotoffsets master_slave.res 1 25000 1 5000 0.5

To see the effect of setting another threshold on the number
and distribution of selected windows.

Better than copying the example input, use the run script.

Good luck and best regards,

Bert Kampes

+++ At 15:48, April 18th, Severino Fernandez wrote:

<SF: I am new to DORIS and I am trying to get used to it with two ERS images
<SF: over Madrid.
<SF: I have taken the scripts from the test data (doris[1234].in), and
<SF: changed the input file names to match those of the images I have,
<SF: leaving everything else unchanged. After downloading the applicable
<SF: orbital data, both step 1 and 2 seem to work correctly, but I get the
<SF: following fatal crash in step 3:
<SF:
WARNING : NOT USED ANYMORE. since v3.8
WARNING : It is highly recommended to use CPM_PLOT to review the estimated model.
PROGRESS: Interpretation inputoptionsfile finished.
total cpu: 0 min 0.12 sec
PROGRESS: Finished initialization
PROGRESS: Orbit: interpolation coefficients computed.
PROGRESS: Orbit: interpolation coefficients computed.
PROGRESS: Start COREGPM.
Segmentation fault (core dumped)

May it be possible that I have to set some parameters to other values?
Did someone had a similar problem before?
Thank you for your advice.
Severino Fernandez

From Bert.Kampes@dlr.de Mon Mar 14 08:37:26 2005
Date: Tue, 29 Jun 2004 11:14:41 +0200 (CEST)
From: Bert Kampes <Bert.Kampes@dlr.de>
To: o.petrik <o.petrik@harsp.rsc.fomi.hu>
Cc: Doris software list <doris_users@geo.tudelft.nl>,
doris_users@yahoogroups.com
Subject: Re: installation Doris without VECLIB

Dear Otto,

Veclib/Sunperlib is an optimized library for specific architectures such as HP/SUN. These are commercial libraries as far as I know.

Fortunately, Doris does not require this library. During execution of the ./configure script, simply confirm "n" when it asks for it. This makes sure the define __USE_VECLIB_LIBRARY__ is not set.

By the way, VECLIB cannot be used together with FFTW lib.

I hope this helps you further. Make sure to use the ./configure script to create a Makefile for Doris. Installation then should be automated. Under cygwin, I linked the shell /bin/tcsh to /bin/csh. Make sure you have installed tcsh.

Best regards,

Bert Kampes

+++ At 10:20, June 29th, o.petrik wrote:
<<o: Dear Bert,
<<o:
Finally we have got all scenes from ESA, so I start the fight with Cygwin&Doris... :)
I kidnapped our sysadmin :), he fixed my Cygwin...

my problem now is "veclib"

http://tc1.chemie.uni-bielefeld.de/doc/veclib link on your homepage is not work...

Google do not find any veclib for Cygwin... the nearest thing I have found is hp_mlib_linux_1.2...

Could you help me to find any veclib for Cygwin?

Regards,
Ott

From Bert.Kampes@dlr.de Mon Mar 14 08:40:48 2005
Date: Fri, 16 Apr 2004 08:43:13 +0200 (CEST)
From: Bert Kampes <Bert.Kampes@dlr.de>
To: Vladimir Serna <vserna@osso.univalle.edu.co>
Cc: Doris software list <doris_users@geo.tudelft.nl>, doris_users@yahoogroups.com
Subject: Re: Doris Input

+++ At 16:43, April 15th, Vladimir Serna wrote:

Dear Miss Berth:
I am a student in topographic at the valley university in Colombia. Right now I am working on my final project, using the Doris programm.
Now I have some question about it. I hope you could help me on that. I would be very thankfull.
I have two radar images in this format: *.ci2, *.cr4, *.byt (multilook 1x1 y 5x1) and *.hdr. This format is different to the image extensions in the doris user manual. My question now is: Is it possible to work with this format, or do I have to change it to a format that only works in this programm? If so, how does format exportation work? Maybe I need a special tool or software for that?
In the attachmet of this e-mail you find a textfile with all the information about my two radar images.
Thank you in advance and many greetings from Colombia,
Vladimir Serna Marin.
Dear Vladimir,

Under unix a file is a file, extention is arbitrary. In Doris the conventions are the store files without a header, a raw binary data stream. Externally the width, height, type, etc. (byte-order) are stored. (in the result files). The name of the files is specified with input cards. I normally use the run script to process data. With that script, template input files are generated. In those files the convention is to use extensions describing the type of the file, ie. cr4 for complex real 4 byte files, etc.

In order for Doris to work with your data you thus need to have (convert) the files as 0 header raw binary streams, and you will have to know the dimensions. See the manual.

If you simply have ESA CEOS files the step CROP creates these files, so you don’t have to do this yourself.

In your case I suspect that you don’t have SLC data. The attached files states it is raw data. In this case Doris cannot handle it. You need a SAR processor to focus the data first. See the webpage, manual, etc.

Greetings,

Bert Kampes

PS. I forwarded your question to the list so it could be helpful to more people. (in future)
Maybe an admin at your department or another doris user can
guide you through the installation procedure. If you know anyone
with Linux/Unix experience there, get that person to help you,
and to explain in detail what it all means.

If there is a Doris user that has a log of exactly what he/she
did in order to install, please share this on a website.

Please also read the files "README" and "INSTALL" in the distribution.

I can only repeat the information contained in those files, the manual,
and the web pages.

Install Doris:
  -fftw is not required, it is recommended.
  -follow steps as described in INSTALL file (i.e. do tools too).

After "make install" the executables are copied to some directory.
Let’s say this directory is "/usr/local/bin".

In order for the shell to find an executable, let’s say "doris", it needs
to look for it. All shells have a "path" in which they look for a typed
command (e.g., if you type "doris", it looks for a file with that name in
all directories specified in the "path").

After installation of doris and the SARtools and ENVISATtools, it is thus
most convenient when the installations directory ("/usr/local/bin") is
in your path. You can check this by doing the following:

after installation of doris,
log out, or perform a shut down of your system.
log in again.
now type: doris -q
run -h

if you get a quote from doris and help for the run script,
you are on the right track.

The other programs I did not write, and I cannot help you with installing them.

Doris uses getorb program for precise orbits for ERS and ENVISAT.

Thus, if you want this capability, you need to install this program.
Thus, download it, for example via the link provided at the doris website.
Then install it. Make sure it is in your "path".

Similar to this, Doris also used GMT. Therefore, also install this
if you want it. Make sure it is in your "path".

Similar to this, Doris also used snaphu. Therefore, also install this
if you want it. Make sure it is in your "path".

For all programs that Doris uses, there should be a link on the website.
Hopefully this is useful to you, and in future I can simply refer to this email if people have questions on installation.

Regards,

Bert Kampes

PS. By the way, "/usr/local/bin" is a local directory to your computer which normally already is in your "path". Then you do not have to do anything extra. But this directory cannot be found if you log through to another machine for processing. Therefore, I do not do install there, but I install in my home directory, in my case "/users/kampes/local/bin". Since this is not a default directory, i need to specify that the shell has to look in that directory. Since I am using a tcsh shell, the appropriate way to do that is to add that directory to the path variable in the tcsh resource file that is read when you start a new shell (i.e., the file "$HOME/.tcshrc"). this is done by a command like set path = ( /users/kampes/local/bin $path ) but if you use another shell, this may be different for you.

From Bert.Kampes@dlr.de Mon Mar 14 08:33:34 2005
Date: Mon, 27 Sep 2004 16:32:01 +0200 (CEST)
From: Bert Kampes <Bert.Kampes@dlr.de>
To: Mahdi Motagh <motagh@gfz-potsdam.de>
Cc: Doris software list <doris_users@tudelft.nl>, doris_users@yahoogroups.com
Subject: Re: BPERP value in the snaphu.conf

+++ At 17:35, September 6th, Mahdi Motagh wrote:

"MM: Dear Bert Kampes
"MM: I think there is a small bug in Doris when we use it
"MM: to transfer the general parameters, required for phase unwrapping, to the snaphu.conf file.
"MM: The problem is that although Doris transfers the value of BPERP to the
"MM: snaphu.conf, it always puts the '#' sign in front of it.
"MM: I am wondering whether you have had any specific reason for commenting out such a value before
"MM: Yours sincerely
"MM: Mahdi Motagh

Dear Mahdi Motagh,

I apologize for not answering before. I just got access to this email address, since I am not located in Delft. Please use <doris_users@tudelft.nl> for questions on Doris.

Your reported behavior is not an error in Doris, I think. See the snaphu manual, where it is specified that the baseline value and orientation is used, before checking the perpendicular baseline value, if I remember correctly. It is in principle equivalent, and I
assume that these values ("B and alpha") are correctly parsed and used. If you have any experiences using/tuning snaphu regarding changes that need to be made in Doris, that would be very valuable to us.

Best regards,

Bert Kampes

---

From Bert.Kampes@dlr.de Mon Mar 14 08:27:50 2005
Date: Fri, 24 Dec 2004 18:56:38 +0100 (CET)
From: Bert Kampes <Bert.Kampes@dlr.de>
To: Manoj Karkee <manoj_bhojpure@yahoo.com>
Cc: Doris software list <doris_users@tudelft.nl>, doris_users@yahoogroups.com
Subject: Re: [doris_users] "bc" command in 'plotoffsets' script

Dear Manoj,

bc is a standard unix command such as ls, dc, tr, etc. Most likely it is located in /bin. Try "man bc". Using which bc gives me at Cygwin:
/usr/bin/bc

and typing at the command prompt:
echo " 5+6" | bc -l
returns 11

Hope this helps you resolve the problem.

Bert Kampes

---

+++ At 11:09, December 21st, Manoj Karkee wrote:

<MK:>
<MK: Dear all,
<MK: I have encountered 'bc' command in "plotoffsets" script but could not locate it in the doris package and other reference tools including "InSAR tools", "GMT" and "ghostview".
<MK:>
<MK: What may be my problem?
<MK:>
<MK: Thanking in advance.
<MK: Manoj Karkee
<MK: AIT
<MK:>

From Bert.Kampes@dlr.de Mon Mar 14 08:29:54 2005
Date: Thu, 18 Nov 2004 17:23:36 +0100 (CET)
From: Bert Kampes <Bert.Kampes@dlr.de>
To: Petar Marinkovic <p.s.marinkovic@lr.tudelft.nl>
Cc: arciniegas06157@itc.nl, Doris software list <doris_users@tudelft.nl>,
doris_users@yahoogroups.com
Subject: Re: newbie question

FYI - -

Below is a bounced post that may be of general interest so I forward it.

<PM: From: owner-doris_users@listserv.tudelft.nl
<PM: To: owner-doris_users@listserv.tudelft.nl
<PM: Subject: BOUNCE doris_users@listserv.tudelft.nl: Admin
<PM: request: /^subject:\s*help\b/i
<PM: Date: Thu, 18 Nov 2004 15:42:36 +0100 (MET)

I think if the word "help" is in the subject the list things
you want help on usage. I am not sure about that. Further see the
question and answer by Petar below.

Bert Kampes

+++ At 16:01, November 18th, Petar Marinkovic wrote:

<PM: Hi Gustavo,
<PM: My name is Petar, and I’m administrator of the doris_user email list.
<PM: It seems like your post has been bounced from the list, under a virus
<PM: suspicion - see below. Please check your email client (and/or system)
<PM: for viruses. However this might be a false alarm, since the subject of
<PM: your post had some strange characters.
<PM: To answer your question quickly:
<PM: Have you looked into the Doris user manual? The manual and the Bam quake
<PM: processing overview might be a good starting point.
<PM: There’s also an excellent paper by Bert at Fringe03 section of the Doris
<PM: web site : enterprise.lr.tudelft.nl/doris/FRINGE_2003
<PM: You might consider registering to the list, if you haven’t yet done so.
<PM: Hope this helps.
<PM: Cheers,
<PM: Petar
<PM:
E.2  Debugging, running Doris, what to do when Doris crashes

From Bert.Kampes@dlr.de Mon Mar 14 08:44:17 2005
Date: Fri, 9 Jan 2004 13:56:24 +0100 (CET)
From: Bert Kampes <Bert.Kampes@dlr.de>
To: kianicka@email.cz
Cc: Doris software list <doris_users@geo.tudelft.nl>,
doris_users@yahoogroups.com
Subject: Re: Strange numbers in fine coregistration

Dear Jan,

I am still unclear on your question. To get familiar with the processing flow,
I would suggest you download and process the test data set and experiment
with it. If you have a tandem pair, process that first.

For your current problem, the information you provide is too limited.
I suspect that you fitted some wild polynomial through the fine offset vectors,
and then indeed RESAMPLE will crash (the buffers are not big enough, variable FORSURE).
Increasing FORSURE may prevent the crash, but is not likely to give a coherent
interferogram. Again, a visual check of (intermediate) data products may
help you. After CPM step, the GMT plots indicate the fitting process. There is
no point in continuing with RESAMPLE if these plots do not look OK.

<: I used now as input into fine coregistration parameters from orbit,
=: and that values in interferogram.out are quite different now.

Instead of randomly trying different input initial offsets, figuring out

the correct coarse offset is advisable. What offset did you obtain by hand?
What does the logfile say on coarse correlation?

Also check the doppler difference, perpendicular and temporal baseline. I fear the

combination of SLCs you try to interfere may be awkward.
How large is your area? What kind of area are you processing? Please provide such

parameters in your next mail.

Best regards,

Bert Kampes

+++ At 13:33, January 9th, kianicka@email.cz wrote:

<:
<: Dear Bert Kampes,
<: Thanks for advice, I used now as input into fine coregistration parameters from orbit, and that value
<: coarsegistration.cc, but I supposed that my results of coregistration is bad. (FORSURE is deafault
<: comment ___COMPUTE COREGISTRATION PARAMETERS___//
<: c c
<: CPM_THRESHOLD 0.3
<: CPM_DEGREE 1
From Bert.Kampes@dlr.de Mon Mar 14 08:35:35 2005
Date: Mon, 16 Aug 2004 09:37:00 +0200 (CEST)
From: Bert Kampes <Bert.Kampes@dlr.de>
To: maryam rahnemon <rahnemon_far@yahoo.com>
Cc: doris_users@yahoogroups.com
Subject: Re: [doris_users] problem of run -s2 for envisat data

Dear Maryam,

The first warnings about environment variable are for your convenience. They are generated by the run script, and have no bearing on Doris. Default values are likely used, but I would set these variables in your startup script (for example, the same place where you have checked and set the "path" variable.) Please read what is reported.

[SNIP]
<mr: PROGRESS: system call may take some time...
<mr: PROGRESS: Finished system call to envisat_dump_data
<mr: PROGRESS: Finished S_CROP.
<mr: total cpu: 0 min 0.01 sec
<mr: WARNING : File: Outdata/6687.raw does not seem to exist (may not be a problem).
<mr:

[SNAP]

This I cannot really explain with the information provided. It seems the data is not correctly cropped, based on the small amount of total CPU time, and that Doris says that some file does not exist. I would suspect that the program "envisat_dump_data" is not in your path, but you were able to process the master scene.

* Does it work if you start over, and as a test use the slave image as master, and the master as slave?

* Please go over the output of Doris. Use "SCREEN debug" setting. Also look at the message on the screen. The run script puts the output in a file, in directory "Outinfo", file called stdout.input.s_initial I think.
With the command run -v2 you should see it with the PAGER specified (for example, "setenv PAGER nedit"); then run -v2 will use nedit to view this file.

* Try to localize the exact point what does not work. In your example, run first only "PROCESS S_READFILES" by commenting out any other PROCESS card in the input file, or by using the ONLYPROCESS card. If this exits OK, then continue with the next processing step, e.g., "PROCESS s_crop".

* Go over your input again, check the meaning of each card. Check whether you have specified a file name that really exists. Check the file size of the data file you specified. Is it say 500 MB?

* Anticipate what should happen with the input you use. Check the output file, and see if you understand and see indeed what you expected.

* Note that you have to edit the input files, changing the provided example setup to meaningful values for your processing.

* Doris uses system calls to do this ENVISAT processing for cropping/reading information. The command that is given from within Doris to do the actual work, should also work from the command line. Look in the output file for this command. For example, here, typing
  ```
  grep envisat_dump_data Outinfo/stdout.input.s_initial
  ```
returns
  ```
  envisat_dump_data 9192.N1 Outdata/9192.raw 1 26897 1 5167
  ```
Then, copying and pasting this command should work exactly the same from the prompt. If it does not work, there lies your problem.
  ```
  envisat_dump_data -help
  ```
should give help.

With the given information I cannot find an error.

Message #363 I have looked at. The problem was too long file names. The suggested solution was to use symbolic links. See #366. Please do that. See this message or any Unix book for how to do that. No changes in code are required.

Maybe downloading the latest Doris version and installing that will help you further. There was a bug fixed, reported at #384, that could be related.

Try to do some tests yourself. Let the list know when you find the problem. If the problem was solved by using a symbolic link, please write a short report to the list, describing step by step what you did, why you did it, what the effect is, and why it works. Or try to be as precise as possible. This may be of help to others. If the solution for you is to follow the solution provided in #366, please let us know.

Regards,

Bert Kampes

PS. It may not be advisable to run Doris as "root". It may be very helpful for you to first spend a week learning about Unix in order to understand how Doris works, and I would encourage you to do so, since it will save you a lot of time in the long run.
+++ At 19:35, August 15th, maryam rahnemon wrote:

<mr:
<mr: Dear Bert
<mr:
<mr: I try to process Envisat data using Doris 3.11. run -s1 is ok. but I have problem with running -s2
<mr:
<mr: [root@localhost Testdoris]# run -s2
<mr:
<mr: environment variable PAGER not set, please set this variable
<mr:
<mr: to specify your prefered viewer. For example in csh shell type:
<mr:
<mr: setenv PAGER more
<mr:
<mr: setenv PAGER less  (adviced)
<mr:
<mr: Missing filename ("less --help" for help)
<mr:
<mr: using PAGER more
<mr:
<mr: environment variable EDITOR not set, please set this variable
<mr:
<mr: to specify your prefered editor. For example in csh shell type:
<mr:
<mr: setenv EDITOR vi
<mr:
<mr: setenv EDITOR nedit
<mr:
<mr: setenv EDITOR xemacs
<mr:
<mr: /usr/local/bin/run: line 1079: nedit: command not found
<mr:
<mr: using EDITOR vi
<mr:
<mr: doris Inputfiles/input.s_initial >> Outinfo/stdout.input.s_initial
<mr:
<mr:
<mr:
<mr: PROGRESS: Interpretation inputoptionsfile finished.
<mr:
<mr: total cpu: 0 min 0.01 sec
<mr:
<mr: PROGRESS: Finished initialization
<mr:
<mr: PROGRESS: Orbit: interpolation coefficients computed.
<mr:
<mr: PROGRESS: Start S_READFILES.
<mr:
<mr: WARNING : Under development, requires envisat_dump_header
PROGRESS: Making system call to envisat_dump_header2doris.csh
PROGRESS: Finished system call to envisat_dump_header2doris.csh
PROGRESS: Finished S_READFILES.
total cpu: 0 min 0.01 sec
PROGRESS: Orbit: interpolation coefficients computed.
PROGRESS: Computing integration constant based on tiepoint
PROGRESS: Start S_CROP.
WARNING: Under development, requires envisat_dump_data
PROGRESS: Making system call to get ASAR SLC data
PROGRESS: system call may take some time...
PROGRESS: Finished system call to envisat_dump_data
PROGRESS: Finished S_CROP.
total cpu: 0 min 0.01 sec
WARNING: File: Outdata/6687.raw does not seem to exist (may not be a problem).
PROGRESS: calling preview for cropped slave
PROGRESS: Start PREVIEW generation.
PROGRESS: SUNraster file created of: Outdata/6687.raw (see also file: ./slave_mag.ras.sh)
total cpu: 0 min 0.01 sec
...Everything should be made as simple as possible, but not simpler.
--- WARNING SUMMARY ---
There were 3 messages:
1: Under development, requires envisat_dump_header
2: Under development, requires envisat_dump_data
3: File: Outdata/6687.raw does not seem to exist (may not be a problem).
Redirected output in file: Outinfo/stdout.input.s_initial
view with: run -v2
Dear Javier,

OK. First comment of my side is that the yahoogroups account is actually only intended for archiving. (to prevent spam, etc.) There is a listserved for Doris, see the website, I encourage you to join. No people should be member of the yahoo list imo. The listserver does not accept attachments, or big postings. Put things on a website if you want to show them, or ftp server, or copy relevant parts. If someone is willing to look at your input files, result files, etc. send the full files in a private email to that person only.

After run -e1, you did not change the card M_ORBDIR appropriately. Note that this requires installing the orbit files and getorb program.

If Doris crashes, please take a systematic approach to find the error. I:
1) limit the processes to one PROCESS card. In your case, change with run -e1 the file to only do the READFILES step.
2) Run Doris with that step. (run -s1)
3) Look only at the first warning/error.

I suspect that this will be OK for you. The warning you get is that your CEOS SLC file has a value 9999 in some field where I did not anticipate that. This could create a problem. Check the create master result file carefully, particularly the Range sampling rate.

II)
1) limit the processes to one PROCESS card. In your case, change with run -e1 the file to only do the CROP step.
2) Run Doris with that step. (run -s1)
3) Look only at the first warning/error.

I hope this will work without problems.

III)
1) limit the processes to one PROCESS card. In your case, change with run -e1 the file to only do the PORBITS step.
2) Run Doris with that step. (run -s1)
3) Look only at the first warning/error.
I think getorb may not be installed. Either do not do the PORBITS step, (requires repeating the readfiles step to get back those orbital data), or install getorb and the ODR files.

Good luck processing. Hopefully Doris grows on you.

Regards,

Bert Kampes

PS. I will be out of the office next 3 weeks.

+++ At 15:38, November 18th, javierurien wrote:

    <j>
    <j>
    <j> --- In doris_users@yahoogroups.com, Bert Kampes <bert.kampes@...> wrote:
    <j> >
    <j> > Dear Javier,
    <j> >
    <j> > Are you an experienced user who switched to Cygwin, or is
    <j> > this one of your first attempts to run Doris? I am not sure
    <j> > whether your problem is related to Cygwin.
    <j> >
    <j> > first time user...
    <j> >
    <j> > If you did not specify
    <j> > <j: M_ORBDIR /home/fmr/d4/delftorbits/ers?
    <j> >
    <j> > then Doris does not use it. Impossible.
    <j> > I suspect you simply typed run -s1 without editing the
    <j> > input file?
    <j> >
    <j> > I went through the run -e1 and replaced lines like these:
    <j> >
    <j> > M_IN_VOL /cdrom/scene1/vdf_dat.001 // slc volume file
    <j> > M_IN_LEA /cdrom/scene1/lea_01.001 // slc leaderfile
    <j> > M_IN_DAT /cdrom/scene1/dat_01.001 // slc data file
    <j> >
    <j> > with something like this:
    <j> >
    <j> > M_IN_VOL /cygdrive/f/scene1/vdf_dat.001 // slc volume file
    <j> > M_IN_LEA /cygdrive/f/scene1/lea_01.001 // slc leaderfile
    <j> > M_IN_DAT /cygdrive/f/scene1/dat_01.001 // slc data file
    <j> >
    <j> > I did not change other things, that is, I replaced /cdrom with
    <j> > /cygdrive/f all over the file, but that’s it.
    <j> >
    <j> > please set the correct input cards, how else
    <j> > is Doris supposed to know the location of data files?
    <j> >
    <j> > I believe that I did.
    <j> >
    <j> > Also see the example section of run -h.
I have problems with this command, will search a little and see if I can fix it.

Is it OK if I post the configuration files?

TIA.

Javier.

Let us know if this was the problem.

Regards,

Bert Kampes

PS.

--- WARNING SUMMARY ---

There were 2 messages:

1: NOT USED ANYMORE. since v3.8

2: Number of invalid samples 9999 not 0: rsr may be wrongly computed.

ERROR: [utilities.cc[214]]: code 905: utilities.c: getorb:
Dear wangtao8869,

please send questions to the email list.

<w: ERROR : [ioroutines.cc[1189]]: Results of step: 15 (coarse_orbits:) already in result file.

Consult the manual, i.e., the introduction, section "outputfiles", currently:
http://enterprise.geo.tudelft.nl/doris/usermanual/node8.html
Each step can only be run once (after successful exit). If you want to run it again, (i) set the process control flag to zero, and (ii) delete the output section in the corresponding result file.

I suspect you had e.g., three PROCESS cards in the input file. Then Doris failed or was stopped during the run while doing the second PROCESS. And then you tried to run Doris again. After each run it is wise to check the result files to see what has happened. Here, I suspect, you had to comment out the first PROCESS card in the input file before re-starting.

Regards,

Bert Kampes

+++ At 06:42, October 1st, wangtao8869 wrote:

<w: Dear sir:
<w: I can’t understand the output of the below. Can you help me ?
<w: [root@sar_pc1 Your_work]# doris doris2.in > stdout.doris2
<w: WARNING : It is highly recommended to use FC_PLOT to plot the computed FINE offset vectors.
<w: PROGRESS: Interpretation inputoptionsfile finished.
<w: total cpu: 0 min 0.09 sec
<w: --- WARNING SUMMARY ---
<w: There was 1 message:
<w: 1: It is highly recommended to use FC_PLOT to plot the computed FINE offset vectors.
<w: ERROR : [ioroutines.cc[1189]]: Results of step: 15 (coarse_orbits:)
<w: already in result file.
<w: !!! ABNORMAL TERMINATION !!!
<w: wangtao
<w: China
<w: October first

+++ At 19:19, March 20th, Severino Fernandez wrote:

<SF: I can not run snaphu from Doris. Even if It seems to generate the correct snaphu configuration file (snaphu.conf), Doris executes an incomplete dos command (snaphu -f snaphu.conf <width of the interferogram>). The complete command sequence accepted by snaphu includes the interferogram file: snaphu -f snaphu.conf <name of the interferogram file> <width of the interferogram>. If I use the snaphu configuration file generated by Doris, and just add the name of the interferogram file, it seems to work OK. I supposed that Doris is able to obtain the interferogram file name from the interferogram.out file it generates. In the source of Doris (unwrap.cc) it seems to
try to do it, but somehow the field interferogram.file it uses is empty.
I could apply the next step (slant2h) if I just knew how to edit interferogram.out, but the example in the manual is specific for the other method (the one from Ramon I do not have access to).
Could you please tell me how I could continue, either fixing something so that Doris calls snaphu with the correct parameters, or that I mimic the unwrap output in interferogram.out?
Best regards

From bert.kampes@dlr.de Mon Mar 21 09:04:18 2005
Date: Mon, 21 Mar 2005 08:56:33 +0100 (CET)
Subject: [doris_users] Re: height from unwrapped phase with snaphu

Dear Severino,

Indeed, the file should be there. I can detail how Doris tries to get it:

Please have a look at the doris web pages, the code is now doxygen-ed online at:
http://enterprise.lr.tudelft.nl/doris/DOXYGEN/html/

"unwrap.cc" [readonly] line 409 of 769 --53%-- col 6

void snaphu_unwrap(
    const input_gen &generalinput,
    const input_unwrap &unwrapinput,
    const productinfo &interferogram,
    const slcimage &master,
    const slcimage &slave,
    orbit &masterorbit,
    orbit &slaveorbit,
    const input_ell &ellips)

i.e., the function is called "snaphu_unwrap", in file "unwrap.cc" and as input it uses an object of the class "productinfo", which is named "interferogram".

The call to snaphu is
"unwrap.cc" [readonly] line 714 of 769 --92%-- col 28

    // ______ Call snaphu! ______
    INFO.print("System call for running snaphu follows:");
    INFO.print(basecmdstring);
    system(basecmdstring);

where

"unwrap.cc" [readonly] line 434 of 769 --56%-- col 25

    // ______ Make commandstring ______
    char basecmdstring[3*ONE27];
    ostrstream basecmdstr(basecmdstring,3*ONE27); // to convert int to char
    basecmdstr << prog << " -f " << configfile << " "
        << interferogram.file << " "
        << filepixels << ends;
In short, if the (public) data member (variable) "file" of the object "interferogram" is empty, then the command string will be incorrect as in your case.

This class is defined in productinfo.cc productinfo.hh. at http://enterprise.lr.tudelft.nl/doris/DOXYGEN/html/go to file list, click on it productinfo.cc (source), etc.

So the real question is; "where is interferogram->file set" (excuse the poor notation here.)

Indeed, this should be read from the product result file. Normally, the last known processing stage would be "INTERFERO" output, i.e., the file specified in that section should have been copied in this variable.

If you use "SCREEN debug", Doris should tell you all about the last known processing stage, the content of all members of "master", "slave", "interferogram", etc. In this output, simply try to find "interferogram.file". In your case it is not set, which seems to indicate the the "INTERFERO" section does not contain a string "Data_output_file: filename"

You can send me the file "interferogram.out" and the output of a run with SCREEN debug level if you cannot figure it out further. I will try to have a look at it.

Best regards,

Bert Kampes

-------------------------------------------------------------------------------
FYI - -

Severino's problem that Doris was unable to find certain parameters in the result files was due to a Unix vs. Windows issue.

The difference is the way an "end-of-line" characters are written. Doris apparently requires the Unix way. Severino used a MS-Window's editor to change the file, which changed it to DOS style EOLs.

The solution that works was to use a "dos2unix" utility after changing the files. Better would be to use an editor under cygwin, such as vim, nedit, xemacs, xedit, etc. so the problem cannot occur.

Note that such effects can also happen when ftp-ing files.

Also, I was informed that this also occurs when Cygwin was installed using DOS-type file format (not recommended to do this.)

Best regards,

Bert Kampes
E.3 Formats files, satellites, orbits

From Bert.Kampes@dlr.de Mon Mar 14 08:22:08 2005
Date: Wed, 16 Feb 2005 16:40:03 +0100 (CET)
From: Bert Kampes <Bert.Kampes@dlr.de>
To: Gustavo Arciniegas <arciniegas06157@itc.nl>
Cc: Doris software list <doris_users@tudelft.nl>, doris_users@yahoogroups.com
Subject: Re: Amplitude, magnitude and intensity

+++ At 15:09, February 16th, Gustavo Arciniegas wrote:

[...]  
<GA: Doris computes magnitude and phase images. Can intensity images be
<GA: calculated from a co-registered pair?
<GA:
[...]

Dear Gustavo,

The answer is "yes", but I am not sure I understand your question. Please try to formulate your question accurately.

Doris computes matrices. These are generally complex files, except if you want something else (see options in manual for each step). The user visualizes, e.g., the phase or magnitude in some scale, using some color mapping. Doris makes some attempt to show quicklook results, using cpxfiddle scripts.

Possible ways to compute the intensity of a matrix in file "q.raw", 1000 pixels width, 2000 pixels high, complex float format would be:

1) load it in matlab using freadbk.m, then use, for example,
   intensity=abs(cpximage).^2;
2) with cpxfiddle, for a SUNraster file (ugly...), use something like,
   cpxfiddle -w1000 -fcr4 -e2.0 -qmag -osunraster q.raw > q.ras
   xv q.ras

You can do this for master and coregistered slave, if that is what you meant.

The magnitude of the complex interferogram is the pointwise multiplication of the magnitudes of the master and coregistered slave, i.e., something resembling the intensity of the master. For visualization this is normally rescaled using -e0.3 or so, i.e., this is a kind of histogram equalization, otherwise your image looks black with some white dots.

see also cpxfiddle -h and matlab help.

Bert Kampes

From Bert.Kampes@dlr.de Mon Mar 14 08:41:22 2005
Date: Fri, 16 Apr 2004 17:23:42 +0200 (CEST)
From: Bert Kampes <Bert.Kampes@dlr.de>
To: Juan Manuel Lpez Snchez <juanma@dfists.ua.es>
Cc: Doris software list <doris_users@geo.tudelft.nl>,
doris_users@yahoogroups.com
Subject: Re: Question about DORIS: interferogram with coregistered images

+++ At 10:04, April 15th, Juan Manuel Lpez Snchez wrote:

<JMLS: Dear Bert,
<JMLS: First of all, receive my compliments for your excellent
<JMLS: DORIS software.
<JMLS: I have an important question about using DORIS for advanced
<JMLS: differential interferometry.
<JMLS: My objective is to generate a number of interferograms of the same scene
<JMLS: with different pairs of images.
<JMLS: First of all, I need to coregister (and resample) every image with
<JMLS: respect to the same master in order to have all of them coregistered.
<JMLS: I have already done it, by using different directories and treating
<JMLS: every image as a slave of the same master.
<JMLS: Next, I would like to compute interferograms between pairs of them.
<JMLS: The problem is that both images in the pair have been treated as slaves,
<JMLS: and now I want to make one of them to be the master.
<JMLS: I have already made a few attempts to trick DORIS, by changing the
<JMLS: *.res files, but I have not succeeded.
<JMLS: I also have looked at the source code. The problem seems to be in
<JMLS: products.cc, because it uses the full master coordinates (total number
<JMLS: of lines and pixels) to reference the calculations.
<JMLS: I have reviewed the DORIS documentation, and you say:
<JMLS: Page 85. Chapter 25. 2nd paragraph: "... or first coregister master
<JMLS: and slave on the master of the deformation pair".
<JMLS: Page 86. Caption Fig 25.1: "This interferogram has been coregistered
<JMLS: on the defo pair by tricking DORIS".
<JMLS: So, it should be possible.
<JMLS: Therefore, my question is: Would you mind to indicate to me the
<JMLS: necessary steps for using two images coregistered on a third master for
<JMLS: generating an interferogram (and then apply the substraction of flat
<JMLS: earth and DEM...)?
<JMLS: Please note that, in principle, I would like to work with conventional
<JMLS: interferograms, not to apply the DINSAR module of your software.
<JMLS: Thank you in advance for your support.
<JMLS: Kind regards,
<JMLS: Juanma
<JMLS: [...]

Dear Juan,

Thank you for your comments. It seems you know quite well how Doris works and
what you want to do. I am glad you like it.

<JMLS:
<JMLS: Therefore, my question is: Would you mind to indicate to me the
<JMLS: necessary steps for using two images coregistered on a third master for
<JMLS: generating an interferogram (and then apply the substraction of flat
<JMLS: earth and DEM...)?
<JMLS: Please note that, in principle, I would like to work with conventional
<JMLS: interferograms, not to apply the DINSAR module of your software.
<JMLS:

I do not remember doing the above.
And I think you are correct that this is not possible with the current Doris
implementation. The trick I think I used probably is correct only if we
assume all orbits are exactly parallel.
This may work for you to trick Doris, but probably better would be to make
this somehow a standard option, particularly since you seem to want to do
something advanced/special.

But let’s start at the beginning. Assume you have the following situation,
as you sketched:

```
+--MASTER---------+
|                 |
|                 |
| +RSLAVE1+       |
| |               |
| |               |
| |               |
| |               |
| +-------+       |
| |               |
| |               |
+-----------------+
```

Where RSLAVE1 is resampled on the master image grid.
```
master.res
slave1.res
ifg1.res
```

```
+--MASTER---------+
|                 |
|                 |
| |               |
| |               |
| +RSLAVE2+       |
| |               |
| |               |
| |               |
| |               |
| +-------+       |
| |               |
| |               |
```

Where RSLAVE2 is resampled on the master image grid.
```
master.res
```
I assume you used DBOW card of resample, such that the resampled SLAVE1 and SLAVE2 have the exact same dimension, grid spacing, location.

Now you want to trick Doris, such that SLAVE1 --> MASTER; SLAVE2 --> SLAVE.

I assume here now that the following holds:

`###SLAVE1############`  
`#`  
`# ..CROPSLAVE1...`  
`# . .`  
`# . | | . #`  
`# . | | . #`  
`# . | | . #`  
`# | | . #`  
`# | | . #`  
`# | | . #`  
`# | | . #`  
`# l.-------+ . #`  
`# . . .`  
`# ..p............`  
`#`  
`################`  

That is, that the resampled slave is only shifted, not skewed, then we may trick Doris. To simplify it a bit I now only consider the one dimensional case in azimuthal direction. The relevant layout is now as follows:

```
1------r-------------R-------M  MASTER coordinate system 1 to M lines
1------Q---------------------S  SLAVE1 coordinate system 1 to S lines
l-----------------------------L  CROPPEDSLAVE1 coordinate system:
\hspace{1cm}l to L in SLAVE1 coordinates
r-------------R  RSLAVE1 coordinate system:
\hspace{1cm}r to R in MASTER coordinate system
```

What you want is that the SLAVE1 becomes the master, and RSLAVE1 becomes the crop of that new master. We thus have to indicate in a new master result file what the first line is of that crop wrt. the original SLAVE1, i.e., what are r and R in SLAVE coordinates. Such we pretend that RSLAVE1 is a cropped master, we have to obtain Q in SLAVE1 coordinates.

So, what you have to do is the following:

1) mkdir RSLAVE1_RSLAVE2
2) cd RSLAVE1_RSLAVE2
3) cp ../slave1.res master.res
4) cp ../slave2.res slave.res
5) vi master.res
   a) delete line "resample: 1" at top
   b) delete section "resample" at end. Assume it contained the following lines:
   First_line (w.r.t. original_master): 7
   Last_line (w.r.t. original_master): 22
   In the last sketch, thus: r=7, R=22 (Doris starts counting lines with 1, not 0)
   c) go to section "crop" in this file. suppose it says
   First_line (w.r.t. original_image): 5
This is annotated in the last sketch in the SLAVE1 coordinate system with
l=5, L=27.

What we have to figure out is the distance between l and r.
Go to the master_slave.res file of the first original interferogram.
Section "*_Start_coarse_correl:
  Coarse_correlation_translation_lines: -3
This means that SLAVE1 is shifted wrt. MASTER approximately 3 lines.
I am not sure about the sign here at the moment, see the manual.

I think Q in slave coordinates now becomes Q=r--3=10 (?)

I hope this clarifies it a bit for you, and you are able to use this.
If I have forgotten something please let the list know your experiences.

Best regards,

Bert Kampes

PS. I forwarded your question to the list so it gets archived and may help others.

PPS.
Obviously this may not be correct in general, since an actual polynomial transformation
would be required. However, if the orbits are parallel, it may be reasonably correct.
I do not see a simple solution to this particular problem.
The consequence when the polynomial is not zero degree (a shift in both directions),
is that Doris will compute the wrong master sensor position (i.e., SLAVE1),
since it thinks it is in line X, while it actually is in line X+dX.
The correct way of dealing with it would be something like
true_line_slave = inv(f)(line_resampled_slave), with f the resampling polynomial ??
See source code and help. See manual:
Also see "http://groups.yahoo.com/group/doris_users/message/264".

kampes@capone[17:50]: envisat_dump_data
argc: 1.000000
Usage: envisat_dump_data envisat-product outputfile [l0 lN p0 pN]
where envisat-product is the input filename
outputfile is the output filename
l0 is the first azimuth line (starting at 1)
lN is the last azimuth line
p0 is the first range pixel (starting at 1)
pN is the last range pixel

Example:
envisat_dump_data ASA_IMS_1PNPAP20021025_175208_000000162010_00356_03416_0005.N1 crop.out 0 10 0 100

From this I thought it would be clear that this program crops part of the
envisat SLC data from the envisat format and dumps it to a output file
that can be read by Doris (see the manual for definition).

Regarding your previous message, Mon Apr 5, 2004, subject
"Does it have a step by step instruction for how to use Doris?",
I would recommend not to write a converter to CEOS format, also see a previous
discussing in the archive. If you do want to pursue that idea, see the manual
and documentation for detailed instructions. But in that case, please consider
making your converter public domain.

Regards,
Bert Kampes

PS. For processing envisat with doris you don't have to run envisat_dump_data.

---

Dear K.K. Mohanty,

Since both First_pixel_azimuth_time (UTC): and Range_time_to_first_pixel
(2way) (ms): are critical parameters, the option I have now is to use scene
centre UTC time and line, pixel values and nominal PSR and RSR to compute
these. Is there a better solution or how bad will be the current approach. Or
is there any where else in Radarsat leader from which I can retrieve these.

I think your approach is sensible. I would like to clarify it a bit.
In the past I have discussed this with other people too, but I could not
find a record of it in the yahoo email list.
(1) First_pixel_azimuth_time:
The formula used by Doris relating azimuth time to line number is:
\[ t(\text{linenumber}) = t(\text{firstline}) + \text{PRF}*(\text{linenumber}-1); \]  
\[ \text{// (i.e., linenumber of firstline=1)} \]

Suppose we know/read the nominal PRF from the leader: \text{PRF}.
And we somehow know the time of the center pixel: \text{t}_{\text{az center}},
and we know the line number of the center pixel: \text{l}_{\text{center}}

Then we can compute the starting time as:
\[ t(\text{firstline}) = t(\text{az center}) - \text{PRF}*(\text{l}_{\text{center}}-1) \]

Of course this needs to be re-formatted to UTC time stamp, e.g.,
"07-JAN-1998 06:13:00.384".
The format of the scene center time is given in the CEOS leader UTC
"YYYYMMDDhhmmssttt". I do not know whether the given scene center time is zero-doppler.

(2) Range_time_to_first_pixel:
The formula relating range time to pixel number is:
\[ t(\text{rangepixel}) = t(\text{firstpixel}) + 2*\text{RSR}*(\text{pixelnumber}-1); \]  
\[ \text{// (i.e, pix=1 is first one)} \]
and range to travel time
\[ \text{range}(\text{rangepixel}) = \text{SpeedOfLight} * t(\text{rangepixel}) \]

Possibly two-way time/range, etc., see source code. But if you somewhere can find the
range to the first range bin, and you know the range sampling rate, then you can
indeed compute the required parameter "Range_time_to_first_pixel (2way) (ms)"
Note that in the source code there are functions like \text{pix2tr()} for pixel number
to range time conversion, so you don’t have to think about this too much once
you know the proper values of the arguments.

Maybe it would be nice if Doris could do this; after finding these parameters
first compute provisional first azimuth/range times, and then later to update
them. I think v3.10 should do this for the PRF, i.e., use the nominal one
if that is the only thing that can be found. It would be a great help if you can
add it this to the code. Also see appendices of the manual for these definitions.

<KM: I am also not able to find details of a leader record in Radarsat leader
<KM: which is about "Detailed Processing parameters" of 7726 bytes, as standard
<KM: ceos leaves its detailed specification open. Will it contain this? I am also
<KM: parallelly trying to explore from RSI.
<KM:

The start of each section contains codes, e.g, first record type, second record
sub-type code, etc. You could go through all record of the leader file by reading
the start, length, print record codes, go the next record, obtain length, print
record codes, etc. which may give you an indication where what record is. I am not
sure how to obtain the meaning of these codes.

Using other software to try to read this CEOS data may be helpful. I used ESA provided
software available on their website once without success (same as Doris). But I think
http://www.asf.alaska.edu/apd/software/
may be of interest too, in the distributed source code there are files like "ceos.h".

I hope you are able to continue your work.
Kind regards,

Bert Kampes
+++ At 12:49, April 2nd, K.K. Mohanty wrote:

<KM>

Dear Doris Users,

I am trying to process a pair of Radarsat CEOS format SLC images using Doris. The standard CEOS format used by Radarsat is different from that of ERS. I have done necessary homework to read Radarsat CEOS format.

All parameters required by Doris can be retrieved from Radarsat CEOS format except the two critical parameters namely,

First_pixel_azimuth_time (UTC):

and

Range_time_to_first_pixel (2way) (ms):

In case of ERS these are derived from the following parameters,

SLC data set summary record: sensor specific local use segment

Zero-doppler range time (two-way)
+of first range pixel (millisec):
+of centre range pixel (millisec):
+of last range pixel (millisec):

Zero-doppler azimuth time
+of first azimuth pixel (UTC):
+of centre azimuth pixel (UTC):
+of last azimuth pixel (UTC):

These parameters in Radarsat leader file are missing and CEOS standard declares entire sensor specific local use segment as spare.

Moreover, Radarsat leader file doesn’t contain a map projection record and Doris intelligently writes the nominal PRF and RSR values as computed values.

Since both First_pixel_azimuth_time (UTC) and Range_time_to_first_pixel (2way) (ms) are critical parameters, the option I have now is to use scene centre UTC time and line, pixel values and nominal PSR and RSR to compute these. Is there a better solution or how bad will be the current approach. Or is there any where else in Radarsat leader from which I can retrieve these.

I am also not able to find details of a leader record in Radarsat leader which is about “Detailed Processing parameters” of 7726 bytes, as standard CEOS leaves its detailed specification open. Will it contain this? I am also parallelly trying to explore from RSI.

Any suggestion or experience of anyone who has processed Radarsat SLC will be highly useful.

Thanking you all

KKMohanty
Dear insar_china,

Doris seems to have a problem reading your data. Doris can handle the CEOS format that is generated by the ESA PAFs. The number 4096 rings a bells somewhere, but I can't remember what exactly. Depending on the time you want to invest working with these data, you can try to create an initial master.res yourself, based on the information Doris was able to read, and by putting in some dummy parameters yourself (e.g., 0 for Doppler parameters if Doris cannot read them, or copying the range starting time from some other image. The starting azimuth time you may set to a reasonable time which may be in the log file, or from when you ordered the data.)

If Doris is then able of doing step M_CROP, chances are you can create at least an interferogram with the help of Doris. Maybe M_CROP does not even require running step READFILES, so you can try to run that step first, and then you can simply check with the quicklook image if the correct data was cropped.

If this is however the first time you try to work with Doris, probably the best thing to do is to order the same data from ESA. I heard SLC data now became quite reasonably priced. The extensions you mention are not standard.

Regards,

Bert Kampes

PS. Please send/join the email list server at geodesy, delft university, do not join/send directly to the yahoo list or to me. see doris home page.

+++ At 01:02, May 9th, insar_china wrote:

<1>: I have some slc files, the filename of which are *vol(volume directory file), *.led(leader file), *.img(i guess it is date set file) and *.nul(null volume file).
<1>: when doris process M_READFIERS, some warning occured:
<1>: WARNING : readvolume: number of pointer records = " 3";
<1>: expected "2" for ESA SLC (full scene).
<1>: WARNING : readvolume: number of records = " 5"; expected "4" for
From Bert.Kampes@dlr.de Mon Mar 14 08:38:26 2005
Date: Mon, 3 May 2004 13:08:25 +0200 (CEST)
From: Bert Kampes <Bert.Kampes@dlr.de>
To: Zbigniew Perski <perski@us.edu.pl>
Cc: Doris software list <doris_users@geo.tudelft.nl>,
    doris_users@yahoogroups.com
Subject: Re: problems with Envisat Tools - still the same....

Dear Zbigniew,

Thanks for sending me the files that had the problem. I found
that the created master result file contained lines like:

Radar_wavelength (m): 0,0562356
instead of:
Radar_wavelength (m): 0.0562356

The reason is that the awk command that computes the wavelength
from the radar frequency and speed of light respects your LOCALE setting.

Observe the following:
kampes@newton[18:22]: setenv LC_NUMERIC pl
kampes@newton[18:22]: echo "5300000000.0" | awk '{print 299792458.0/$1}"
0,0562356
kampes@newton[18:22]: setenv LC_NUMERIC C
kampes@newton[18:22]: echo "5300000000.0" | awk '{print 299792458.0/$1}"
0.0562356

(Who knew?)

Anyway, at the start of the script "envisat_dump_header2doris.csh" now this
environment variable is set (i.e., at line 46 I added: "setenv LC_NUMERIC C")
This should take care of it, at least for this script. Please report if
this change seems required elsewhere.

The next release will incorporate this change. Until then users will have to
make this change themselves.
Best regards,
Bert Kampes

PS. See also: man locale

+++ At 16:04, April 23rd, Zbigniew Perski wrote:

<ZP: Dear Bert,
<ZP:
<ZP: I have still the same problem. I have tried to change the compiler flag
<ZP: and also forced to use gcc-2.95 instead of 3.33. The result is the
<ZP: same..
<ZP:
<ZP: > <ZP: #There were 3 messages:
<ZP: > <ZP: # 1: wavelength is smaller than 1 centimeter?
<ZP: > <ZP: # 2: wavelength seems to deviates more than 1 cm from ASAR nominal.
<ZP: > <ZP: # 3: rsr deviates more than 0.1 MHz from ASAR nominal.
<ZP: > <ZP: #ERROR : [orbitbk.cc[155]]: orbit time axis: require distinct, time
<ZP: > <ZP: #sorted data
<ZP:
<ZP: The same for all Envisat data which I have also for BAM dataset!
<ZP:
<ZP: > You did
<ZP: > not by chance re-install your compiler, or upgraded linux somehow?
<ZP: > It may be a difference between my gcc and your gcc. I use 2.95.1 under SUN.
<ZP:
<ZP: Here is probably the case I have re-installeed it again today but it
<ZP: works exactly as before.
<ZP:
<ZP: Best regards
<ZP:
<ZP: Zbigniew
<ZP:

From Bert.Kampes@dlr.de Mon Mar 14 08:35:44 2005
Date: Mon, 16 Aug 2004 09:41:50 +0200 (CEST)
From: Bert Kampes <Bert.Kampes@dlr.de>
To: YouLiang Jia <jyl69@mail.china.com>
Cc: Doris software list <doris_users@geo.tudelft.nl>,
doris_users@yahoogroups.com
Subject: Re: [doris_users]: getorb problem

+++ At 11:44, August 16th, YouLiang Jia wrote:

<YJ: Dear Bert,
<YJ:
<YJ: I try to do according your way, But doris can’t keep the result in '*.res', so when I do next step
<YJ:
Dear Jia,

Your files looks fine to me. In the attached master.res the information was (see below):

First_pixel_azimuth_time (UTC): 02-OCT-1997 02:55:44.624

It seems getorb does not work for this time? You can thus maybe better do the following:

1) Start over and first process it without using precise orbits.
   i.e., do not use PROCESS M_PORBITS.
   use polyfit 3 for orbit interpolation.
   This will use the orbit data that is given in the SLC leader file.

2) if (1) works, but the orbit in the SLC is not precise enough,
   i.e., you have an residual orbit trend in the interferogram, then
   investigate why it does not work.
   to the screen/stdout, there should be a message what command
doris uses to call getorb, something like
getorb 19971002025540,19971002025620,5 /data/orbits/ERS1/
try to understand this command.

-read the help of getorb by typing getorb at the prompt
and visit the homepage of getorb.
-does the given directory exist, in my example: "/data/orbits/ERS1/"
-give this command from the prompt, not using Doris.

Does this work?

For me the following works (I used ERS2 here) do you get the exact same?

kampes@franklin[9:38]: getorb 19971002025540,19971002025620,5 /home/capone02/kampes/DELFTORBITS/ERS2/dgm-e04/
(getorb: orbit file /home/capone02/kampes/DELFTORBITS/ERS2/dgm-e04//ODR.256)
402375340.000  0 26.2979212 -70.6302385 787299.552 2131796.385 -6063776.575 3157494.610
402375345.000  0 26.5933851 -70.7823785 787371.931 2104827.659 -6038383.363 3223330.688
402375350.000  0 27.1842124 -70.8594184 787409.124 2091271.880 -6025427.240 3256117.643
402375355.000  0 27.4795749 -70.9363932 787446.976 2077669.145 -6012298.525 3288816.021
402375360.000  0 27.7749025 -71.0136676 787485.481 2064019.948 -5998997.540 3321424.931
402375365.000  0 28.0701948 -71.0912468 787524.630 2050324.784 -5985524.611 3353943.487
402375370.000  0 28.3654511 -71.1691364 787564.416 2036584.148 -5971880.068 3386370.806
402375380.000  0 28.6607100 -71.2473420 787604.833 2022798.536 -5958064.250 3418706.006

But for ERS1, what your image is, it seems there is no orbit available:

kampes@franklin[9:40]: getorb 19971002025540,19971002025620,5 /home/capone02/kampes/DELFTORBITS/ERS1/dgm-e04/
(getorb: reading arclist /home/capone02/kampes/DELFTORBITS/ERS1/dgm-e04//arclist)
402375340.000  0 0.0000000 0.0000000 0.000 6378137.000 0.000 0.000
402375345.000  0 0.0000000 0.0000000 0.000 6378137.000 0.000 0.000
402375350.000  0 0.0000000 0.0000000 0.000 6378137.000 0.000 0.000
402375355.000  0 0.0000000 0.0000000 0.000 6378137.000 0.000 0.000
402375360.000  0 0.0000000 0.0000000 0.000 6378137.000 0.000 0.000
402375365.000  0 0.0000000 0.0000000 0.000 6378137.000 0.000 0.000
402375370.000  0 0.0000000 0.0000000 0.000 6378137.000 0.000 0.000
402375375.000  0 0.0000000 0.0000000 0.000 6378137.000 0.000 0.000
402375380.000  0 0.0000000 0.0000000 0.000 6378137.000 0.000 0.000

And this seems to be the problem. Apparently, there is no precise orbit for ERS 1 for October 2, 1997. If you look in the arclist file, it also states that. The arclist I have goes only until June 1996, and the fd orbits start at June 1999.

Doris should have reported you that there is a problem with getorb, since the second column is not 0. If you continue making an interferogram using option (1) above, you likely will have residual orbital fringes.

To remove those, consider the trick with the reference phase computation, see eg. (currently)
http://www.geo.tudelft.nl/fmr/research/insar/sw/doris/Usersmanual/node85.html

Greetings,

Bert Kampes

+++ At 11:29, August 10th, YouLiang Jia wrote:
<YJ: Bert,

<YJ: I have a trouble to process a scene of radar data, you can reference the attachment to get the information.

<YJ: Thanks

<YJ: Jia youliang

---

the relevant information from the attached file you send:

---

*******************************************************************
*_Start_readfiles:
*******************************************************************

Product type specifier: PRODUCT:ERS-1.SAR.SLC

First_pixel_azimuth_time (UTC): 02-OCT-1997 02:55:44.624

*******************************************************************
*_End_readfiles:_NORMAL
*******************************************************************

*******************************************************************
*_Start_precise_orbits:
*******************************************************************

t(s) X(m) Y(m) Z(m)

NUMBER_OF_DATAPOINTS: -1

*******************************************************************
*_End_precise_orbits:_NORMAL
*******************************************************************

From Bert.Kampes@dlr.de Mon Mar 14 08:35:07 2005
Date: Wed, 11 Aug 2004 10:36:26 +0200 (CEST)
From: Bert Kampes <Bert.Kampes@dlr.de>
To: YouLiang Jia <jyl69@mail.china.com>
Cc: Doris software list <doris_users@geo.tudelft.nl>, doris_users@yahooogroups.com
Subject: Re: format of heights.raw and interpolation to grid

+++ At 00:10, August 11th, YouLiang Jia wrote:
...

<YJ: another question is what is the format of your heights.raw? How can I convert it to other format?

<YJ: Thank you,
Dear Jia,

Please refer to the manual, appendix of file formats. My convention normally is to use the format of the file as extension for the name. For the "height.raw" a better name would thus might have been "heights.r4".

Almost all files used by Doris are "raw". This does not refer to RAW radar data, but to a raw binary stream. That means, the data is simply stored without any header, footer, etc. Such information is available from the result files, e.g., "master.res".

The data are mostly written row major (row-by-row), pixel interleaved (for complex data: real part pixel 1, imag part pixel 1, real part pixel 2, imag part pixel 2, etc.).

"cpxfiddle -h" gives the following information:

```
1  range (X) -->
1  ---------------------
|                   |
| <Complex file>    |
| e.g. float (2x4B) | azimuth (Y)
| format            |
|                   |
|                   |
| (major row order) |
| (pixel interleav.)|
\/
---------------------
```

There is a band-interleaved format used, called "hgt". That is because some unwrapping software uses that format to store row-by-row first a band for amplitude, then for unwrapped phase, second row, etc. Visually this can also be seen as, e.g.,

```
+----------+----------+
+ + | row 1     + +
+ BAND 1 + BAND 2 | row 2
+ + | .          + +
+ + | .          + +
+ + |             + +
+----------+----------+
```

I think the raw format is common for GIS systems, and I expect any software to be able to read such files. You must however tell the program yourself then how many lines the file has, and exactly how the data is stored. As you know however, it is not very accurate to simply give the lon/lat for the corners of such an image. The radar derived information needs to be geocoded.

After geocoding, you may have files for latitude, longitude, and height. To interpolate this to a regular grid, please see/edit the example scripts provided, e.g.,
These scripts are not very efficient, but they are easy to understand, and work on all platforms. GMT is used for the actual interpolation. I hope users will supply their own scripts that do more complex operations, such as a manual how to import Doris processed data to, e.g., GRASS, and then to combine it with LANDSAT data. See the fringe 2003 presentation for more information on that too.

Also see the GMT help pages, and search the FAQ for this issue.

I hope this clarifies things for you,

Bert Kampes

---

Dear Marius,

Doris indeed uses focused data (SLC) as input (all sensors).

The main reason/advantage for this decision was that when we started developing the Doris software, we did not want to write yet another implementation of an existing algorithm, without having experience with it. We felt there are enough software that do that, and we cannot improve on for example the processor used by ESA. (Doris is developed at the departement of Geodesy of Delft University of Technology, not, e.g., at the dep. of electric engineering.)

Price used to be the main consideration to use RAW data. RAW data used to be half the price for ERS. It seems ERS data recently came available at low cost for scientific purposes (40 euro/scene?), in RAW or SLC format, so there is no difference here anymore.

The main difference is that the ESA processed SLC format is "general purpose". It is filtered using a HAMMING window, to reduce side lobes. Starting from RAW data it would be possible to change these filters. The SLC format is useful to people in interferometry, but also used by people not interested in the phase information.

I do not see an advantage over using RAW data for normal interferometric applications. The SAR processor developed for ESA likely is more sufficated/calibrated/robust/tested, and was more expensive than, e.g., ROI_PAC or a few lines Matlab implementation.

For geocoding purposes, the annotated orbit, starting line and times, etc. are quite important. I trust ESA to do this correctly, but when you
perform the focusing yourself, such information may be less accurate.

I do not know about studies comparing using the ESA produced SLC format or starting with RAW data and focusing it yourself. As an intermediate step, the data I assume is focused, say in image SLC'. Then an easy test would be to perform a point target analysis, i.e., compare how well a point scatterer or transponder is focused in SLC vs. SLC'. Comparing the quality of the created interferogram is not that easy, since it depends also on, e.g., coregistration.

Best regards,

Bert Kampes

PS. I forwarded your question to the user community. Maybe other people have other ideas, or know about studies.

+++ At 22:30, August 4th, Marius Necsoiu wrote:

<MN:
<MN: -----Original Message-----
<MN: From: marius@iprimus.com [mailto:marius@iprimus.com]
<MN: Sent: Wednesday, July 28, 2004 5:22 PM
<MN: To: kampes@geo.tudelft.nl
<MN: Subject: SLC vs raw data
<MN:
<MN: Dear Mr. Kampes,
<MN:
<MN: Recently I learned about DORIS capabilities in processing ERS/ ENVISAT data. Regarding ERS data, my understanding is that the software is using ERS SLC datasets, having some provisions for processing raw data via external software.
<MN:
<MN: I am wondering what are the advantages of creating interferograms starting from SLC data. I know, for example, that ROI_PAC is using raw ERS data.
<MN: Are there any studies available that compare the quality of interferograms obtained from both methods?
<MN:
<MN: Thank you.
<MN:
<MN: With regards,
<MN:
<MN: Marius
<MN:

From Bert.Kampes@dir.de Mon Mar 14 08:33:15 2005
Date: Thu, 2 Sep 2004 08:37:51 +0200 (CEST)
From: Bert Kampes <Bert.Kampes@dir.de>
To: Severino Fernandez <severinofer@recol.es>
Cc: Doris software list <doris_users@tudelft.nl>, doris_users@yahoogroups.com
Subject: Re: NaN in file

Dear Severino,

I do not generally experience a problem with NaN values in Doris output files. Indeed, the coherence image did contain NaN values, and I thought a small utility (C program) that can replace them by a specified value would be very helpful.

I did not write this, but think it would not take very long (once you know how to detect a NaN value). But if you have Matlab, it is a few lines, described also in the manual and previously on this forum. I have heard people were writing a utility in C, and I may add it to cpxfiddle in a next release. (usage something like:
   cpxfiddle -w1 -fcr4 -qnormal -ofloat -N0.001 inputfile > outputfile

Where -N would replace Nan with the specified value.

Maybe a test in C for a NaN is the following, instead of isnan():
   isnan = ( a==a ) ? false : true;

Besides maybe the function finite() is better than isnan(), since this also signals infinity.

For the coherence, I thought a NaN value could occur if the amplitude of master or slave is equal to zero. This test is performed in Doris when computing the coherence image. For some reason, still sometimes NaN values occur. I did not investigate this further. It may be more likely to happen if the windows used for multilooking and coherence estimation are of different size maybe.

For the complex interferogram I cannot think of a reason, also after subtraction of a reference phase (i.e., complex conjugate multiplication). Strange that snaphu signals this. You could look in the code of snaphu to see how snaphu recognizes such anomalous values. By not using the coherence as input to snaphu, but using a configuration where for example the amplitude of the complex interferogram is used, you may also be able to avoid these problems with snaphu.

I hope another user will react with a program, or knows in more detail when in Doris a NaN value is generated during step COHERENCE and SUBTR_REFPHA.

Best regards,

Bert Kampes

+++ At 18:32, September 1st, Severino Fernandez wrote:

<SF: I have been using Doris to generate the wrapped interferogram and I
<SF: wanted to use snaphu to unwrap it.
<SF: I have also the already mentioned problem of NaN values in the
<SF: interferogram generated by Doris.
<SF: I have written a simple program in C, using the _isnan function thinking
it would detect the Nan value to replace it by the recommended value in the email from Petar Marinkovic, but to my surprise no Nan is detected by the program.

By the way, the Nan value or values are not detected in the coherence file, but in the complex file with the earth surface phase subtracted. Strangely, the file without earth surface phase subtracted does not seem to have any Nan value in it, at least Snaphu does not detect any anomalous value.

Has anybody tried using a C program to correct the problem? If anybody tried, is it not possible to include this in Doris as an option?

Is it known why these Nan values are generated by Doris?

Best regards

Severino Fernandez
Departamento de Observacion de la Tierra, Teledeteccion y Atmosfera
Instituto Nacional de Tecnica Aeroespacial
28850 Torrejon de Ardoz, Spain
Tel +34916487800
Fax +34916774646
Email severinofer@recol.es, fdezas@inta.es

From Bert.Kampes@dlr.de Mon Mar 14 08:30:43 2005
Date: Tue, 23 Nov 2004 17:09:38 +0100 (CET)
From: Bert Kampes <Bert.Kampes@dlr.de>
To: Gustavo Arciniegas <arciniegas06157@itc.nl>
Cc: Doris software list <doris_users@tudelft.nl>, doris_users@yahoogroups.com
Subject: Re: Envisat ASAR orbits

Dear Gustavo,

Correct. To get precise orbits for ERS or ENVISAT you have to

1) download and install getorb program;
2) download the ODR files (and "arclist" database file, approx. 200MB for all);

See also the link to the orbits at:
http://www.deos.tudelft.nl/ers/precorbs/

The getorb package is automatically called by Doris by running step M_PORBITS or S_PORBITS. In the input file you have to specify a path to the ODR files.

Alternatively you can use the orbital data annotated in the distributed SLC file (mph/sph format for ENVISAT). It is highly recommended to use DELFT precise orbits. If you do so, please acknowledge usage appropriately in publications (as for using Doris).

Regards,

Bert Kampes
At 16:34, November 23rd, Gustavo Arciniegas wrote:

<GA: How do I obtain orbit information for asar (envisat) data? Do I have to use getorb package?
<GA: Thanks for your help
<GA: Gustavo Arciniegas

Dear Gustavo,

Please send questions to the user list and not directly to me. Refer to the manual for this question; also see the template input files. I added a search option to the website, so that hopefully the manual can be searched for such index terms (does not seem to work yet, maybe google first needs to index the enterprise site?) Please use an appropriate subject of your emails so it becomes more useful in the FAQ.

Anyway,
http://enterprise.lr.tudelft.nl/doris/usermanual/node11.html currently shows the general input cards: DUMPBASELINE allows you to compute baseline parameters on a user specified grid. Step COARSE_CORR_ORBIT dumps some values for the center of the crop to the interferogram result file. (Both require the orbits to be in the master and slave result file, so run steps READFILES, and advisably PRECISE_ORBITS first.) Setting SCREEN to DEBUG level may show some more intermediate parameters that could be useful to you.

Bert Kampes
Subject: Re: [doris_users] unwrap problem

Dear Luo,

<lx: NaN or infinity found in correlation data

The problem is that there are Not-a-Numbers in the coherence data file. The answer is to remove the NaNs from the coherence file, or do not use coherence during unwrapping (try this!). I thought this issues was fixed in Doris v3.12. See previous posts in archive. See manual (ps version). Running snaphu by hand may help you further, the command is logged as INFO or DEBUG level, and I may have written it in the snaphu conf file and source code.

Is your slave larger than your master?
Did you use RS_DBOW card?
Can you speculate what the reason for the NaNs could be?

Are you sure you use the latest version of Doris? Maybe the fixed code is not released yet, I cannot find it at the status page: http://enterprise.lr.tudelft.nl/doris/status.html although I am sure at home I resolved the issue. Check the code for routine compcoherence, there is a line that Doris checks for zero division, which I forgot for the first block. There may be a post in the FAQ by me telling about this, with the recommendation to change this line in the code, instead of releasing a new version with only such a minor change. Note this check is performed multiple times, and was forgotten the first time. Let me know if this helped you. I will add this info to the status page, or make a new release (I wanted to wait until the COMPREFDEM step was implemented using a new algorithm).

<lx: When I remove the NaN from correlation file and
<lx: re-perform unwrap, the same problem still exists(no

The coherence file (?) still contained NaNs; make sure you remove it, check e.g., with Matlab. How did you remove the NaNs? In what file?

<lx: By the way, when I perform sanphu standalone,
<lx: there is not output file yet.

Please clarify. Basic input is wrapped ifg, output is unwrapped. options are in a conf file. If you use coherence as additional input and there are NaNs in that file, sanphu crashes and no output results. If it works, you will have to create the Doris result file unwrap section yourself.

<lx: What I am very doubt of is that there is no
<lx: unwrapped phase file. So I can’t process furthermore
<lx: such as SLANT2H, GEOCODE and DINSAR.

Correct, you have to resolve this issue first and make sure the unwrapped result is reasonable by checking the files.

Bert Kampes

+++ At 20:01, December 17th, Luo Xiaojun wrote:

<lx:
<lx: Dear users:
As if there is a problem when I try to unwrap the interferogram with snaphu 1.4.2 through doris 3.12. The messages emerge on screen are as following:

```
PROGRESS: Interpretation input options file finished.
total cpu: 0 min 0.03 sec
```

```
PROGRESS: Finished initialization
PROGRESS: Orbit: interpolation coefficients computed.
PROGRESS: Orbit: interpolation coefficients computed.
PROGRESS: Computing integration constant based on tiepoint
PROGRESS: Start UNWRAP.
PROGRESS: Configuration file for snaphu: snaphu.conf created.
NaN or infinity found in correlation data
Abort
PROGRESS: Finished snaphu_unwrap.
PROGRESS: Finished UNWRAP.
total cpu: 0 min 0.11 sec
```

```
WARNING : File: Outdata/unwrp1096.uint does not seem to exist (may not be a problem).
PROGRESS: calling preview for unwrapped interferogram
```

```
...Why did the blond stare at the orange juice?
...it said concentrate.
```

```
--- WARNING SUMMARY ---
There was 1 message:
1: File: Outdata/unwrp1096.uint does not seem to exist (may not be a problem).
```

When I remove the NaN from correlation file and re-perform unwrap, the same problem still exists (no output file is resulted).

What I am very doubt of is that there is no unwrapped phase file. So I can’t process furthermore such as SLANT2H, GEOCODE and DINSAR.

By the way, when I perform sanphu standalone, there is not output file yet.

Why ????????

Will you help me please?

Thanks alot and Merry Christmas.

yours

Xiaojun Luo
From Bert.Kampes@dlr.de Mon Mar 14 08:42:53 2005  
Date: Tue, 16 Mar 2004 08:21:39 +0100 (CET)  
From: Bert Kampes <Bert.Kampes@dlr.de>  
To: Ziyadin Cakir <ziyadin.cakir@eost.u-strasbg.fr>  
Cc: Doris software list <doris_users@geo.tudelft.nl>,  
edoris_users@yahoogroups.com  
Subject: Re: cygwin cpxfiddle float bug?

+++ At 21:01, March 15th, Ziyadin Cakir wrote:

<ZC:
<ZC: Dear Doris users,
<ZC: Has anybody had a problem of writing float data using
<ZC: cpxfiddle running on cygwin? Ascii output is fine and
<ZC: correct. But I cannot get a float file with a correct
<ZC: size.
<ZC: Any idea?
<ZC:
<ZC: Cheers,
<ZC: Ziya
<ZC:

Dear Ziya,

I create a zero binary "complex float" input file named "out.fcpx"
of width 4 (pixels), height 10:

kampes@capone[8:13]: dd if=/dev/zero of=out.fcpx count=80 bs=4
kampes@capone[8:13]: ls -l out.fcpx
-rw-r--r-- 1 kampes MFAP 320 2004-03-16 08:13 out.fcpx

(size 320B=4*10* 2*4 =width*height*bytespercomplexpixel: is correct)
Then with cpxfiddle I printed the amplitude to screen

kampes@capone[8:13]: cpxfiddle -w 4 -o ascii -q mag out.fcpx

kampes@capone[8:13]: cpxfiddle -w 4 -o ascii -q mag out.fcpx
(©)Doris software, $Revision: 1.1 $, $Author: kampes $
cpxfiddle: INFO: Number of output pixels: 4
 0 0 0 0
 0 0 0 0
 0 0 0 0
 0 0 0 0
 0 0 0 0
 0 0 0 0
 0 0 0 0
 0 0 0 0
 0 0 0 0
 0 0 0 0

Ziya,
Then I create the binary float output and redirected it to a file "out.r4":

```bash
kampes@capone[8:13]: cpxfiddle -w 4 -o float -q mag out.fcpx > out.r4
kampes@capone[8:13]: ls -l out.r4
-rw-r--r-- 1 kampes MFAP 160 2004-03-16 08:17 out.r4
```

This is as expected, i.e., 160B.

If you get another number, maybe text send to stderr is redirected somehow to the file, for example, if you use cpxfiddle -w 4 -o float -q mag out.fcpx >& out.r4 this would cause the strings:

```bash
@(#)Doris software, $Revision: 1.1 $, $Author: kampes $
cpxfiddle: INFO: Number of output pixels: 4
```

To be in the created output file.

Please test the above and let me/the list know.

Best regards,

Bert Kampes

---

From Bert.Kampes@dlr.de Mon Mar 14 08:42:36 2005
Date: Mon, 15 Mar 2004 08:22:06 +0100 (CET)
From: Bert Kampes <Bert.Kampes@dlr.de>
To: Dr.Y.S. Rao <ysrao@care.iitb.ac.in>
Cc: Doris software list <doris_users@geo.tudelft.nl>,
doris_users@yahooogroups.com
Subject: Re: bam earthquake (cookbook)

Dear Dr. Rao,

Doris can handle any DEM if it is supplied in the coordinate system of the orbits. Limitations are the memory usage and speed of the algorithm as implemented in Doris.

As far as I know, the SRTM C-band data set is distributed in WGS84, and in same or similar format as GTUP030. Simply download it, create input cards, and try it? Admittedly I have not used SRTM data with Doris myself.

Concerning differences when using external DEM (first thoughts):
- in topo pair there is also noise, possible displacement, and atmospheric signal.
- ratio Bperp_defo/Bperp_topo defines how this scales after subtraction.
  Here: 500:500 ---> 120 deg of (noise+atmo) phase thus causes error of 120 deg error in differential interferogram.
- unwrapping errors will propagate to DEM error, thus to dem correction.
  An error of 2pi in unwrapped DEM maybe is not so bad in this case since the baseline is equal for both images. I guess even when I had not unwrapped the topo-pair it would have worked for identical baselines (this case).
- errors in the SRTM DEM also propagate of course, somehow. The heightambiguity is something like 20 meters. Thus error of 10 meter causes half a fringe error?
- coregistration of DEM from external is not supported by default. After computation of a syntethic phase image using SRTM, most likely it is mis-aligned a little bit with the interferogram. The user has to figure out these shift, for example using the shifting cards provided in step subtract ref dem. The topo-pair DEM is aligned automatically in 3 pass, since it has the same master.

Regards,
Bert Kampes

PS. There may be a trend over the displayed result. Using subtr_ref_pha this trend could be taking out, which I had no time for.

+++ At 11:23, March 14th, Dr.Y.S. Rao wrote:

<DR:
<DR: Dear Bert,
<DR: <DR: Many thanks for giving info about the data sets over Bam earthqauke
<DR: region. Suppose that June 11, 2003 data set is not available. In place of
<DR: that if one uses SRTM data, what difference can we expect in differential
<DR: fringes. I want to know whether DORIS supports SRTM data set for doing
<DR: differential interferometry.
<DR: <DR: best regards,
<DR: y.s.rao

From Bert.Kampes@dlr.de Mon Mar 14 08:39:23 2005
Date: Fri, 7 May 2004 08:16:51 +0200 (CEST)
From: Bert Kampes <Bert.Kampes@dlr.de>
To: Markus Roman <slcikkt@hotmail.com>
Cc: Doris software list <doris_users@geo.tudelft.nl>, doris_users@yahoogroups.com
Subject: Re: integration constant unwrapping and geocoding displacement

Dear Markus,

You can easily test this by doing it, so please do... If i was wrong, please report the correct way to do it to the list.

I expect that the input of S2H is unwrapped phase.
and that the output of interest for you are lat/lon values (method ambiguity).
I expect that the output of COMPREFDEM is unwrapped phase.

Thus, what is wrong with DEM-->COMPREFDEM-->S2H?
regardless how these routines are called or what you may think they do,
i don't see a problem with this. Of course, what you have to do is to trick doris
to use the COMPREFDEM output as input for the S2H step. Indeed, do this
in a separate run. The easiest way seems to create a dummy UNWRAP section,
and copy the relevant lines of S2H output to the UNWRAP section. Then run doris
with PROCESS S2H and it will take the unwrapped phase values that you just
created (and set the process control flag).

The geocoding of the data can thus be done without even having the data!
(but based on a DEM). i.e., in your case, simply forget you even have
displacement for a minute, only consider you have a master SLC with a certain
geometry. The question is, where are the range,azimuth pixels in this
geometry located on Earth. Using zero-doppler iteration intersection method
with a known DEM will give you this. The output is then for each pixel in
the interferogram its corresping lat/lon/hei values. Since you already had the
displacement you actually have lat/lon/hei/displ values. You can do with these
things whatever you want, e.g., making 3d models, etc., but that is up to you and cannot be done with Doris.

I would greatly appreciate if you could make a little cookbook describing this method with exactly what you did and how you did it, examples of output files, etc. I think this would be very valuable to many people. I don’t mind putting it up at the doris pages, add it to the manual, or link to it.

Regards,
Bert Kampes

PS. I had not time to look at the formulas you mention. a report is due today.

+++ At 19:49, May 6th, Markus Roman wrote:

<MR: Hi Bert,
<MR: Thanks again for taking the time to answer to my questions.
<MR: The first answer provides a clear approach for adding the phase bias to the unwrapped phase.
<MR: Regarding the second answer: If after (1) and (2) the displacement values are in radians, the method suggested at PS1 and PS2 is correct. However, the assumption seems to be inaccurate. The equation 28.10 (Chapter 28, SLANT2H, p 97 in Doris manual) provides a formula similar with the one in PS1 which converts phase to height values. I also recall one of your previous statements:
<MR: >From your last (...) email I understand that you agree that SLANT2H step >cannot be used if you have defo-phase.
<MR: Could you please clarify this issue?
<MR: I wonder if the right approach will not require running DORIS twice, first with COMPREF_DEM set to get the displacement, and later without COMPREF_DEM to get the height. In the former case we could use SLANT2H and GEOCODE to get the lambda.raw and phi.raw (i.e., the latitude and longitude of each pixel). Then, instead of offering the height values to the script "lonlathei2ascii", we could offer the displacement values (as suggested in http://groups.yahoo.com/group/doris_users/message/303 )
<MR: Markus
<MR:

From Bert.Kampes@dlr.de Mon Mar 14 08:38:59 2005
Date: Thu, 6 May 2004 14:36:47 +0200 (CEST)
From: Bert Kampes <Bert.Kampes@dlr.de>
To: Markus Roman <slcikkt@hotmail.com>
Cc: Doris software list <doris_users@geo.tudelft.nl>,
doris_users@yahoogroups.com
Subject: Re: integration constant unwrapping and geocoding displacement
Having a pair of SAR SLC images and a 30m DEM model, what are the steps following the creation of the unwrapped interferogram, which will conduct to a final geocoded displacement map?

Dear Roman,

It would be great if there would be a step by step example on the web how to do this. Maybe the people in Delft can do this as an exercise for BAM using another set of images?

still remains:

a) correction of the height value
b) geolocation of a displacement map.

a) do this by hand using Matlab.

A good place may be to add a phase bias to the unwrapped phase such that a pixel of known height has the correct unwrapped phase value based on the height ambiguity factor for the interferogram. After you do this, simply continue with doris. Somewhere there should be a couple of postings on the subject.

example:

Suppose your baseline=200 meter. Look interferogram.res what the height ambiguity is, lets say this is h_amb=-47.0. Now assume you have a pixel H and you know that the height of P is h(P)=530 meters. The correct number of wraps is thus: 530/47 = 11.28 cycles. The correct unwrapped phase is thus: phi=2pi* 530/47 = 70.85 rad.

Now look up the pixel in the unwrapped interferogram. suppose it is located at (x,y) and it has the unwrapped value z=2.42423 Then you know that you have to add 70.85-2.42423 to all pixels of the unwrapped ifg.

It may be interesting to have an option with unwrapping to put the mean terrain height in, i.e., that the integration constant is set such that the average of the unwrapped phase corresponds to this, or that one can say pixel x,y has the height H. This is easy enough to implement, and may be very useful, particularly in combination with an interactive clicking tool and a databank of known points for automation.

b) Also, this has been addressed before, but it seems to be a returning topic.

What I quickly found was

- http://groups.yahoo.com/group/doris_users/message/331
- http://groups.yahoo.com/group/doris_users/message/303
- http://www.geo.tudelft.nl/fmr/research/insar/sw/doris/Usersmanual/node121.html

If you cannot assume that the terrain has a constant height, you may want to use an external DEM to geocode the displacement map. I have not done this with Doris before. But in principle it sounds easy enough:
(1)  
**COMPREF_DEM, card CRD_OUT_FILE:**
Computes the DEM in radar coordinates in unwrapped phase values (i think).

2)  
**SLANT2H, method ambiguity, cards S2H_OUT_HEI, S2H_OUT_LAM, S2H_OUT_PHI;**
Transform phase [rad] to height values [m] in range,azimuth pixels.
(also gives you the lat.lon values.
(of course this has been done in (1) as well, but i don’t think there is
output of the dem in radarcoordinates in meters, only in rad.)
**S2H_METHOD ambiguity**
**S2H_OUT_HEI Outdata/hei.ambi**
**S2H_OUT_LAM Outdata/lam.ambi**
**S2H_OUT_PHI Outdata/phi.ambi**

This should be it. alternative you can probably find other ways, or
hack the code. I am aware that the COMPREF_DEM step takes a long time.
We need a faster algorithm, which exist, but someone needs to implement it (not me).
It may be possible to speed it up in the current implementation by setting the
MEMORY card to say, 5000. (set it as high as possible, until you get memoria
culation problems. the value 5000 would mean you have 5GB available, however,
this is not strictly true, it is more a relative card that defines the size of
the buffers used, and I have heard that there is a small error in how this is
done in this step (which is changed for the next release). A good way is thus to
dummy run doris, with MEMORY 1000 set, see with e.g., “top” howmuch memory is actually
used. If you still have memory left, keep increasing the value of the
MEMORY card.)

Best regards,

Bert Kampes

**PS1.** After you have done this, you know your displacement in radians for
triples [lat/lon/defo].
To obtain the displacement in meters, divide by 2pi and multiply with 0.5*wavelength.
\[ \text{defo}_m = \text{defo}_\text{rad} \times (0.028/\pi) \] (for ERS)

**PS2.** This is still in the line of sight to the radar.
To approximately map this to vertical displacemnt, divide this by cos(theta),
(theta is local incidence angle)
\[ \text{defo}_\text{vert}_\text{meters} = \text{defo}_\text{los}_\text{m}/\cos(\theta) \]
\[ = \text{defo}_\text{rad} \times (0.028/6.28) / \cos(0.401) \]
\[ = \text{defo}_\text{rad} \times 0.00489000 \]

(using ERS wavelength=0.0565646, theta=23deg, pi=3.14159)
This can be best done in the scripts using awk, etc.
or in Matlab, depending on your unix skills.
Unfortunately, I don’t think the local incidence angle is computed anywhere.

+++ At 21:40, May 5th, Markus Roman wrote:
Dear Bert,

Thank you for wonderful help and guidance.

While from documentation the process of creating and georeferencing TOPO maps is clear, I still have some questions regarding the process of creating DISP maps. I looked at your BAMS earthquake example and the DORIS setup files for the 3-pass approach are clear. My question specifically addresses the case of using DORIS when you just have one pair of SAR data.

Therefore, here's the question:

Having a pair of SAR SLC images and a 30m DEM model, what are the steps following the creation of the unwrapped interferogram, which will conduct to a final geocoded displacement map?

To obtain the unwrapped interferogram I assume done the:
- Registration, resampling and filtering of the pair of SAR images
- Creation of interferogram
- Flattening with orbits
- Simulation of interferogram from DEM
- Registration of simulated interferogram
- Flattening with simulated phase
- Filtering of flattened interferogram
- Offset adjustment for the unwrapped phase (as suggested in your previous email message)

So, what is (are) the next step(s)? I assume that lat/long information is available via the satellite ephemeris's data. I suspect a step in which pairs of lat/long are calculated and associated to each phase deformation value. Can somehow DORIS do that, or is there an utility available to do this function?

It has been a long day... Looking forward to your reply.

Thanks again. With best regards

Markus

Date: Wed, 05 May 2004 23:22:02 -0500
From: Markus Roman <slcikkt@hotmail.com>
To: Bert.Kampes@dlr.de
Subject: addendum to my previous message

Dear Bert,

I am starting to get quite familiar with the software however two problems still remains:
- Correction of the height value
- Geolocation of a displacement map.

Based on your previous answers and documentation (I just reviewed some of your messages) I see a good connection between these problems (yes, it would be wrong applying slant2height and geocode procedures to the deformation values). My interest is in obtaining displacement maps from a pair of SAR data therefore I could not use 3 or 4 ways approaches. I also have a good USGS 30m DEM for the area.
Dear Mahdi,

The SRTM C-band data apparently use as a vertical datum the EGM gravity model. This is thus the geoid, not the ellipsoid which is logically used by Doris.

See e.g.,
http://www.personal.psu.edu/users/b/e/beal121/Project3.htm

For the BAM area, this very well could imply a practically constant offset between the geoid and ellipsoid heights. This constant offset translates to shifted fringes.

Hope this clarifies it. If you find software to deal with this, e.g., to evaluate the EGM model then I would appreciate it if you could share this with the other users.

Best regards,

Bert Kampes

+++ At 10:16, January 5th, Mahdi Motagh wrote:

MM: I have processed the BAM data set in both two-pass and three-pass method. In the two-pass method I used 3-arc second SRTM as my topography reference. The result is more or less the same except for the geocoded one.
MM: Indeed, in the two-pass method the geocoded fringes are shifted a bit to the right compared with the corresponding ones in the three-pass method. This certainly affects the result of inversion when we want to obtain information regarding the dislocation behind this earthquake. I do not know whether this problem is due to the SRTM data or not.
MM: I was wondering if any of you had any experience in this regard.
MM: Best wishes
MM: Mahdi
Dear Mahdi,

I think this relates directly to the integration constant after unwrapping. For deformation measurements, I would say the observations are relative. If I understand you correctly, "shifted fringes" relates to the wrapped phase. In my opinion, one can add an arbitrary bias to the wrapped phase, consider it an absolute signal delay, or absolute orbit error. If you agree this is allowed, then this means that I can "shift" the wrapped fringes by adding 0.5 pi (for example). It also means that I can shift the fringes on top of eachother in this way.

But maybe better; if you unwrap both phase images then it is easy. Assume that in the far field there is no deformation (or a known amount) and correct the unwrapped phase so that for the same pixel at the edge of the image the unwrapped phase is zero. Then you force that both have the same reference. The way you do it, i.e., straightforward interferogram generation and look at the phase, you cannot compare them in an absolute sense (it is logical that the fringes are shifted).

Regarding the effect on inversion, I guess you are right. Maybe you can explain a little bit how it works because I never actually did it. Do you simulate a (lot of) wrapped interferogram(s) and then compare/fit with the observed wrapped data? (if so, why not with the unwrapped data?) Indeed there would be an effect on the estimated parameters. But I think it should always be the case that the phase is corrected somehow based on some apriori information. It would be very surprising to me if this is not common practice. Maybe using 3 pass there is no need for it normally? But still, the measurements are inherently relative, so somehow you have to choose a reference, e.g., a pixel with known position/deformation. I would be interested in hearing how you do it and what the common approaches are.

I have no experience with SRTM C-band data with Doris. Maybe other people can comment on absolute accuracy of it (i.e., if the DEM would be 10 meter to low as a bias, the result is a shift of the fringes you see!) For a 500 meter baseline, the height ambiguity is ~20 meter, thus even a 5 meter bias (within SRTM specs) would "shift the fringes" by 0.25 fr.

Best regards,

Bert Kampes
+++ At 10:16, January 5th, Mahdi Motagh wrote:

<MM: I have processed the BAM data set in both two-pass and three-pass method. In the two-pass method I used 3-arc second SRTM as my topography reference. The result is more or less the same except for the geocoded one. Indeed, in the two-pass method the geocoded fringes are shifted a bit to the right compared with the corresponding ones in the three-pass method. This certainly affects the result of inversion when we want to obtain information regarding the dislocation behind this earthquake. I do not know whether this problem is due to the SRTM data or not. I was wondering if any of you had any experience in this regard.

<MM: Best wishes
<MM: Mahdi
<MM:

From Bert.Kampes@dlr.de Mon Mar 14 08:28:36 2005
Date: Mon, 27 Dec 2004 17:23:10 +0100 (CET)
From: Bert Kampes <Bert.Kampes@dlr.de>
To: maryam rahnemon <rahnemon_far@yahoo.com>
Cc: Doris software list <doris_users@tudelft.nl>, doris_users@yahoogroups.com
Subject: Re: [doris_users] Accuracy of DEM

Dear Maryam,

1-Is there any method that be independent from phase unwrapping method?

1 - The only way that may be independent of phase unwrapping I can think of at the moment are (i) do not do phase unwrapping (but you will have to compare the interferogram then, not the DEM) (ii) use the exact same phase unwrapping algorithm and take the difference of the results. I understand this is not what you want, but all other ways are dependent on the unwrapping method. However, I think this is not a big issue. Simply, as suggested before, set the same pixel to the same value in both DEM’s.

Precision is a relative measure, Accuracy is an absolute measure. At least that is how I understand it. For example, orbit of satellite may be very precise, but not accurate. That means that the absolute position of the orbit is wrong, say 10 meters, but the difference between a point at time t=0 and t=1 seconds is really well known.

Since unwrapping is integration there is an unknown integration constant. This is what I called a bias. This bias cannot be determined without using ground control points (GCP). An unwrapping algorithm starts at a certain seed location. There it is assumed that the true unwrapped phase value is the observed wrapped value. Then in principle the unwrapped phase of the neighboring pixels is found by adding the wrapped phase difference (i.e., integrate along a line). (Note that the trick with unwrapping is however to find the most likely path based on assumption of smoothness.) You see, it depends where you start what the unwrapped phase is. If I start at the left, the unwrapped phase at the right may be, say 100, and if I start at the right the phase at the left becomes -100. It is all relative. Interferometry thus is very precise, but not accurate.
I guess it depends on the definition of these terms.

<mr: 3-According to "Accuracy of Topographic Maps Derived from ERS-1 Interferometric Radar" by Zebker, I would suggest following this paper and not get confused about accuracy vs. precision. Probably this paper is exactly what you need!

<mr: 2-How can I compute unwrapping error of snaphu method?

If you could, you would add the error to the result and obtain an error free solution! You have to assume unwrapping went correct. For some algorithms there are masks but I think snaphu does not give you this. Download some papers from Curtis Chen about snaphu, and read his thesis. E.g., via http://www.stanford.edu/group/radar/group.html

Sometimes a region is unwrapped incorrectly. You can often see it with your eyes, but it is not that easy for a computer apparently. What you see is in the DEM a region that has a multiple of the height ambiguity a deviating height. For example, an island of the coast. There is no way the height of the island is correct (since the unwrapping cannot cross the random phase in the sea.) Only if you have a pixel of which you know the height on the island (a GCP) you can compute the bias and "lift" the island to the correct height. This can also happen if you have a river, or an incoherent region.

<mr: 4- How many GCPs are required for removing all the errors in DEM (including orbit error)? You do not per se remove the error but make it smaller by modeling it as a bias plus linear trend in range and azimuth. This are three parameters. To estimate them, you need at least three heights at three locations. They should have a good geometric distribution and the more the better (e.g. fit on an existing low quality DEM like gtopo30 or globe).

Best regards,

Bert Kampes

+++ At 02:16, December 27th, maryam rahnemon wrote:

<mr: 2-How can I compute unwrapping error of snaphu method? And how can I sure that this is a reliable
<mr: 3-According to "Accuracy of Topographic Maps Derived from ERS-1 Interferometric Radar" by Zebker, Could you please introduce me a paper for computing them? (We unfortunately haven\'t the book of
4- How many GCPs are required for removing all the errors in DEM (including orbit error)?

Best regards,

Maryam.

PS. In previous mail lambda/phi was longitude/latitude. Sorry for this terminology.

Bert Kampes <Bert.Kampes@dlr.de> wrote:

Dear Maryam,

For error propagation of phase error to height error see, e.g.,
the book by Ramon Hanssen. In principle, using a single interferogram
with a perpendicular baseline Bperp, the height of ambiguity is given as
\[ H_{\text{amb}} = -\frac{\lambda}{2} \frac{r \sin(\theta)}{B_{\text{perp}}} \]
(for ERS, about 100 meter baseline, 100 meter height ambiguity,
200 m baseline, 50 meter \( H_{\text{amb}} \), etc. reciprocal relation).

Thus, an unwrapping error of 2\( \pi \) exactly gives this error in the DEM.

If you can assume there are no unwrapping errors, then
A phase uncertainty with, say, standard deviation is 0.2 fringe,
propagates thus to a height uncertainty of 0.2*\( H_{\text{amb}} \).

This is precision. Accuracy is infinitely bad, since InSAR is relative.
Using more images I am not sure how error propagates in simple words.

Specifics to Doris I do not know regarding this point, except that
no ground control points are used to somehow control the error. Also,
note that the DEM by Doris is relative, not absolute. i.e., you can add
any bias to it, values are all relative to eachother.

I did not understand your table but saw values around 900 and around 50.
I am assuming 900 is the "true" value and 50 is the "doris" value.
In my opinion, all that InSAR can give you w/o apriori information is
that this difference should be constant (it gives you the "shape"
of the topography, if you like that term.)

If you want to obtain a DEM that is correct in an absolute sense, have a look at the
tiepoint card. Identify the tiepoint in the radar coordinates. After
unwrapping, add the phase bias to the whole unwrapped interferogram and
continue as normal. (phase bias is difference between value for that pixel
after unwrapping and what tiepoint reports as INFO.)

You will have to add this bias by hand (e.g., use Matlab). The simplest way is probably
to do it at the end. Even simpler, for the points you have, compute the mean difference between
doris and your reference. Say this is 900-50=850m. Then to make the result
for Doris absolute, add this bias to the DEM, i.e., simply add 850 meter to it.
This is identical to the "complicated" way described above. If you want to
remove effects due to orbit errors too, do something like the following in
Matlab. 1) load the original DEM by doris. 2) load the reference DEM.
3) somehow put them in the same grid by interpolation, e.g., using Matlab or GMT.
4) find parameters to best fitting transformation (e.g. least-squares norm) of the Doris DEM to the reference DEM accounting for a offset, a linear trend in latitude and a trend in longitude (three parameter fit). 5) apply transformation to Doris DEM.

What you do in this way is to use the long wavelength information of the reference DEM to remove these errors from the Doris DEM, but you keep the short wavelength information InSAR gives you.

I am sure this email may not be completely clear. I hope you can use it however to figure out yourself what to do for your purpose. These things are currently not automated by Doris and requires e.g. Matlab. The script latlon2qscii, etc. can be easily edited to add a bias to the DEM values using awk. If I remember correctly, there are some comments in one of these scripts how I did this for the Las Vegas DEM created for the fringe2003 presentation using ENVISAT data.

Best regards,

Bert Kampes

PS. lambda I use for wavelength, which is assumed constant for each sensor. phi is for (unwrapped) phase. I think you may have reported latitude/longitude which are irrelevant?
memory consumption a lot (since Doris reads data line-by-line, a shorter line length (width) reduces RAM requirements maybe).

If you could try to use 10x2, maybe CRD_DENSE 1 is acceptable and gives you some results you could work with.

*Ia:* I also try to use the interpolated SRTM (so as no NODATA pixels are present but it had the same resolution) but it didn’t help.

Interpolation of the input DEM, e.g., with GMT surface program, or a Matlab or IDL function to say 25x25 meter resolution could reduce the need for Doris to use a higher CRD_DENSE value. I mean interpolation to a denser grid, not to fill NODATA values.

Best regards,

Bert Kampes

PS.
*Ia:* I have a problem with the step COMPREFDEM. It always ends up with SIGSEV. I do not know what SIGSEV signals.

+++ At 08:32, October 7th, Ivana apkov wrote:

*Ia:* I have a problem with the step COMPREFDEM. It always ends up with SIGSEV.
*Ia:* I use SRTM-3 (100m resolution) with the ERS data (approx. 20 by 4 m resolution). That’s why I don’t use CRD_DENSE 1, as recommended before.
*Ia:* CRD_DENSE 1 made an image with many zeros and just few usable points.
*Ia:* I also try to use the interpolated SRTM (so as no NODATA pixels are present but it had the same resolution) but it didn’t help.
*Ia:* I also tried to reduce the buffermmsize variable in the radarcodedem function, also tried to reduce the memory in a general card but it didn’t help. When running the step with the same amount of memory, it falls down at the same time - at the beginning of buffer 3.
*Ia:* My cards and the end of the error stream are listed below.
*Ia:* Thanks in advance,
*Ia:* Ivana.
*Ia:* SCREEN debug // level of output to standard out
Dear Luo,

I assume you have read previous posts on this issue. Specific answer to your question

\(<1>\): a. How to calculate the numbers of buffers before compute the reference dem?

is to look in the code. The RAM specified by MEMORY is divided over a few large
matrices. Depending on resolution of DEM, oversampling, multilooking, etc.
See the code.

\(<1>\): b. Why are there the above warning messages? Will they affect the result?
see the manual, implementation section and the code. currently:
http://enterprise.lr.tudelft.nl/doris/usermanual/node93.html
i.e., the DEM is oversampled to a certain resolution, say 10x10m when the
interferogram has 20mx20m. Then these values are radar coded (see appendix of
manual) and the nearest neighbor interpolator is used in that domain to obtain
values at the integer rg,az grid. The warnings mean that the nearest neighbor
was not within a "cell". Of course results are not optimal in that case. But I guess
it is not that important, since I suspect that the resolution of the DEM you use
is much worse than that of the interferogram grid anyway, i.e., topographic signal
is approximate anyway (bilinear interpolation of DEM used by Doris is not very realistic
for 1000m DEM grid spacing).

A smaller area, larger multilooking (interferogram resolution), larger MEMORY card
usage may help you run it faster, decrease DENSE (but use an oversampled DEM using
GMT, matlab or so).

Regards,
Bert Kampes

PS. please use appropriate subject for your questions.

+++ At 10:28, December 15th, lxj7495 wrote:

Dear users:
Now I come across the following problems. Could you help me? Note: my doris version is DorisV3.8.
When I try to compute and subtract the reference DEM with gtopo30 DEM, there is many buffers
"WARNING: No nearest for pixel at:1559,5132 (i.e. 8924,5166)". At last the perform fails when it compute
"/usr/local/bin/run: line 1315: 6698 abort". So I am doubt of the two problems:
a. How to calculate the numbers of buffers before compute the reference dem?
b. Why are there the above warning messages? Will they affect the result?
Waiting for your help.
Good luck with your work.

Luo Xiaojun
From Bert.Kampes@dlr.de Mon Mar 14 08:28:05 2005
Date: Sun, 26 Dec 2004 02:13:19 +0100 (CET)
From: Bert Kampes <Bert.Kampes@dlr.de>
To: maryam rahnemon <rahnemon_far@yahoo.com>
Cc: Doris software list <doris_users@tudelft.nl>, doris_users@yahooogroups.com
Subject: Re: [doris_users] Accuracy of DEM

Dear Maryam,

For error propagation of phase error to height error see, e.g.,
the book by Ramon Hanssen. In principle, using a single interferogram
with a perpendicular baseline Bperp, the height of ambiguity is given as
$$H_{amb} = -\frac{\lambda}{2} \cdot \frac{r \cdot \sin(\theta)}{B_{perp}}$$
(for ERS, about 100 meter baseline, 100 meter height ambiguity,
200 m baseline, 50 meter H_amb, etc. reciprocal relation).

Thus, an unwrapping error of 2\pi exactly gives this error in the DEM.
If you can assume there are no unwrapping errors, then
a phase uncertainty with, say, standard deviation is 0.2 fringe,
propagates thus to a height uncertainty of 0.2*H_amb.

This is precision. Accuracy is infinitely bad, since InSAR is relative.
Using more images I am not sure how error propagates in simple words.

Specifics to Doris I do not know regarding this point, except that
no ground control points are used to somehow control the error. Also,
ote that the DEM by Doris is relative, not absolute. i.e., you can add
any bias to it, values are all relative to eachother.

I did not understand your table but saw values around 900 and around 50.
I am assuming 900 is the "true" value and 50 is the "doris" value.
In my opinion, all that InSAR can give you w/o apriori information is
that this difference should be constant (it gives you the "shape"
of the topography, if you like that term.)

If you want to obtain a DEM that is correct in an absolute sense, have a look at the
tiepoint card. Identify the tiepoint in the radar coordinates. After
unwrapping, add the phase bias to the whole unwrapped interferogram
and continue as normal. (phase bias is difference between value for that pixel
after unwrapping and what tiepoint reports as INFO.)
You will have to add this bias by hand (e.g., use Matlab). The simplest way is probably
to do it at the end. Even simpler, for the points you have, compute the mean difference between
doris and your reference. Say this is 900-50=850m. Then to make the result
for Doris absolute, add this bias to the DEM, i.e., simply add 850 meter to it.
This is identical to the "complicated" way described above. If you want to
remove effects due to orbit errors too, do something like the following in
Matlab. 1) load the original DEM by doris. 2) load the reference DEM.
3) somehow put them in the same grid by interpolation, e.g., using Matlab
or GMT. 4) find parameters to best fitting transformation (e.g. least-squares norm)
of the Doris DEM to the reference DEM accounting for a offset, a linear trend in latitude
and a trend in longitude (three parameter fit). 5) apply transformation to Doris DEM.
What you do in this way is to use the long wavelength information of the reference DEM
to remove these errors from the Doris DEM, but you keep the short wavelength information
InSAR gives you.

I am sure this email may not be completely clear. I hope you can use it however
to figure out yourself what to do for your purpose. These things are currently
not automated by Doris and requires e.g. Matlab. The script latlon2qsci1,
etc. can be easily edited to add a bias to the DEM values using awk. If I remember
correctly, there are some comments in one of these scripts how I did this for the
Las Vegas DEM created for the fringe2003 presentation using ENVISAT data.

Best regards,

Bert Kampes

PS. lambda I use for wavelength, which is assumed constant for each sensor.
phi is for (unwrapped) phase. I think you may have reported latitude/longitude
which are irrelevant?

+++ At 03:01, December 25th, maryam rahnemon wrote:

<mr:
<mr:
<mr: Dear Bert,
<mr:
<mr:
<mr: Could you please let me know what the accuracy of DEM generated by Doris is?
<mr:
<mr: I know it depends to various parameters. However I used the same input file that you have used for
<mr:
<mr: The results of comparison for some points are:
<mr:
<mr:
<mr: Lambda
<mr:
<mr:
<mr: Phi
<mr:
<mr:
<mr: Height by Doris
<mr:
<mr: Real Height
E.5 Step snaphu unwrapping

From Tim.Wright@earth.ox.ac.uk Wed Mar 16 09:39:59 2005
Date: Fri, 21 Jan 2005 17:27:49 +0000 (GMT)
From: Tim Wright <Tim.Wright@earth.ox.ac.uk>
To: doris_users@tudelft.nl
Subject: snaphu experience

++---------------------------------+
| Doris Listserver                 |
| (Delft Object-oriented Radar Interferometric software) |
|                                 |
++---------------------------------+

Hi

I thought I would share some experience with snaphu.

It will unwrap every pixel, including areas with zero coherence. This is bad (it produces unwrapping errors and wastes cpu time).

A fix for this is:

1. Create a masked coherence file to use as the weights: e.g. create a file with 0's for coherence < 0.2 and regular coherence elsewhere.

2. With this mask file, set the complex values (real and imag parts, or amplitude and phase) to zero.

3. Unwrap using snaphu as usual.

4. It will still unwrap the areas with 0 amplitude and phase, but the results will not influence the areas with good coherence. So as a final step you need to mask the unwrapped result file with the coherence file.
For the Bam area (preseismic pair with ~500 m baseline) this procedure makes a much nicer unwrapped image (essentially no errors) AND runs almost twice as fast.

Hope this is helpful information. I will be interested to hear if it works in other areas.

Cheers

Tim

*********************************************************************
Tim J Wright
Royal Society University Research Fellow
Department of Earth Sciences
Parks Road, Oxford, OX1 3PR
tel : +44 (0)1865 272068 [fax:272072]
email : tim.wright@earth.ox.ac.uk
*********************************************************************

From Bert.Kampes@dlr.de Mon Mar 14 08:25:10 2005
Date: Wed, 5 Jan 2005 10:14:18 +0100 (CET)
From: Bert Kampes <Bert.Kampes@dlr.de>
To: Kinshuk Kulshreshtha <kkulshreshtha@yahoo.com>
Cc: Doris software list <doris_users@tudelft.nl>, doris_users@yahoogroups.com
Subject: Re: Phase unwrapping with Snaphu

Dear Kinshuk,

The advantage in my opinion is that snaphu gives good results, it is the best. Complaints users commonly report is that snaphu is slow for larger files. Moreover, stronger phase filtering, more multilooking(!) will reduce processing times. At most I think it is about 24h for a difficult large interferogram on a 1GHz. There exist faster algorithms, like Zebker’s residue branch-cut or least-squares. For example, the least-squares may be faster, but as a user you should not be happy with it. Only if there are no residues it works, i.e., it simply is integration along any path. I discovered that pressing CTRL-C in snaphu dumps the current solution. So, if you are impatient, press ctrl-c and continue with Doris. I think that the parts of the interferogram that are easy to unwrap are then correct (i.e., wont change in later iterations) and noisy parts may not be optimal, but maybe they cannot be unwrapped correctly anyway. But to be sure, better wait.
I would like to report my experiences, and would like to ask other users to do the same:

<table>
<thead>
<tr>
<th>test area size</th>
<th>coh</th>
<th>fringes</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>BAM topo [2567,2577]</td>
<td>good, some areas worse</td>
<td>quite a lot</td>
<td>took quite long</td>
</tr>
<tr>
<td>BAM topo [1000,1000]</td>
<td>good, some areas worse</td>
<td>quite a lot (filtered)</td>
<td>1h:15m</td>
</tr>
<tr>
<td>SRTM ~[15000,5000]</td>
<td>reasonable, steep topo</td>
<td>a lot (filtered)</td>
<td>24H</td>
</tr>
</tbody>
</table>

I used snaphu log:
Elapsed processor time: 1:14:16.71

As you can see, I do not have much experience with it.

Hopefully some people are willing to do some experiments changing the input file regarding size (multilook with e.g., cpxfiddle or doris), filtering (no filt, phase filt, etc.), and snaphu options (e.g., change sign of Bperp, initialization method, using from command line repeatedly, e.g.:

snaphu -f snaphu.conf Outdata/9192_6687.cint.filtered 1285

and changing the file snaphu.conf

Best regards,

Bert Kampes

+++ At 20:13, January 4th, Kinshuk Kulshreshtha wrote:

<KK: Hi
<KK: Can anyone please tell what are the limitations and drawbacks of using snaphu for phase unwrapping
<KK: Regards
<KK: Kinshuk
<KK:

E.6 Step coherence

---

From Bert.Kampes@dlr.de Mon Mar 14 08:27:02 2005
Date: Fri, 3 Dec 2004 08:55:21 +0100 (CET)
From: Bert Kampes <Bert.Kampes@dlr.de>
To: Gustavo Arciniegas <arciniegas06157@itc.nl>
Cc: Doris software list <doris_users@tudelft.nl>, doris_users@yahoogroups.com
Subject: Re: coherence computation
Dear Gustavo,

Geocoding has been addressed, see FAQ, question of Zbigniew Perski, answer by me, I think. Otherwise, simply take the WGS84 corner coordinates from the log file/stdout and stretch the images. If your area is flat and your desired resolution is ca. 100x100 m, this is probably fine and simple to do.

I would be very careful with the coherence computed by Doris for this purpose. There is no correction for (i) the expected bias due to taking a finite number of samples (ii) signal.

Ad (i): You can take a large estimator window using COH_WINSIZE 64 64 to counter this at the cost of resolution, still use COH_MULTILOOK 10 2 if desired. alternative, subtract some value of the computed coherence to counter this. This will be the same bias for the whole image. (i.e., if you take only a few samples, the computed coherence will be too high)
However the variance of the bias is also higher.

Ad (ii): You will have to implement some answer to this. the problem is that if the data you use for coherence computation contain, e.g., a phase ramp due to signal, e.g., topography, the computed coherence for a window 64x64 will be affected by this. One way is to include in the coh. computation routine a phase detrend-ing for each patch of COH_WINSIZE, which may take a long time and may not be reliable. You can take a smaller COH_WINSIZE, which has Ad (i) problem. Or you can offer to Doris a file corrected for as much signal as possible. I guess this is the best solution for you. I.e., in principle the input is M,S,R where M is the complex master image, S the slave and R the polynomial for the reference phase. Instead of offering S, it should be possible to correct S for a DEM somehow = S', and save that new file under a name of which Doris thinks it is S. Running COHERENCE then will cause Doris to compute coh = [M*conj(S')*conj(R)] / [abs(M)abs(S')] or something like it. This is possible, but not really easy, since how to obtain matrix S'? Probably this requires some manipulation. Therefore, easier would be to add an additional input to Doris for a matrix with correction phase, C, then to change the coherence routine that if that matrix is specified, that coh = [M*conj(S)*conj(R)*conj(C)] / [abs(M)abs(S')]
should be computed. This phase matrix I think you can output using step COMPREFDEM for the topography.

The interferogram shows you how what to expect for this bias (what signal phase slopes). Thus, an option could be to filter the interferogram using an spatial averaging kernel, then to take the difference with the original which gives you the high-pass part (the noise) which gives you the coherence. I have no idea if this method has been proposed before, but it seems to make sense to me.

Write some scripts in Matlab with this in mind to show these effects may be very helpful for you and could be an interesting topic for a paper for you.

If your topographic signal and displacement signal is small, ad(ii) is less severe (In your thesis, anticipate putting some examples studying these effects).

Alternatively, select you study area, or region of interest such that no displacement and no topography is there, also use a small Bperp.

In literature other methods are proposed as well, but that is more your field, apparently.

In the latest code I fixed some small bugs regarding this. I am not sure if they are in the latest released version yet (check file "newsince...".) Doris has not been written for absolute coherence computations, it is more intented as a check for coregistration.

I noticed some effects after adaptive range filtering that the computed coherence was "blocked", probably due to abrupt changes of the filtered spectrum. I would not use this computed coherence (the phase I think is OK however).

Do use azimuth and range filtering before computing the coherence.

For the BAM earthquake data, interesting results where presented by Joern Hoffmann and Eric Fielding (among others) about this at ENVISAT/ERS symp. Salzburg 2004.

Regards,

Bert Kampes

PS. the ps version of the manual may be more updated than the online on. Please print this one out for your reference.

PS. <QA: In you paper "RADAR INTERFEROMETRY WITH PUBLIC DOMAIN TOOLS" I read that <QA: the coherence image gets automatically georeferenced; however, I would <QA: like to be sure about how to do this.

I don’t think so. If that is in the paper, it is incorrect. see the FAQ. there is a trick that when you know the mapping of radar coordinates to geo-coordinates you can transform of course an arbitrary matrix (image). I think that I may have tried to say that.

PS. please use approriate subject of emails for saerching the FAQ.

PS. I will be out of the office until Jan 2, 2005. Happy new year!
Dear Gustavo,

The coherence image is the same dimensions as the interferogram by default as far as I know in the run scripts generated templates. See manual currently at:
http://enterprise.lr.tudelft.nl/doris/usermanual/node87.html
also see the implementation section. Note that figures that are not displayed online are visible in the postscript/pdf version.

The output size is related to the COH_MULTILOOK card. using "1 1" should give full resolution. The estimator window size is given by the COH_WINSIZE card. This is the number of pixels used during the estimation of the coherence. Of course, using "1 1" for multilook and say "10 10" for estimator window size means the estimated coherence is correlated, i.e., the estimated coherence of neighboring pixels is not independent since partly the same pixels are used. The best would be to use an infinite amount of pixels for the estimator window. But this has two drawbacks:
1-a systematic phase is not corrected for. Thus, larger windows may lead to worse estimated coherence. Note that the ellipsoid reference phase is corrected for, if it is computed before doing the coherence.
2-"resolution" of coherence map decreases note that coherence is not corrected for bias, e.g., if 3x15 pixels are used to estimated the coherence you expected a too large estimate with maybe 0.15 or so.

I think this question has been addressed before, so I summarized the main points here. In any case, if you want to use the coherence computed by Doris for something useful, very likely you have to re-write the code.

Best regards,

Bert Kampes

+++ At 10:15, January 27th, Gustavo Arciniegas wrote:

<GA: Happy new year,
<GA: I am computing coherence images from Envisat ASAR data and I have the following question: How can I obtain a coherence image at the best resolution, with the less noise? How can I modify the script (parameters) that generates this image? Or this (changing parameters) must be done in the COHERENCE step?
<GA: The coherence map that Doris computes of a scene by default is a rather small image, compared to the size of a the original data, or a computed interferogram.
<GA: Thanks for your answer,
<GA: Gustavo
E.7 Visualization of results

From Bert.Kampes@dlr.de  Mon Mar 14 08:40:33 2005
Date: Thu, 15 Apr 2004 16:33:07 +0200 (CEST)
From: Bert Kampes <Bert.Kampes@dlr.de>
To: Mahdi Motagh <motagh@gfz-potsdam.de>
Cc: Doris software list <doris_users@geo.tudelft.nl>,
< doris_users@yahoogroups.com
Subject: Re: displaying with DORIS

+++ At 17:17, April 15th, Mahdi Motagh wrote:

<MM: Dear Bert Kampes
<MM: I have just finished processing of the BAM area
<MM: with Doris.I have a problem in displaying my final result. In fact, the color cycle of my result
<MM: you have put in the net, although the general pattern and, particularly, the number of fringes
<MM: Therefore, I would be grateful if you could tell me how I should interpret the color cycle in DORIS
<MM: Yours sincerely
<MM: Mahdi Motagh

Dear Mahdi,

The meaning of a color cycle depends on definition of the colorbar. Since I don’t know what colorbar you use, I cannot tell you the answer to your in itself interesting question.

The unwrapped phase is ultimatively important. If we have a pixel somewhere that has the unwrapped phase \( P(X) = 0 \)
And we have another pixel that has the unwrapped phase \( P(Y) = +2\pi \)

Assume that point \( X \) is stable.
Then with the definitions used by Doris this theoretically would mean that point \( Y \) has moved away from the radar (wrt. point \( X \); 2.8 mm) in the time between the acquisitions. That is, an increase of the unwrapped phase means displacement away from the sensor.

Suppose your slave image is acquired later than the master image. I assume here for simplicity that we deal with vertical displacement only. Then this means that point \( Y \) has "subsided".

I hope this helps you. It is not that easy to guarantee that the above is correct. If, for example, your RAW data processor as a convention would multiply the phase with -1, than consequently point \( Y \) has "uplifted". In any case, once we figure out what it means it should always be the same in all interferograms.
I don’t have much practical experience/applications with Doris, and it has been a couple of years since I used it last. Therefore, the easiest way to check this is to ask this list what the experience are of people.

Is there anybody who has created an interferogram of a large displacement signal of known direction, and can that person thus tell us whether it is correct that if the unwrapped phase increases that this means that the point moves away from the sensor, i.e, if the slave is acquired later, that this is “subsidence”?

Best regards,

Bert Kampes

PS. I have forwarded your question to the list server for their help and for the benefit of all users.

---

From Bert.Kampes@dlr.de Mon Mar 14 08:26:46 2005
Date: Wed, 1 Dec 2004 08:56:15 +0100 (CET)
From: Bert Kampes <Bert.Kampes@dlr.de>
To: Gustavo Arciniegas <arciniegas06157@itc.nl>
Cc: Doris software list <doris_users@tudelft.nl>, doris_users@yahoogroups.com
Subject: Re: Visualizing output of Doris

+++ At 19:43, November 30th, Gustavo Arciniegas wrote:

<GA: Greetings
<GA: Sorry for the disturbance. I have generated my first interferogram and coherence map from my topography pair. However, I haven’t been able to properly visualize and export these data, so that I can paste them on text editors or slideshows. I have xv and ghostview installed but, for some reason, I can’t display them.
<GA: I have checked the online manual and it does refer to this software tools, but I’m still a bit clumsy with them.
<GA: I am a new beginner user of Doris, hoping to more and more learn about it. So, I again apologize for the inconveniences and appreciate the help I have received from this user list.
<GA: Sincerely,
<GA: Gustavo
<GA:

Dear Gustavo,

Indeed the FAQ and the manual do not very much explain visualization. (This email is intended to serve as reference for future questions on this subject). This list is basically intented for questions, so good of you to ask. I may not always answer this quickly however on some of your questions in future due to lack of time.

Hopefully you will answer questions from other users, or write a report on how to use “Doris for Dummies” (no offense intended) that can be included
in the manual. That would be great!

Try searching the FAQ, there is a specific post on the format of the files. (also in an appendix). However, the output of Doris are not image files that can be viewed directly with most tools. They do not contain information on colormap, dimensions, etc. Therefore, you could look up the format, dimensions, etc. and load your data in, e.g., Matlab, and then visualize it how you want. An alternative are the GMT tools. There are examples for both, see the Matlab Delft InSAR toolbox and the script cpx2ps.

The cpxfiddle utility may be an alternative for visualization. This should be installed if you followed the installation instructions. Doris actually uses this if you specify card PREVIEW ON in your input files. Please refer to "cpxfiddle -h" for help since it is too complex to explain here in a few words.

The cpxfiddle tool can be used off-line for many purposes, but you have to tell the program at least the format and the linelength of the file (in pixels).
-\texttt{w} : width (line length) of the file in pixels
-\texttt{f} : input format, e.g., 4B float values, complex 4B floats, etc.

Then there are options what you want to do and where the output should go to. For example, if you want to see the phase value of a few pixels of a complex file you will have to tell the program where to crop, to compute the phase from complex float input, and to display it on the screen. For visualization there is a special output format, "SUNraster" files. You can also (exponentially) scale the data to make it look brighter, etc., and use different colormaps.

This may sound very difficult but once you work with UNIX filter kind of programs you will appreciate the powerful things you can do with it. One more feature of cpxfiddle is that files can be manipulated off-line, and then continued with Doris, e.g., mirrored/cropped/scaled/subsampled, etc.

It gets much simpler though due to the integration with Doris. This work like this:

1) Use PREVIEW ON in input file
2) Doris creates little cpxfiddle script files with appropriate options in the working directory
3) Doris runs these scripts.

The advantage is that you now only have to change the scripts a little bit to tailor them to your needs. For example, if you use PREVIEW with M_CROP step, a little script is generated to output the magnitude. This script is called "master_mag.ras.sh" and if it is run with the command "master_mag.ras.sh" or "sh master_mag.ras.sh" it creates the output file "master_mag.ras" which can be viewed in any image browser, e.g., use "xv master_mag.ras"

This script looks like (open it with your favorite editor):

```
#!/bin/sh
# Doris software generated script to generate preview.
# Program cpxfiddle is used to create a SUNraster file
# from a complex input file. (For unwrapped interferograms
# we try to fool cpxfiddle.)
# Send comments to: Bert Kampes, kampes@geo.tudelft.nl
# to run, type at prompt: ./master_mag.ras.sh

cpxfiddle -w 5167 -e 0.5 -s 1.0 -q mag -o sunraster -b -c gray -M 2/10 -f ci2 -l1 -p1 -P5167 Outdata/9192.raw
```

```
You see, it simply is a one-line cpxfiddle command. Giving this exact command from the command prompt will do the same as execution of the script. In this case, the relevant options are:

- \texttt{-w 5167} : linelength
- \texttt{-e 0.5} : make magnitude a little brighter to equalize the distribution
- \texttt{-q mag} : output the magnitude using
- \texttt{-f c12} : complex 2B integer input
- \texttt{-o sunraster} : output a SUNraster file with a little header
- \texttt{-M 2/10} : multilook the data to get approx. square pixels.

The redirection using the "\texttt{>}" is important. Under UNIX this means that the output of the first program is put in a file that is pointed to, i.e., create file "master_mag.ras"

If the PREVIEW card is ON during PROCESS INTERFERO, actually three scripts are generated. One to create a SUNraster file for the magnitude, one for the phase and for the phase transparently overlayed on the magnitude (called "interferogram_mag.ras.sh", "interferogram_pha.ras.sh", "interferogram_mix.ras.sh"). The latter one looks like:

```bash
#!/bin/sh
# Doris software generated script to generate preview.
# Program cpxfiddle is used to create a SUNraster file
# from a complex input file. (For unwrapped interferograms
# we try to fool cpxfiddle.)
# Send comments to: Bert Kampes, kampes@geo.tudelft.nl
# to run, type at prompt: ./interferogram_mix.ras.sh

cpxfiddle -w 5134 -e 0.3 -s 1.0 -q mixed -o sunraster -b -c hsv -M 2/10 -f cr4 -l1 -p1 -P5134 Outdata/9192_9693.cint
```

Note that the exponential is smaller since the interferogram contains master and slave magnitude. The colormap is changed using "\texttt{-c hsv}". Multilooking is set to square pixels already since it is assumed this is done by Doris during the interferogram generation. The rest is pretty much intuitive once you understand the previous example.

Hope this helps you and others. Note that cpxfiddle can also be used to manipulate SUNraster files and almost any data. I use it a lot also for non-Doris things.

Regards,

Bert Kampes

PS. Using PREVIEW XV actually calls xv automatically.

PS. <GA: properly visualize and export these data, so that I can paste them on "exporting" I do not understand? the text you can cut and paste, the data format is defined and easy to import in any GIS, etc. using the information in the result files on dimension and format.
Dear Richy,

<RB: I am working on an area in western New York state, USA. The area
<RB: ranges from urban to suburban to rural. We have a research project
<RB: trying to map out some areas of subsidence so that we can better
<RB: understand the underlying mechanisms from the ground.

We have good results in rural areas ~20x20km in Germany with the PS technique.
Some villages/housing say ~2-4 km apart. Atmosphere was limited for
these areas. (However, we have 50-80 images available and the displacement
signal was linear and upto 70 mm/y.)

<RB: oversampling in azimuth (outside of Doris or modifying Doris) help
<RB: as well?

The main reason for oversampling is to avoid problems with aliasing
since PS processing with large baselines is hard enough as it is. I don;t
think the sub-pixel position one so obtains is important for point scatterers
(or the interpolated phase).

Here we oversample both in range and azimuth by a factor two. Since
doing this won;t be bad, we simply do it, and I did not really look into
consequences of not doing so. It made kind of sense. Oversampling in
azimuth is a bit more cumbersome due to the shifted spectrum, and I guess
that is why it has not been implemented. The sampling prf ~1680,
bandwidth ~1380 in azimuth, i.e, ratio .82 and there is a hamming weighting
on the azimuth spectrum of SLC data.
For range, RSR=18.96, BW=15.55, ratio=0.82 (and Hamming).
Therefore, I see no reason why oversampling in azimuth should not be
done if oversampling in range is done. I once wrote some comments how
Doris can be tricked to do oversampling in azimuth using a dummy
resampling processing of a shifted grid by half a pixel, then stitching
the output. This automatically would take care of the IDC problem since the
resampling routine does. However, I have never done it. (Very) secondary
would be that the pixels get approximately square if e.g., factor 4 in range is used.
I have no experience with this step, except that some generated test interferograms
looked very nice with OVS range=4.

<RB: How much will getting more images help? What methods are worth
<RB: trying without them? Also, I realize that statistics get much better
<RB: for the PS method at 30 scenes. Does it make any sense to just get
<RB: up to 20 scenes, or do I need to get all the way to 30 scenes? (I
<RB: might be able to get 20 total, but probably not 30 total)
<RB:
Since it is quite a non-linear inversion (wrapped data), less than 20 is difficult with an single master PS approach. We have some results with 16 scenes for Bangkok data agreeing reasonably well with an adjacent stack with 20 scenes, but an algorithm such as ensemble coherence maximization or integer least-squares has a certain "false alarm rate". I estimate this is 50% for 15 images, 10 percent for 30 images (depending on noise level of course, but also on signal and baseline distribution, i.e., if the DEM error and linear displacement rate difference between nearby points is small, the search space can be limited stricter, reducing the false alarm rate.) There is a graph in Kampes and Hanssen 04 "Integer least-squares", TGARS, that supports this more or less.

The problem is that one can estimate between nearby points, and certainly find some correct estimations, but it is difficult to get the overall picture. The amplitude dispersion index is not a very reliable measure if you only have 20 images, but one could resort to signal-clutter-ratio in a spatial estimation window, or even select points based on spatially estimated coherence.

It also depends on the size of your area, i.e., 5x5 km would be easier than 20x20 km, 40x40 km gets difficult with PS, except if you have years experience maybe or suited data. I would limit with 15 images to 8x8 km or so.

Alternatives are the small baseline approaches such as SBAS and the algorithm by Hooper et al. I mentioned. Depending on your baseline distribution this may be good for you. PS in principle allows to study the phase of a single point on a roof, etc., and allows in principle to use acquisitions with large baselines (say ~1000 m) and delta_fDC, say ~1000 Hz basically because corner reflectors are not affected by decorrelation. If you have patches of coherent information, then you could try to work with multilooked maybe even filtered interferograms, aiming for distributed scatterers not affected by temporal decorrelation. I am not very familiar with this, but with Doris you could try this easier than a PS approach (since that requires quite precise algorithms). In general I think the topography can be removed if you have 15 images, so you may want to aim at estimation of topographic information somehow, than remove that, and interpret the residual. Again depends whether you use multilooked distributed scatterers or PS points.

One way with \( N=15 \) SLC images is to generate all \( 0.5 \times N \times (N-1)=105 \) interferograms and simply see how it looks. Compare this with now areas of deformation. (Note that all information however is in 14 ifgs.) With Doris it should not be difficult to automate this, but will require all SLC on hard disk and writing a little script and template processing input files.

First plot your baseline diagram (Bperp vs. Btemp) and this will help you further to decide if a single master stack or some cascade processing strategy is possible. Easier I think is such an approach creating the small Bperp interferograms since at least you will see some phase in the interferograms.

If your max. signal is less than 1 fringe in 2 years (15 mm/y) I guess with 15 interferograms spanning 8 years it is difficult to retrieve since atmospheric signal can be expected to be sigma ~1 rad or a fringe in your area could happen. If your displacement is more than 15 fringes (240mm/8 years ~1 inch/year) signal aliasing likely will prevent correct estimation (depending on temporal distribution, assuming equi-temporal, each half year an interferogram). If your signal is linear, a PS approach is more suited than if it is not.

Hope this helps you somewhat. I cannot guarantee all my comments are accurate and apply to your situation. It may be smart to look for a partner, e.g.,
Delft, Bari, (or Stanford) somehow to discuss specifics with people actually having done some things with Doris.

If you have the budget, the commercial insar packages promise some results and PS-like processing. You could look into capabilities/pricing etc. With Doris, I think, some development still needs to be undertaken and "support" is not guaranteed. However, people are working on it.

Best regards,

Bert Kampes

+++ At 15:52, November 29th, Richard Becker wrote:

<RB: Dear Bert,
<RB: 
<RB: Thanks for the response,
<RB: 
<RB: I appreciate the specific processing suggestions.
<RB: 
<RB: At first I was processing to try 3 or 4-pass DINSAR, with small temporal baselines, but, as anticipated, the coherence was so variable across a section of the scene that I started looking at other DINSAR techniques. Right now I am just trying to get a handle on what methods can best be used for my problem.
<RB: 
<RB: I am working on an area in western New York state, USA. The area ranges from urban to suburban to rural. We have a research project trying to map out some areas of subsidence so that we can better understand the underlying mechanisms from the ground.
<RB: 
<RB: I know in Doris it is possible to oversample in range, would oversampling in azimuth (outside of Doris or modifying Doris) help as well?
<RB: 
<RB: How much will getting more images help? What methods are worth trying without them? Also, I realize that statistics get much better for the PS method at 30 scenes. Does it make any sense to just get up to 20 scenes, or do I need to get all the way to 30 scenes? (I might be able to get 20 total, but probably not 30 total)
<RB: 
<RB: BTW, I found the Hooper paper on his website. I'm not sure if the GRL issue is published yet.
<RB: 
<RB: Thanks again for your help,
<RB: 
<RB: -Ricky Becker
<RB: 
<RB: Ricky Becker
<RB: Principal Research Associate
<RB: Earth Sciences Remote Sensing Lab
<RB: http://www.esrs.wmich.edu/
<RB: Western Michigan University
<RB: (269) 387-5446
<RB: richard.becker@wmich.edu
Dear Ricky,

Thanks for your question. This list is intended for such discussions.

As it happens the PS technique is my PhD research topic. I am working at the DLR helping to develop a scientific PS processor. I do not use Doris for this work, but I know there are groups doing so. Hopefully they will react and give you some more information. (TU Delft, particularly Ramon Hanssen, Petar Marinkovic and Freek van Leijen who maybe can share some Matlab scripts for PS estimation; and POLIBA (/?)Alberto Refice and Raffaele Nutricato who created OVS step and multi_run.csh (and found some other small bugs:)))

Also there are some papers by these authors on this topic, consider searching http://enterprise.lr.tudelft.nl/doris/bibliography/

I think the following may be of interest for you, but it depends on your processing strategy and signal if this is relevant:

- Do not perform spectral filtering. I also would not do phase filtering personally except if your points are spatially correlated (height (DEM error) and displacement). This depends whether your are interest in information on isolated point scatterers.
- Use oversampled data (range, step OVS by Raffaele Nutricato)
- Use a script for the processing, multi_run.csh is somewhere, i do not have it in the archive at the moment but I am sure you can get it from people working with it. Else, I have an older version.
- COMPREFDEM is slow, for PS processing it is currently being upgraded.
- I think you do not need a DEM anyhow for your application, since you can estimate it from your data simultaneously with the displacement.
- Buy more images... (20+:

Furthermore, recently/soon a GRL paper will be published by A.Hooper et al.
with a similar processing strategy for spatially correlated
displacement
signals that you may find interesting for your work.
Best regards,
Bert Kampes

+++ At 12:08, November 29th, Richard Becker wrote:

Hi,

I was wondering if any of you had any experience in taking
interferograms produced in DORIS (i.e flat earth corrected,
filtered
complex interferograms), and using techniques such as those
Ferretti
et. al describe for point scatterers, or that Gamma
implements as
Point Target Analysis. I am looking for some pointers for
attempting
some variation on these techniques.

I am working on a landform deformation problem (differential
subsidence in a residential area), and because the area has
too much
vegetation, it is difficult to get a good coherence across
the
entire area of interest, even with pairs one day apart. I am
able to
get locally good coherence numbers (> 0.85) over a large
temporal
baseline in areas which are more developed. I have about a
dozen
scenes over 6 years covering the small area that I am working
with.
I have a good DEM at a better horizontal resolution than the
ERS
data (nominally 1 or 5m horizontal) which I can use to remove
the
topo phase if necessary.

I know this isn't really a specific technical question on
the
software, but I was wondering if anyone could give me some
pointers
in this kind of processing, what pitfalls to avoid, or if
anything
like this was planned for DORIS in the future.
Thanks for your input,
Ricky Becker
Principal Research Associate
Dear Ricky,

Thanks for your question. This list is intended for such discussions.

As it happens the PS technique is my PhD research topic. I am working
at the DLR helping to develop a scientific PS processor. I do not use
Doris for this work, but I know there are groups doing so. Hopefully
they will react and give you some more information. (TU Delft,
particularly Ramon Hanssen, Petar Marinkovics and Freek van Leijen who
may be can share some Matlab scripts for PS estimation; and POLIBA (?) (Alberto
Refice and Raffaele Nutricato who created OVS step and multi_run.csh
and found some other small bugs:)))

Also there are some papers by these authors on this topic, consider searching
http://enterprise.lr.tudelft.nl/doris/bibliography/

I think the following may be of interest for you, but it depends on
your processing strategy and signal if this is relevant:

- Do not perform spectral filtering. I also would not do
  phase filtering personally except if your points are spatially
  correlated (height (DEM error) and displacement). This
  depends whether your are interest in information on isolated point
  scatterers.
- Use oversampled data (range, step OVS by Raffaele Nutricato)
- Use a script for the processing, multi_run.csh is somewhere, i do not
  have it in the archive at the moment but I am sure you can get it
  from people working with it. Else, I have an older version.)
- COMPREFDEM is slow, for PS processing it is currently being upgraded.
  I think you do not need a DEM anyhow for your application, since you
  can estimate it from your data simultaneuously with the displacement.
- Buy more images... (20+:

Furthermore, recently/soon a GRL paper will be published by A.Hooper et al.
with a similar processing strategy for spatially correlated displacement
signals that you may find interesting for your work.

Best regards,

Bert Kampes
At 12:08, November 29th, Richard Becker wrote:

Hi,

I was wondering if any of you had any experience in taking interferograms produced in DORIS (i.e flat earth corrected, filtered complex interferograms), and using techniques such as those Ferretti et. al describe for point scatterers, or that Gamma implements as Point Target Analysis. I am looking for some pointers for attempting some variation on these techniques.

I am working on a landform deformation problem (differential subsidence in a residential area), and because the area has too much vegetation, it is difficult to get a good coherence across the entire area of interest, even with pairs one day apart. I am able to get locally good coherence numbers (> 0.85) over a large temporal baseline in areas which are more developed. I have about a dozen scenes over 6 years covering the small area that I am working with. I have a good DEM at a better horizontal resolution than the ERS data (nominally 1 or 5m horizontal) which I can use to remove the topo phase if necessary.

I know this isn't really a specific technical question on the software, but I was wondering if anyone could give me some pointers in this kind of processing, what pitfalls to avoid, or if anything like this was planned for DORIS in the future.

Thanks for your input,

Ricky Becker
Principal Research Associate
Earth Sciences Remote Sensing Lab
http://www.esrs.wmich.edu/
Western Michigan University
(269) 387-5446
richard.becker@wmich.edu

---

E.9 Items not yet filed

From Bert.Kampes@dlr.de Mon Mar 14 08:25:00 2005
Date: Tue, 4 Jan 2005 18:33:18 +0100 (CET)
From: Bert Kampes <Bert.Kampes@dlr.de>
To: Zbigniew Perski <perski@us.edu.pl>
Cc: Doris software list <doris_users@tudelft.nl>, doris_users@yahoogroups.com
Subject: Re: doris_3.13-questions and comments

At 12:08, November 29th, Richard Becker wrote:

Hi,

I was wondering if any of you had any experience in taking interferograms produced in DORIS (i.e flat earth corrected, filtered complex interferograms), and using techniques such as those Ferretti et. al describe for point scatterers, or that Gamma implements as Point Target Analysis. I am looking for some pointers for attempting some variation on these techniques.

I am working on a landform deformation problem (differential subsidence in a residential area), and because the area has too much vegetation, it is difficult to get a good coherence across the entire area of interest, even with pairs one day apart. I am able to get locally good coherence numbers (> 0.85) over a large temporal baseline in areas which are more developed. I have about a dozen scenes over 6 years covering the small area that I am working with. I have a good DEM at a better horizontal resolution than the ERS data (nominally 1 or 5m horizontal) which I can use to remove the topo phase if necessary.

I know this isn't really a specific technical question on the software, but I was wondering if anyone could give me some pointers in this kind of processing, what pitfalls to avoid, or if anything like this was planned for DORIS in the future.

Thanks for your input,

Ricky Becker
Principal Research Associate
Earth Sciences Remote Sensing Lab
http://www.esrs.wmich.edu/
Western Michigan University
(269) 387-5446
richard.becker@wmich.edu

---

From Bert.Kampes@dlr.de Mon Mar 14 08:25:00 2005
Date: Tue, 4 Jan 2005 18:33:18 +0100 (CET)
From: Bert Kampes <Bert.Kampes@dlr.de>
To: Zbigniew Perski <perski@us.edu.pl>
Cc: Doris software list <doris_users@tudelft.nl>, doris_users@yahoogroups.com
Subject: Re: doris_3.13-questions and comments

At 18:12, January 4th, Zbigniew Perski wrote:

Hi Richard,

I am working on a landform deformation problem (differential subsidence in a residential area), and because the area has too much vegetation, it is difficult to get a good coherence across the entire area of interest, even with pairs one day apart. I am able to get locally good coherence numbers (> 0.85) over a large temporal baseline in areas which are more developed. I have about a dozen scenes over 6 years covering the small area that I am working with. I have a good DEM at a better horizontal resolution than the ERS data (nominally 1 or 5m horizontal) which I can use to remove the topo phase if necessary.

I know this isn't really a specific technical question on the software, but I was wondering if anyone could give me some pointers in this kind of processing, what pitfalls to avoid, or if anything like this was planned for DORIS in the future.

Thanks for your input,

Ricky Becker
Principal Research Associate
Earth Sciences Remote Sensing Lab
http://www.esrs.wmich.edu/
Western Michigan University
(269) 387-5446
richard.becker@wmich.edu
<ZP: Bert,
<ZP:
<ZP: Thanks a lot for this explanations. I am sure that it will be usefull
<ZP: for many people as well.
<ZP:
<ZP: > Using a point target analysis in Matlab or so you can compute the subpixel
<ZP: > coordinates of the CR in the radar image.
<ZP: It sound very interesting.
<ZP: I have no access to matlab in this moment yet. I have requested the
<ZP: trial version. The point target analysis is a part of matlab or it is
<ZP: separate toolbox?
<ZP: I found something similar in ENVI but it is dedicated to hyperspectral
<ZP: data.

Dear Zbigniew,

Sorry, the term point target analysis may sound a bit fancy. I do not know of a toolbox publicly available. What I meant was interpolation of the amplitude. Maybe with matlab you could do something like the following:

Matlab> crop = freadbk('slc.raw',numlines,'cpxint16',101,132,1001,1032);% read data from file
where the crop is centered on a corner reflector (CR);

Matlab> cropovs = oversample(crop,8,8);% interpolate data, assume fDC=0
Matlab> [maxampl,idx] = max(abs(cropovs(:)));% find max amplitude
Matlab> [ri,ci] = ind2sub(size(abscropovs),row column of max
Matlab> ri = 101+ri/8;
Matlab> ci = 1001+ci/8;% float pixel coordinate in original
Matlab> phase_cr = angle(cropovs(idx));%

However, it has been years I worked with Matlab. If fDC not 0, then you have to shift the spectrum working with complex data due to the Doppler centroid. Another way maybe to interpolate (oversample) the amplitude image, not the complex image. Then once you have the sub-pixel peak position, you load the unwrapped phase and interpolate in it with some matlab standard function. Since you anyway need the unwrapped phase of the CR for tiepointing, I guess this is easier, but not sure whether it is fully correct.

Would be nice to have an interactive tool. The data is read and displayed in multilooked, but is full-res in memory. You click on a point and such an analysis is performed for the brightest point in a local window. Shown are the spectrum, etc. sub-pixel coordinates.

Have a nice evening,

Bert Kampes

From Bert.Kampes@dlr.de Mon Mar 14 08:24:43 2005
Date: Tue, 4 Jan 2005 16:07:39 +0100 (CET)
From: Bert Kampes <Bert.Kampes@dlr.de>
To: Dr.Y.S. Rao <ysrao@care.iitb.ac.in>
Cc: Doris software list <doris_users@tudelft.nl>, doris_users@yahoogroups.com
Subject: Re: [doris_users] Re: Reference DEM programme

Dear Dr. Rao,
I tried to understand your code in the file “referencephase.cc.” I know a little in C, but not C++. There are several matrices in your code. I think the correct matrix where I get DEM in radar co-ordinates is in the following code. Is there any way to convert the dem phase file (output) into height instead of modifying your code?

Since the height_to_phase conversion factor is different for each pixel, you cannot simply convert the phase output back. Second, I think doing the transform twice is not recommended. If you want to do it, compute the height_to_phase conversion factor for a number of points on the image using the DUMP_BASELINE card. Then model the factor as function of rg and az. Then evaluate it for the whole image and divide the phase by it. This gives you the DEM heights in radarcoordinates. As a crude approximation, use a single factor for the whole image (the "height ambiguity").

Some software register DEM accurately with master image after simulating radar image with DEM and taking crosscorrelation between simulated image and actual image. I do not know what process goes in your case of aligning DEM with master image. I think you are taking Lat and long of SAR image and correlating with that of DEM.

I agree this is a very smart approach. Doris does not do any coregistration. Foreseen was a process by hand, shifting the synthetic phase image at pixel level before subtraction and then visual inspection of the result. There is a card for this, but it is quite some work. However, if the DEM has a resolution of 1000m and the interferogram has 20m, then an offset of 5 pixels is not so important. Moreover, the definition where the DEM value is, in the middle, at the edge of the pixel is unclear to me, introducing a further error of half the DEM resolution. Another option is to use the general input card M_RG_T_ERROR which in effect does the same. Doris uses zero doppler iteration for geocoding/radarcoding, see annex of manual.

Unfortunately I did not have time to look at the code in detail and will have no time in near future for it. I think the code now uses the following steps:
1) interpolated DEM, put it in matrix "DEMinterpolated";
2) find out which pixel in DEMinterpolated corresponds to a Line,Pixel of the interferogram;
3) for all pixels, use the range times to compute the phase.

To find the DEM value, indeed is a bit hidden. Actually, the DEM value is not there. Instead the slave range time is used (master range time is given by the pixel number). There are several options for it. I would think directly after finding the nearest neighbor it should be possible to index the DEMinterpolated and write out the value. But probably better is in the loop3), e.g., storing the interpolated DEM value in a new vector temporarily and writing it to a file the same way the phase is.

I agree these issues should be dealt with. The best way is a new implementation, using the algorithm of Einder, 2003, TGARS "Efficient Simulation of SAR interferograms of large Areas and of Rugged Terrain" and to use the trick proposed by you to check the coregistration of DEM and interferogram, since there could be a timing error. A modular approach could be a new routine that transforms the DEM from [lat,lon] to [rg,az] in meters, and a second routine that transform the DEM in meters [rg,az] to interferometric phase, and a final routine that subtracts this phase from the interferogram.
Best regards,

Bert Kampes

+++ At 16:23, January 4th, Dr.Y.S. Rao wrote:

<DR: Dear Bert,
<DR: Thanks for your kind reply.
<DR: > incidence angle masks. DORIS could register images upto baseline of 2276 km. with one pixel accuracy, but with little warning. I first converted
<DR: > 2276 meter?
<DR: For the Bperp baseline of 43 meters, I got shift 3 range pixels, for 678 meters, the shift is 21 pixels, and finally 2276 meters, I got shift 109 pixels. So they are in the order.
<DR: > You are correct. The OUT_FILE is in phase as far as I understand from
<DR: > the documentation (and recall from previous posts).
<DR: > http://enterprise.lr.tudelft.nl/doris/usermanual/node93.html
<DR: > You need the DEM in meters but in the radarcoordinates (rg,az) (instead
<DR: > of in WGS84). This is not output of Doris currently, and you will
<DR: > have to code it as additional output. Internally it is somewhere
<DR: > in a matrix I think, find this and write it to a file (in buffers).
<DR: > I tried to understand your code in the file "referencephase.cc." I know a
<DR: little in C, but not C++. There are several matrices in your code. I
<DR: think the correct matrix where I get DEM in radar co-ordinates is in the
<DR: following code. Is there any way to convert the dem phase file (output)
<DR: into height instead of modifying your code?
<DR: >
<DR: // ====== Nearest neighbor to get reference phase to OBUFFER ======
<DR: PROGRESS << "Start nearest neighbor for buffer: " << buffer+1;
<DR: PROGRESS.print();
<DR: int32 indexobufferL;
<DR: int32 indexobufferP;
<DR: real8 indexobufferLr8;
<DR: real8 indexobufferPr8;
<DR: real8 mlLr8 = real8(mlL);
<DR: real8 mlPr8 = real8(mlP);
<DR: real4 squareddistance; // in matrix index
<DR: syste
<DR: m
<DR: const int32 numlinesOBUFFER = OBUFFER.lines();
<DR: const int32 numpixelsOBUFFER = OBUFFER.pixels();
<DR: // tested PNT2 2b a real8* matrix, no gain in speed... BK 15-feb-00
<DR: matrix<int32> PNT2LPPhrase(numlinesOBUFFER,numpixelsOBUFFER);//
<DR: 'pointers'..
<DR: .
<DR: // matrix<real8*> PNT2LPPhrase(numlinesOBUFFER,numpixelsOBUFFER);//
<DR: pointers
<DR: // Store squared distances in DISTANCES, OBUFFER could be used if
<DR: cleaned.
<DR: matrix<real4> DISTANCES(numlinesOBUFFER,numpixelsOBUFFER);
<DR: const real4 DEFAULTDIST = 99999.; // must be a positive large number
<DR: DISTANCES.setdata(DEFAULTDIST); // store here the pixel distance for
<DR: nn.
<DR: for (i=0; i<numpointsDEMinterpolated ;++i)
<DR: {
<DR: if (i%500000 == 0)
<DR: {
<DR: // ______ Extra info ______
<DR: PROGRESS << "Nearest neighbor point number: " << i << " (" << floor(.5+(100.*real8(i)/real8(numpointsDEMinterpolated)))
<DR: << ")%");
<DR: PROGRESS.print();
<DR: }
<DR: }
<DR: simulating image using COMPREFDEM. In MATLAB tools, something is mentioned
<DR: about this.
<DR: I don't recall any remarks. To simulate the amplitude, consult, e.g.,
<DR: paper by M Eineder TGARS 2003 or 2002 titled "fast simulation of interferograms
<DR: in rugged terrain" (or so). Could be a starting point for you, I think
<DR: there was a simple formula in it for this purpose. To implement it in
<DR: doris requires some work however. Likely they are better papers on it
<DR: with more sophisticated scattering models.
<DR: Some software register DEM accurately with master image after simulating
<DR: radar image with DEM and taking crosscorrelation between simulated image
<DR: and actual image. I do not know what process goes in your case of aligning
<DR: DEM with master image. I think you are taking lat and long of SAR image
<DR: and correlating with that of DEM.
<DR: best wishes,
<DR: Y.S. Rao
<DR:

From Bert.Kampes@dlr.de Mon Mar 14 08:24:29 2005
Date: Tue, 4 Jan 2005 16:06:16 +0100 (CET)
From: Bert Kampes <Bert.Kampes@dlr.de>
To: Zbigniew Perski <perski@us.edu.pl>
Cc: Doris software list <doris_users@studelft.nl>, doris_users@yahoogroups.com
Subject: Re: doris_3.13-questions and comments

Hi Zbigniew,

Quite a long email. I will try to answer your questions.

<ZP: WARNING : SWST was not constant
<ZP: WARNING : c4SWSTchange: 1

This is a new check I added. It means that the SWST has changed, i.e.,
that the range first pixel shifts in the raw data. This is done to keep
the same range sensor/terrain, i.e., due to ellipsoid or topography.
Doris does not compensate for this, e.g., by writing zeros for the first range pixels
(20 or so) in all following lines. I do not know for sure whether this is
important, or should be done. It is a thing of ESA processed SLCs. As far
as I know, ESA VMP processor compensates for this by writing zeros. I remember
BAM envisat data where the left is black, with shifting windows. This
is probably SWST. There should be no effect on InSAR, except that you should be aware that a few zeros are added.

<ZP: c S_A2_T_ERR 0.0 // optional timing error
<ZP: It is clear what does it means, but I don't know how to practically use
<ZP: this in doris. If someone could write a brief explanation it would be
<ZP: nice...
<ZP:

Basically you can shift the master and slave image with these cards. The goal would be to have an absolute reference frame, e.g., a DEM you trust, or a corner reflector tiepoint with known WGS coordinates. Using the TIEPOINT card you can then compute the time from the CR to the master (and slave). Using a point target analysis in Matlab or so you can compute the subpixel coordinates of the CR in the radar image. These coordinates you can convert to azimuth and range time. The difference between the two is the error, which you can then put in the input file. Once you have done this, your image is "absolutely" correct.

If you do not have a corner reflector, it is still useful. For example, we may find out that always data processed by I-PAF before 1996 has a systematic shift of 6ms or so. Then we can account for that using this card. Mainly for multi-image analysis with data from different processors this may become handy.

Mainly the M_RG_TIME is important to play with since that is the geometry of the interferogram affected most by timing errors due to e.g., electronic delay etc. The M_AZ_TIME is ok if the orbits and annotation is ok.

One could compare geometric coregistration using precise orbits with coreg based on correlation (the terrain). Suppose now there is a range timing error equivalent to 10 pixels. The coreg based on orbits should tell you, e.g., that the images are exactly on top of each other. Then the correlation should tell you that the shift is 10 pixels. Maybe one can uses this to make a relative correction only using the M_* cards.

<ZP: 3. I am experimenting a bit with OVS.
I did it once, Raffaele Nutricato has implemented it. I used OVS=4 in range.
Results were quite nice looking.

The plots you show are relative plots, i.e., they are normalized and cannot be compared with eachother. How is the fit of the CPM? I assume the interferogram does look noisy?

The numbers on top of the ps files look short of similar?
<ZP:
<ZP: 4. I have manually edited run script to have my favorite settings as
<ZP: defaults (mainly path to orbit files) but I did not found the way how to
<ZP: change default settings for preview generation with cpxfiddle. I would
<ZP: like to generate scripts which 1/1 multilooking as default. How to do
<ZP: it?

Good suggestion to make it input card.
For now, you could do it by hand, by editing the .sh files, e.g., change the file interferogram_srp_mag_ras.sh from cpxfiddle -w 2567 -e 0.3 -s 1.0 -q mag -o sunraster -b -c gray -M 2/2 -f cr4 -l1 -p1 -P2567 Outdata/91
to

cpxfiddle -w 2567 -e 0.3 -s 1.0 -q mag -o sunraster -b -c gray -M 1/1 -f cr4 -l1 -p1 -P2567 Outdata/9192_6687.srp

and run it again by typing:

    sh ./interferogram_srp_mag.ras.sh

or automatically by doing something like (not very robust)

    sed s/'2\'/1\'/ interferogram_srp_mag.ras.sh > q; sh q

you can even detect in the run file what are the latest created .sh files, and then re-run them using the sed as above. Personally, I would simply edit the files by hand, since there are only a few you need. They are meant to be quick-look images.

Bye,

Bert Kampes

+++ At 16:32, January 3rd, Zbigniew Perski wrote:

  <ZP: Hi,
  <ZP: Two weeks ago I have installed doris 3.13 and I have some questions
  <ZP: regarding new functionality.
  <ZP: 1. During processing ERS1/2 data i found strange warning during
  <ZP: M_READFILES and S_READFILES:
  <ZP: WARNING : SWST was not constant
  <ZP: WARNING : c4SWSTchange: 1
  <ZP: As I understand, it is something wrong with Sampling Window Start Time I
  <ZP: found this for the first time and I don't know if this could affect the
  <ZP: results.
  <ZP: 2. I found very interesting new cards:
  <ZP: c M_RG_T_ERR 0.0 // optional timing error
  <ZP: c M_AZ_T_ERR 0.0 // optional timing error
  <ZP: c S_RG_T_ERR 0.0 // optional timing error
  <ZP: c S_AZ_T_ERR 0.0 // optional timing error
  <ZP: It is clear what does it means, but I don't know how to practically use
  <ZP: this in doris. If someone could write a brief explanation it would be
  <ZP: nice...
  <ZP: 3. I am experimenting a bit with OVS.
  <ZP: I found in documentation that "The oversampling step is also useful for
  <ZP: the coregistration. In particular oversampling by a factor of 2 of the
  <ZP: cropped images could improve the computation of the offset between
  <ZP: master and slave magnitudes".
  <ZP: OK. I have performed coregistration without oversampling and with
  <ZP: "default" doris settings. The result you may see here (ps file):
Next, I have performed coregistration with OVS\_RANGE=2:

The vectors seem to be much worst that in the first run

Next, I have performed coregistration again, but with OVS\_RANGE=4 and increased number of windows (FC\_NWIN=901):

And the results are bad again.

All test was performed using magfft method and the same dataset within the same area (D\_BOW).

Baseline of my data is:

*******************************************************************
*_Start\_coarse\_orbits:
*******************************************************************

Some info for pixel: 22945, 10338 (not used):

\text{Bperp} [m]: 501.8

\text{Bpar} [m]: 233.7

\text{Bh} [m]: 552.1

\text{Bv} [m]: -40.6

\text{B} [m]: 553.6

\text{alpha} [deg]: -4.1 // baseline orientation

\text{theta} [deg]: 20.8 // look angle

\text{inc\_angle} [deg]: 23.5 // incidence angle

\text{orbitconv} [deg]: 0.00578354 // convergence

\text{Height\_amb} [m]: 17.1

\text{Btemp: [days]}: -71

Estimated translation slave w.r.t. master (slave-master):

\text{Coarse\_orbits\_translation\_lines}: -212

\text{Coarse\_orbits\_translation\_pixels}: -118

*******************************************************************
*_End\_coarse\_orbits:_NORMAL
*******************************************************************

4. I have manually edited run script to have my favorite settings as defaults (mainly path to orbit files) but I did not found the way how to change default settings for preview generation with cpxfiddle. I would like to generate scripts which 1/1 multilooking as default. How to do it?

Best regards

Zbigniew

| Please send contributions for the Doris mailinglist to: |
| doris\_users@tudelft.nl |
| |
| to unsubscribe send a message with the phrase: |
| "unsubscribe doris\_users" |
| to listserv@tudelft.nl |
| |
| http://enterprise.lr.tudelft.nl/doris |

--
Dear Dr. Rao,

<DR: incidence angle masks. DORIS could register images up to baseline of 2276 km. with one pixel accuracy, but with little warning. I first converted 2276 meter?

<DR: cCRD_OUT_DEM /Output/DEM.raw //request output
<DR: cCRD_OUT_DEM /output/DEMi.raw // request output
<DR: CRD_OUT_FILE /output/refdem.raw

I think one is called CRD_OUT_DEMI for interpolated DEM.

<DR: The size of the files DEM.raw and DEMi.raw are not matching to any of the files in my results. Output of COMPREFDEM is not DEM and it contains phase values (if I am wrong please correct me).

You are correct. The OUT_FILE is in phase as far as I understand from the documentation (and recall from previous posts).

http://enterprise.lr.tudelft.nl/doris/usermanual/node93.html
You need the DEM in meters but in the radar coordinates (rg,az) (instead of in WGS84). This is not output of Doris currently, and you will have to code it as additional output. Internally it is somewhere in a matrix I think, find this and write it to a file (in buffers).

The card CRD_OUT_DEM simply is the DEM as is, but cropped (for debug purposes).

The card CRD_OUT_DEMI is the CRD_OUT_DEM as is, but bilinear interpolated with factor DENSE. This explains why the file sizes are not matching and why they are no use to you (they are not in radar coordinates).

<DR: Now I have a difficulty of finding of the sizes of DEM.raw and DEMi.raw. They should be reported as DEBUG to stdout channel (screen or redirected to file) for each buffer. Note that if there is more than one buffer they do not have to align. Note again they are not in radar coordinates.

<DR: angle masks for the master image. I am also interested for generating simulated image using COMPREFDEM. In MATLAB tools, something is mentioned about this. I don't recall any remarks. To simulate the amplitude, consult, e.g.,
paper by M Eineder TGARS 2003 or 2002 titled “fast simulation of interferograms in rugged terrain” (or so). Could be a starting point for you, I think there was a simple formula in it for this purpose. To implement it in doris requires some work however. Likely they are better papers on it with more suffisticated scattering models.

It could be useful to have optional output of files containing in the same size as the interferogram (i.e., radarcoordinates) the per pixel local viewing angle height of DEM incidence angle (???) height_to_phase conversion factor perpendicular baseline

Using the first two files you can create the shadow mask, layover mask, etc. for your purposes I think. The other files are just for completeness.

Best wishes for 2005,

Bert Kampes

PS. For multi-image processing with the same master the height of DEM in radar coordinates is very useful to achieve a speed-up.

PS. One more comment on previous emails on integration constant/bias of unwrapped phase: using the synthetic phase image as outputted by CRD_OUT_FILE this bias could be estimated from a coarse DEM by mimizing the difference between both in a reasonable convenient way.

+++ At 12:52, December 31st, Dr.Y.S. Rao wrote:

<DR:
<DR: I am applying DORIS for other purposes i.e registering images automatically and generating snow cover maps, layover, shadow and incidence angle masks. DORIS could register images upto baseline of 2276 km. with one pixel accuracy, but with little warning. I first converted all SLCs of ENVISAT into backscattering coefficient (power and not dB) values. I added imagery part 0 to the backscattering file and gave it DORIS in real4 format. DORIS gave similar shifts that I got with original SLC data. I checked with about 5 pairs of data.
<DR: For generating layover, shadow and incidence angle maps, I tried to run COMPREFDEM by removing the comment before the lines.
<DR: cCRD_OUT_DEM /Output/DEM.raw //request output
<DR: cCRD_OUT_DEM /ouput/DEM1.raw // request output
<DR: CRD_OUT_FILE /output/refdem.raw
<DR: The size of the files DEM.raw and DEM1.raw are not matching to any of the files in my results. Output of COMPREFDEM is not DEM and it contains phase values ( if I am wrong please correct me).
<DR: For my case, I am giving the file sizes of COMPREFDEM task outputs
<DR: 9954980 Dec 31 11:31 DEM.raw_buffer_0
<DR: 59654016 Dec 31 11:31 DEM1.raw_buffer_0
<DR: 27599040 Dec 31 12:10 refdem1.raw ( with size 2664 linex x 2590)
Now I have a difficulty of finding the sizes of DEM.raw and DEMi.raw. These files may be useful for generating layover, shadow and incidence angle masks for the master image. I am also interested for generating simulated image using COMPREFDEM. In MATLAB tools, something is mentioned about this.

I hope some of you may give clues in this regard.

I take this opportunity to wish you all a very happy and prosperous new year 2005.

-y.s.rao-

---

From Bert.Kampes@dlr.de Mon Mar 14 08:22:53 2005
Date: Mon, 28 Feb 2005 12:33:35 +0100 (CET)
From: Bert Kampes <Bert.Kampes@dlr.de>
To: Gustavo Arciniegas <arciniegas06157@itc.nl>
Cc: Doris software list <doris_users@tudelft.nl>, doris_users@yahoogroups.com
Subject: Re: Interferogram magnitude

Dear Gustavo,

Doris calls cpxfiddle for visualization. It creates a little script, e.g., interferogram_mag.ras.sh

```
#!/bin/sh
# Doris software generated script to generate preview.
# Program cpxfiddle is used to create a SUNraster file
# from a complex input file. (For unwrapped interferograms
# we try to fool cpxfiddle.)
# Send comments to: Bert Kampes, b.m.kampes@lr.tudelft.nl
# to run, type at prompt: ./interferogram_mag.ras.sh
cpxfiddle -w 5134 -e 0.3 -s 1.0 -q mag -o sunraster -b -c gray -M 2/2 -f cr4 -l1 -p1 -P5134 Outdata/9192_9693.cint
```

Please read the help of cpxfiddle for meaning. Basically, the complex file 9192_9693.cint is transformed to values between 0 and 255 as follows and displayed using a gray color map;

- `mag`: output = abs(re+im);
- `e 0.3`: output = output^0.3;
- `sunraster`: output = rescale to 0:255 (?)

For more info, e.g., whether cpxfiddle actually uses the average to rescale, please see cpxfiddle.cc source code and internal documentation.

To practice, do the following on a piece of paper and on your input file:

1) what are the complex values of the first 4 pixels in the upper corner?
cpxfiddle -w 5134 -e 0.3 -s 1.0 -q normal -L2 -P2 9192_9693.cint

2) what are the magnitudes?
cpxfiddle -w 5134 -o ascii -q mag -L2 -P2 9192_9693.cint

3) what is the multilooked magnitude?
cpxfiddle -w 5134 -o ascii -q mag -L2 -P2 -M2/2 9192_9693.cint
4) what is the effect of -e option:

cpxfiddle -w 5134 -o ascii -e0.3 -q mag -L2 -P2 -M2/2 9192_9693.cint
e tc.

Doris does not compute anything with magnitudes I think, it simply leaves it as is. I think you will also find this info in the previous post if you re-read it.

Bert Kampes

---

From Bert.Kampes@dlr.de Mon Mar 14 08:22:42 2005
Date: Thu, 24 Feb 2005 13:00:52 +0100 (CET)
From: Bert Kampes <Bert.Kampes@dlr.de>
To: Gustavo Arciniegas <arciniegas06157@itc.nl>
Cc: Doris software list <doris_users@tudelft.nl>, doris_users@yahoogroups.com
Subject: Re: Computing magnitude correlation

Dear Gustavo,

Seems trivial by resetting the phase to zero of the complex files? Simply process upto interferogram. Then step COHERENCE would take the latest processing stage as input, i.e., file of master 9192.raw and for slave 6683.resampled.raw. Or if you performed filtering it would take, e.g., 9192.rf (range filtered).

Anyway, figure out which files are taken as input. The latest version of Doris logs the "latest known processing stage" which could be helpful.

One way to do it would be to create a complex file with a constant phase:

- I assume the input of step COHERENCE is 9192.raw and 6683.resampled
- mv 9192.raw 9192.original
- mv 6683.resampled 6683.resampled.original
- open matlab (i don’t see an easy way using cpxfiddle)
MATLAB> master=freadbk('9192.original',numlines,'cpxfloat32');
MATLAB> fwritebk(complex(abs(master)),'9192.res');// put abs in re

Do the same for the slave. I am not a 100% sure about the syntax.

There are more ways I guess, but I hope this gives you some ideas.
Best regards,

Bert Kampes

+++ At 11:37, February 24th, Gustavo Arciniegas wrote:

<GA: Hello,
<GA: Is it possible to trick Doris in such a way that I can compute
correlation of two co-registered magnitude images?
<GA: Process COHERENCE does that for the interferogram, by specifying
COH_WINSIZE and CO_MULTILOOK and computes correlation (complex
coherence).
<GA: I wonder if that can be possible, but now taking the magnitude file of
both master and slave (9192.raw and 6683.raw) as an input.
<GA: Hope this is not a crazy idea. I don’t know of other software that can
do this.
<GA: Thanks for your help.
<GA: Gustavo
<GA: