approach Documentation

Release 1

Samuel Bowers

May 24, 2022

Contents

1 How do I get set up? 3

2 Who do I talk to? 5

3 Contents: 7

3.1 Background . . . 7

3.2 Setup instructions . . . 8

3.3 Preprocessing ALOS data on the command line . . . 9

3.4 Mosaicking ALOS data on the command line . . . 13

3.5 Under the bonnet . . . 14

4 Indices and tables 19

The Alos Pre-PROcessing CHain (a.k.a. approach… sorry) is a repository to aid in the pre-processing of level 1.1 data from ALOS-1 and ALOS-2 with the SNAP graph processing tool.

Backscatter from the L-band ALOS-1 and ALOS-2 RADAR sensors is strongly associated with aboveground woody biomass in African woodlands and savannahs. This repository performs the pre-processing elements for producing similar maps. The gamma0 output geotiffs can be futher calibrated to measure aboveground biomass.

CHAPTER 1

How do I get set up?

This script is written in Python for use in Linux. You will need to have:

- A linux terminal
- Installed the SNAP graph processing tool
- Python, with snappy installed.

For more detailed instructions, see setup page.

CHAPTER 2

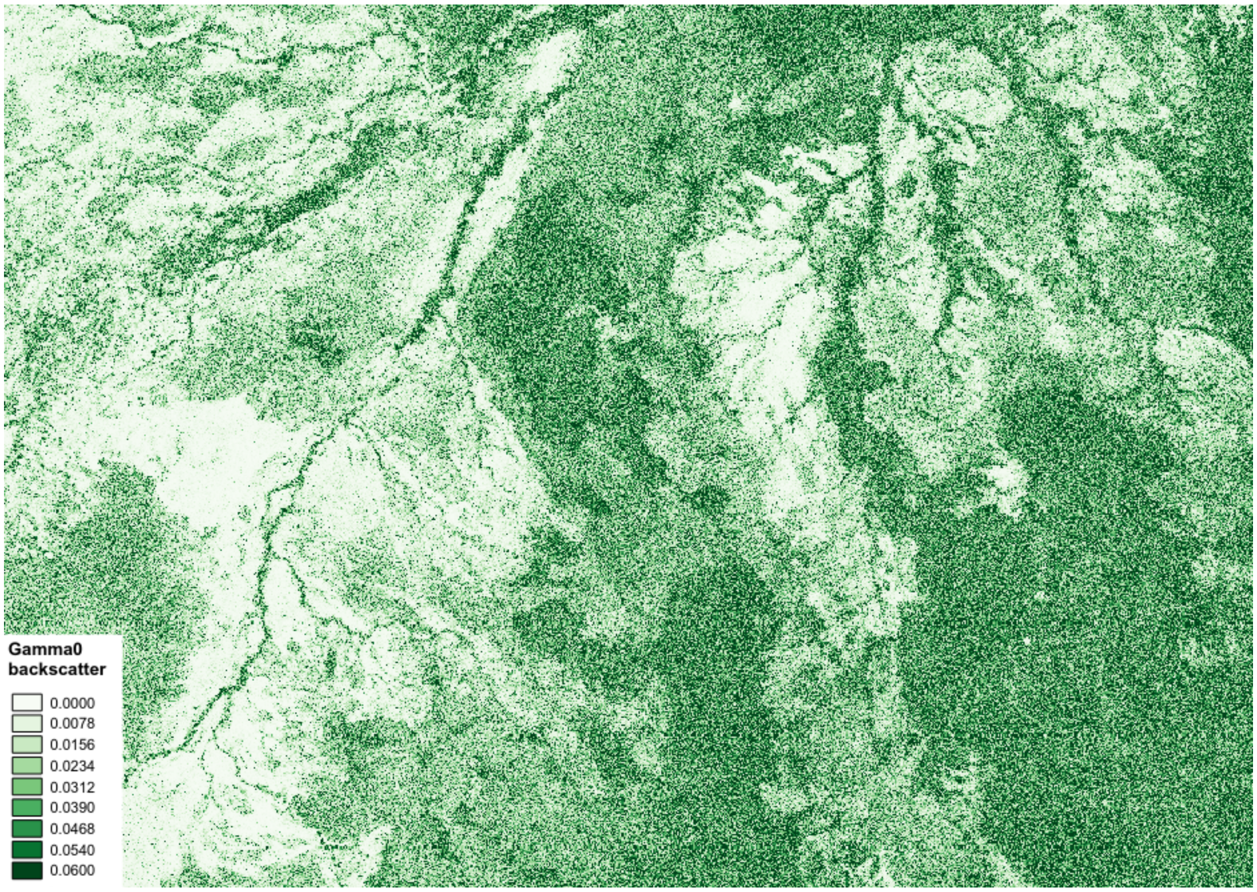
Who do I talk to?

sam.bowers@ed.ac.uk

CHAPTER 3

Contents:

3.1 Background



This repository contains scripts to perform pre-processing on level 1.1 data from the Phased Array L-band Synthetic

Aperture Radar (PALSAR/PALSAR-2) instruments, which is part of Japan's Advanced Land Observing Satellites (ALOS-1/ALOS-2). Radar satellite systems offer a number of advantages compared to traditional optical satellite systems (e.g. Landsat). For example, radar sensors are able to operate at any atmospheric conditions and equally either at day- and night-time. Furthermore, and most importantly in relation to forestry, long wavelength radar signal penetrates the canopy foliage and primarily interacts with forest aboveground biomass (i.e. tree trunks and stems), from which forest aboveground carbon can be calculated. The radar signal over forested areas that return to the sensor (backscatter) will be stronger compared to the backscatter measured in correspondence of a flat surface, as most of the signal bounces off in the direction opposite to the satellite. As a result of this, radar images will have brighter pixels in correspondence to forested areas, and darker pixels in correspondence to flat surfaces such as water bodies and bare soil. For these reasons radar can be considered a more suitable approach, compared to optical systems, to quantify forest carbon stocks, forest carbon change as well as forest carbon stock loss due to deforestation and forest degradation in tropical dry forests and African woodlands.

Further reading:

- Ryan, Casey M., et al. "Quantifying small-scale deforestation and forest degradation in African woodlands using radar imagery." Global Change Biology 18.1 (2012): 243-257.
- Mitchard, Edward TA, et al. "Measuring biomass changes due to woody encroachment and deforestation/degradation in a forest–savanna boundary region of central Africa using multi-temporal L-band radar backscatter." Remote Sensing of Environment 115.11 (2011): 2861-2873.

3.2 Setup instructions

3.2.1 Requirements

This toolset is written for use in Linux.

You will need access to a PC or server with at least:

- 32 GB of RAM to run SNAP.

For your images you might get away with as little as 8 GB, but in our experience 32 GB is required for reliable outputs.

3.2.2 Installing SNAP

SNAP is an ESA toolset used for (amongst other things) pre-processing data from Sentinel-1. SNAP for Linux can be downloaded from http://step.esa.int/main/download/.

To install SNAP, open a terminal window, change directory to the location you'd like to download SNAP (we recommend your home directory, at~/), and run the following commands:

```
wget http://step.esa.int/downloads/6.0/installers/esa-snap_sentinel_unix_6_0.sh
bash esa-snap_sentinel_unix_6_0.sh
```

...and follow the instructions. *When prompted, be sure that you install the 'snappy' module for integration with python.*

Note: If using ALOS data from ESA, make sure that your version of SNAP is at least 6.0.

It's a good idea to increase the memory allocation to SNAP. This is controlled by the text file `~/snap/bin/gpt.vmoptions`. This can be done with the following line:

```
echo '-Xmx32G' >> ~/snap/bin/gpt.vmoptions
```

For further details and up-to-date installation instructions, see the SNAP website.

3.2.3 Installing snappy

Snappy is a Python module that we used to extract metadata from ALOS datasets. It should work out-of-the-box following installation of SNAP, but in case not, here are some things to try:

- Find the directory that you istalled snap (default: `~/snap`), and run `./path/to/snap/bin/snappy-conf` and follow the instructions it prints out.
- Make sure that Python can find the snappy package. You can do this by moving the snappy module to your Python `site-packages` directory.
- If that fails, try editing your `~/.bashrc` file to include the line `export PYTHONPATH=$PYTHONPATH:/PATH/TO/snappy/`.
- If still in doubt, try consulting the ESA STEP forum.

3.2.4 Installing approach

approach can be downloaded to a machine from its repository. To do this, open a terminal window and input:

```
git clone https://sambowers@bitbucket.org/sambowers/approach.git
```

3.2.5 Where do I get help?

For help installing SNAP, it's best to refer to the ESA STEP forum. For assistance in setting up and using approach, email sam.bowers@ed.ac.uk.

3.3 Preprocessing ALOS data on the command line

Approach should be used on the Linux command line. Here the functionality of the command line tool is explained. In the next section we show how it works in more detail.

3.3.1 Showing help

Help for `preprocess.py` can be viewed by typing `python /path/to/approach/preprocess.py --help`:

```
usage: preprocess.py [-h] [-p HH/HV [HH/HV ...]] [-f] [-o PATH] [-g PATH]
                     [-n N_PROCESSES] [-v] [-e]
                     N [N ...]

Perform pre-processing on ALOS-1 and ALOS-2 level 1.1 data to produce a
terrain-corrected Gamma0 backscatter image.

Required arguments:
N                     Input files. Either specify a valid ALOS-1/2 input
```

(continues on next page)

(continued from previous page)

```
                        file, or multiple files through wildcards.

Optional arguments:
-p HH/HV [HH/HV ...], --polarisation HH/HV [HH/HV ...]
                        Process either HH or HV polarisation. Defaults to
                        both.
-f, --filter          Apply a refined lee filter to the processing chain.
-o PATH, --output_dir PATH
                        Optionally specify an output directory. If nothing
                        specified, files will be output to current working
                        directory.
-g PATH, --gpt PATH   This script uses the SNAP graph processing tool (gpt).
                        If not installed in your home directory, you'll need
                        to specify it's location. Defaults to ~/snap/bin/gpt.
-n N_PROCESSES, --n_processes N_PROCESSES
                        Choose a number of paralell processes, defaults to 4.
-v, --verbose         Like a chatty script? Use this flag.
-e, --show_error      Print SNAP gpt output, including detailed error
                        messages.
```

3.3.2 An example run

Let's say we wanted to pre-process an ALOS-2 level 1.1 file from JAXA (`0000149017_001001_ALOS2117217000-160724.zip`), , considering only the HV polarisation. The command to submit would be:

```
python /path/to/approach/preprocess.py -p HV 0000149017_001001_ALOS2117217000-160724.
↪zip
```

If we wanted to specify an output directory that is different to the present working directory, and to print script progress, the command might be:

```
python /path/to/approach/preprocess.py -p HV -o /path/to/output_directory/ -v␣
↪0000149017_001001_ALOS2117217000-160724.zip
```

To apply a speckle filter, use the `-f` flag:

```
python /path/to/approach/preprocess.py -p HV -f -o /path/to/output_directory/ -v␣
↪0000149017_001001_ALOS2117217000-160724.zip
```

If we had multiple files to process, we can use wildcards to select all the files we want to pre-process as follows:

```
python /path/to/approach/preprocess.py -p HV *ALOS*.zip
```

For some flavours of ALOS data (e.g. from ASF), you may need to unzip the directory for SNAP to understand the data. In this case, point SNAP to the *VOL* file.

Note: The SNAP graph processing tool can throw up a lot of errors. There shouldn't be much reason to be concerned by many of these, but do check that outputs look reasonable.

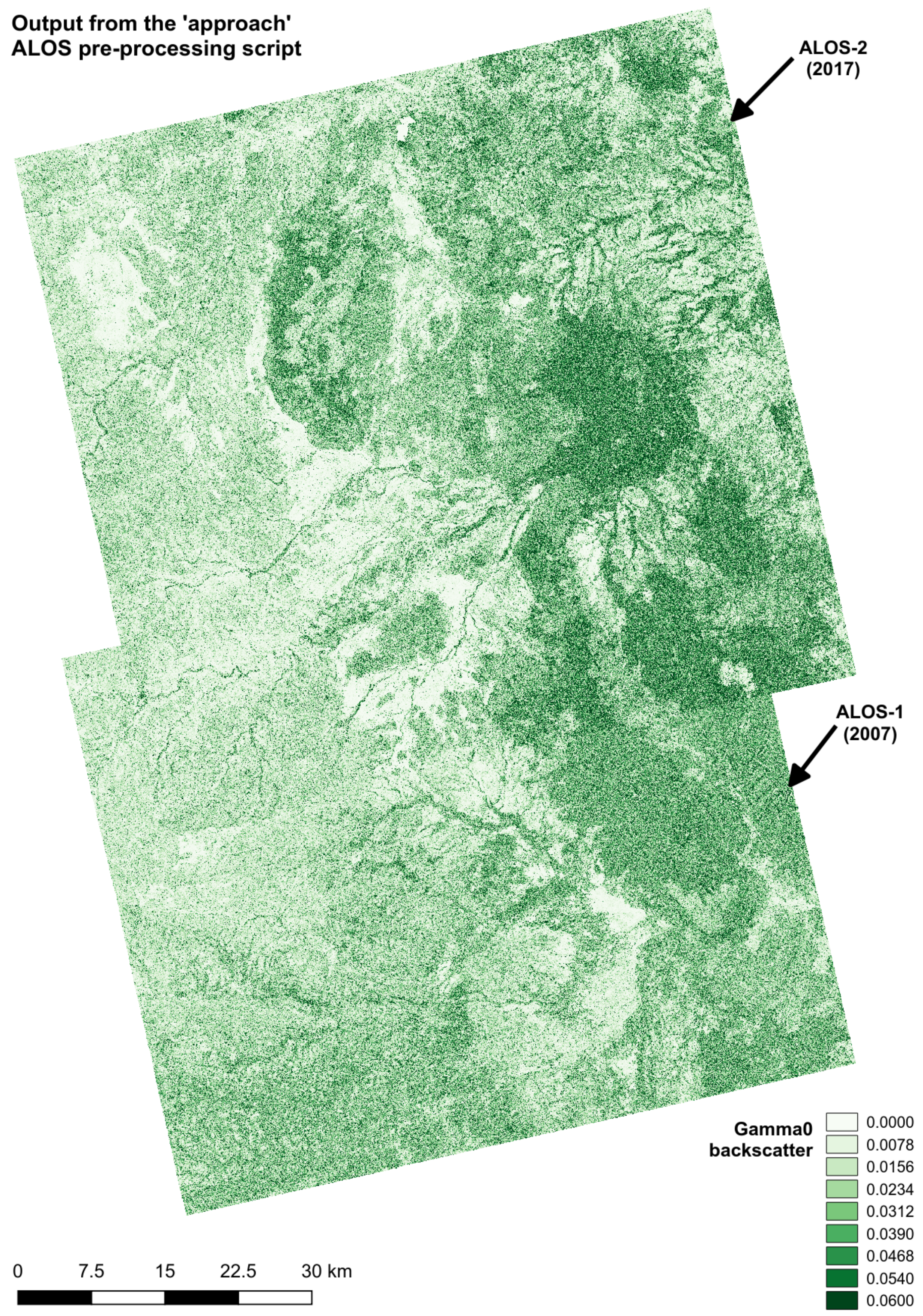
The script will take between 5 mins and 12 hours to run. The reasons for the disparity between tiles is still very unclear, but somehow relates to the underlying SNAP preprocessing chains.

3.3.3 Output

The script outputs geotiff files, and will process files separately for each polarisation.

3.3.4 Visualising in QGIS

You should expect outputs that look a bit like this when viewed in a GIS:


Output from the 'approach'
ALOS pre-processing script
ALOS-2
(2017)
ALOS-1
(2007)
Gamma0
backscatter
0.0000
0.0078
0.0156
0.0234
0.0312
0.0390
0.0468
0.0540
0.0600
0
7.5
15
22.5
30 km

3.4 Mosaicking ALOS data on the command line

Approach also has a script to generate mosaics of ALOS data, which has a fairly small tile size. Data from radar instruments is often very nosiy, so averaging data from multiple scenes can also be beneficial. Showing help ————

Help for `mosaic.py` can be viewed by typing `python /path/to/approach/mosaic.py --help`:

```
usage: mosaic.py [-h]
                [-te TARGET_EXTENT TARGET_EXTENT TARGET_EXTENT TARGET_EXTENT]
                [-tr TARGET_RESOLUTION] [-e EPSG] [-o PATH] [-v] [-y YEAR]
                N [N ...]

Generate a mosaic of scenes from ALOS-1 or ALOS-2 imagery.

Required arguments:
N                      Input files. Either specify a valid ALOS-1/2 input
                        file, or multiple files through wildcards.
-te TARGET_EXTENT TARGET_EXTENT TARGET_EXTENT TARGET_EXTENT, --target_extent TARGET_
↪EXTENT TARGET_EXTENT TARGET_EXTENT TARGET_EXTENT
                        Specify image bounds as xmin ymin xmax ymax.
-tr TARGET_RESOLUTION, --target_resolution TARGET_RESOLUTION
                        Specify an image resolution for the output image
-e EPSG, --epsg EPSG  Specify the EPSG code of the output projection..

Optional arguments:
-o PATH, --output PATH
                        Optionally specify an output directory or file. If
                        nothing specified, we'll apply a standard filename and
                        output to the present working directory. Note: you
                        cannot process multiple input files with a single
                        output file name.
-v, --verbose          Like a chatty script? Use this flag.
-y YEAR, --year YEAR  Optionally specify a single year to process. If not
                        used, all years represented in infiles will be
                        processed.
```

3.4.1 An example run

Let's say we had processed two ALOS scenes, with filenames `ALOS1_HV_20070702_ALPSRP076546990.tif` and `ALOS1_HV_20070702_ALPSRP076547000.tif`, and that we wanted to combine these two adjacent scenes into a single image. An example command to submit would be (given a known output extent):

```
python /path/to/approach/mosaic.py -te 1100000 8900000 1210000 9050000 -tr 25 -e␣
↪32736 ALOS1_HV_20070702_ALPSRP076546990.tif ALOS1_HV_20070702_ALPSRP076547000.tif
```

If we wanted to specify an output directory that is different to the present working directory, and to print script progress, the command might be:

```
python /path/to/approach/mosaic.py -te 1100000 8900000 1210000 9050000 -tr 25 -e␣
↪32736 -o /path/to/output_directory/ -v ALOS1_HV_20070702_ALPSRP076546990.tif ALOS1_
↪HV_20070702_ALPSRP076547000.tif
```

If we had multiple files to process, we can use wildcards to select all the files we want to pre-process as follows:

```
python /path/to/approach/mosaic.py -te 1100000 8900000 1210000 9050000 -tr 25 -e␣
↪32736 ALOS1*.tif
```

If we had multiple files to process, but only wanted to process those from a single year:

```
python /path/to/approach/mosaic.py -te 1100000 8900000 1210000 9050000 -tr 25 -e 32736 -y 2007
ALOS1*.tif
```

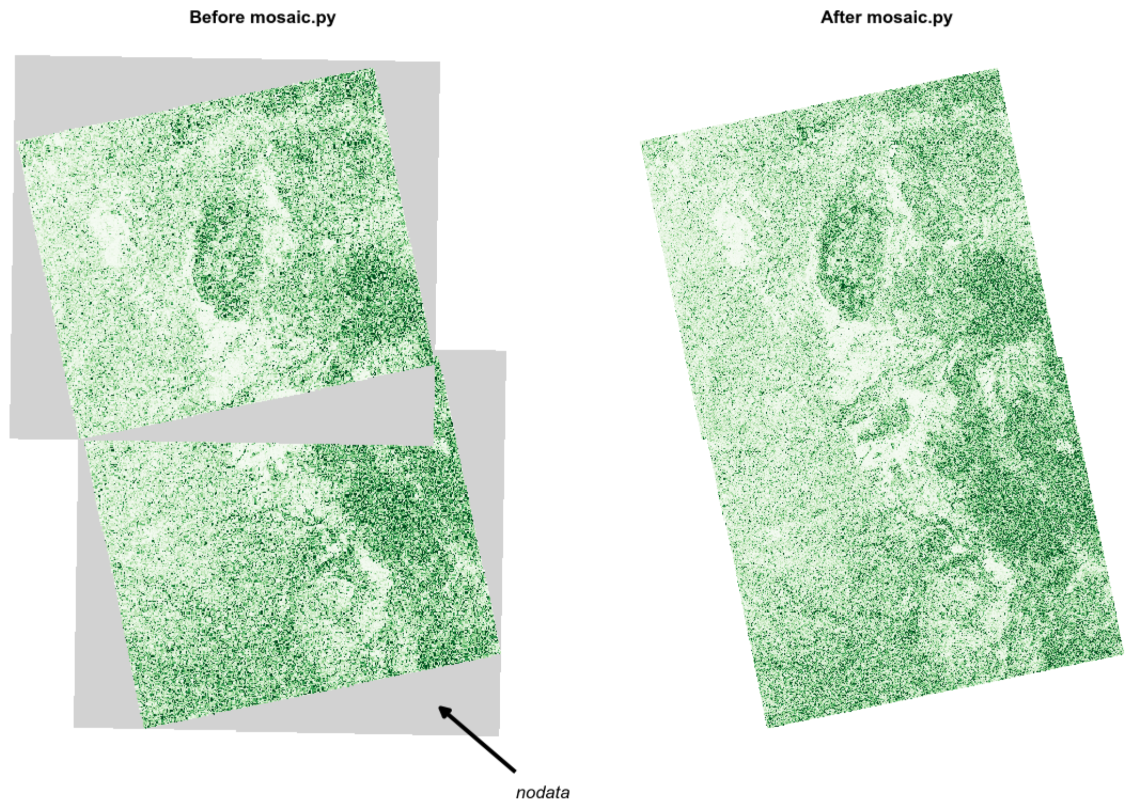
Note: Adjacent tracks of ALOS data are often poorly calibrated, the result of variability in the scene, particularly soil moisture. This package doesn't fix this, but consider histogram matching as a potential solution.

3.4.2 Output

The script outputs two geotiff files. The first (`g0_av*.tif`) records the mean gamma0 backscatter from all input scenes. The second (`nviews*.tif`) records the number of scenes contributing to each pixel in the output image.

3.4.3 Visualising in QGIS

You should expect outputs that look a bit like this when viewed in a GIS:



3.5 Under the bonnet

If you're interested in how this script works, or would like modifications, here I'll describe how it works.

3.5.1 What is going on?

`preprocess.py` is simply an elaborate way of calling the SNAP graph processing tool (gpt) with pre-determined processing chains. The Python element deals with different input data (ALOS-1/ALOS-2 data), and automates the creation of new file names. The processing chains are stored in the `approach/cfg/` directory, and these are used by the Python script by specifying the flags `-Pinputfile` and `--Poutputfile`. For example, to process an ALOS-1 file the Python script may submit the command:

```
~/snap/bin/gpt approach/cfg/SNAP_chain_A1.xml -x -Pinputfile=~/ALOS_data/
↪ALPSRP076546990-L1.1.zip -Poutputfile=ALOS1_HV_20070702_ALPSRP076546990.tif
```

and to process an ALOS-2 file the Python script you would submit:

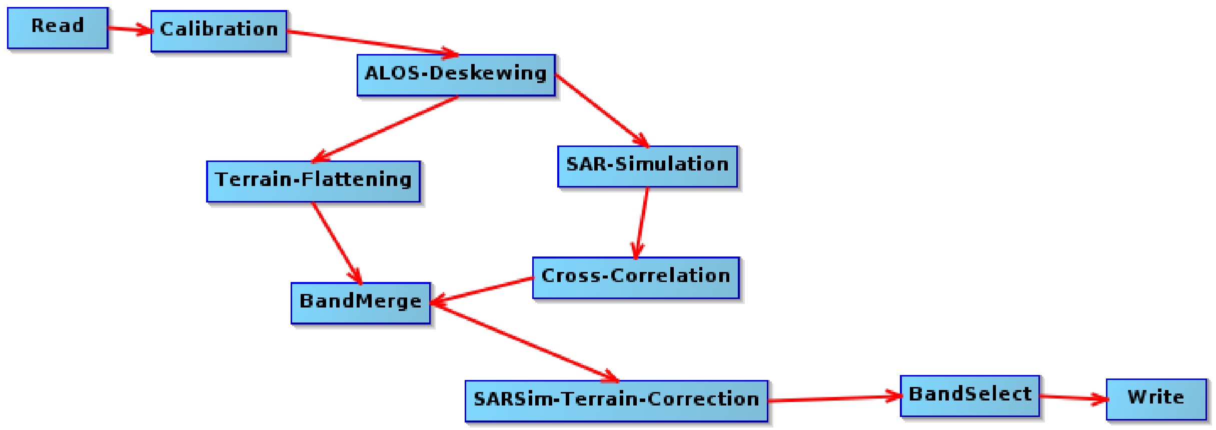
```
~/snap/bin/gpt approach/cfg/SNAP_chain_A2.xml -x -Pinputfile=~/ALOS_data/0000149017_
↪001001_ALOS2117217000-160724.zip -Poutputfile=ALOS2_HV_20160724_ALOS2117217002-
↪160724.tif
```

If you want to use a different pre-processing chain, the place to begin would be to modify the files `approach/cfg/*.xml` using the SNAP Graph Builder. I can recommend a look through the SNAP forum to get a better idea of how the graph processing tool works.

`mosaic.py` is a set of commands to assist in the automation of generating mosaic products. The script relies heavily on GDAL/OGR to reproject images to a single output CRS, and uses numpy to generate a mean output image.

3.5.2 What is the processing chain for ALOS-1?

The processing chain for ALOS-1 looks like this:



The steps are:

1) *Calibration*: Radiometrically corrects the SAR image so that pixel values represent the radar backscatter of the reflecting surface.

2) *ALOS deskewing*: Required only for ALOS-1 data, as the data is distributed in a squinted geometry. Deskewing transfers the data into a zero Doppler geometry, which is necessary before applying standard SAR processing.

3) *Terrain-Flattening*: This step improves radiometric calibration by removing topographic effects. This will, for example, correct for the tendency for surfaces that area angled towards the sensor to be recorded as brighter.

4) *SAR-Simulation*: This step simulates the image that would be expected given topographic effects alone.

5) *Cross-Correlation*: The recorded backscastter is correlated with that expected from topography. Step 4 must be conducted in parallel with step 3, as radiometric terrain flattening removes the very terrain features that SARSim requires to align images to the DEM.

6) *BandMerge*: Images from steps 3 and 4/5 are added to a single image.

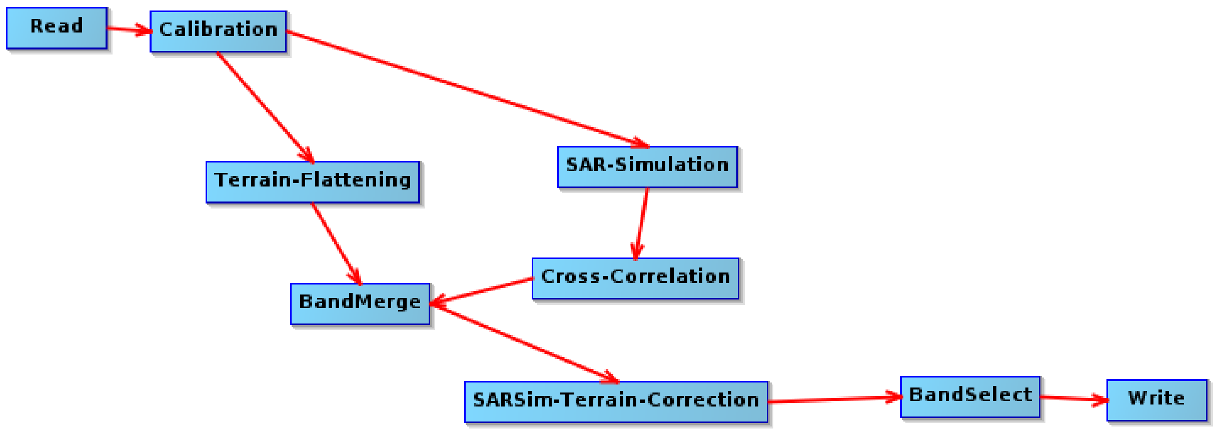
7) *SARSim-Terrain-Correction*: Terrain Correction will geocode the image by correcting SAR geometric distortions using a DEM and producing a map projected product. We use SARSim TC as the geometric accuracy when using Range-Doppler TC (which uses dead reckoning) is otherwise poor. SARSim TC warps the image to the expected topographic features. It may not work well given very flat topography.

8) *BandSelect*: Selects the band to output.

9) *Write*: Writes output to a geotiff file.

3.5.3 What is the processing chain for ALOS-2?

The processing chain for ALOS-2 looks like this:



The processing chain we have selected for ALOS-2 is identical to ALOS-1, but omits the ALOS-deskewing step which is not required for ALOS-2.

We still perfrosm SARSim terrain correction, though note that Range Dopper terrain correction is probably more appropriate for ALOS-2 with its better geolocation accuracy. The reason we still use SARSim terrain correction is so that the outputs better match those of ALOS-1. If we compare the outputs of Range Doppler TC for ALOS-2 and SARSim TC for ALOS-1, we find that shifts between the two images are much greater than if we accept the additional error and processing time introduced by SARSim TC.

3.5.4 What about speckle filtering?

Speckle filtering is handled by two additional .xml files (SNAP_chain_filter_A1.xml/SNAP_chain_filter_A2.xml). These insert a lee-filter operator before the terrain flattening step. In some cases this can dramatically increase the processing time, for reasons that aren't clear yet.

3.5.5 How can this be improved?

This is by no means the perfect solution. A more complete processing chain for pre-processing ALOS-1 and ALOS-2 data might:

- Include an additional step to warp images to a reference image to improve comparability.

- Include a method to cross-calibrate backscatter from ALOS-1 and ALOS-2, which are at present do not appear to be calibrated identically (histogram matching?).
- Do something about the profusion of error messages from SNAP and snappy :/.

3.5.6 What other options do I have?

ALOS data can also be processed using ASF MapReady (no longer updated) or GAMMA (commercial). Both are probably more straightforward than this approach, but this script is free and (at the time of writing, Jan 2018) up-to-date.

A pre-processed ALOS mosaic product is made freely available from JAXA, and geolocation accuracy is excellent. However you will have to accept that the dates imagery come from are not always ideal, and that you cannot take the average of multiple images to reduce the impacts of speckle.

If you are interested in using the ALOS mosaic product, we've also built a Python module to assist in the ingestion and processing of this data. Its available at:

https://bitbucket.org/sambowers/biota/

CHAPTER 4

Indices and tables

- genindex
- modindex
- search