How to make an InSAR time series from Sentinel-1 TOPS data: Kilauea

Introduction

This document presents a step-by-step recipe for creating InSAR time series from Sentinel-1 TOPS data. We aim to guide you, the user, through system set-up, downloading necessary data and orbit files, coregistering chosen images, creating interferograms, unwrapping those interferograms, and running the Small BAseline Subset Analysis (SBAS) to create the final time series. In the example below, we chose a region of interest on the big island of Hawaii to look at the deformation surrounding the eruption of Kilauea and the ensuing earthquake that occurred there in 2018. This recipe can be adapted to any region of interest, and we include tips and tricks that apply to various types of areas (both areas of good coherence and poor coherence).

To begin this tutorial, we wish to prepare the user for the path ahead with three tips:

(1) This document assumes you have a working knowledge of UNIX or LINUX systems. While we provide quick commands to accomplish the required steps in most cases, if you are new to UNIX/LINUX, we suggest you run through a tutorial of these systems so you are prepared (http://www.ee.surrey.ac.uk/Teaching/Unix/).

(2) All updated software and help pages are located at Github under GMTSAR (https://github.com/gmtsar/gmtsar). If you find yourself troubleshooting, or having any problems, check out the issues tab at this site to see if someone else has encountered your issue (https://github.com/gmtsar/gmtsar/issues).

(3) We also suggest that you make an account with the Alaska Satellite Facility (ASF) and with Copernicus (for Sentinel-1 orbit files) ahead of time for downloading data (you will need your username and password in the steps below).

    If you haven't already done so, visit the ASF EarthData page and click on their VERTEX tool to find the sign in/register page:
    https://asf.alaska.edu/

    For European Space Agency's Copernicus System, visit:
    https://scihub.copernicus.eu/dhus/#/home

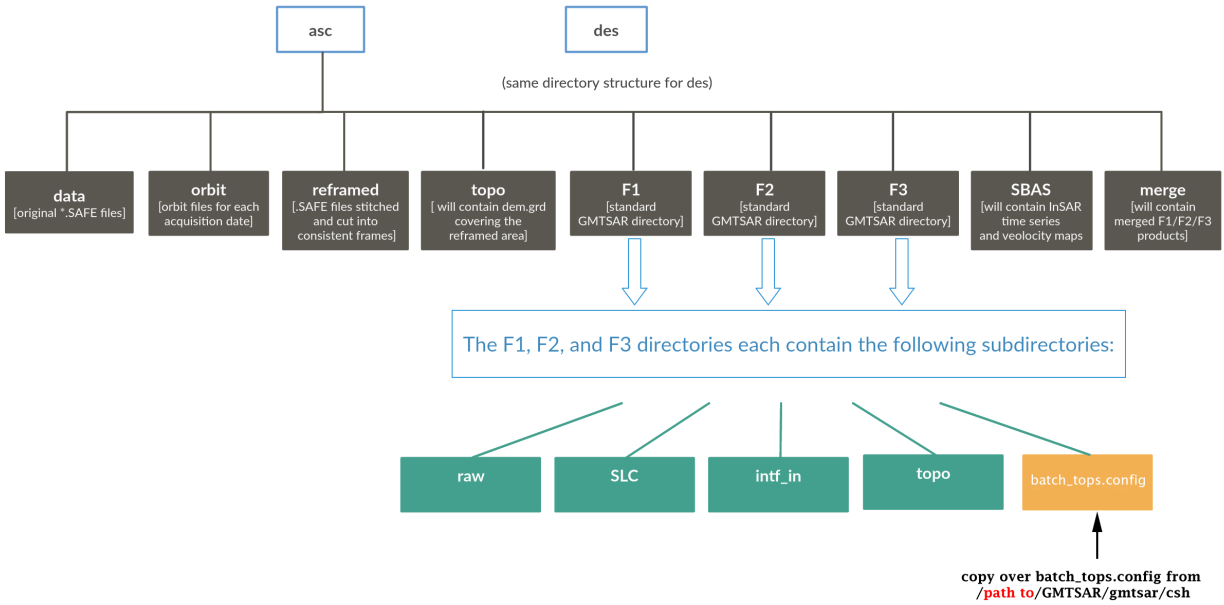
Lastly, remember to enjoy the fact that we can visualize minute displacements on the Earth's surface through the use of orbiting satellites that measure radar range pulses like a bat from outer space. There is a remarkable amount of high-quality data these days that we can gain access to, and it is amazing how we can use these data to study our planet's various changes. Creating an InSAR time series is just one of those ways we can study our planet from space, so go forth and experiment and discover!

This document was updated and prepared by Katherine Guns and Julie Gevorgian in October 2020, with assistance from Xiaohua Xu and David Sandwell. [Last updated 02/2022.]

Table of Contents

1. **General Setup**

   We need to start by setting up our directories in the necessary pattern. Make a top level **directory structure** for ascending or descending data all the following directory names:



[*Note:* Depending on how GMTSAR is installed, a copy of batch_tops.config can be found in: /usr/local/GMTSAR/gmtsar/csh, though the path to the GMTSAR directory can vary. If you cannot locate it, visit the Github repository for GMTSAR to download a copy to place in your F1,F2,F3 directories.]

2. **Preparing the Topography DEM Grid Files**

   Prepare a **topography grid** (dem.grd)

   a. Using the DEM generator web interface (http://topex.ucsd.edu/gmtsar/demgen/), prepare a topography grid for your area of interest (lon -157.0 to -154.2, lat 18.0 to 20.4 for Kilauea). This will download as dem.grd, along with a kml companion file. View the kml file in Google Earth and verify that it completely covers the area (as shown in the image following 3c).

   [*Note:* The DEM generator can only generate 4 degree wide topography areas at a time. In addition, if your area is near the equator, you cannot have a dem area straddling the equator--you need to download above and below the equator areas separately. To stitch multiple dem grids together, we recommend using gmt grdpaste (see examples here: http://gmt.soest.hawaii.edu/doc/latest/grdpaste.html)]

   b. Move the dem.grd file to your upper level **topo** directory (inside **asc** or **des** directories).

c. Next link the dem.grd file to a **topo** directory inside the F1/F2/F3 directories.. Create a symbolic link in each:

```
cd asc/F1/
mkdir topo
cd topo
ln -s ../../topo/dem.grd .
```

d. Also link the dem.grd to the **merge** directory (doesn't need to be inside a topo folder here).

```
cd asc/merge/
ln -s ../topo/dem.grd .
```

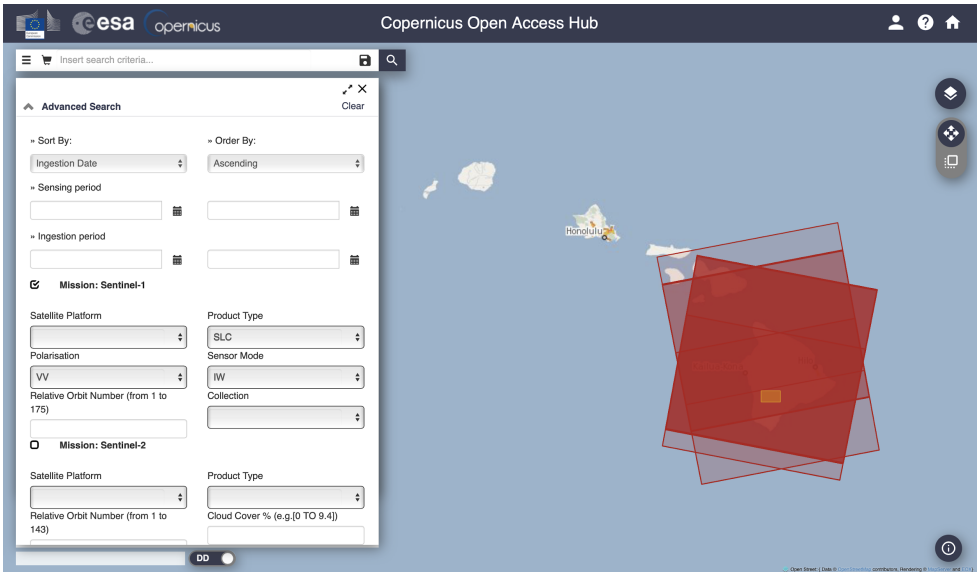
3. Data Selection and Download for Region of Interest

a. Data Selection: Plan your overall processing strategy

i. Go to the Sentinel open Copernicus data hub GUI, login or make an account, and search for data: (https://scihub.copernicus.eu/dhus/#/home)

ii. Use the Pan and Box tools (top right corner) to zoom in on the area of interest, for example, The Big Island, Hawaii (lon -157.0 to -154.2, lat 18.0 to 20.4, an area that spans the entire frame for making the topography). Draw a box over this area.

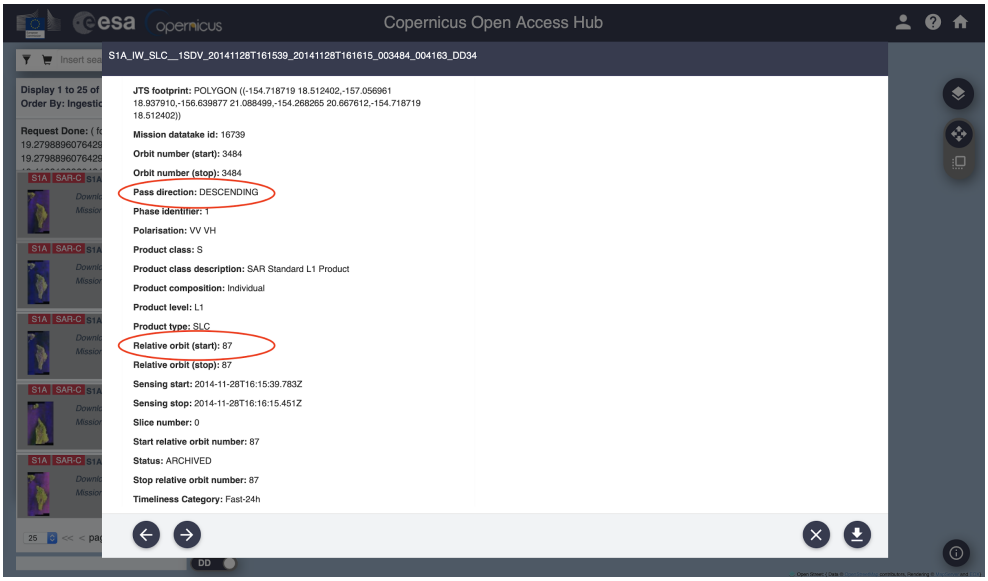
iii. Using the drop down search menu (three horizontal lines icon), set the search parameters for the sensing dates, product type (SLC), polarisation (VV) and the sensor mode (IW). As you can see in the example below, we had ordered the search results by ascending frames (this can be changed to descending). Ascending frames are oriented NW/SE while descending frames are oriented NE/SW.



iv. From the search results, select one descending frame and one ascending frame and take note of the relative orbit numbers of each (in this case, 124 for ascending and 87 for descending). This is found if you click on the eyeball icon on the individual scene (left hand side of the screen) and click on the "product" drop-down menu (see screenshot below). You'll want to download data from ascending and descending tracks separately, since the data files for each will go into the different **asc** and **des** directories you created before.

v. All instructions below are for downloading and processing ascending data specifically, but they can be replicated for descending data.

[*Note*: Images taken in the afternoon may have more ionospheric/atmospheric turbulence and noise, as the sun has had longer to heat up the ionosphere and atmosphere over the course of a day. Therefore, images taken in the morning often are less noisy. Keep track of when your region of interest images are measured by noting the time ("Sensing start" and "Sensing end") shown in the "product" drop down menu.]



b. Download Data (2 Options):

i. **Direct Download:** To **download directly** from the Copernicus hub, do the data search again but this time include in your search one of the relative orbit numbers that you just found (for either ascending or descending), along with the same parameters as before (desired sensing dates, polarisation= VV (HH for Antarctica), product type = SLC, sensor mode = IW). Then manually click on each result and download. Put these .zips

into the corresponding data directory that you created earlier (**asc/data** or **des/data**).

Note: Go take a break and get a coffee because downloading these may take awhile (~1 hour per file) depending on your connection--a wired connection should go faster.

OR

ii. **Automated download** method using ASF API (Alaska Satellite Facility)

1. If you are a new user, you will need to set up a user account and accept the license agreement.
2. Generate URL command with web API Query (https://asf.alaska.edu/api/) and use the wget command to download the results (see example scripts below for Kilauea data).

[*Note:* the ASF API is one choice you can use to automate data download--in addition you can also use the European Space Agency's (ESA) Copernicus API or the SSARA UNAVCO API systems. We only include an example for the ASF API below. We choose the ASF API mainly because we are located in the United States and so is ASF, so the download times are shorter than if using ESA's Copernicus API].

Sample scripts for Kilauea data: You will need to enter your own intended path to **data** storage location (**asc/data**) and your own username and password (highlighted in the scripts)

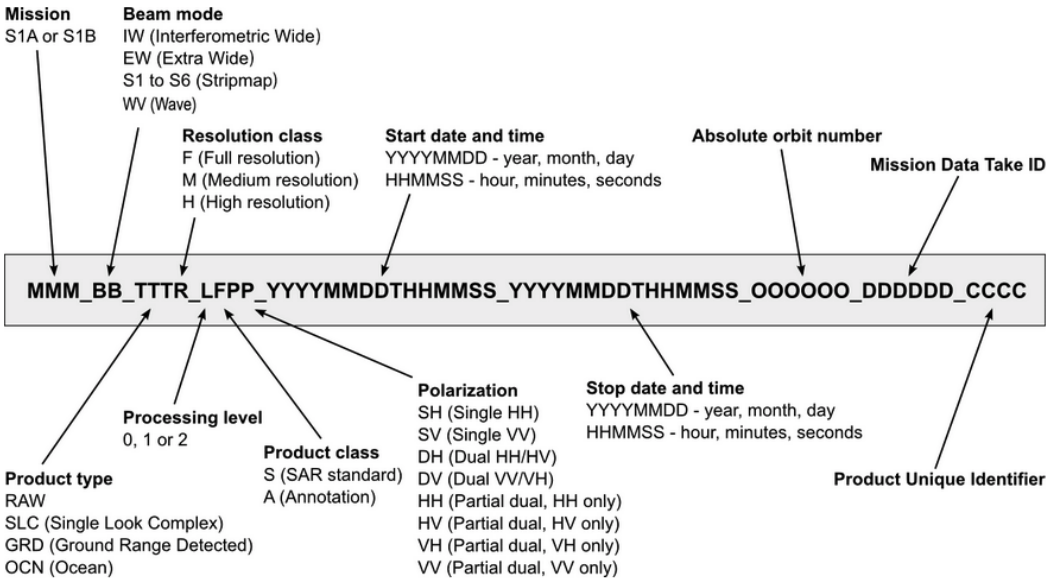
```
#!/bin/bash
#
# search the ASF site to determine all data inside of a polygon and make a csv
    file listing file names
#

cd  path to /asc/data
cmd="https://api.daac.asf.alaska.edu/services/search/param?platform=Sentinel-
    1A,Sentinel-1B&polygon=-155.64625964,19.3937790141,-155.52834329,19.
    1922612441,-154.994620338,19.4236985176,-155.135252149,19.64533461
    1,-155.64625964,19.3937790141&processingLevel=SLC&relativeOrbit=124&
    output=csv"
#
echo $cmd | xargs curl  > asf.csv
#
cat asf.csv | awk -F"," '{if (NR>1) print $27}' | awk -F"'" '{print $2}' > data.list
```

```
#!/bin/csh
#
#  download the scenes from the data.list file that was created by above script
#
cd  path to /asc/data
foreach file (`awk '{print $1}' data.list`)
 set name = `echo $file | awk -F'/' '{print $6}' | awk -F'.' '{print $1".SAFE"}'`
 if ( ! -e $name ) then
  echo $name
  wget --http-user=**** --http-password=***** $file
 endif
end
```

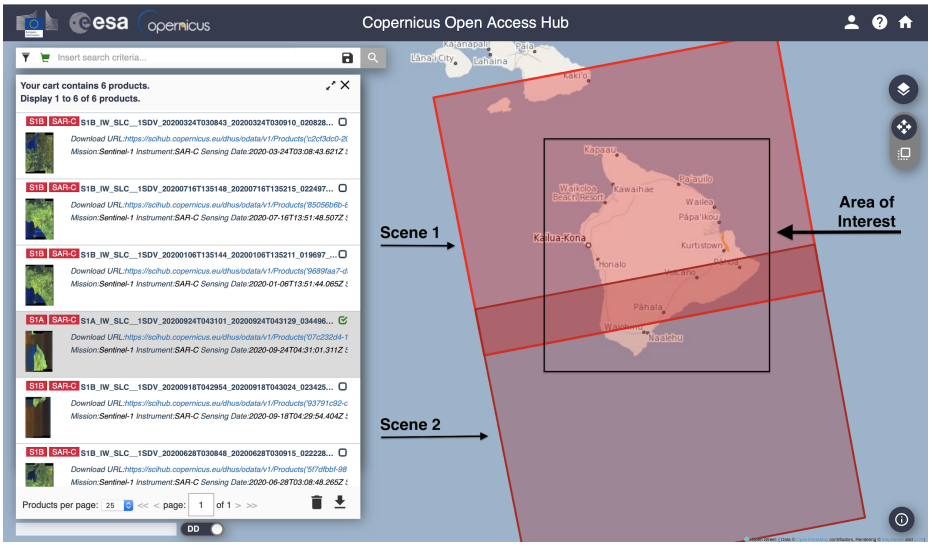
3. Unzip all files and remove the .zip files. Note, all Sentinel-1 directories have a **.SAFE** suffix
4. When you download the data, each *.SAFE file will have a very long name. The below screenshot from the ASF website (https://asf.alaska.edu/data-sets/sar-data-sets/sentinel-1/sentinel-1-data-and-imagery/) explains what is in this name:

Naming Convention

The figure below illustrates the Sentinel-1 granule naming convention. For more information, see the Sentinel-1 Technical Guide. Additional resources are listed in Documents and Tools.



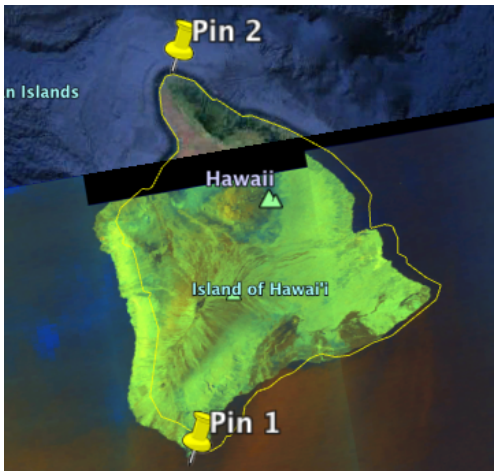
c. **Adjusting Region of Interest:** Stitching multiple frames and/or cropping one frame to define a smaller region.



i. You may need to **re-assemble frames** together because the area of interest spans two frames. If you have *.SAFE files with names that share a common date and must be stitched, stitch them by using pins in Google Earth.

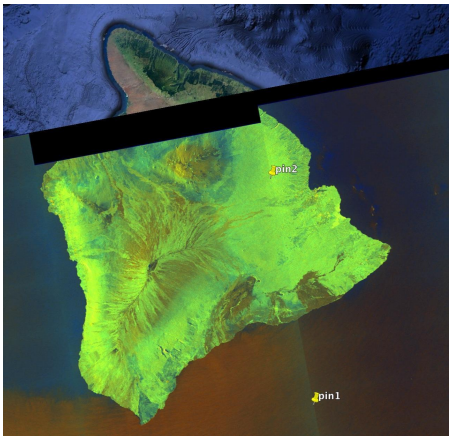
1. Open the kml preview found in data/*.SAFE/preview/map-overlay.kml

2. Drop two pins ordered in the **along-track direction.** Note that this will differ for **asc** vs. **des** directories. The example photo below is the pin order for ascending, but a descending orbit would require swapping pins 1 and 2. **Please note, that your pins must fall within your data frames**, the image below just shows the order of the pins, and is not accurately representing data coverage.



3. If you are only interested in a small area within a single frame, the pins can be placed within one frame instead (see example below), and the regions above and below these constraints will be cut off. This is helpful for making processing run more quickly.

   In this case, pin 1 could be somewhere south of Kilauea (ocean), and pin 2 could be on land, in the ascending orbit path direction (see below). For descending, pin 1 could be somewhere due-North of Hilo (ocean), and pin 2 could be somewhere on land near Kilauea.



iii. Export coordinates of the pins to a file called pins.ll and save this file in the **reframed** directory.

Example file coordinates for ascending might be:

-155.13 18.95
-155.30 19.75

Example file coordinates for descending might be:

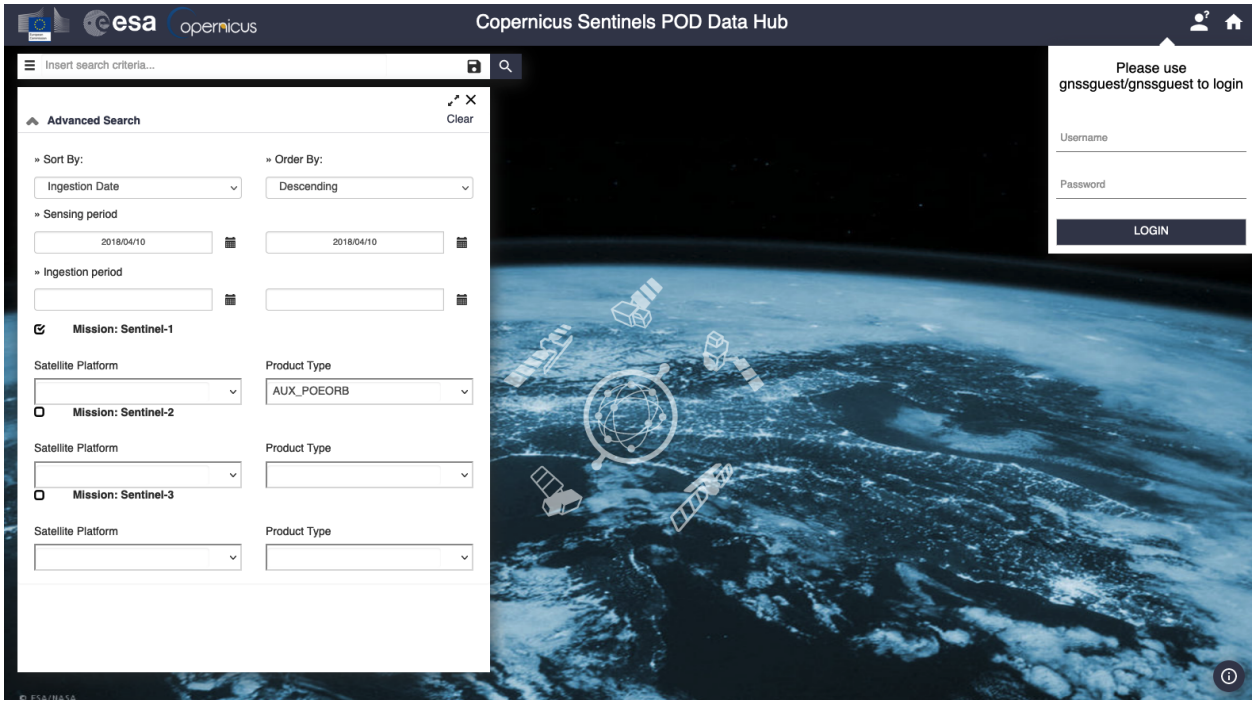
-155.50 19.65
-155.68 18.93

4. Downloading Precise (and/or Restituted) Orbits

a. Navigating the ESA Website

The ESA Copernicus HUB website has the orbit files.
(https://scihub.copernicus.eu/gnss/#/home)
The type of orbit file (in this case, precise orbit files) can be chosen by filtering a search (see screenshot below) by choosing the AUX_POEORB from the dropdown menu. Also input your sensing date window (the time period of the scene you are looking for). When you click the search button, the system will ask you to sign

in -- the signin information is provided (username = gnssguest password = gnssguest)



b. Downloading Orbit Files through Command Line

[*Note:* This example uses the ASF service, but a similar approach can be used to download orbit files through the ESA Copernicus service.]

i. Create a file in the **data** directory called "SAFE_filelist", and input a list of full paths to all of your .SAFE files in order of acquisition date:

```
cd data
ls -d $PWD/*.SAFE > SAFE_filelist
less SAFE_filelist
```
(to look at it and check, should look like:)

```
/ full path to /asc/data/S1A_IW_SLC__1SDV_20180219T043025_20180219T043053_020671_023679_5817.SAFE
/ full path to /asc/data/S1A_IW_SLC__1SDV_20180315T043025_20180315T043053_021021_024191_5F9B.SAFE
/ full path to /asc/data/S1A_IW_SLC__1SDV_20180327T043025_20180327T043053_021196_024722_A9DB.SAFE
...
```

ii. The next step requires use of the organize_files_tops.csh script, which searches for and downloads the appropriate **precise** orbit files for you. To use this script, add the following to your .cshrc or .tcshrc file:

1. `vi ~/.cshrc`
2. Copy and paste the below line to create an alias called "wgetasf", but use your credentials. Don't forget the ' quotes '. Beware of special characters in your password as well.

```
alias wgetasf ' wget --http-user=**** --http-password=***** '
```

where the username and password are those for your ASF account. **Remember to close and open a new terminal after doing so, or to source your cshrc file like so:**

```
source ~/.cshrc
```

*We have encountered issues with previous users that are unable to use this alias. If this alias term does not work for your shell, try adding this alias line in your organize_files_tops.csh right at the beginning or, if that still doesn't work, navigate to where the wgetasf lines appear in your organize_files_tops.csh and replace those lines manually with:

```
wget --http-user=***** --http-password=*****
```

Where the **** are your actual username and password for your ASF account.

iii. Run the script in two steps in the **data** directory.

- MAC USERS:

First run with mode 1 (~5 min run time):

```
organize_files_tops.csh SAFE_filelist ../reframed/pins.ll 1 &> oft_mode1.log &
```

Then with mode 2 (~35 min run time):

```
organize_files_tops.csh SAFE_filelist ../reframed/pins.ll 2 &> oft_mode2.log &
```

- LINUX USERS:

First run with mode 1 :

```
organize_files_tops_linux.csh SAFE_filelist ../reframed/pins.ll 1 &> oft_mode1.log &
```

Then with mode 2:

```
organize_files_tops_linux.csh SAFE_filelist ../reframed/pins.ll 2 &> oft_mode2.log &
```

- ALL USERS:

To create the log file in the above commands we are sending them to the background processing. To see the screen output for these commands, just tail the log file like this:

```
tail -f filename.log
```

[**Note:** We strongly recommend running all commands with an output log file (e.g. `command &> yourlogfile.log &`). This makes it much easier to debug, if you encounter an issue. If you want to print everything to standard screen output, just leave off the "&> oft_mode.log &" part of the commands above.]

iv. Now a new directory within the **data** directory named Fxxxx_Fxxxx has been created and populated with new stitched/trimmed *SAFE data. Sometimes, when the timing is not exactly the same for each satellite fly

by, multiple Fxxxx_Fxxxx directories will be created. In this case, just move all the files into one of the Fxxxx_Fxxxx directories.

c. Downloading Restituted Orbits (optional)

Data that is from within the past 20 days will not yet have precise orbit files available. If you want to use newer data, then you need to download the appropriate restituted (RESORB) orbit files and use the **create_frame_tops.csh** script instead.

i. Using ESA (https://scihub.copernicus.eu/gnss/#/home search for AUX_RESORB) or ASF (https://s1qc.asf.alaska.edu/) download the appropriate orbit files that correspond to each .SAFE data file. Note that ESA orbit files will likely be shifted by 1 day compared to ASF's orbit files, so we recommend that ASF orbits be used.

Orbit files must span the acquisition date and time. For example, for SAR data file

S1A_IW_SLC__1SDV_20180514T043027_20180514T043055_021896_025D31_C20C.SAFE ,

Where the start date is 05/14/2018, and the start time is 04:30:27, and the end date and end time are 05/14/2018, 04:30:55.

Download orbit file:

S1A_OPER_AUX_RESORB_OPOD_20180514T065858_V20180514T024649_20180514T060419.EOF

Which has the start date of 05/14/2018 and start time of 02:46:49 and the end date and time of 05/14/2018 and 06:04:19, and these times completely encompass the time that the InSAR scene was collected.

ii. Do this for each *.SAFE you have that requires a restituted orbit file and place these orbit files (*.EOF) in the **asc** or **des orbit** directory .

iii. cd to the **reframed** directory and run the commands below for each *.SAFE and its corresponding *.EOF (each command is all on one line):

1. To create the **SAFE_list** for the second command, print out the full path name for the particular *.SAFE file you are working on for that orbit file, and repeat as necessary for each of the files you need restituted orbits for.

```
ln -s
/full-path-to/asc/orbit/S1A_OPER_AUX_RESORB_OPOD_20180514T065858_V20180514T024649_2018
0514T060419.EOF .

create_frame_tops.csh  SAFE_list
S1B_OPER_AUX_POEORB_OPOD_20180609T110546_V20180519T225942_20180521T005942.EOF
```

iv. Move the resulting *.SAFE file into the same Fxxxx_Fxxxx folder (see above step c) where the rest of the precise *.SAFE data was placed.

v. Repeat steps iii and iv for each *.SAFE you have that requires a restituted orbit file.

5. Choosing Master/Reference Image and Aligning

a. Selecting a Reference Image from a Baseline Plot

i. Go to the **F1/raw** directory. Link the data and orbit files that belong to this (F1) subswath. Also link the dem.grd.

```
ln -s ../../data/F*/*.SAFE/*/*iw1*vv*xml .
ln -s ../../data/F*/*.SAFE/*/*iw1*vv*tiff .
ln -s ../../data/*EOF .
ln -s ../topo/dem.grd .
```

Note: "iw1" files correspond to the F1 subswath, "iw2" files correspond to the F2 subswath, etc. When you repeat this step for the **F2**/**F3** directories, change the 1's to 2's or 3's!

ii. Now you will need to prepare a file called **data.in** that has the data file names (no suffix) and orbit file names organized as follows, separated by a colon:

```
<DataFileName:DataFileName2:...:OrbitFileName>
E.g., for just one data file for one orbit name:
s1a-iw1-slc-vv-20180207t043037-20180207t043048-020496-0230e3-005:S1A_OPER_AUX_
POEORB_OPOD_20180227T120553_V20180206T225942_20180208T005942.EOF
s1a-iw1-slc-vv-20180219t043037-20180219t043048-020671-023679-005:S1A_OPER_AUX_
POEORB_OPOD_20180311T120551_V20180218T225942_20180220T005942.EOF
s1a-iw1-slc-vv-20180303t043037-20180303t043048-020846-023c05-005:S1A_OPER_AUX_
POEORB_OPOD_20180323T120847_V20180302T225942_20180304T005942.EOF
s1a-iw1-slc-vv-20180315t043037-20180315t043048-021021-024191-005:S1A_OPER_AUX_
POEORB_OPOD_20180404T120811_V20180314T225942_20180316T005942.EOF
s1a-iw2-slc-vv-20180327t043038-20180327t043049-021196-024722-005:S1A_OPER_AUX_
POEORB_OPOD_20180416T120736_V20180326T225942_20180328T005942.EOF
…
```

iii. There is a script **prep_data.csh (mac) or prep_data_linux.csh (linux)** that will do this for you but only for the precise (not restituted) orbits. Simply run the command "prep_data.csh" from the **raw** directory. After running the script, open and manually provide the name of any missing restituted orbits (if you are using them).

iv. Next preprocess and align the images specified in data.in

b. Aligning Secondary Images to Reference Image

i. From the **raw** directory, run preproc_batch_tops.csh with mode = 1. This generates a table of baselines (baseline_table.dat):

```
preproc_batch_tops.csh data.in dem.grd 1 &> pbt_mode1.log &
```

[To see the screen output for this command, just tail the log file like this:]

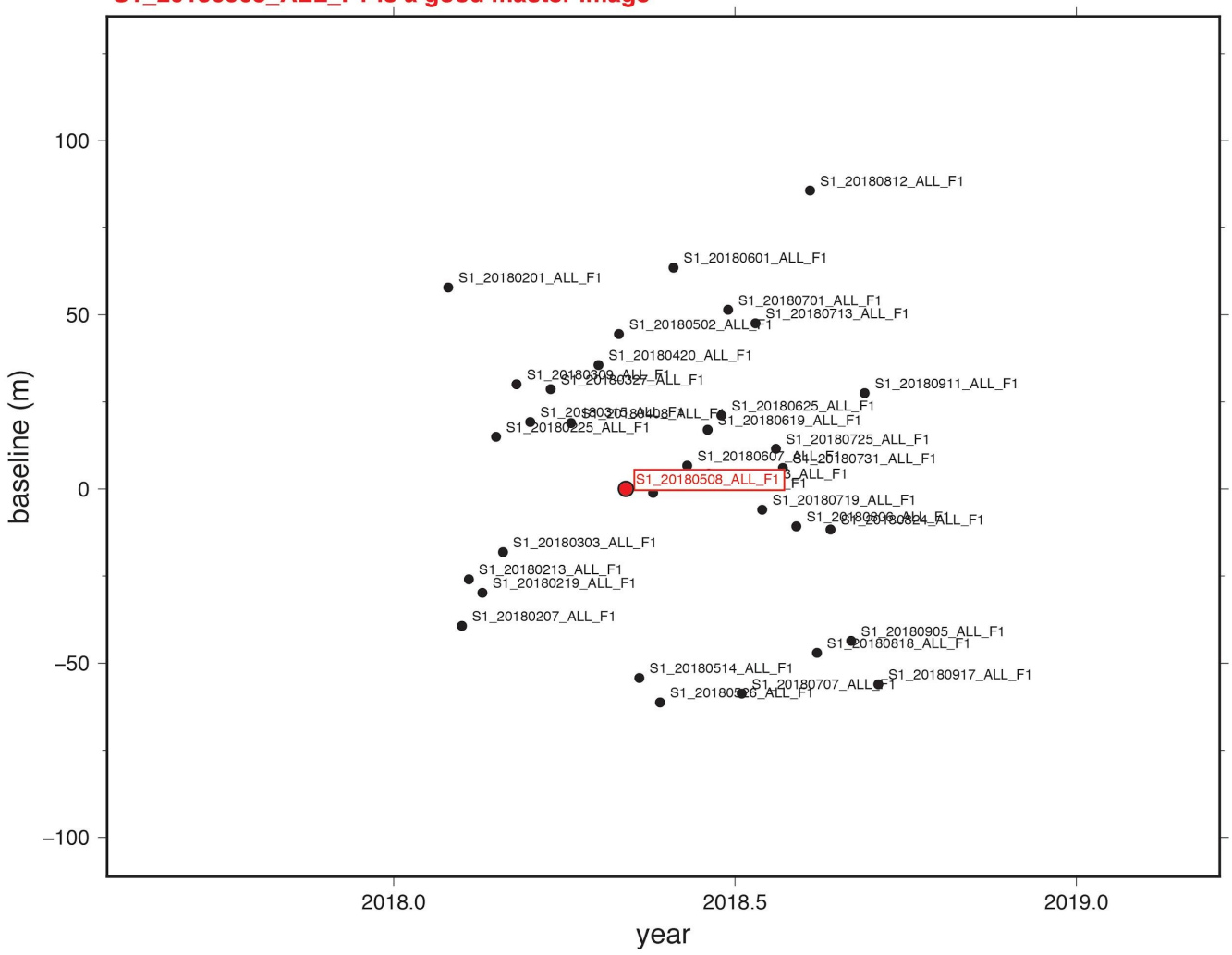
```
tail -f filename.log
```

ii. Save (move) baseline_table.dat for later use. Make sure you do this at this stage to ensure you produce the correct list of interferograms in later steps:

```
mv baseline_table.dat ../
```

iii. Check on the baseline.ps (figure showing perpendicular baselines plotted over time) to determine a master image for the stack of data. Select an image in the middle of your field of baselines. For the case below, image 20180508 is a good master since it is located in the middle of the cloud of baselines.

S1_20180508_ALL_F1 is a good master image



iv. The master you choose needs to get promoted to the **first line** in **data.in**. Open/edit this file and move (copy/delete/paste) the line designating the master (in this case, S1_20180508_ALL_F1) to the first line of data.in, so the program will treat it as the master.

v. Re-run preproc_batch_tops.csh with mode = 2. This will take roughly 1 - 3 hours per F* directory, depending on your processing power.

```
preproc_batch_tops.csh data.in dem.grd 2 &> pbt_mode2.log &
```

[To see the screen output for this command, just tail the log file like this:]
`tail -f filename.log`

vi. Repeat steps **5a** and **5b** for directories **F2** and **F3**, changing iw1 to iw2/iw3 in the links of step **5a(i)**. The same master image you selected in subswath 1 should be selected and promoted in both subswaths 2 and 3, to maintain consistency across all subswaths, but each subswath directory needs a baseline_table.dat file calculated from its specific subswath.

Note: **Re-creating Baseline Table**

vii. In the event that your **baseline_table.dat** file has the long data file format (the format from a previous version of preproc_batch_tops.csh), you need to create a **baseline_table.dat** that uses the correct data names for later processing. For this, we call the get_baseline_table.csh tool. To use this, we need to give it a list of .PRM files that we are including, as well as the name of our master .PRM file. In the F1/raw/ directory:

```
ls *ALL*PRM > prmlist     (this creates the prmlist input file)
get_baseline_table.csh prmlist S1_20180508_ALL_F1.PRM
```

This creates a new baseline_table.dat file that uses the correct file names (which we will need in the next step).

The new baseline_table.dat looks like:

```
S1_20180201_ALL_F1 2018031.1874165505 1491 19.088688436880 56.434825714407
S1_20180207_ALL_F1 2018037.1879020806 1497 -55.518698696406 -34.273719507112
S1_20180213_ALL_F1 2018043.1874162052 1503 -41.389270891689 -22.026755662146
S1_20180219_ALL_F1 2018049.1878994363 1509 -50.211739839015 -25.142105633455
S1_20180225_ALL_F1 2018055.1874136333 1515 -20.370088939888 16.909205059498
S1_20180303_ALL_F1 2018061.1878994068 1521 -45.495652914617 -13.865048265740
                                        …
```

Alternatively, you can visit the Github page and download the fresh version of preproc_batch_tops.csh and rerun step **5b**.

viii. Repeat for F2 and F3 (make sure to change the F1 in the second command line to F2/F3 as needed.)

6. Running Interferograms -- Let's see those fringes!

a. Preparing List of Interferograms

i. Go to the **F1** directory and run:

```
select_pairs.csh baseline_table.dat 50 100
```

[*Note:* Here, 50 is the temporal baseline (days) and 100 is the perpendicular baseline (meters). The output file **intf.in** then contains interferometric pairs with a timespan smaller than 50 days and a perpendicular baseline smaller than 100 meters. The contents of **intf.in** look as follows:

```
S1_20180508_ALL_F1:S1_20180514_ALL_F1
S1_20180508_ALL_F1:S1_20180526_ALL_F1
S1_20180508_ALL_F1:S1_20180607_ALL_F1
S1_20180508_ALL_F1:S1_20180520_ALL_F1
                  ...
```

To view their baselines, open up baseline.ps. To obtain a count of the interferogram pairs specified in **intf.in**, use `wc -l intf.in` where there should be 194 interferograms.]

ii. Copy the **intf.in** file from the **F1** directory to the **F2** and **F3** directories (this overwrites the **intf.in** files created by step (ii), and ensures you have the same number of interferograms in each subswath). From the **F1** directory:

```
cp intf.in ../F2
cp intf.in ../F3
```

iii. Once you have copied the **intf.in** file over, move to the **F2** and **F3** directories and edit the **intf.in** file to ensure it refers to the **F2** and **F3** directories respectively (instead of **F1**). An easy way to do this is, in the **F2** directory:

```
vi intf.in
:%s/F1/F2/g
```
(enter this command while in vi; for subswath **F3**, change **F2** to **F3**)

iv. Ensure there is a batch_tops.config file in each of the **F1**, **F2**, and **F3** directories.

v. Edit the batch_tops.config file in each directory to ensure that the master image name is set to the appropriate **F1**, **F2** or **F3** folder, e.g.:

```
master_image = S1_20180508_ALL_F2
```
(for F2 processing)

b. Run One Interferogram to Test Settings

We recommend running a single test interferogram in each subswath to (1) make sure everything is set up properly, but also to (2) optimize processing by calculating a topo_ra.grd file once, instead of many times for each interferogram. This is not mandatory, this is just a recommendation. If you decide to not complete this step, make sure you run all your interferograms in step (c) using stage 1 in the batch_tops.config file.

[*Note:* For this Kilauea example specifically, subswath **F1** happens to contain only ocean. Running the interferograms for **F1** therefore isn't necessary, but it's good practice and keeps the steps consistent between subswath directories. Two ways of checking if you'll need all 3 subswaths are by 1) reviewing the quick-look png file in the original data folder, or 2) checking the display_amp.pdf file after running step b(ii) below in each **F1/2/3** directory - if it is just noise, you can skip processing for that subswath. You can also look at the topo_ra.pdf in the **F1/2/3** topo directory]

i. Edit your batch_tops.config file to set up your first run
In **F1**, use a text editor to set some parameters:

1. Ensure the master image is edited to match your subswath, e.g.:
```
master_image = S1_20180508_ALL_F1
```

2. Set processing parameters to:
```
Proc_stage = 1
shift_topo = 0
filter_wavelength = 200
range_dec = 8
azimuth_dec = 2
threshold_snaphu = 0
threshold_geocode = 0
```

This will decimate the image 8 in range and 2 in azimuth. Keep these numbers consistent across sub-swaths (i.e., when you do this for the **F2** and **F3** directories). The two zeros for threshold_snaphu and threshold_geocode indicate that we want to skip unwrapping and geocoding as this will be done after merging the subswaths.

ii. Generate your test interferogram and necessary run files

1. We need to map topography into phase to be used by the master so edit batch_tops.config by setting proc_stage = 1 and make a single interferogram. This makes a large trans.dat file, as well as a topo_ra.pdf file that you can check.

```
head -1 intf.in > one.in
intf_tops.csh one.in batch_tops.config
```
(~5 min)

2. Once that finishes, edit batch_tops.config again and set proc_stage = 2, which will enable a full run of all interferograms, and will skip making the topo_ra file which has already been created in step 6(b)(ii)(1) above. Move to section 6(c) below.

c. Run All Interferograms

[*Note:* To use the intf_tops_parallel.csh script below, you'll need to have GNU parallel installed. If you would like to install this, first run the command "sudo port install parallel". This will allow multiple runs of intf_tops.csh to be run in parallel (see command below), with '6' being the number of threads that are being sent out. If you are working on a larger server, this number can be increased (but for each thread, allow one core and 5-8 GB of memory. If you are on a lower memory machine (e.g. 64GB) we'd recommend not going over 4-5 jobs).
Alternatively, feel free to use the intf_tops.csh script instead and remove the '6' from the end of the below command. This will just take longer processing time.]

i. Run this from **F1**, to produce all interferograms listed in the **F1** **intf.in** list:

```
intf_tops_parallel.csh intf.in batch_tops.config 6 &> itp.log &
```

Take a break, this may take a minute (~1 hr or more, depending on whether it's running in parallel or not)

[*Note:* Output will go to the **intf_all** directory. This will also produce a log file for each interferogram pair that will end up in this directory.]

ii. Repeat steps 6(b) and 6(c) above to edit the batch_tops.config files and run all interferograms for subswaths 2 and 3 in directories F2 and F3. Double check that you have edited the **intf.in** lists and batch_tops.config files to point to **F2** and or **F3** file names (step 6(a)(iv.) and 6(a)(vi.) above). Check that the other parameters are set correctly and that proc_stage = 1 for the first interferogram, and then modified to 2 for the full set.

7. Merging Across Subswaths

a. Prepare the merge_list input file

i. (skip to the Note below to use a pre-made script to do this) To see what is needed to run the merge, type merge_batch.csh into the command line to see its usage. Go to the **merge** folder and prepare a **merge_list** for merge_batch.csh in the following format:

```
IF1_Swath1_Path:master.PRM:repeat.PRM,IF1_Swath2_Path:master.PRM:repea
t.PRM,IF1_Swath3_Path:master.PRM:repeat.PRM
```

If we interpret this briefly for clarity:

```
IF1_Swath1_Path:master.PRM:repeat.PRM,IF1_Swath2_Path:master.PRM:repea
t.PRM,IF1_Swath3_Path:master.PRM:repeat.PRM
```

In yellow is the `IF1_Swath1_Path =` "path to interferogram #1 (IF1) in swath #1 (Swath1)": `master.PRM =` "name of master PRM file for that interferogram pair": `repeat.PRM =` "name of the repeat PRM file for that interferogram pair", all separated by colons.

In green is the same, but for swath #2

In blue is the same, but for swath #3. Each swath's information is separated by commas.

For example, in our example case, the first line could read:

```
../F1/intf_all/2018127_2018133/:S1_20180508_ALL_F1.PRM:S1_20180514_A
LL_F1.PRM,../F2/intf_all/2018127_2018133/:S1_20180508_ALL_F2.P
RM:S1_20180514_ALL_F2.PRM,../F3/intf_all/2018127_2018133/:S1_2
0180508_ALL_F3.PRM:S1_20180514_ALL_F3.PRM
```

Because we choose not to run the F1 swath in this example, we can skip that section of the **merge_list**, so the first line in our **merge_list** would read:

```
../F2/intf_all/2018127_2018133/:S1A20180508_ALL_F2.PRM:S1A20180514_A
LL_F2.PRM,../F3/intf_all/2018127_2018133/:S1A20180508_ALL_F3.P
RM:S1A20180514_ALL_F3.PRM
```

You will need a line for every interferogram in intf.in (N=194)

NOTE: In the latest distribution (2021) we include a script that will organize and prepare a merge_list file for you. This command is called create_merge_input.csh and needs a list of interferogram directories in the intf_all directory of F1 (intflist), a path to the F* directories, and a chosen mode number. Type create_merge_input.csh into the command line to see the example and usage. We run a command that looks like this:

```
create_merge_input.csh intflist .. 2 > merge_list
```

where mode 2 implies we want to merge the interferograms in F2 and F3. Remember to redirect the output of this command to a file named merge_list, otherwise it will just print to the screen. There are two other options for running all subswaths or just the F1/F2 subswatch.

ii. Place a line which includes the master image as the master (first) of the pair of images, as the first line in the **merge_list**. Make sure to do this, as it will ensure that all images are processed with the same coordinates and are the same final grid size.

b. Prepare necessary run files

i. Copy the batch_tops.config to the merge directory. Make sure threshold_snahpu = 0 to skip unwrapping (we will unwrap later).

```
cp ../F2/batch_tops.config .
```

ii. Add dem.grd link too

```
ln -s ../topo/dem.grd .
```

c. Run merge_batch.csh

i. From the merge directory:

merge_batch.csh merge_list batch_tops.config (~20 min)

The output will be interferometric directories for each interferogram in the merge directory that each contain merged coherence, mask, and phase files.

[*Note:* If troubleshooting, be aware that merge_batch.csh will not overwrite an existing trans.dat file when it runs the next time. If you're sure that your trans.dat file is computed correctly, no need to delete before running again.]

8. Unwrapping Interferograms

Unwrapping is one of the most time-consuming and computationally intensive steps of the InSAR time series process, and because of that we seek out whatever steps we can to limit the processing of unnecessary areas or pixels in our region of interest. One critical thing to determine before unwrapping is how coherence varies in your region of interest, because low coherence can lead to lower accuracy in displacement estimates and more difficulty in unwrapping. In areas of good coherence, such as regions of flat, dry, unvegetated landscapes (e.g. the Mojave Desert in California, U.S.) unwrapping is fairly straightforward (the steps in 8(a) will guide you through what to do). This is because radar waves can reflect back to the SAR instrument well on dry flat terrain.

On the other hand, in areas of poor coherence, such as wet, highly vegetated landscapes (e.g. the rainforest wet side of the island of Hawaii, as in our example!), unwrapping can be a little trickier and requires some forethought as to how to avoid the areas of worst coherence. These regions often have poor coherence because the radar waves reflect in all directions and

do not have as strong of a return signal to the SAR instrument. Luckily, there are a few things we can do to minimize how much this poor coherence affects the visualization of our signal of interest--see steps in 8(b)).

LARGE BODIES OF WATER (Ocean/Lakes): Please note that we recommend that if you have large areas of open water in your region of interest, you should mask out those wet areas using a land mask. This will prevent the snaphu program from attempting to unwrap those areas of near zero coherence. If you leave those areas in, the unwrapping process may take longer than it should. To make a landmask, use `landmask.csh` and the boundaries of your phase.grd like so:

`cd` to one of your merged interferogram directories

`gmt gridinfo phasefilt.grd`

[the output will list the grid features of this file -- look for the x_min, x_max, y_min, y_max values]

```
cd ../
landmask.csh minX/maxX/minY/maxY
```

Then, once you have created your landmask_ra.grd, link it to all your interferogram directories in your unwrap_intf.csh script below.

a. Unwrapping in Regions of Good Coherence

i. Unwrap the phase for each interferogram

[Note: We suggest you unwrap a single interferogram to start with, in order to make sure your settings are appropriate]

1. In the merge directory, first make a list of interferogram directories to feed to the script:

```
ls -d 201* > intflist
```

2. Here is a sample script that does the unwrapping; name this "unwrap_intf.csh":

```
#!/bin/csh -f
# intflist contains a list of all date1_date2
directories.

cd $1
ln -s ../landmask_ra.grd .
snaphu_interp.csh 0.01 40
cd ..
```

Make sure this script is executable by chmod-ing it:

```
chmod +x unwrap_intf.csh
```

If your system cannot find this script make sure it is in your $PATH.

To run:

```
unwrap_intf.csh [name of interferogram directory to unwrap]
```

[*Note:* Type snaphu_interp.csh into the command line, to see usage. To run snaphu to unwrap, we need to choose a correlation threshold and a maximum discontinuity threshold. The correlation threshold describes the minimum correlation value in your unwrapping; if a value in a pixel is less than the chosen threshold, it will be set to zero. The smaller the threshold however, the longer it will take to run. Maximum discontinuity threshold is usually set to zero for interseismic motion, where there are no large displacements. A number greater than zero will allow for phase jumps along discontinuities such as earthquake ruptures.]

LARGE BODIES OF WATER (Ocean/Lakes): Please note that we recommend that if you have large areas of open water in your region of interest, you should mask out those wet areas using a land mask.

3. Another option is to run snaphu_interp.csh with a region cut, which will only unwrap the area given in the bounding box you set. For example the command:
   ```
   snaphu_interp.csh 0.10 40 1000/3000/24000/27000
   ```

   would only unwrap the area inside the 1000-3000 in range, and 24000-27000 in azimuth. (format minRange/maxRange/minAzi/maxAzi)

4. Start unwrapping!

   [*Note.* In order to save time, there is also an option to instead run this in parallel with the unwrap_parallel.csh script. This calls the script you just created (unwrap_intf.csh), so make sure you named it correctly. Again, 6 is the number of threads (see step 6(c) for an explanation of how to install parallel). This means that multiple interferograms would be unwrapping at once, decreasing the time it takes to complete all of them. For example, running one by one takes ~ 1hr per interferogram. For lower memory machines, be aware that a 5000x5000 pixel image at 8/2 decimation can take ~8GB of memory on one core to unwrap; E.g. if your computer memory is 64 GB, we might recommend you stick to a lower number of jobs/threads (3-5).]

   ```
   unwrap_parallel.csh intflist 6
   ```
   (~1-2 days depending on processing power and number of cores in your machine, as well as how much noise there is in your interferogram--definitely go take a break. Get a meal. Take a nap. Come back in a few hours to check progress).

***Note:If one interferogram has not been processed within 24 hours (you don't have a finished unwrap.grd file), double check that you are not trying to unwrap large areas of ocean or large swaths of bad data, as those can cause issues. Do this by taking a look at the phasefilt.grd.)

****Note: If trouble-shooting, note that unwrap.cmd gets produced by unwrap_parallel.csh, but does not get overwritten if this is run again. You must change it to a different name (mv unwrap.cmd old_unwrap.cmd) or delete the file "unwrap.cmd" to ensure you run everything with the correct command file.

b. Unwrapping in Regions of Poor Coherence

There are a couple steps we can take to minimize the effect of poor coherence pixels on our unwrapping. The first is stacking coherence grids, and the second is to creak a mask to mask out low coherence pixels (which occur often in water body areas).

i. Stacking Coherence Grids

1. You can either create a **corr_stack.grd** file by (1) averaging a selection of coherence files (**corr.grd**) you put in, or by (2) running the provided stacking script in the distribution.
    a. Averaging a selection of coherence files that you choose; this is an example for a different set of interferograms:

```
gmt grdmath 2006355_2007036/corr.grd 2006355_2009179/corr.grd
2006355_2009225/corr.grd 2007036_2009225/corr.grd
2008039_2010136/corr.grd ADD ADD ADD ADD 5 DIV = corr_stack.grd
```

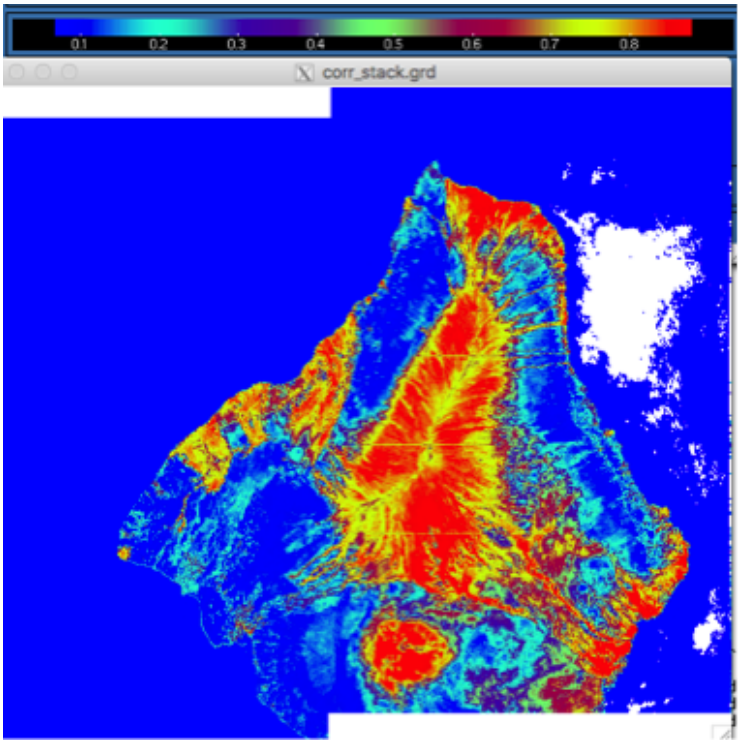
    b. Using stack.csh; from the **merge** directory:

```
ls 201*/corr.grd > corr.grd_list
stack.csh corr.grd_list 1 corr_stack.grd std.grd
```

    [*Note:* "1" is the scaling factor (in this case, no scaling). "corr_stack.grd" and "std.grd" are the output files (we are only going to use "corr_stack.grd")].

    After creating corr_stack.grd, take a look at it:

```
ncview corr_stack.grd
```



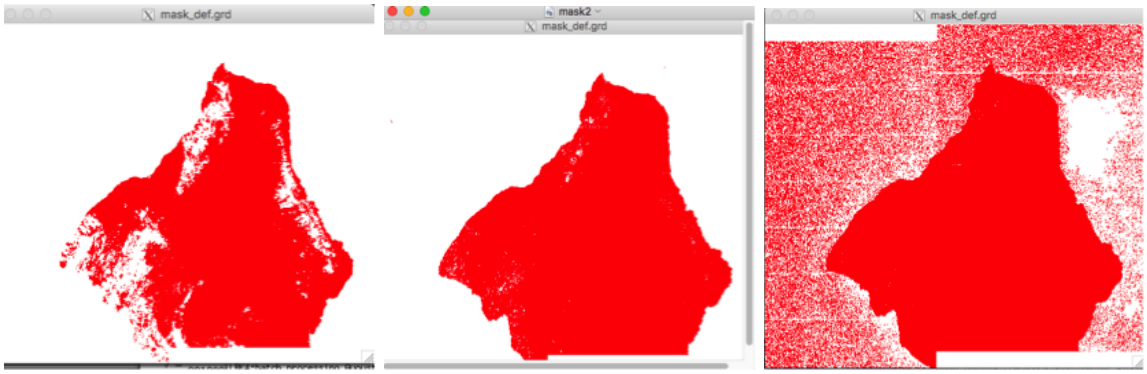
Here we can see that most of the old lava flows show high coherence, while the more vegetated regions are low.

ii. Creating a Coherence Mask

1. Now we'll create an ocean mask based on the stacked coherence. This will mask out any data that doesn't meet the threshold that we set (here we chose 0.075, but feel free to experiment) The value we chose masks most of the ocean and only a little bit of the land (see figure below).

```
gmt grdmath corr_stack.grd 0.075 GE 0 NAN = mask_def.grd
```

Use ncview again to view the mask_def.grd file to inspect the mask (see below images). If it masks too much (lots of white patches on land, see leftmost figure), then lower the coherence threshold. If it masks too little (for example, if there is speckle in the ocean, see rightmost figure), raise it. The middle figure here shows a 0.075 threshold:



(Illustration: when the coherence threshold is too high, just right (0.075), and too low).

2. Link the new mask_def.grd to all /date1-date2/ directories

iii. Unwrapping the phase for each interferogram

This is the same as unwrapping in section 8(a), except we are editing the script above to include a link to mask_def.grd added in.

1. In the merge directory, first make a list of interferogram directories to feed to the script:
   `ls -d 201* > intflist`

2. Make the following script and name it "**unwrap_intf.csh**".

   **See note in above section 8(a) regarding choices of snaphu threshold values. Alternatively, just type snaphu_interp.csh into the command line and see the usage information.

```
#!/bin/csh -f
#  intflist contains a list of all date1_date2 directories.

cd $1
ln -s ../mask_def.grd .
snaphu_interp.csh 0.001 40
cd ..
```

   Make sure this script is executable by chmod-ing it:
   `chmod +x unwrap_intf.csh`

   If your system cannot find this script make sure it is in your $PATH.

   To run:

```
unwrap_intf.csh [name of interferogram directory to unwrap]
```

3. Another option is to run snaphu_interp.csh with a region cut, which will only unwrap the area given in the bounding box you set. For example the command:
   `snaphu_interp.csh 0.10 40 1000/3000/24000/27000`

   would only unwrap the area inside the 1000-3000 in range, and 24000-27000 in azimuth. (format minRange/maxRange/minAzi/maxAzi)

4. Start unwrapping!

   [*Note.* In order to save time, there is also an option to instead run this in parallel with the unwrap_parallel.csh script. This calls the script you just created (unwrap_intf.csh), so make sure you named it correctly. Again, 6 is the number of threads (see step 6(c) for an explanation of how to install parallel). This means that multiple interferograms would be unwrapping at once, decreasing the time it takes to complete all of them. For example, running one by one takes ~ 1hr per interferogram.]

   `unwrap_parallel.csh intflist 6` (~1-2 days depending on processing power and number of cores in your machine, as well as how much noise there is in your interferogram--definitely go take a break. Get a meal. Take a nap. Come back in a few hours to check progress).

   ***Note: If one interferogram has not been processed within 24 hours (you don't have a finished unwrap.grd file yet), double check that you are not trying to unwrap large areas of ocean or large swaths of bad data, as those can cause issues. Do this by taking a look at the phasefilt.grd.)

   ****Note: If trouble-shooting, note that unwrap.cmd gets produced by unwrap_parallel.csh, but does not get overwritten if unwrap_parallel.csh is run again. You must change "unwrap.cmd" to a different name (mv unwrap.cmd old_unwrap.cmd) or delete the file "unwrap.cmd" to ensure you run everything with the correct command file.

9. Running Small BAseline Subset (SBAS) Analysis

a. Prepare the Input Files

i. If you want this automatically done for you, skip to (iii); In the SBAS directory, prepare scene.tab file***. The first column is the seven-digit scene ID and the second column is the number of days. These values can be found in your

baseline_table.dat file. You must ensure that these records are in time, or chronological order to proceed.

```
awk '{print substr($2,1,7),$3}' ../F1/baseline_table.dat > scene.tab
```

Example of what scene.tab should look like:

```
<scene_id> <days>
2018031 1491
2018037 1497
2018043 1503
2018049 1509…
```

[*Note.* Double check that the scene.tab file is in chronological order, to prevent issues later].

ii. If you want this automatically done for you, skip to (iii); In the SBAS directory, prepare the intf.tab file***. In the intf.tab file we need to list the path to the unwrapped interferogram, the path to the correlation grid of that same interferogram, the scene_id for the reference scene of the interferogram, the scene_id for the secondary scene of that interferogram, and the difference in perpendicular baseline (secondary - reference) between those interferograms (in that order), so we get a file that looks as follows:

```
< path to unwrap.grd >                < path to corr.grd >           <ref. scene_id>  <sec. scene_id>  < b_perp diff >
../merge/2018031_2018037/unwrap.grd ../merge/2018031_2018037/corr.grd 2018031 2018037 -97.1762
../merge/2018031_2018043/unwrap.grd ../merge/2018031_2018043/corr.grd 2018031 2018043 -83.7672
../merge/2018031_2018049/unwrap.grd ../merge/2018031_2018049/corr.grd 2018031 2018049 -87.637
../merge/2018031_2018055/unwrap.grd ../merge/2018031_2018055/corr.grd 2018031 2018055 -42.8708
```

The baselines and scene_id's come from the baseline_table.dat file, and the scene_id's must match the ones in scene.tab exactly. In addition, they (the scene_id's and baseline values) must be integer values, because sbas reads them in as integers.

iii. There is a script that will accomplish this for you that was recently added to the GMTSAR distribution called prep_sbas.csh. This script creates the scene.tab file, the intf.tab file, and outputs a default SBAS command structure for the user. Check and see if you have this in your distribution by typing "prep_sbas.csh" into your command line and see if the usage pops up. If you do not have it, feel free to check github and download it there.

```
prep_sbas.csh intf.in baseline_table.dat ../merge unwrap.grd corr.grd
```

[*Note:* need to give it the list of interferograms (intf.in), the list of baselines (baseline_table.dat), the path to the merge directory where it will find the unwrapped interferogram grids and correlation grids, the name of the unwrapped file (unwrap.grd) file, and the name of the correlation grid file (corr.grd)]

b. Run SBAS

i. Type “sbas” into the command line to read about the usage and see example commands. Here is an example command for this Kilauea example (all one line):

```
sbas intf.tab scene.tab 194 34 4612 3300 -range 888122 -incidence 40
-wavelength 0.0554658 -smooth 5.0 -rms -dem &> sbas.log &

Usage: sbas intf.tab scene.tab N S xdim ydim [-atm ni] [-smooth sf]
[-wavelength wl] [-incidence inc] [-range -rng] [-rms] [-dem]
```

+ 194 is the number of interferograms (N) being run (it is 194 here, because that is how many are in subswath F1)
+ 34 is the number of scenes (S) being included (type wc -l scene.tab to get this number)
+ To find xdim (here is 4612) and ydim (3300) (x and y dimension of interferograms):
  `gmt grdinfo ../merge/date1_date2/unwrap.grd` (where date1_date2 is replaced with an interferogram date--they should all have the same dimensions).
  “x n_columns” is the value for x dim
  “y n_rows” is the value for y dim

+ wavelength: wavelength of the radar wave in m. To find this out, look in supermaster.PRM for “radar_wavelength”
+ incidence: the default value is 37 degrees, though the value used here is largely irrelevant (the default is fine in most cases)
+ range: range distance from radar to center of the interferogram. To get this number for Sentinel-1:
  Range = ({[(speed of light) / (rng_samp_rate) / 2] * ((x_min+x_max)/2)} /2) + near_range
  Speed of light is ~3 x 10^8 m/s
  “rng_samp_rate” is in supermaster.PRM
  X_min and x_max come from the gmt grdinfo command above
  “near_range” is in supermaster.PRM

  Range = ({[(3e8) / (64345238.125714) / 2]*36995.5895372} / 2) + 845000
  Range = 888,121.5937
  Range ~ 888,122

+ -rms tells the program we want it to output a grid of rms values (in mm), see usage
+ -dem tells the program we want it to output the dem error (in m), see usage

[To see the screen output for this command as it’s running, just tail the log file like this:]
`tail -f filename.log`

c. Post-SBAS Results: Projecting into Latitude/Longitude

i. After running SBAS, you get a **vel.grd** file (in radar coordinates, and in mm/yr). To transform it into lat/long coordinates, do the following:

```
ln -s ../merge/trans.dat .
ln -s ../F1/intf_all/2018127_2018163/gauss_* .
```

(wildcard there to include any filter wavelength one specifies in their config file. We used a wavelength of 200, so our file is gauss_200)

```
proj_ra2ll.csh trans.dat vel.grd vel_ll.grd
gmt grd2cpt vel_ll.grd -T= -Z -Cjet > vel_ll.cpt
grd2kml.csh vel_ll vel_ll.cpt
```

[*Note:* proj_ra2ll.csh creates two files called raln.grd and ralt.grd. However, once they exist, they will not be overwritten by successive runs, so if you find yourself troubleshooting at this step, make sure to delete these between runs].

You will also see various displacement grid files of the form disp_*.grd. These are the displacements measured at that time stamp (measured in mm) so you can make a time series of displacements from points from each of those grid files. See **Section 10** below for details about how to extract specific points.

ii. View the vel_ll.kml file in Google Earth © (see below) and admire the velocity grid! Anything in cooler colors here means that the ground is moving away from the satellite, in the LOS direction (so the two blue areas you see are areas of subsidence). Anything in warm colors means the ground is moving towards the satellite in the LOS direction.
[*Note.* Remember that you need both the *.kml and the *.png file in the same directory in order to view it in Google Earth ©].

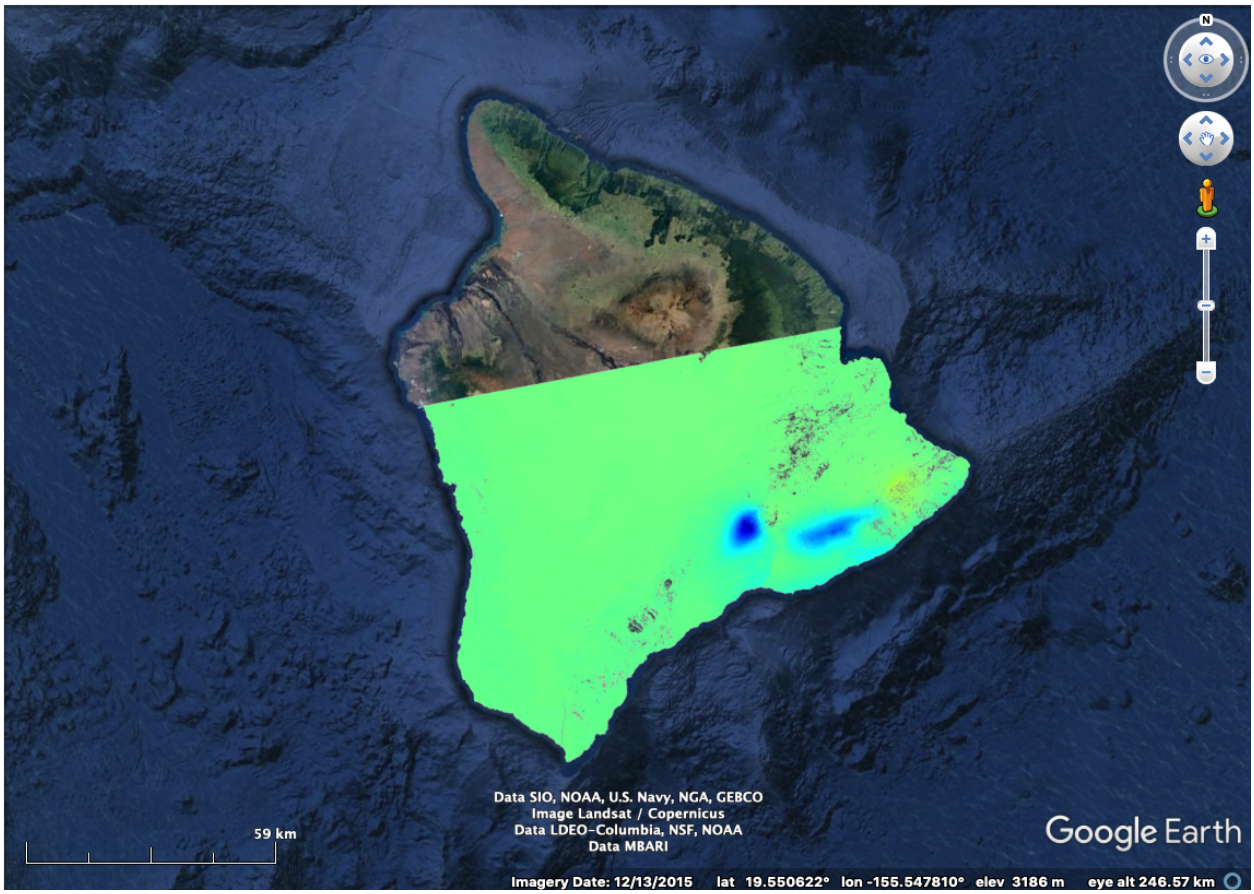


Figure: vel_ll.kml, as viewed in Google Earth ©

10. Converting Displacements to LOS Projection for Comparison with GNSS Data

This is a section for those of us who have GNSS data that we would like to compare with the output of the above InSAR time series analysis. We start by assuming you already have GNSS velocities or time series displacements in hand for your area of interest.

[*Note.* If you would like to find your own GNSS data, there are several locations where you can obtain time series including the University of Nevada Reno (UNR) Nevada Geodetic Laboratory, UNAVCO Derived Data Products, the Scripps Orbit and Permanent Array Center (SOPAC), and the U.S. Geological Survey (USGS). The easiest to use resource for newer users is probably:

University of Nevada Reno (UNR) Nevada Geodetic Laboratory
(http://geodesy.unr.edu/NGLStationPages/gpsnetmap/GPSNetMap.html)

One word of warning--make sure to read the documentation provided so you understand, for example, which column is which and what reference system the data are processed in].

a. Projecting GNSS Velocities/Time Series into LOS

i. First we need to construct a list of GNSS locations (longitude, latitude, and height). Put the longitude and latitude information in a file called **GNSS.ll** . We will pull the height information for these GNSS station locations from our dem.grd, below.

1. Generate height from the dem.grd file in ../topo

```
gmt grdtrack GNSS.ll -G../topo/dem.grd > GNSS.llh
```

[*Note:* we need to extract the heights from our **dem.grd** file because this gives us the height above ellipsoid, while an elevation/altitude from a GNSS station may give you height about the geoid].

ii. Then construct a file with, for example, the velocities from each station in east, north, up components. Call this **GNSS_vel.enu**. These velocities need to be estimated from the same time period as is covered by the InSAR time series. If you are using GNSS time series, make sure that the time series cover the same time period as the InSAR time series. For time series, each station will have its own ***.enu** file; for example, we construct a file for GNSS station MKAI with time series (in east, north, up format) from UNR called **MKAI_ts.enu**.

iii. Either move this to one of your interferogram directories in your merge folder, or copy or link your supermaster.PRM and *.LED file from the interferogram directories in your merge folder to your current location (in this case, we made a new directory to calculate everything in).
E.g. in asc/

```
mkdir GNSS2LOS
cp ../merge/2018127_2018133/supermaster.PRM .
cp ../merge/2018127_2018133/S1_*_ALL.LED .
```

iv. Calculate the look vector (ground to satellite look direction) for each GNSS station

```
SAT_look master.PRM < GNSS.llh > GNSS.llhenu
```

This creates a file that has a look vector for each GNSS station point listed in your GNSS.llh file (output as file GNSS.llhenu), in east, north, up components.

v. Take the dot product of these look vectors with the individual GNSS stations' velocities or displacement time series to get an LOS velocity or displacement time series for each station. In this case:

```
paste file1 file2 | awk '{print [multiplying  e with e, n
with n, up with up and then adding them]}' > LOS_GNSS.vel
```

E.g., for velocities:

```
paste GNSS_vel.enu GNSS.llhenu | awk '{print ($1*$7) +
($2*$8) + ($3*$9)}' > LOS_GNSS.vel
```

For time series, you need to accomplish this dot product for every observation time in the dataset, so the command gets a little more involved:

```
awk '{print ($1*elook)+($2*nlook)+($3*ulook) }' MKAI_ts.enu > MKAI_ts.los
```

where elook, nlook, and ulook are the look vector components from the particular station line in your GNSS.llhenu file (the last three columns, in that order).

Note: In the July 2021 GMTSAR Distribution, we include a script called `gnss_enu2los.csh` which will accomplish the projection of GNSS displacements into LOS for you.

b. Extract Point Velocities/Displacements from InSAR LOS Grids

i. Run gmt grdtrack on the chosen InSAR displ_*.grd or vel.grd files to extract LOS displacements or velocity at chosen GNSS station points. In this instance, we are extracting the final InSAR LOS velocities from the vel_ll.grd file, at the point locations listed in file GNSS.llh.

```
gmt grdtrack GNSS.llh -G../SBAS/vel_ll.grd > LOS_InSAR.vel
```

You can, instead, extract points out of the individual disp_*.grd files to get a displacement time series, but remember that these are in radar coordinates, and need to be projected to latitude, longitude coordinates first (see section **9c(i)** for how to do this). Alternatively, you can also project your GNSS locations into radar coordinates, and extract the information from the disp_*.grd files directly, without having to project them first, like so:

```
SAT_llt2rat supermaster.PRM 1 < GNSS.llh > GNSS.rahll
```

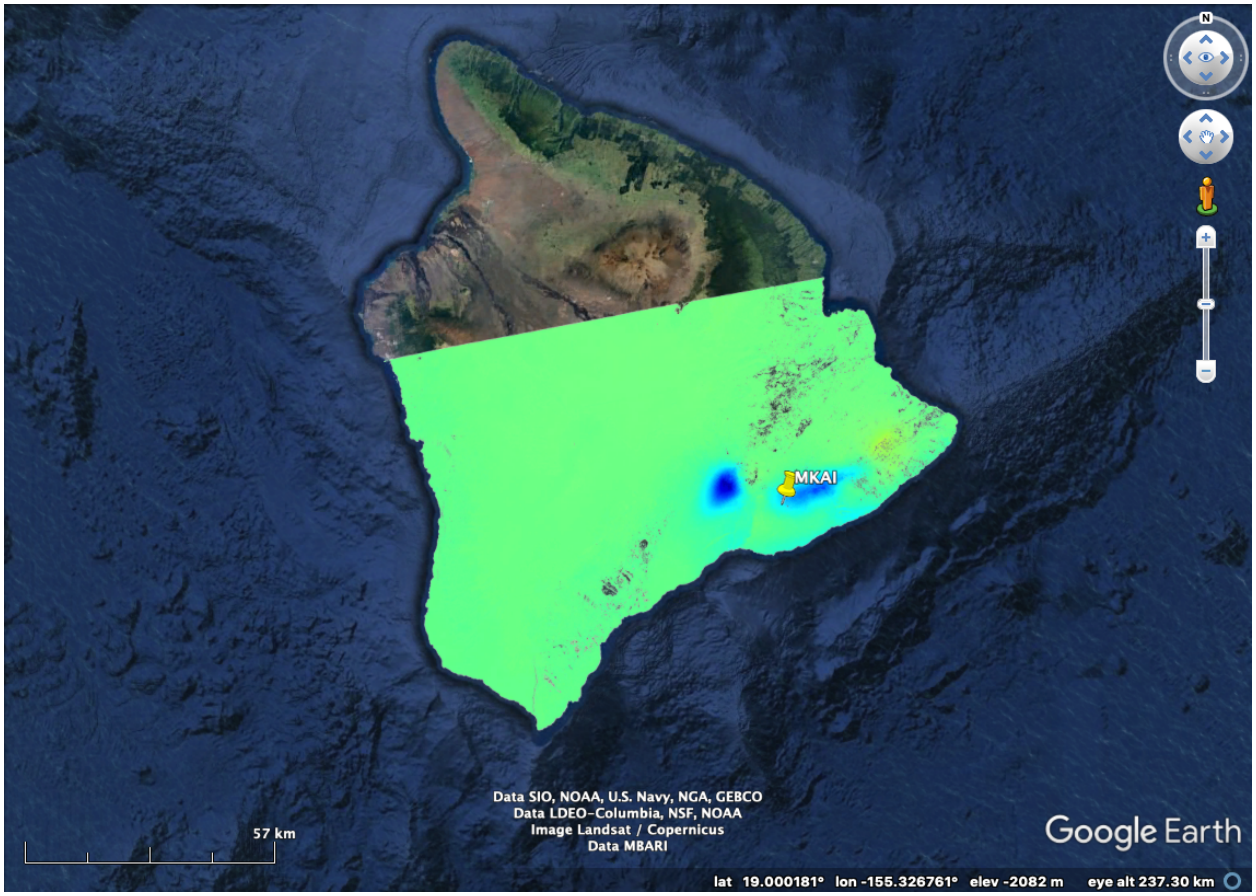
(GNSS.rahll comes out as, range, azimuth, height, long, lat; we use a precision here of 1)

Then to extract:

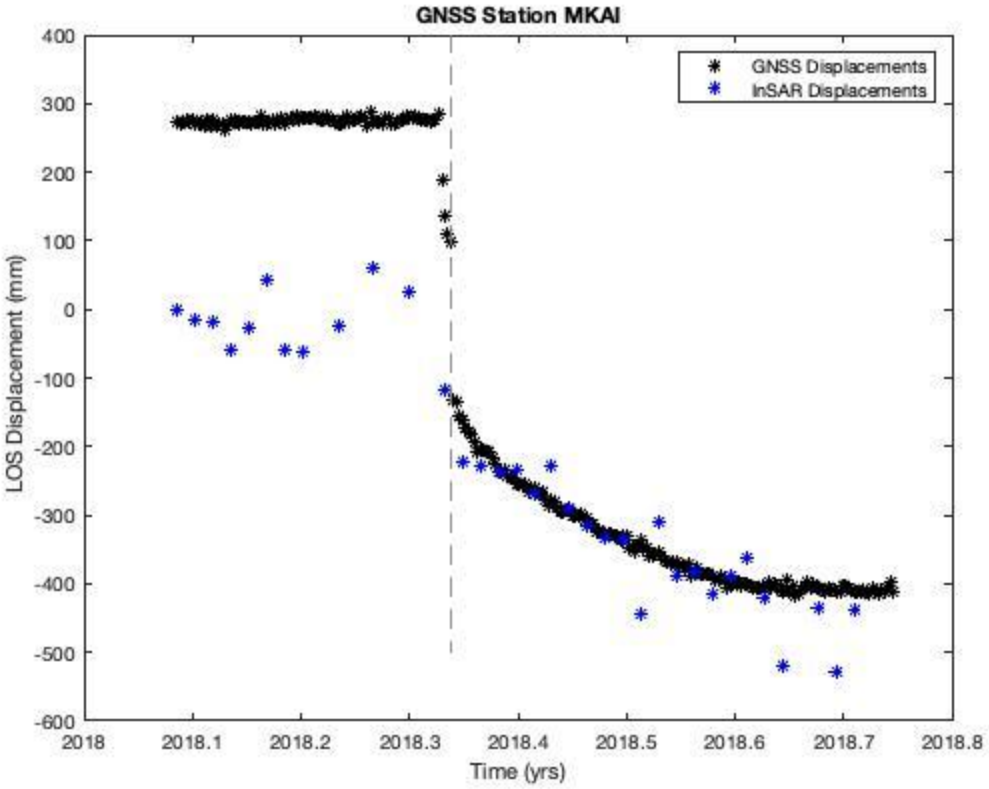
```
gmt grdtrack GNSS.rahll -G../SBAS/disp_2018031.grd > LOS_InSAR.disp
```

This now gives you the InSAR line of sight displacements at every point in your GNSS.rahll list. To get the displacements for just station MKAI, we extract them separately from all disp_*.grd files in the manner above.

ii. Compare the LOS_GNSS.vel and the LOS_InSAR.vel files or the separate time series displacements files (see below) with the plotting mechanism of your choice. As an example, here we plot a comparison between the GNSS displacements in LOS for station MKAI, and the InSAR displacements in LOS for the MKAI station location:



Location of GNSS station MKAI, near Kilauea.



Station MKAI lies near the Kilauea crater and observed large amounts of deformation during the eruption and subsequent earthquakes. The trends of each time series match well here, indicating a successful analysis. If we were to perform a quick back-of-the-envelope velocity comparison here, we can estimate that the MKAI GNSS time series gives us a velocity of about -1078 mm/yr for this time period of 0.65 year, while our **LOS_InSAR.vel** file tells us that the measured velocity from InSAR is -889 mm/yr, which agrees fairly well.

However, there is a clear mismatch between the pre-earthquake displacement levels. This is most likely due to the fact that we have not pinned our InSAR interferograms to any GNSS points, and in addition we have not detrended the InSAR data before running SBAS, so there could be ramps in the data. There are currently a variety of methods that are currently under research that aim to improve the scatter of individual InSAR displacement points, as well as improve fit to the GNSS data. Since these methods are active topics of research, there is no one recommended way of proceeding, ergo we do not stipulate one here. This is simply an example of the product that you would get using this analysis explained in this document, and we refer you to the literature from here.

11. Correcting InSAR Interferograms with GNSS Displacements

With the recent July 2021 GMTSAR distribution we have included a new script called `correct_insar_with_gnss.csh` which allows you to use your own GNSS displacements to correct InSAR interferograms for additional atmospheric effects. This also ties your

interferogram to GNSS motions and helps provide an underlying reference frame. This script uses the general approach of Neely et al. (2020) (e.g. GInSAR), but instead of solving for a best-fit residual correction grid, this script applies a Gaussian filter to the correction grid before removing it from the interferogram. This gaussian filter is user-specified (in units of meters), and we recommend using the value of average station spacing in your GNSS data set (in the example line below, we have an average station spacing of 40 km, so we use 40,000 meters as our gaussian filter wavelength).

```
Usage: correct_insar_with_gnss.csh master.PRM phase.grd
gnsslos.rad filter_wavelength

 master.PRM         -  PRM file for the master SAR image
 phase.grd          -  phase file to be corrected
 gnsslos.rad        -  GNSS displacements in LOS in millimeters
 filter_wavelength -  wavelength of the filter in meters (0.5
gain)

Example: correct_insar_with_gnss.csh supermaster.PRM
unwrap_dsamp.grd gnss_los.rad 40000

See gnss_enu2los.csh for converting GNSS ENU displacement to LOS
```

12. Note on Incorporating Ascending and Descending Time Series

The above process needs to be completed separately for both ascending and descending images if you have both types of images in your region of interest and want to include them. So once you get to the end of the SBAS step (Section **9**), you can return to Section **3(c)** or Section **4** to complete everything for the descending images in the des/ directory.

When it comes to incorporating the information from both sets of time series, this can only be completed through modeling once all processing is done. The kind of modeling performed will depend on your signal of interest, but a first step will be to make sure everything is projected into the same LOS direction (depending on the satellite).

13. References

Berardino P., G. Fornaro, R. Lanari, and E. Sansosti, “A new algorithm for surface deformation monitoring based on small baseline differential SAR interferograms,” IEEE Trans. Geosci. Remote Sensing, vol. 40, pp. 2375–2383, Nov. 2002.

Schmidt, D. A., and R. Bürgmann 2003, Time-dependent land uplift and subsidence in the Santa Clara valley, California, from a large interferometric synthetic aperture radar data set, J. Geophys. Res., 108, 2416, https://doi.org/10.1029/2002JB002267, B9.

Neely, W.R., Borsa, A.A., and Silverii, F., 2020. GInSAR: A cGPS Correction for Enhanced InSAR Time Series, IEEE Transactions on Geoscience and Remote Sensing, 58(1), 136 - 146, https://doi.org/10.1109/TGRS.2019.2934118.

Tange, O., 2020, GNU Parallel 20200522 ('Kraftwerk'): Zenodo, https://doi.org/10.5281/zenodo.3841377.

Tong, X. and Schmidt, D., 2016. Active movement of the Cascade landslide complex in Washington from a coherence-based InSAR time series method. Remote Sensing of Environment, 186, pp.405-415.

Tymofyeyeva, E. and Fialko, Y., 2015. Mitigation of atmospheric phase delays in InSAR data, with application to the eastern California shear zone. Journal of Geophysical Research: Solid Earth, 120(8), pp.5952-5963.

Xu, X., Sandwell, D. T., Tymofyeyeva, E., González-Ortega, A., & Tong, X. (2017). Tectonic and anthropogenic deformation at the Cerro Prieto geothermal step-over revealed by Sentinel-1A InSAR. IEEE Transactions on Geoscience and Remote Sensing, 55(9), 5284-5292.