StaMPS Persistent Scatterer Exercise

UNAVCO InSAR Course, August 1-5, 2016

Andy Hooper, University of Leeds
a.hooper@leeds.ac.uk

This exercise consists of working through an example data set acquired by the Sentinel-1A satellite over Mexico City between October 2014 and August 2015. StaMPS can be used for PS or small baseline analysis (or both combined) but in this exercise you will use StaMPS to carry out PS processing only.

The StaMPS manual can be found at:
http://homepages.see.leeds.ac.uk/~earahoo/stamps/StaMPS_Manual_v3.3b1.pdf

N.B. The version you will use today is enhanced beta version and not all of the functionality is yet detailed in the manual

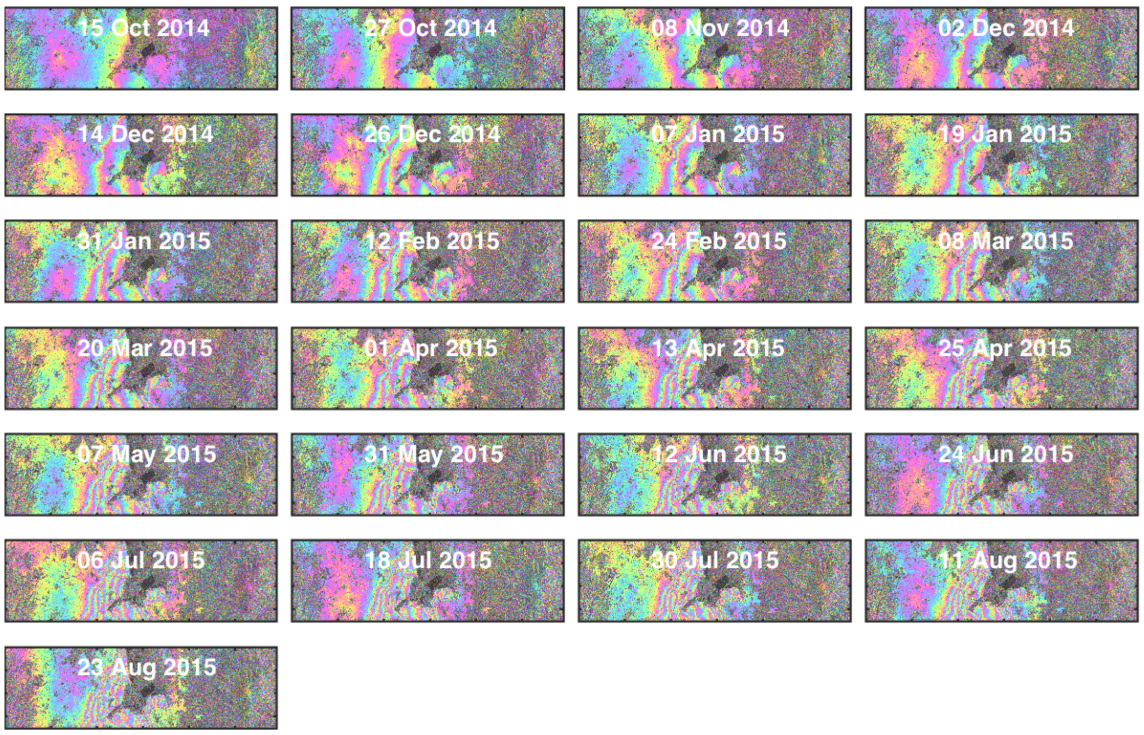

Figure 1. Single master interferograms (20 looks in range, 4 looks in azimuth). The master data is 3 Oct 2014.

1. Pre-processing

The StaMPS package includes scripts to focus raw data using ROI_PAC and to form interferograms using Doris, all in a PS friendly format. Instructions to run the scripts to form interferograms are detailed in Chapters 3 to 5 in the StaMPS manual. There is also now a set of scripts to form interferograms using ISCE (ISCE2StaMPS), developed by David Bekaert. Interferograms can also be formed with other softwares, and imported into StaMPS. For this tutorial, we will use 25 single master interferograms (produced by Yngvar Larsen of Norut using GSAR) that have already been imported into StaMPS. These are plotted in Figure 1.

Candidate PS pixels have also already been selected from the interferograms using **mt_prep** (see beginning of Chapter 6 in the StaMPS manual) The pixels selected have amplitude dispersion < 0.4, where amplitude dispersion is the ratio of amplitude standard deviation to mean amplitude. For efficiency reasons, StaMPS divides the region in the interferogram crop into overlapping patches, each of which are processed independently for the initial 5 steps. In this case we have selected 8 patches in range and 3 in azimuth, giving 24 patches in total.

Now you can get going with the time series analysis. StaMPS consists of command line binaries and matlab functions, but for this exercise you will only need to execute the matlab functionality.

Now you can get going with the time series analysis. First set up your linux/unix environment. Edit StaMPS_CONFIG.bash, if using the bash shell or StaMPS_CONFIG.tcsh if using csh (both can be found in the directory where you installed StaMPS), to set the correct paths to the softwares required by StaMPS. Then

`>source <stampsdir>/StaMPS_CONFIG.<shell>` `Replace` *`<stampsdir>`* `with your path to your StaMPS directory and` *`<shell>`* `with 'bash' or 'tcsh'.`

N.B you might like to put the above command in your .bashrc or .cshrc file so that it is correctly sourced every time.

Next, change directory to the location of the preprocessed data:

`>cd` ***`<datadir>`*** **`/STAMPS_S1_MEXICO/INSAR_20141003`** `Replace` *`<datadir>`* `with your path to the data`

List the files with:

```
>ls
```

Most of the input data are in the individual PATCH_ directories, but a few common input files are in the INSAR_20141003 directory itself.

2. PS Selection

Next start Matlab by entering in the terminal:

```
>matlab
```

N.B Matlab should always be started from a terminal in order source the correct files.

The rest of the exercise is controlled from the Matlab window that opens up.

2.2 Load PS candidiates

By default StaMPS will cycle through all patches. However, it is a good idea to run Steps 1 to 5 for one patch only to start with, in order to optimise the processing parameters for your data set. You can do this by changing directory to the PATCH_14 directory before continuing by entering in Matlab:

```
>> cd PATCH_14
```

Step 1 of StaMPS loads the data for the selected candidate PS pixels into matlab, and stores them in matlab workspace files. To save time, this has been run for you. Several workspace files are created with a '1' appended indicating that they contain data relating to all initial candidates. You can list these by entering:

```
>> ls *1.mat
```

Check the phase of the candidate PS:

```
>> ps_plot('w')
```

You will notice that a lot of pixels have been selected as initial PS candidates, and many of them look quite noisy. The first interferogram displayed uses the master as slave and so has zero phase throughout.

The ps_plot command has a lot of functionality, which you can check by typing

```
>> help ps_plot
```

The help command can be used to get more info for any of the StaMPS commands.

Plot the 2nd interferogram only using the mean amplitude of all images as background:

```
>> ps_plot('w',5,0,0,2)
```

The defaut size of the plotted points (120 m) means that they merge into one another. You can change the size to 30 m by entering:

```
>> setparm('plot_scatterer_size',30)
```

The first input given to setparm is a string with the parameter name you want to change, and the second input is the value you want to set it to. The parameter name string given to setparm only has to be long enough to make it unique (you will be warned if ambiguous), so the following command would work as well (try it):

```
>> setparm('plot_s',30)
```

You can list all of the StaMPS parameters by entering:

```
>> getparm
```

Plot the 2nd interferogram again with smaller plotted points (the 'up' cursor key retrieves earlier commands entered). You will notice that many candidate pixels are selected in a dark rectangular patch, which is a body of water. This demonstrates that amplitude dispersion is not a perfect proxy for phase noise.

2.3 Calculate temporal coherence

This step estimates the spatially correlated phase for each PS candidate, subtracts it, estimates the spatially-uncorrelated DEM error for each pixel from the remaining phase, subtracts this and finally estimates the temporal coherence for each pixel from the residual phase. This process is iterated three times by default. To run this step enter:

```
>> stamps(2,2)
```

The two parameters (2,2) given with the stamps command represent the step you want to start with (first parameter) and the step you want to end with (second parameter). The above command will thus only run the 2nd step.

2.3 Initial PS selection

This step makes a selection of PS based on probability, by comparison to results for data with random phase. This is usually done twice. After the first selection, the temporal coherence of each selected pixel is re-estimated more accurately, by dropping the pixel itself from the estimation of the spatially-correlated phase. Then the selection process is repeated. However, in order to speed things up, I have set a parameter to skip the re-estimation (reest_gamma_flag='n'). Run step 3:

```
>> stamps(3,3)
```

The main parameter that controls the number of PS selected is 'density_rand', which refers to the average density of PS pixels per square kilometre that are selected in error. Normally it is good to leave this high, as these random phase pixels can be weeded out in Step 4, or in Step 5 if the data are to be downsampled. Try changing it to values between 1 and 100 and rerunning Step 3, and note the effect on the number of pixels selected. E.g:

```
>> setparm('dens',1)
>> stamps(3,3)
```

2.4. PS weeding

In Step 4, some PS pixels selected in the previous step are dropped based on their proximity to each other and the smoothness of their deformation in time. The proximity weeding is turned off however ('weed_neighbours' = 'n').

```
>> stamps(4,4)
```

2.5 Phase correction

The phase of the remaining selected pixels is corrected for spatially-uncorrelated DEM error and stored as 'version 2' (PS candidates were version 1).

```
>> stamps(5,5)
```

This produces several new workspace files with a '2' appended indicating that they contain data relating to version 2"

```
>> ls *2.mat
```

Now it is possible to plot the phase of the selected pixels

```
>> ps_plot('w')
```

Check the coherence by visually scanning the phase of each interferogram. How does this relate to the estimated phase standard deviations output by Step 5? Try plotting an individual interferogram, e.g. to plot the tenth one:

```
>> ps_plot('w',1,0,0,10)
```

Have most of the pixels in the water been dropped? Check the amplitude of points remaining in the water – could here be something solid projecting from the water acting as a PS (N.B. You can zoom in and out)? Note that some PS selected are actually sidelobes of a very bright point. We have not attempted to weed them out in this instance.

2.6. Iteration of Steps

Try changing the 'weed_standard_dev' parameter to 1.2 (this is the threshold standard deviation in radians) and rerunning steps 4 to 5. How does this change things?

Steps 4 and 5 are quick, so try playing with 'weed_standard_dev'. You want to end up choosing a value that balances selecting the maximum amount of points with reducing the noise levels. Increasing the threshold from 1 to 1.2 increases the number of selected points without increasing the noise levels by too much. The increased number of points is useful, since it improves the coverage and the amount of deformation information.

3. Merging Patches

You can now add patches 7 to 24 to the processing chain. By default, StaMPS processes all patches. To tell stamps to use only these patches, edit the file named 'patch.list' in the INSAR_20141003 directory and list the patches you want to include in this file:

```
>>cd ..
>>edit patch.list
```

For the other patches, Steps 1 to 4 have already been run for you. Now that you have run PATCH_14, you are ready to merge them together. You can achieve this by running step 5 in INSAR_20141003. You can now plot the wrapped phase for the merged patches by running ps_plot within INSAR_20141003. Note, set 'plot_scatterer_size' to 200 m before plotting.

3.1 Downsampling

By default, PS points are retained at full resolution. However, for the current application, we do not need such a high sampling rate in space. To reduce computation time, the data have been resampled the data to a coarser grid, with a 200 m spacing. An additional effect of the resampling is to reduce the noise, and resampling also offers another chance to reject the noisiest pixels.

To change the downsampling, set the parameter 'merge_resample_size' (units are m, default is 0, meaning no resampling) and rerun Step 5. Check how the wrapped phase looks now with ps_plot. In particular, check areas with high fringe rates – are the fringes sampled well enough, or is a smaller grid size necessary? Try also varying 'merge_standard_dev' (threshold standard deviation in radians) and check the effect that it has.

4. Phase unwrapping

Once happy with the PS selection you can unwrap the phase using StaMPS Step 6. During unwrapping, the data are temporarily resampled to a grid. The size of the grid is controlled by parameter 'unwrap_grid_size', and should be at least as large as 'merge_grid_size'. Go ahead and set it to the same value (200 m), then run Step 6.

Once Step 6 is finished, plot the results using

```
>> ps_plot('u')
```

The main parameters to play with to improve unwrapping are 'unwrap_grid_size' – making it larger reduces noise but may also alias (undersample) the signal, and 'unwrap_time_win', which is the filter length (in days), used to estimate the noise contribution for each phase arc (phase difference between neighbouring pixels). But for now continue to the next step.

4.1 Estimation of spatially-correlated errors

Step 7 estimates the total DEM (a.k.a look angle) error, and the master atmosphere and orbit error. Run Step 7 and view the DEM error

```
>> ps_plot('d')
```

This is given as radians per m of baseline. To convert to height errors, use the function 'K2q'. Also look at the master atmosphere using

```
>> ps_plot('m')
```

You can plot unwrapped phase after subtracting DEM error and master atmosphere with

```
>> ps_plot('u-dm')
```

Check for unwrapping errors i.e., inconsistent jumps in phase. Unwrapping errors are more likely to occur in interferograms with longer perpendicular baselines. This is for two reasons, firstly there is more noise associated with each PS, and secondly, the phase due to any spatially-correlated DEM error is larger, as it is proportional to perpendicular baseline. To check the baselines, enter

```
>>ps_info
```

You can also see the estimated noise for each interferogram, which is calculated from the phase that is not correlated spatially and not correlated with baseline. Anything above ~80 degrees is likely to give trouble for unwrapping. In this case we are using Sentinel-1 data and the baselines are all small.

The plot for 'u-dm' should also be smoother than 'u'. Note that the default colour scales will be different and if smoother the phase range should be smaller. If not generally smoother, one or more interferograms has probably unwrapped incorrectly. It is also worth comparing the 'u-dm' to the wrapped phase corrected for 'd' and 'm':

```
>> ps_plot('w-dm')
```

If there are unwrapping errors in some images, you can drop them from the spatially-correlated error estimation. E.g. to drop the 2nd and 24th interferograms

```
>> setparm('scla_drop_index',[2,24])
```

Then rerun Step 7. To plot unwrapped phase with DEM error and master atmosphere subtracted:

```
>> ps_plot('u-dm')
```

Removing the spatially-correlated error from the wrapped phase can actually help with the unwrapping, and it is a good idea to run Step 6 again after running Step 7 (estimates from Step 7 are automatically subtracted from the wrapped phase before unwrapping).

If all interferograms have been unwrapped successfully, reset 'scla_drop_index' to the default value and rerun Step 7. If it is still not possible to unwrap all interferograms without errors you can drop some completely by setting 'drop_ifg_index' and rerunning from any Step 2, 3 or 4.

N.B If you want to rerun Step 6 at any time without subtracting the results from Step 7, enter

```
>> scla_reset
```

4.2. Ramp estimation

It can be interesting to look at unwrapped phase with a linear phase ramp subtracted, especially if you are interested in local signals only. You can do this by setting 'scla_deramp' to 'y' and rerunning Step 7. To visualise the unwrapped interferograms with the ramps subtracted:

```
>> ps_plot('u-dmo')
```

Note that you can plot unwrapped phase with any combination of estimated errors subtracted, e.g.

```
>> ps_plot('u-o') or >> ps_plot('u-do')
```

Subtracting a ramp can also help with phase unwrapping, so it may be worth rerunning Steps 6 and 7 if you suspect that errors remain.

5. Velocity estimation

You can plot the mean velocity using

```
>> ps_plot('v')
```

The large subsidence is due to water extraction beneath the city. Most of the city is built on old lake sediments, which subside the most.

To subtract DEM error and oribital ramps before estimating mean velocity, run

```
>> ps_plot('v-do')
```

(Note that there is no option to subtract 'm' first, as this would not change the mean velocity.)

The velocity values are relative to the mean velocity of the whole image, by default. You can set a circular reference area using the ‘ref_centre_lonlat’ and ‘ref_radius’ parameters, or a rectangular reference area using the ‘ref_lon’ and ‘ref_lat’ parameters. Pick an area in the west of the image and set this as your reference area.

5.1 Velocity standard deviation and time series

To estimate the standard deviation of the velocity estimates (relative to the reference area) with DEM errors and ramps subtracted, enter

```
>> ps_plot(‘vs-do’)
```

Areas with larger standard deviation indicate either larger errors (due perhaps to atmosphere or unwrapping errors) or deformation that is non-linear. You can get a handle on this by plotting time series of a few points.

```
>> ps_plot(‘v-do’,’ts’)
```

Click on the ‘TS plot’ button and select an area of the image using the cross hairs. The time series for the nearest resampled PS pixels will be plotted.

You can set the search radius within the plotted figure. Try changing the radius to 500 m and reselecting a point. Now all PS within 500 m of your selected position are plotted, together with the average of these PS.

Try plotting various positions with different standard deviations. Can you comment on the cause of the areas with higher standard deviations?

6. Images with strong atmosphere

If any images contain strong atmosphere, they can bias the estimate of velocity. You can see the effect that each image has on velocity estimation using

```
>> ps_plot(‘vdrop-do’)
```

This plots the mean velocity estimated by dropping each interferogram in turn. Is it a strong effect?

7. Output to Google Earth

You can plot output on Google Earth using ps_gescatter. It is best to plot at full resolution, but pick only a subset of the area to avoid crashing GE with too many points. Go into the PATCH_14 directory, run steps 6 and 7, then:

```
>>ps_plot('v-d',-1)

>>load ps_plot_v-d

>>ps_gescatter('ps.kml',ph_disp,1,0.4,[-10,50],[-98.95,-
98.98],[19.44,19.48])
```

This will create a file ps.kml in PATCH_14 directory, which can be opened in Google Earth.

7. Other

You can play with separating the atmospheric and deformation signals using StaMPS Step

8.