Homework 3 : Radar Imaging

Prof. Andrea Virgilio Monti Guarnieri & Dr. Marco Manzoni

Thibault Fievez 10729319
Group members : Hiva Amiri,Kimia Kiyan ,Vahid Rajabi

December 2020

For this third homework , the goal of the project was to form an interferogram in order to asses the deformation on ground caused by an earthquake.

1 Task 1

1.1

In this subsection , we extract the files from the master picture, as the real and imaginary part of each pixel are stored separately , we add them back together to create the full complex number.It is important to keep in mind that we sub-sample that picture to alleviate the computational load. In order to remove any outlier and being able to colour scale efficiently to an acceptable figure , we calculate the 0.98 percent and set any value above this to the 98'th quantile's value.

On top of that we decide to do a multilook approach where we average a pixel with its surrounding in order to tackle the noise contribution. In this example we tried with many values and settled with 20 surrounding pixels. Taking more pixels reduces the noise but eats up the definition and the sharpness of the picture , it becomes more blurred.

You can see the code on figure 1 .

```
%% Task1.1
subs = 3;
infosFileMaster = h5info("C:\Users\Thibault Fievez\Documents\radar\hw3\sardata\sardata\20190704.h5");
realPartImage = h5read(infosFileMaster.Filename, "/i_VV");
realPartImage = realPartImage(1:subs:end, 1:subs:end);
imagPartImage = h5read(infosFileMaster.Filename, "/q_VV");
imagPartImage = imagPartImage(1:subs:end, 1:subs:end);
master = single(realPartImage)+1j.*single(imagPartImage);
figure;
imagesc(abs(master));
figure;
imagesc(angle(master));

quant=(quantile(abs(master),0.98,'all'));

indices=find(abs(master)>quant);
%just to be able to keep our varialbe master untouched for later on
newmaster=abs(master);
newmaster(indices) = quant;

figure;
imagesc(abs(newmaster));

%Averaging (Multilook)

Meanmaster= movmean(abs(newmaster),20);
figure;
imagesc(Meanmaster);
%Geocoding
```

Figure 1: Code 1.1

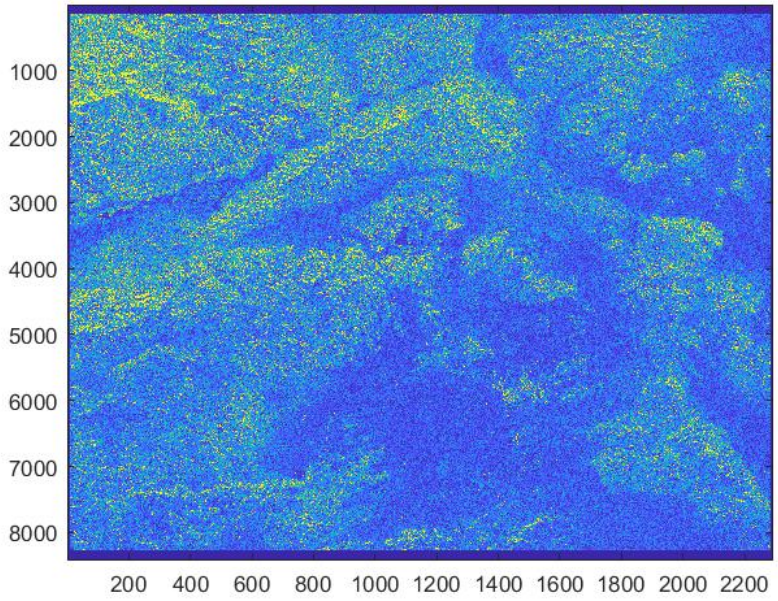

Figure 2: Absolute value of the master without outliers

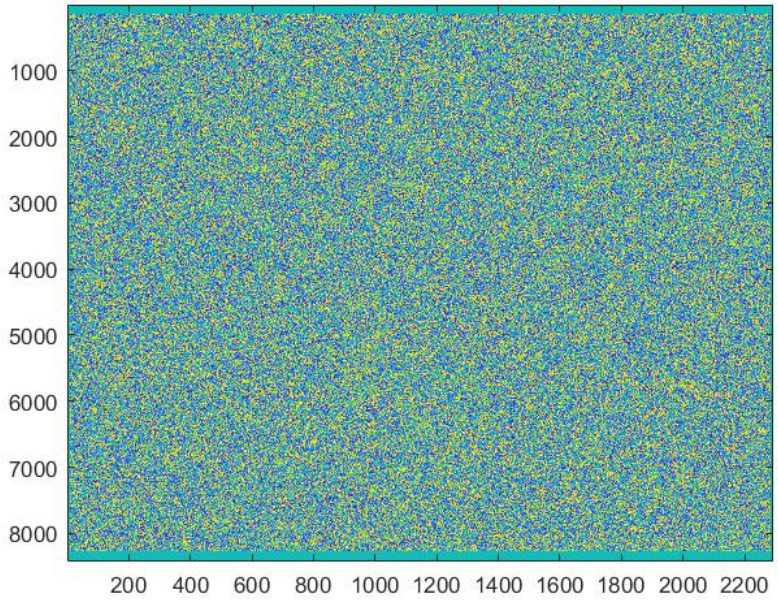

Figure 3: Phase of master

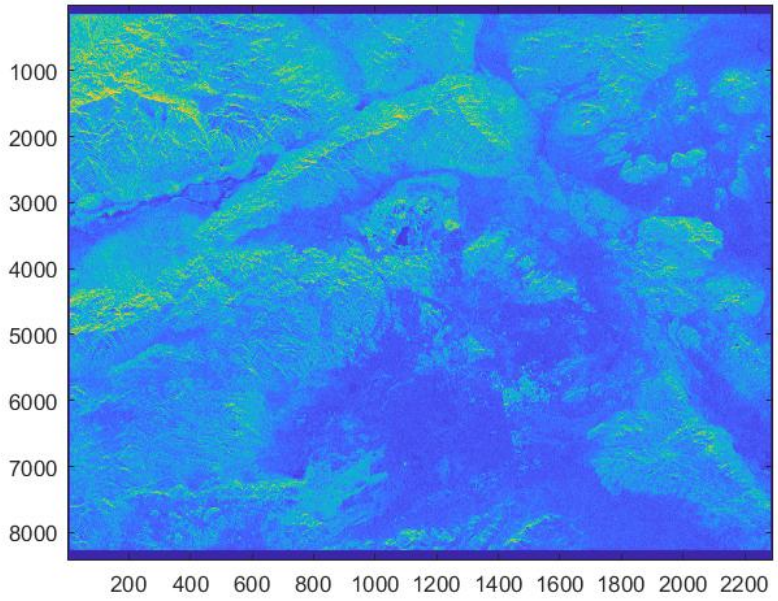

Figure 4: Absolute value of master with averaging over 20 pixels

For interesting results we can geocode and export our result to map it and visualise differences. Let's visualise it on QGIS.

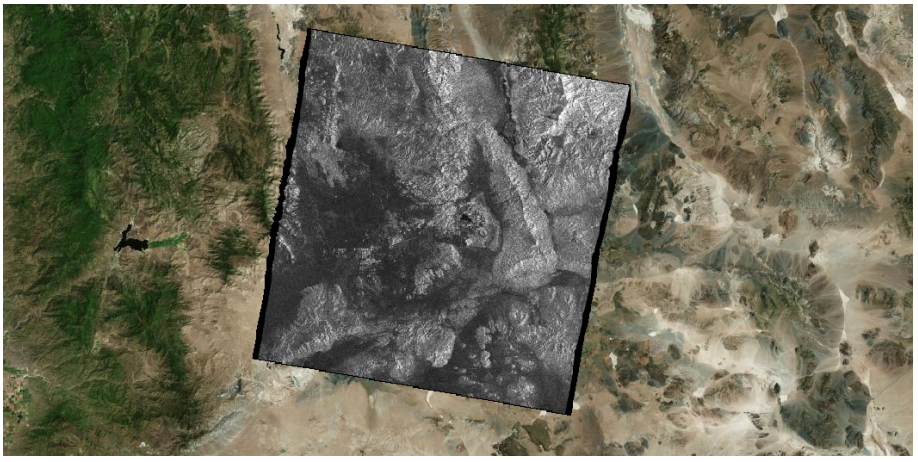

Figure 5: QGIS mapping

1.2

In this subsection , we shall load the the slave file , similarly and apply the same procedures as we did with the master file. Once done , we start working towards the main objectif of this homework which is the interferogram. An interferogram allow us to determine the difference between two pictures, and thus allow us to do a mapping of changes between the two. To do an interferogram we multiply the first (master) but the conjugate of the second(slave) as showcased on the code. Once again we could remove the outliers with the 98 percent quantile but unlike the first case , for interferogram, the information of difference between the two pictures is stored in the phase of the interferogram. As such we can plot the phase of the interferogram as shown on figure 9, and this phase difference represents the variation of distance between the satellite and the ground on

the two pictures. As we get two phase noise contribution(one from the master and one from the slave), the picture is more noisy.

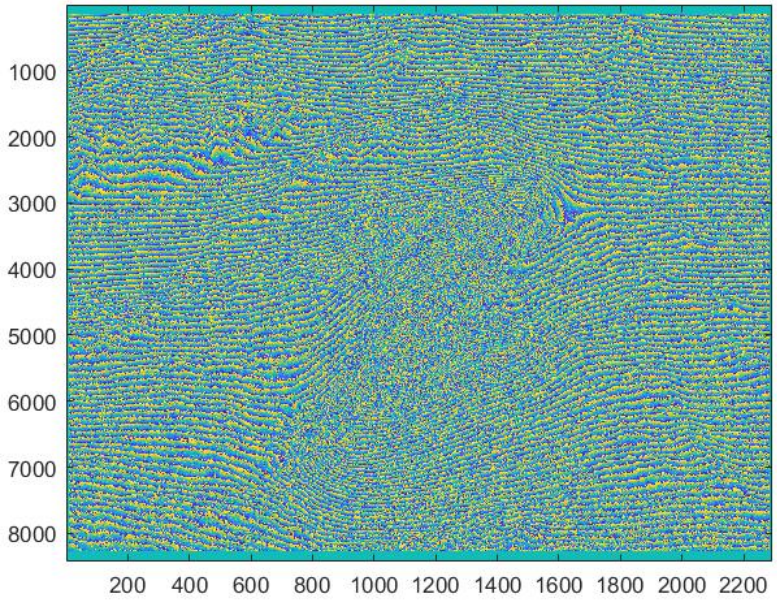

Figure 6: Phase of the interferomgram

```
%Creating interferogram
interferogram = master.*conj(slave);
%Ploting the phase of interferogram , REMAKR , should i alos remove the 98
%quantile ? in both case i get a pretty disapointing result .
indices=find(angle(interferogram)>quant);
%just to be able to keep our varialbe master untouched for later on
newinterferogram=angle(interferogram);
newinterferogram(indices) = quant;
% not used in the end

figure
imagesc(angle(interferogram));

                % reading the distances for the master
read_distance_m = h5read(infosFileMaster.Filename, "/topoDistance");

sub_distance_m = read_distance_m(1:subs:end, 1:subs:end);% subsampling the distance matrix

new_master = master.*exp(1j.*(4*pi*f/c).*sub_distance_m); % compensating the msater

                 % reading the distances for the slave %
read_distance_s = h5read(infosFileSlave.Filename, "/topoDistance");

sub_distance_s = read_distance_s(1:subs:end, 1:subs:end);% subsampling the distance matrix

new_slave = slave.*exp(1j.*(4*pi*f/c).*sub_distance_s); % compensating the msater

                % computing new interferogram
new_interferogram = new_master.*conj(new_slave);
figure
```

Figure 7: Code 1.2

To further improve our design we can compensate the phase shift induced by the distance for the master and slave. This is done by knowing the exact location of the satellite when talking the picture. This has the property to remove the fast fringes in the range direction and to compensate the fringes due to topography.

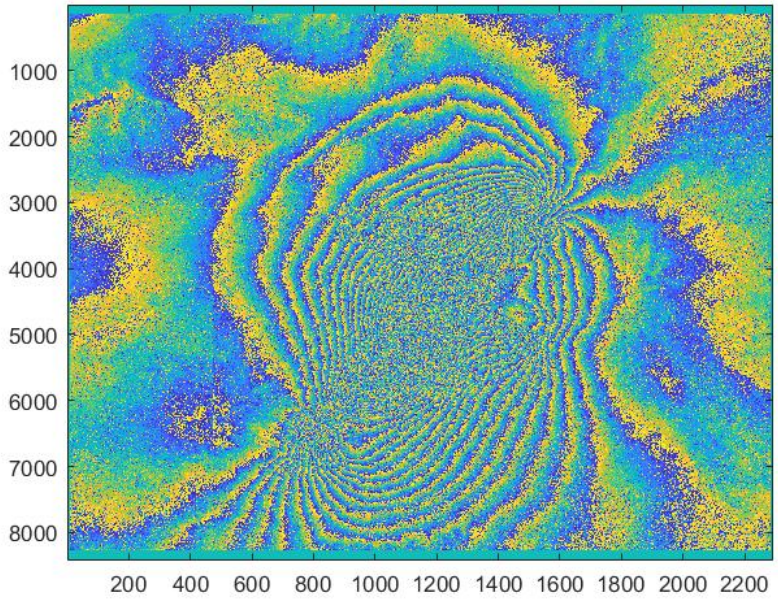

Figure 8: Phase of the compensated distance interferogram, we can clearly see the shape of the topography.

Finally let's see what it does on QGIS.

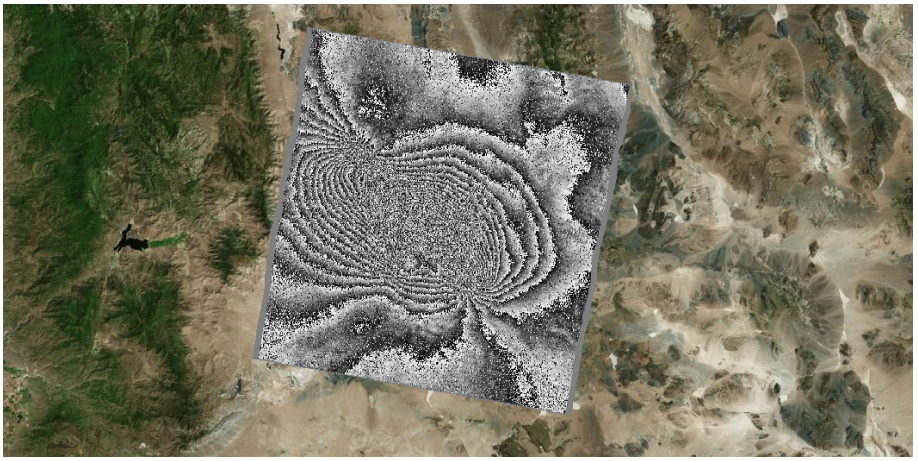

Figure 9: Phase of interferogram on QGIS

1.3

Now , as we have done previously, we can average each pixel with it's neighbours to minimise the noise contribution.It is asked to do a window of 43*11 pixels , but because we have downsampled by 3, this means we use a window of 15*4=60 pixels. we therefore use movmean over 60 pixels.

Results and code can be seen on the following figures.

```
%% task 1.3



Mean_new_master= movmean(abs(new_master),60);
Mean_new_slave= movmean(abs(new_slave),60);
% why 60? we sould compute the equivalent # of pixels in subsamping so:
% a = zeros(43,11)
% size(a(1:subs:end, 1:subs:end)) = [15,4]
% 15*4 = 60 : # of equivalent pixel
figure;
imagesc(Mean_new_master);
figure;
imagesc(Mean_new_slave);

%****//// decode the image ... ////****%


% Geocode
```

Figure 10: Code 1.3

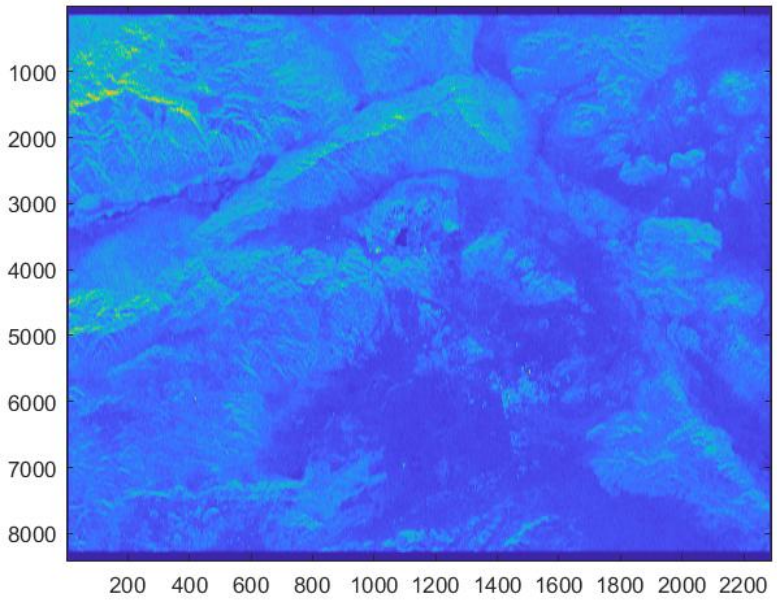

Figure 11: After averaging/multilook on 60 pixels

Once again we export it to QGIS.

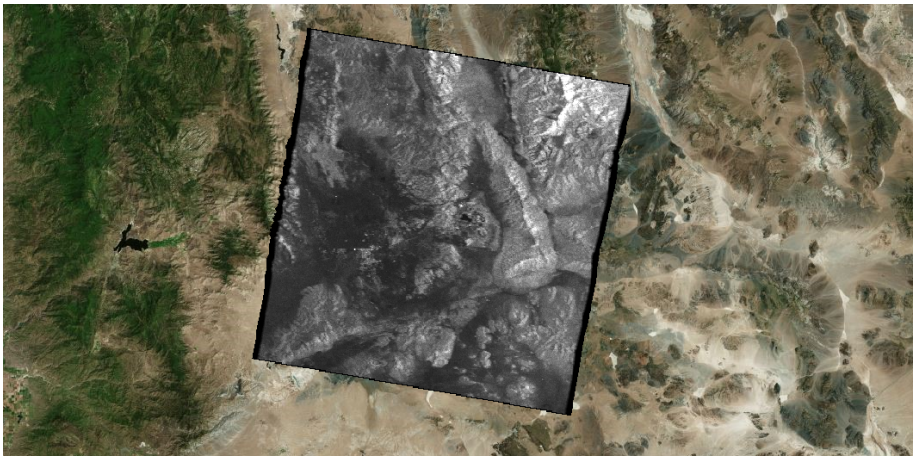

Figure 12: Imported on QGIS

2

2.1

Mapping the coherence of the scene will allow us the measure the quality of the interferometric phase, if the absolute value of the coherence value is high , it means that the phase is reliable whereas a low coherence means that the phase is corrupted by noise. The coherence can be determined by the following equation.

$$\gamma(P) = \frac{E(x_1.(x_2)^*)}{\sqrt{E(abs(x_1)^2).E(abs(x_2)^2)}} \tag{1}$$

The plot of that coherence map is showcased on figure 14.

```
%% task 2.1 :Computing a coherence map
clc
range_step = 20;
azimuth_step = 5;

coherence_map = (movmean(new_interferogram,60)) ./  (sqrt(movmean((new_master).^2,60) .* movmean((new_slave).^2,60) ) ); % The mean is done with 60 surrounding data
sub_coherence_map = coherence_map(1:range_step:end, 1:azimuth_step:end); %this is sumsampled twice

figure
ax1 = subplot(2,1,1)
imagesc(abs(sub_coherence_map))
ax2 = subplot(2,1,2)
imagesc(angle(sub_coherence_map))
linkaxes([ax1,ax2],'x');


%****//// decode the image ... ////****%
figure
load('C:\Users\Thibault Fievez\Documents\radar\hw3\sardata\sardata\geocoding_infos.mat');
geocodingInfos = GecSubs(geocodingInfos, [subs*20 subs*5]);
geocodedImage = geocodeRadarImage(abs(sub_coherence_map), geocodingInfos); % by using coherance map and not the sumbampled one we dont need to interpolate

grid2image(geocodedImage, geocodingInfos.xref);

geotiffwrite("task21.tif",geocodedImage, geocodingInfos.xref);
```

Figure 13: Code of section 2.1

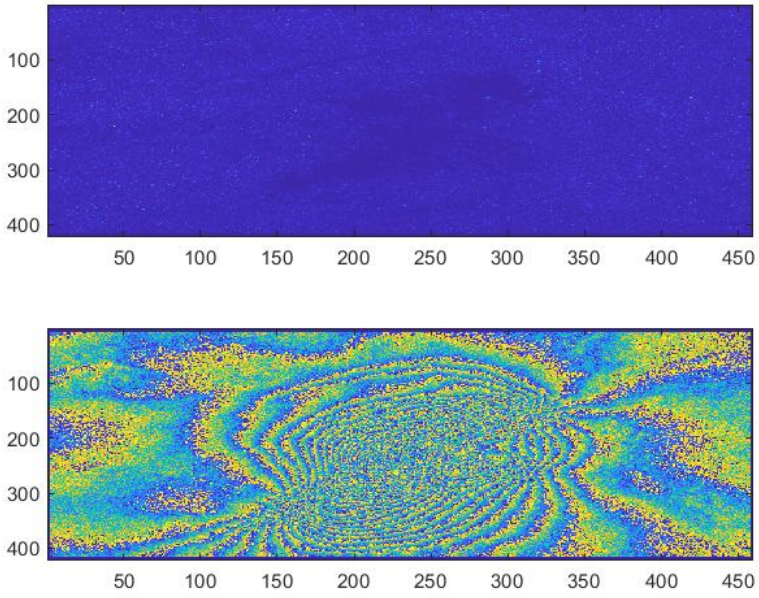

Figure 14: Coherence map and interferometric phase , we can see the the center lower area has a poor coherence and therefore noisier data.

And once again the coherence map placed on QGIS.

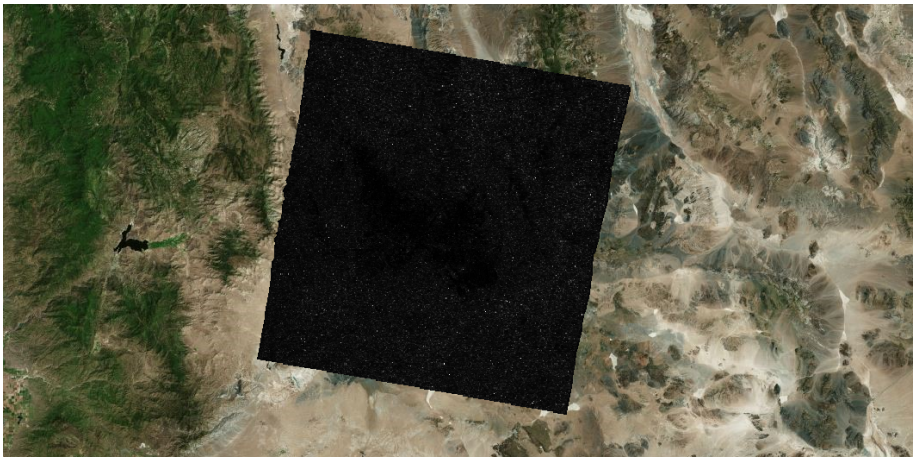

Figure 15: QGIS for 2.1

2.2

Because the phase is only know by modulo 2π , we can use several methods that unwraps the phase field ,but before that and to ease the calculation process , we sub-sample even more, up to 60 in range and 15 in azimuth. Without that we were having difficulties to run the code on any computer even by clearing out the un-used variables.[1] As hinted in the homework , we use a unwrapping function that takes the angle of the coherence (angle of interferogram), as well as its absolute value as a set of "weights" for the unwrapping algorithm.

[1] Some of our group mates had only 8Gb or 4GB of RAM which would show to be insufficient.

```
%% task 2.2

clear all;
load 'subcoherence.mat';


unwrappedInterferogram = unwrap_IRLS(double(angle(sub_coherence_map)),abs(sub_coherence_map), 100, [], 1); % coherence map represent the quality of the phase at each point


subs=3;
figure;
load('C:\Users\Thibault Fievez\Documents\radar\hw3\sardata\sardata\geocoding_infos.mat');
geocodingInfos = GecSubs(geocodingInfos, [subs*20 subs*5]);
geocodedImage = geocodeRadarImage(abs(unwrappedInterferogram), geocodingInfos); %

grid2image(geocodedImage, geocodingInfos.xref);

geotiffwrite("task22.tif",geocodedImage, geocodingInfos.xref);
```

Figure 16: Code 2.2

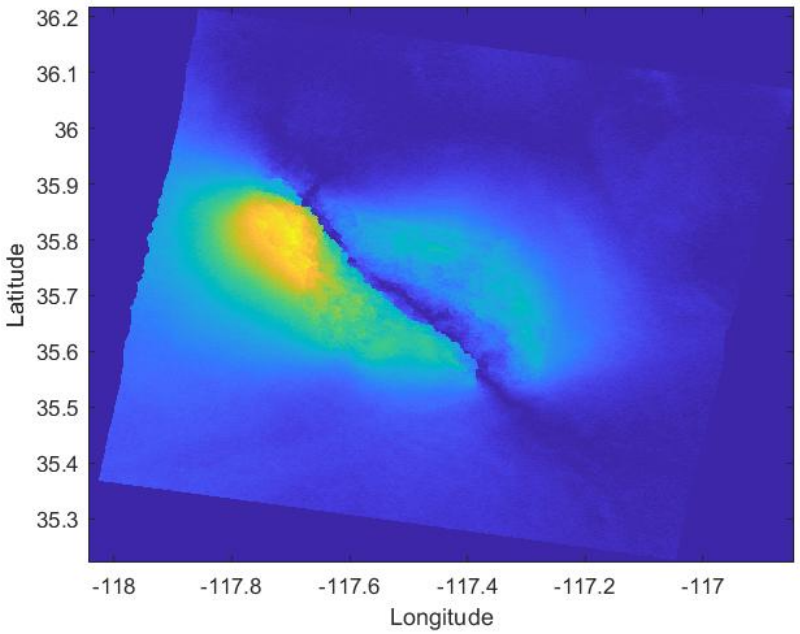

Figure 17: Output of interferogram goecoded

Once again we can place it in QGIS.

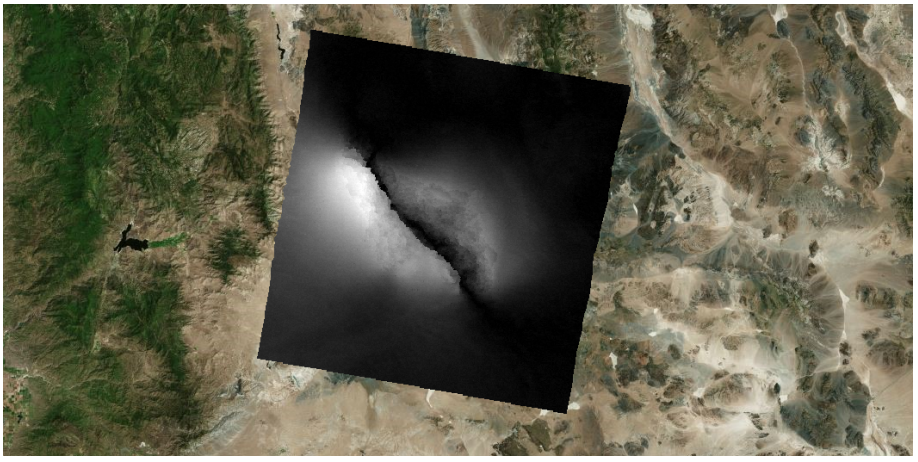

Figure 18: Output of interferogram goecoded and palce on QGIS

3

3.1

In this section we calculate the gradient of the image of the unwrapped interferometric phase in radar coordinates. Those gradients are in latitude and longitude direction.

```
%% task 3.1
clc
angle_unwrappedInterferogram = angle(unwrappedInterferogram);
gradiant_azimuth_range = gradient(gradient(unwrappedInterferogram)')';

subs=3;
figure;
load('C:\Users\Thibault Fievez\Documents\radar\hw3\sardata\sardata\geocoding_infos.mat');
geocodingInfos = GecSubs(geocodingInfos, [subs*20 subs*5]);
geocodedImage = geocodeRadarImage(abs(gradiant_azimuth_range), geocodingInfos); %

grid2image(geocodedImage, geocodingInfos.xref);

geotiffwrite("task31.tif",geocodedImage, geocodingInfos.xref);
```

Figure 19: Gradient of unwrapped interferometric image

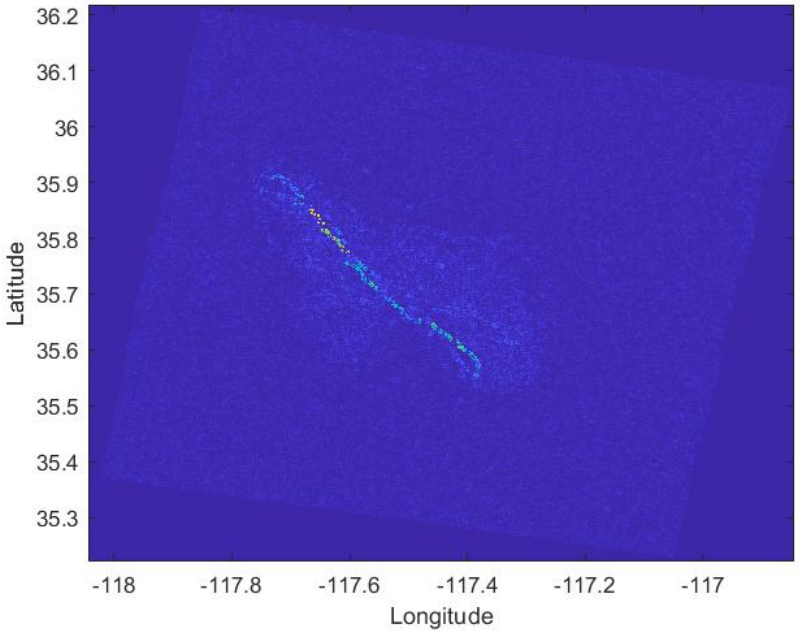

Figure 20: Gradient of unwrapped interferometric image

3.2

We can use improfile to select a line in the data figure (the one produced in 2.2) and compute the profile of that line.By doing so we can analyze the gradient and /or the data by doing a "cut" across it.

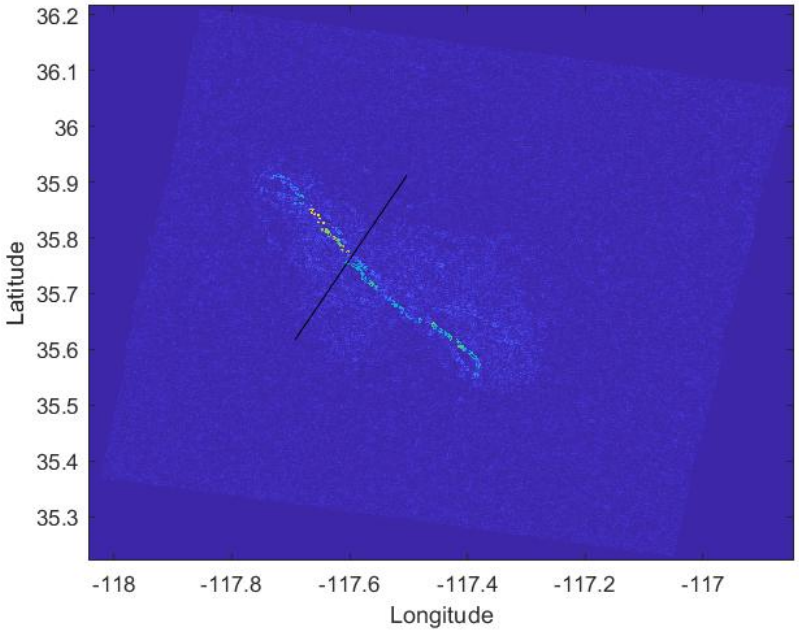

Figure 21: Gradient of unwrapped interferometric image and line of cut.

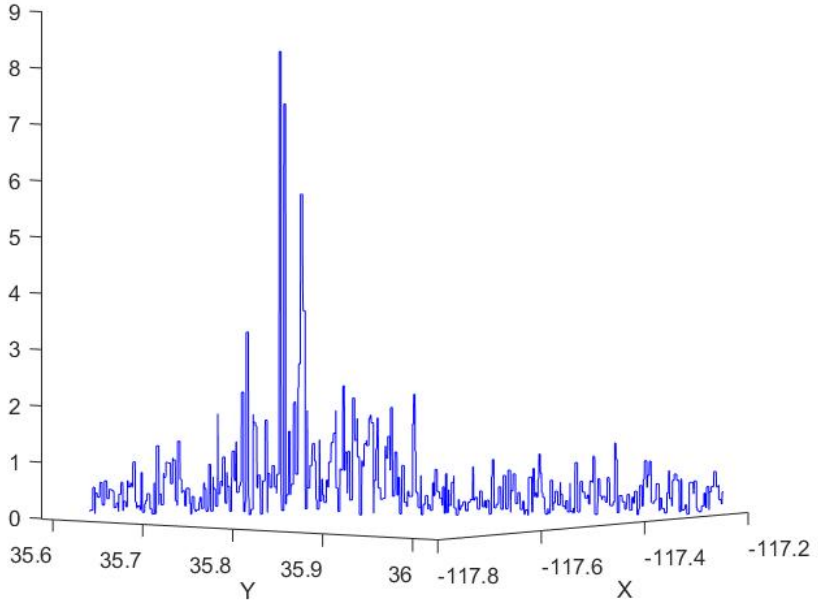

Figure 22: Gradient of unwrapped interferometric image across the line.

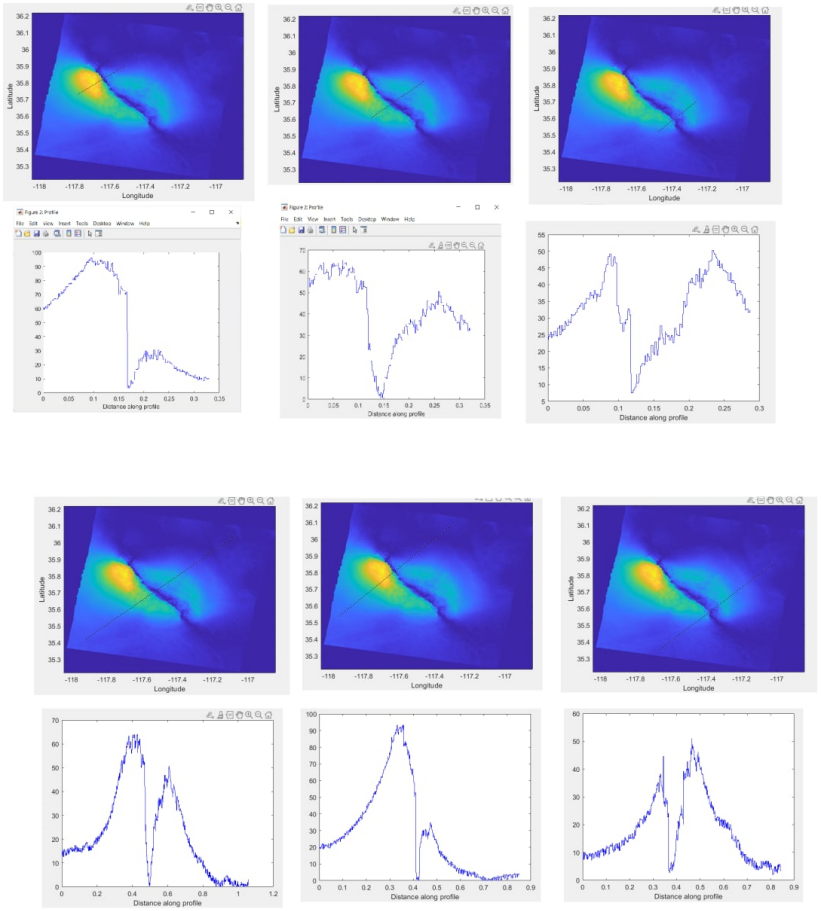

Figure 23: Analysing the data with Improfile, enables us to cut through the data and 2D plot it.

With those we can asses that 2 tectonic plates are meeting and going upwards.