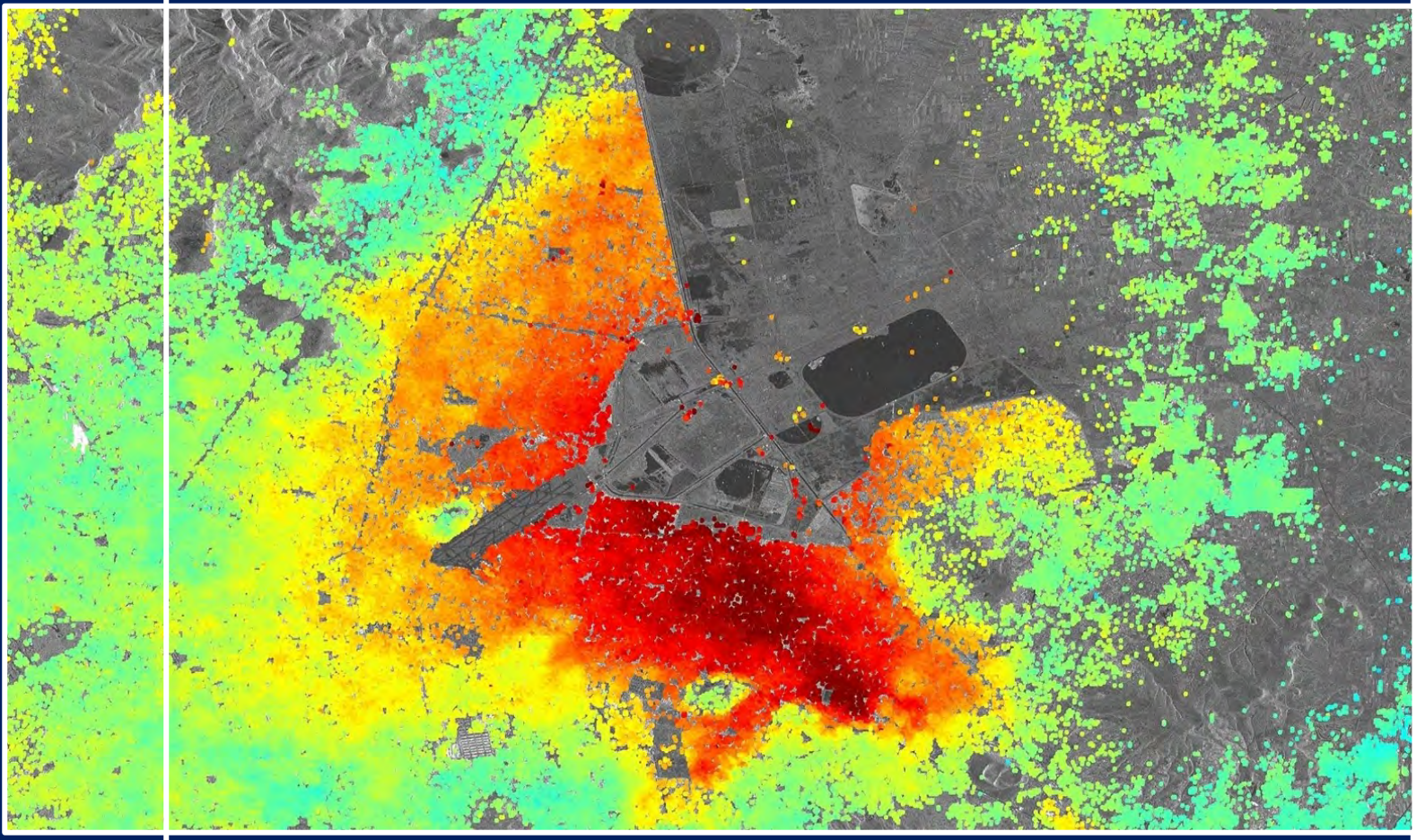


## TRAINING KIT – HAZA09

# SNAP2StaMPS: Data preparation for StaMPS PSI processing with SNAP

## Case Study: Mexico City, Nov. 2019 - Nov. 2020

Research and User Support for Sentinel Core Products

The RUS Service is funded by the European Commission, managed by the European Space Agency and operated by CSSI and its partners.

Authors would be glad to receive your feedback or suggestions and to know how this material was used. Please, contact us on training@rus-coperenicus.eu

Cover images produced by RUS Copernicus

The following training material has been prepared by Serco Italia S.p.A. within the RUS Copernicus project.

Date of publication: December 2020

Version: 1.1

Suggested citation:

Serco Italia SPA (2020). *SNAP2StaMPS: Data preparation for StaMPS PSI processing with SNAP - Mexico City 2020 (version 1.1).* Retrieved from RUS Lectures at https://rus-copernicus.eu/portal/the-rus-library/learn-by-yourself/

# Table of Contents

# 1 Introduction

The Research and User Support for Sentinel core products (RUS) service provides a free and open scalable platform in a powerful computing environment, hosting a suite of open-source toolboxes pre-installed on virtual machines, to handle and process data derived from the Copernicus Sentinel satellites constellation. In this tutorial we will employ RUS to identify and map land subsidence in Mexico City using Sentinel-1 data.

*Mexico City's buildings are seriously leaning due to land subsidence. Photo credit: JOSH HANER/THE NEW YORK TIMES (http://www.sciencemag.org)*

Land subsidence in Mexico City caused by groundwater overexploitation over the last century has been more than 9 meters, resulting in damages to buildings, streets, sidewalks, sewers, storm water drains and other infrastructure [1].

Due to the fact that the city is partially built on the area of a former lake (Lago Texcoco), it rests on the heavily saturated clay which is collapsing due to the over-extraction of groundwater. Current subsidence rates using Sentinel-1 SAR data approximate 2.5 cm/month [3].

Persistent Scatterer Interferometry (PSI) is a powerful advanced DInSAR technique able to measure and monitor displacements of the Earth's surface over time with hight accuracy. Hooper et al. (2004). proposed a novel PS selection using phase characteristics, which is suitable to find low-amplitude natural targets with phase stability that cannot be identified by amplitude-based algorithms. This work originated one of the most widely used PSI software packages, StaMPS. [1]

SNAP2StaMPS is a Python workflow developed by José Manuel Delgado Blasco, Michael Foumelis in collaboration with Prof. A. Hooper to automate the pre-processing of Sentinel-1 SLC data and their preparation for ingestion to StaMPS.

# 2 Training

Approximate time needed to complete this training session is two hours.

## *2.1 Data used*

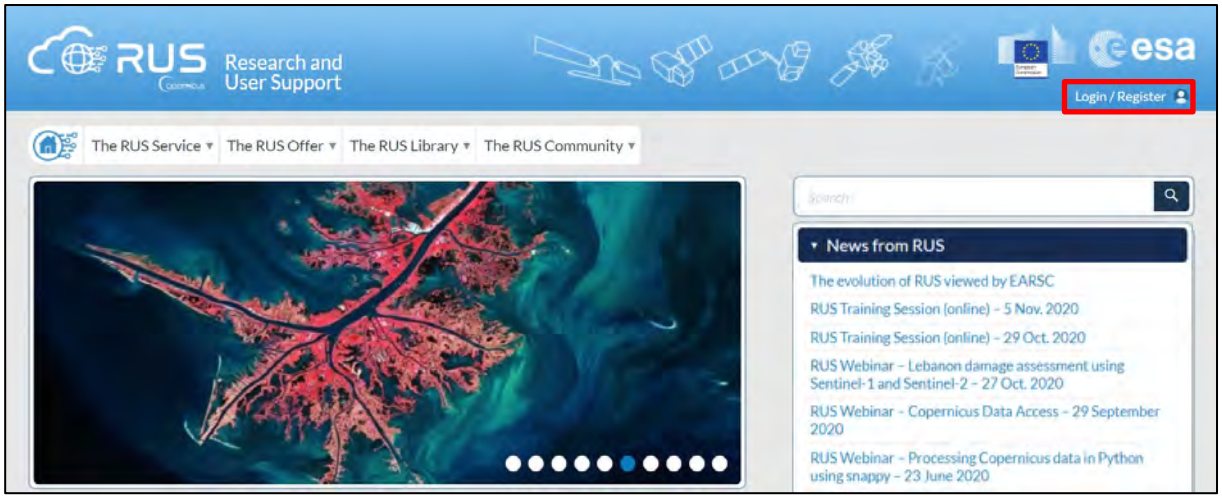
- 31 Sentinel-1A images acquired between November 1, 2019 and November 1, 2020. [downloadable @ASF Data Search portal – Vertex]
- Auxiliary data stored locally @ /shared/Training/HAZA09_SNAP2StaMPs_DataPrep_TutorialKit/AuxData

## *2.2 Software in RUS environment*

Internet browser, SNAP + Sentinel-1 Toolbox, QGIS, Google Earth

# 3 Register to RUS Copernicus

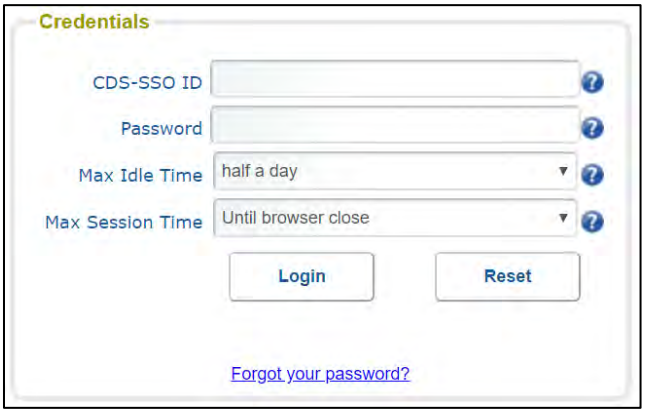
To repeat the exercise using a RUS Copernicus Virtual Machine (VM), you will first have to register as a RUS user. For that, go to the RUS Copernicus website (www.rus-copernicus.eu) and click on ***Login/Register*** in the upper right corner.



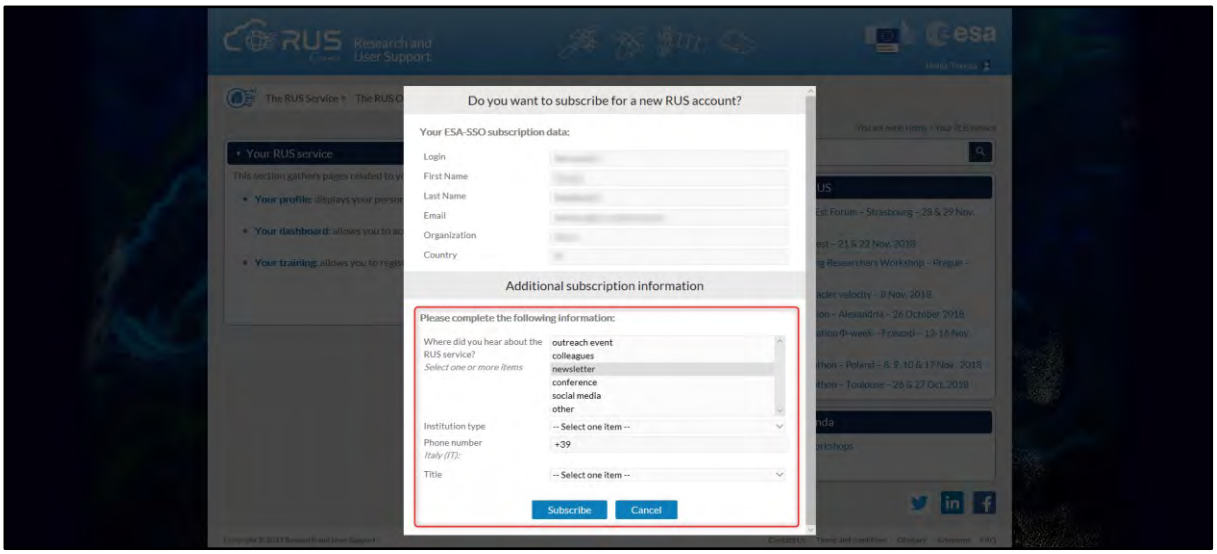

Select the option ***Create my Copernicus SSO account*** and then fill in ALL the fields on the **Copernicus Users' Single Sign-On Registration**. Click ***Register***.



Within a few minutes you will receive an e-mail with activation link. Follow the instructions in the e-mail to activate your account. You can now return to https://rus-copernicus.eu/, click on ***Login/Register***, choose ***Login*** and enter your chosen credentials.

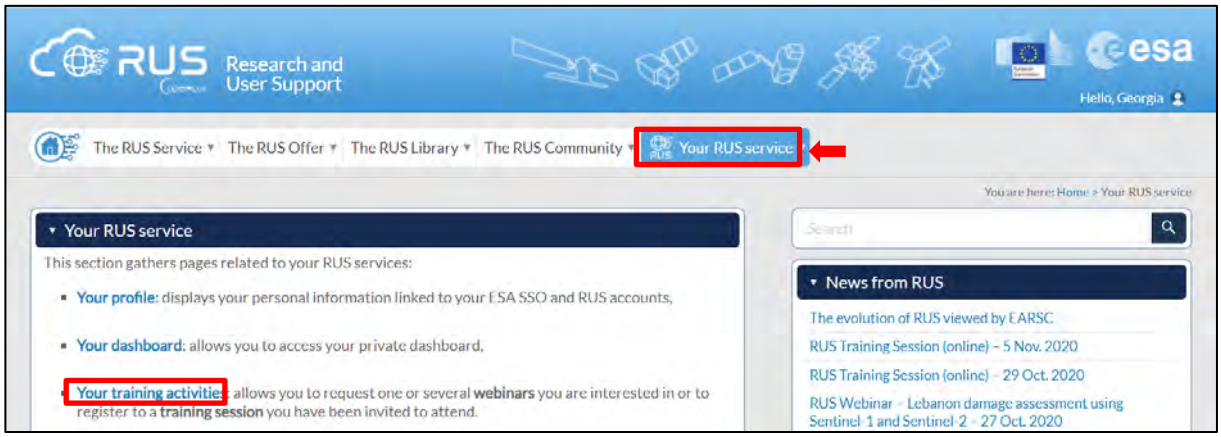



Upon your first login you will need to enter some details. You must fill all the fields.

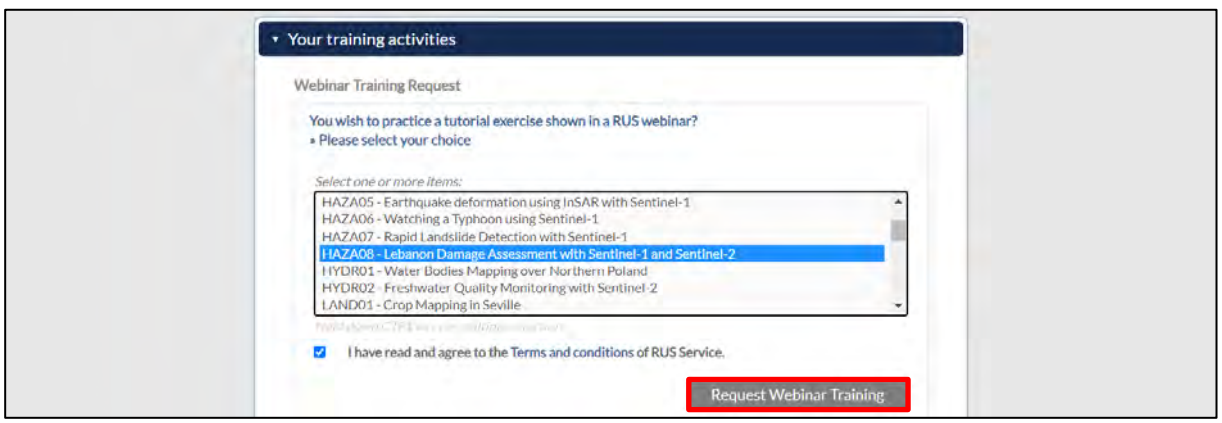



## 4 Request a RUS Copernicus Virtual Machine to repeat a Webinar

Once you are registered as a RUS user, you can request a RUS Virtual Machine to repeat this exercise or work on your own projects using Copernicus data. For that, log in and click on ***Your RUS Service → Your training activities***.

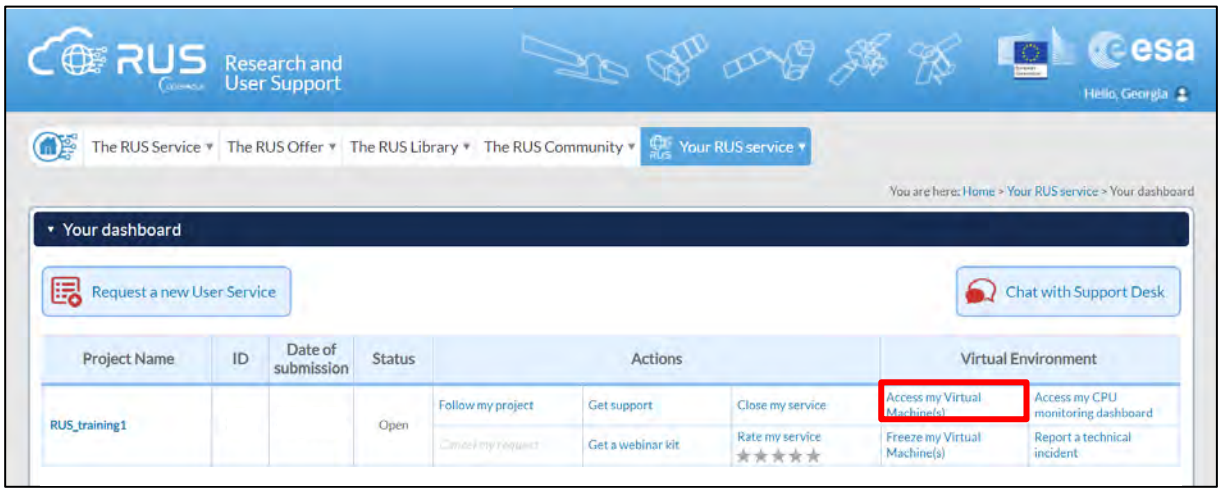



Select **HAZA09**, check the field *"I have read and agree to the Terms and conditions of RUS Service"* and then click on ***Request Webinar Training*** to request your RUS Virtual Machine.



Further to the acceptance of your request by the RUS Helpdesk, you will receive a notification email with all the details about your Virtual Machine.

To access it, go to ***Your RUS Service → Your Dashboard*** and click on ***Access my Virtual Machine***.




Fill in the login credentials that have been provided to you by the RUS Helpdesk via email to access your RUS Copernicus Virtual Machine.

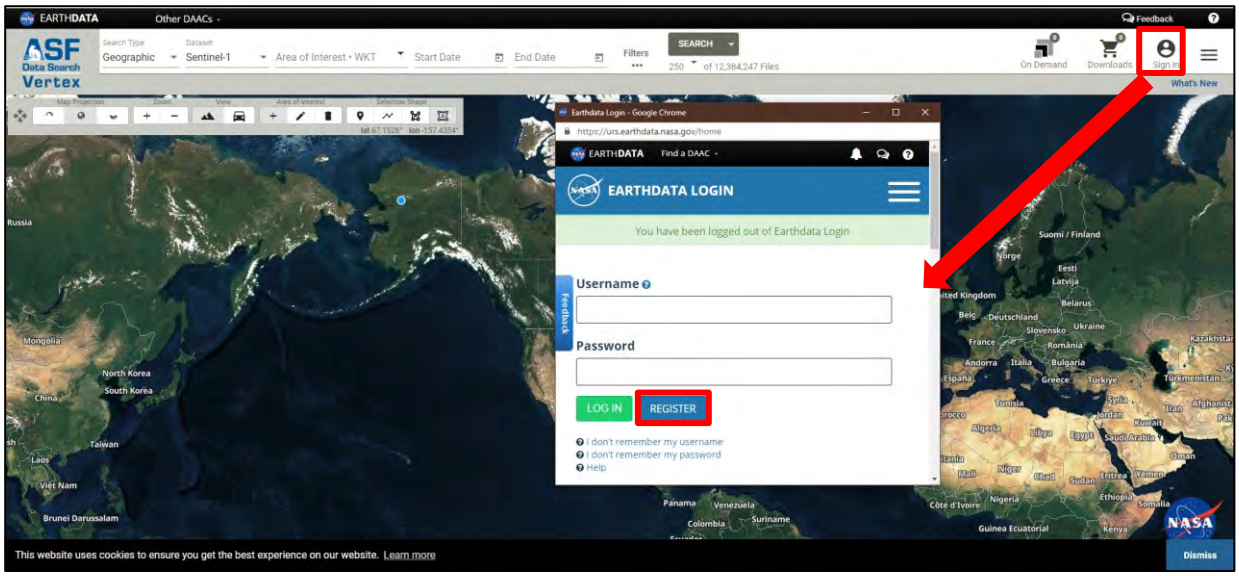

This is the remote desktop of your Virtual Machine.

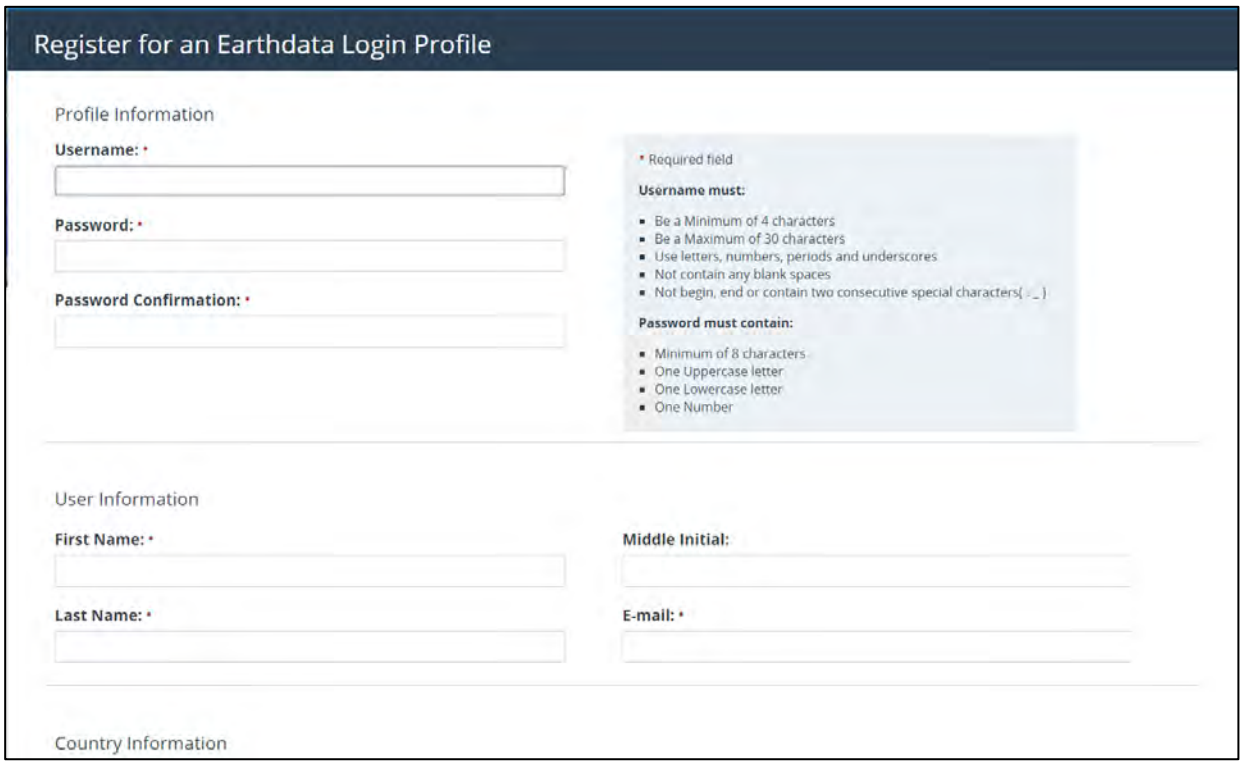
# 5 Data download

For this exercise we will introduce another portal for the download of Sentinel-1 SLC data. We will use the Alaska Satellite Facility portal which offers the full Sentinel-1 SLC archive. Many other portals (including Copernicus Open Access Hub) move data older than certain period to long term archive which makes their retrieval more time demanding.

We will work with 1 year of data from 1 November 2019 to 1 November 2020 and concentrate on acquisitions only by Sentinel-1A satellite in descending pass. In total this amounts to timeseries of 31 images in 12-day interval.

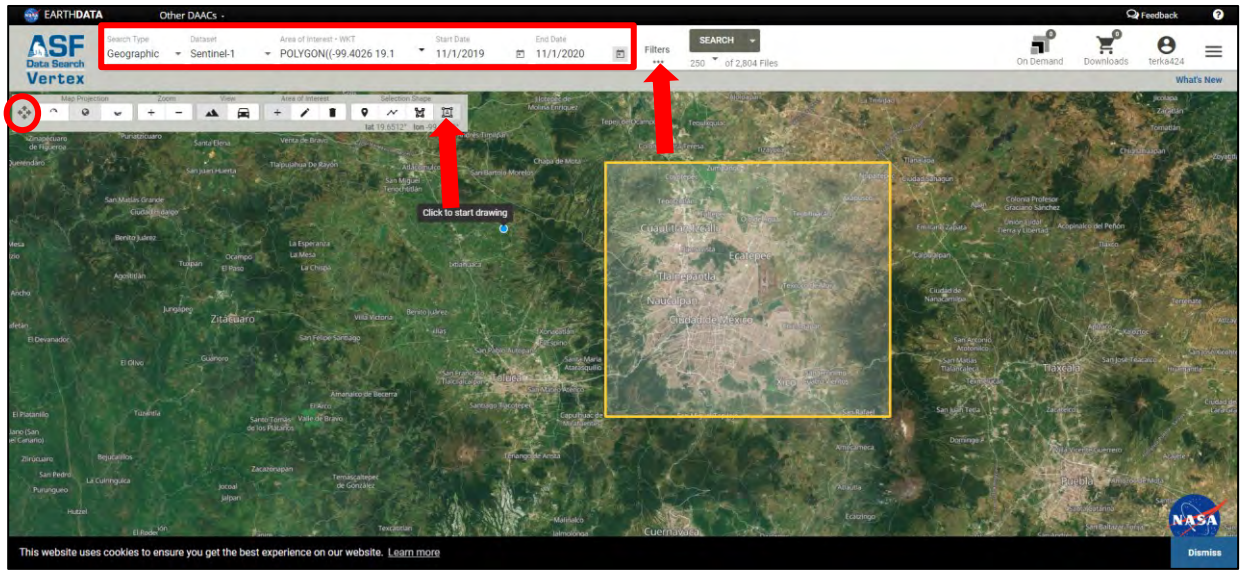
Go to the ASF Data Search portal – Vertex. Click on **Sign in**, a new window will appear. Click **Register** and fill your details.



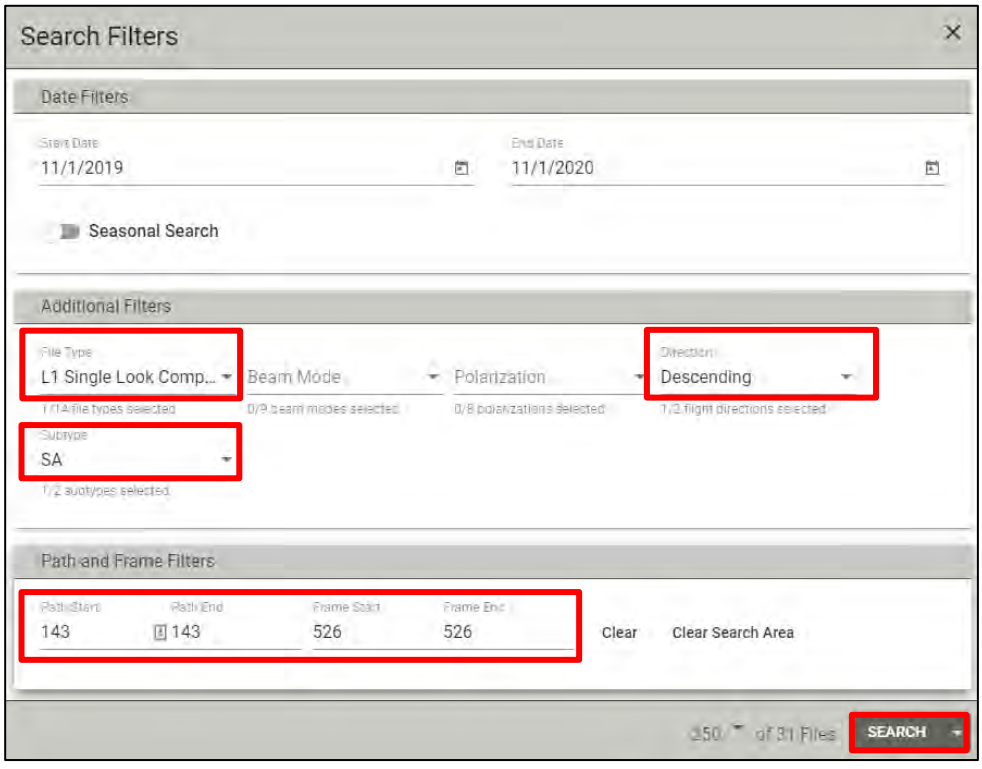
Once you complete your registration and validate your e-mail address, log in. Navigate to Mexico City, select the Box drawing tool and draw a rectangle over the area of interest.

Then in the top panel select:
**Start Date:** 1 November 2019
**End Date:** 1 November 2020

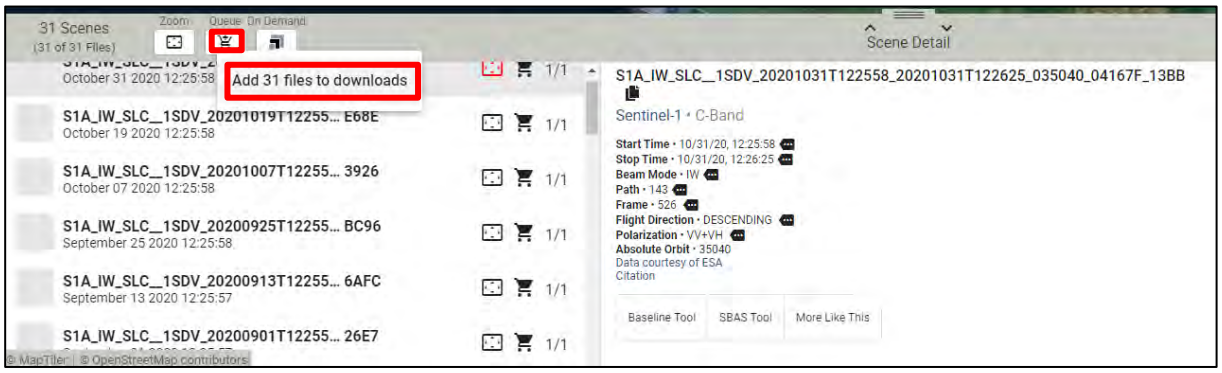



Then go to **Filters** and set:

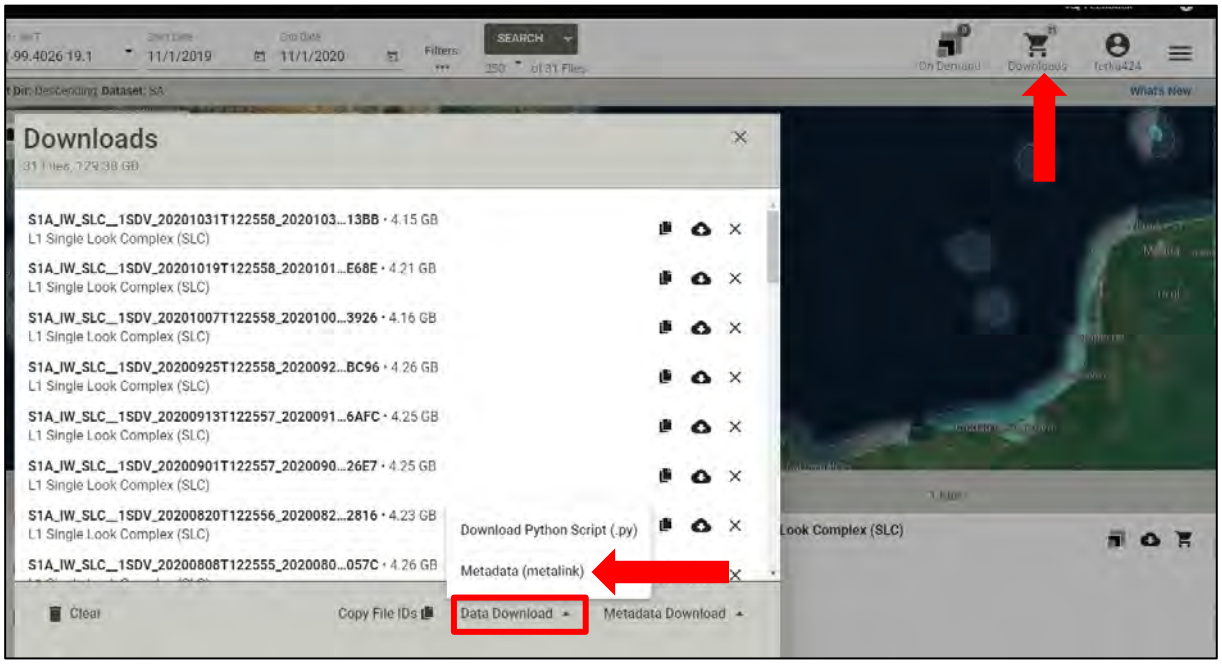
**File Type:** L1 Single Look Complex
**Direction:** Descending
**Subtype:** SA
**Path Start:** 143 **Path End:** 143 (corresponds to relative orbit)
**Frame Start:** 526 **Frame End:** 526



Click **Search**. The search returns 31 products. Above the results list, click on **Add all results to downloads**. You need to click on **Add 31 files to downloads**.




Now go to the **Downloads** tab next to the account button and in the window that appears click on **Data Download**. Now you can either choose a python script or metalink to download the data. We will use **Metadata (metalink).** Click on it and a ***.metalink*** file will be downloaded.




Go to the **/home/rus/** folder and move the metalink to:

***/shared/Training/HAZA09_SNAP2StaMPs_DataPrep_TutorialKit/Original/***

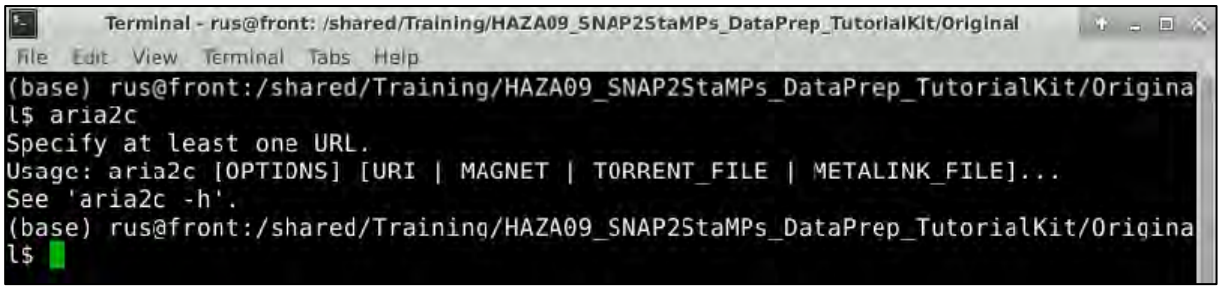
You can also rename it to: ***MC_desc.metalink***

To download our data, we will use **aria2** tool. To do this, go to the ***/Original*** folder where we have placed the metalink file and right-click on the white space in the folder. Select Open Terminal Here. A terminal window will open.

First, let's test our aria2 installation. Type:

```
aria2c
```

The correct response should be as follows:




NOTE 1: If (**and only if**) the response is "-bash aria2c: command not found". Then we have to install the tool, to do this in command line type: `sudo apt-get install aria2`
When requested type: Y
Then you can test your installation again.

If the response is "**-bash aria2c: command not found**" see NOTE 1. If you have received the correct response, then we can run the tool by typing the following commands in the command line (replace **<username>** and **<password>** with your login credentials for Copernicus Open Access Hub):

```
aria2c --http-user='<username>' --http-passwd='<password>' --check-certificate=false -M MC_desc.metalink
```

The first line changes our directory to the target directory. The second line runs the download tool (Type the text all in a single line). All twelve products will be downloaded to the ***Original*** folder, automatically.

# 6 Step by step

## *6.1 SNAP2StaMPS download and installation*

Persistent Scatterer Interferometry (PSI) is a powerful advanced DInSAR technique able to measure and monitor displacements of the Earth's surface over time with hight accuracy. Hooper et al. (2004). proposed a novel PS selection using phase characteristics, which is suitable to find low-amplitude natural targets with phase stability that cannot be identified by amplitude-based algorithms. This work originated one of the most widely used PSI software packages, StaMPS. [1]

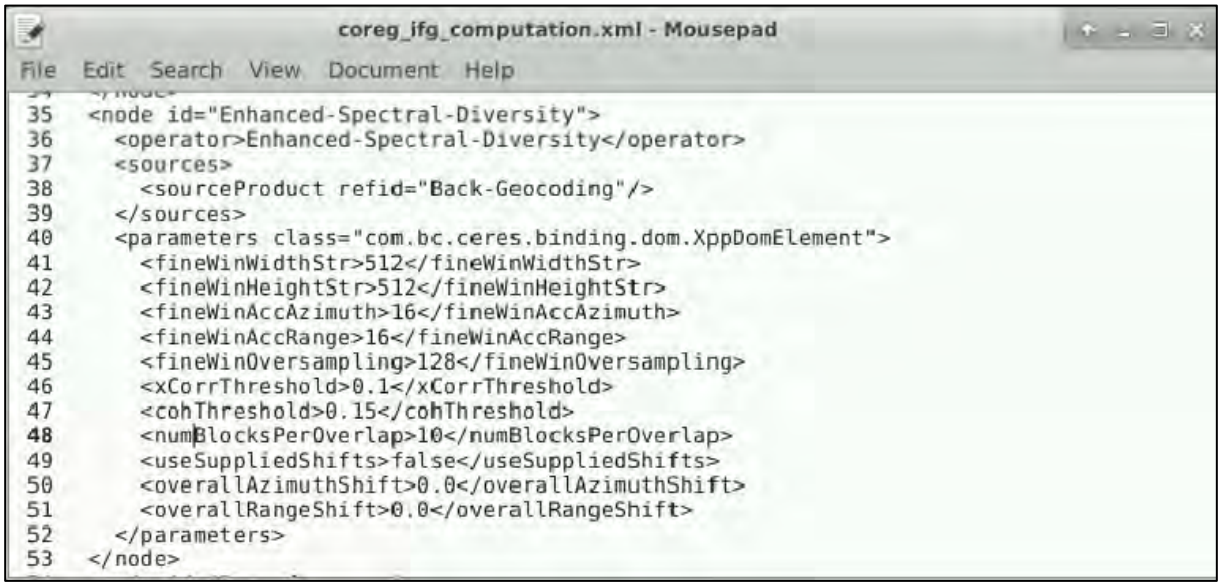
SNAP2StaMPS is a Python workflow developed by José Manuel Delgado Blasco, Michael Foumelis in collaboration with Prof. A. Hooper to automate the pre-processing of Sentinel-1 SLC data and their preparation for ingestion to StaMPS. The running of StaMPS SW requires MATLAB licence.

*The Following part is directly taken from the SNAP2StaMPS - manual and StaMPS manual.*

`StaMPS` is compatible with the output generated by the ESA SentiNel Application Platform (SNAP) after the version 6.0. SNAP allows the user to define a series of xml files which contain user defined processing workflow by using its Graph Builder. These files can be used to run SNAP processing in batch mode by using the GPT command (Graph Processing Tool).

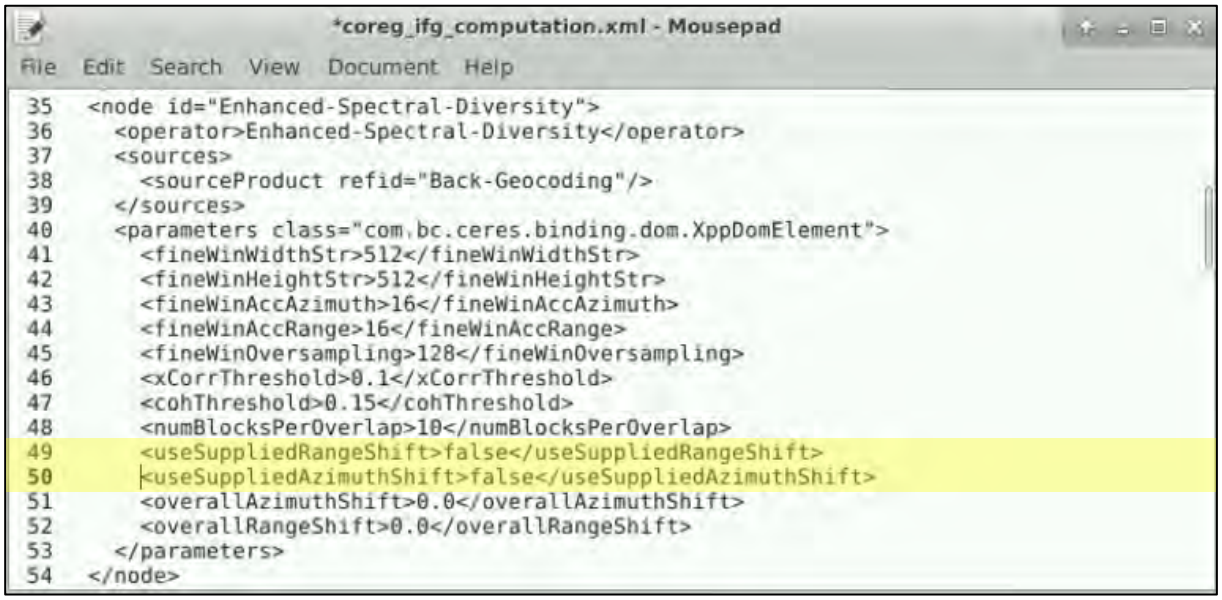
`snap2stamps` contains a set of graphs, together with python wrappers that allow you to automatise the interferogram processing chain for single master interferograms compatible with StaMPS PSI. Information about the provided functionalities and their instructions can be found in the user manual provided within snap2stamps, which already plans newer releases increasing functionality and compatibility maintenance of SNAP-StaMPS chain. A reference to the software package (Foumelis et al., 2018) can be found in the manual.

## 6.2 SNAP v7 compatibility fix




```
<useSuppliedShifts>false</useSuppliedShifts>
```

```
<useSuppliedRangeShift>false</useSuppliedRangeShift>
<useSuppliedAzimuthShift>false</useSuppliedAzimuthShift>
```




***coreg_ifg_computation_subset.xml*** – if subset
***coreg_ifg_computation.xml*** - if no subset

## 6.3 Hardware requirements

There are no specific hardware requirements but note that Sentinel-1 data are quite large, and their processing requires significant resources. For example, the interferogram generation step, which is the most computationally demanding will likely require a machine with minimum of 16 GB RAM.

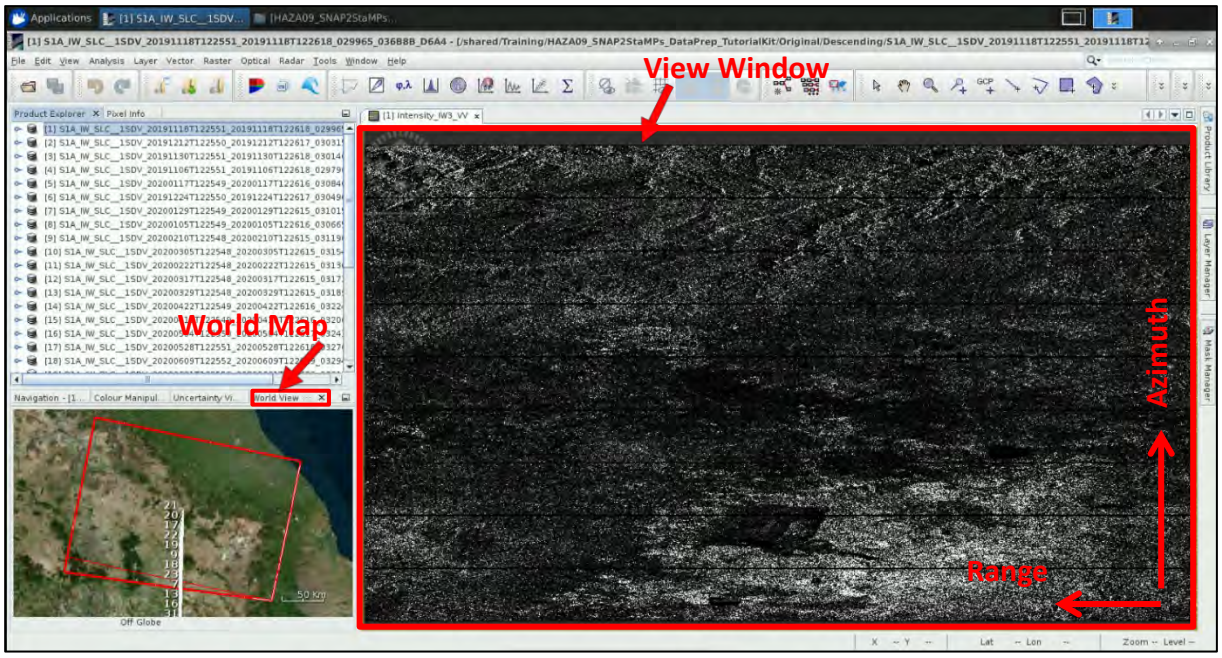
## 6.4 SNAP – open and explore data

Open **SNAP** software from the icon located on the desktop or go to **Applications → Processing → SNAP Desktop**. Click the **Open Product** icon, navigate to:

***/shared/Training/HAZA09_SNAP2StaMPs_DataPrep_TutorialKit/Original/***

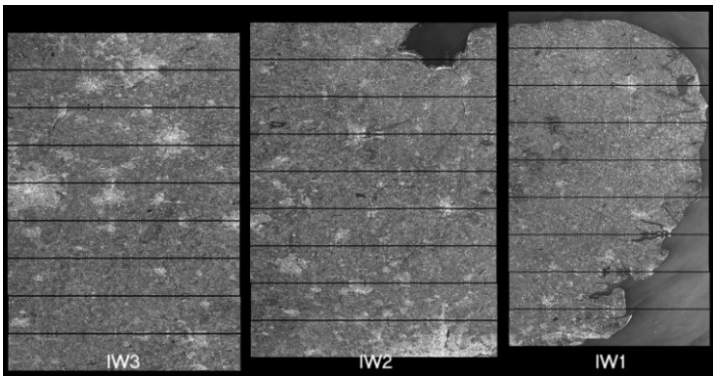
Select all 31 downloaded products and click **Open**.

The opened products will appear in **Product Explorer** window. Click or to expand the contents of first product **[1]** in the list (it may not correspond to the first date), then expand **Bands** folder and double click on ***Intensity_IW3_VV*** band to visualize it in the **View** window. You can go to the **World Map** tab (**View → Tool Windows → World Map**) and zoom-in to see the location of the opened products (See NOTE 1 & 2). You can also see that some products have slightly different footprint.




NOTE 1: The RADAR instrument onboard Sentinel-1 carries an antenna that is looking always to the right during its pass. These two scenes were acquired during descending pass (the satellite was moving in direction from north to south) and in this case while looking to the right it was looking towards the west. That is why we see that the view of the image appears as if "mirrored", because the view shows the pixels in order of the data acquisition.

NOTE 2: The Interferometric Wide (IW) swath mode captures three sub-swaths using Terrain Observation with Progressive Scans SAR (TOPSAR). Each sub-swath image consists of a series of bursts. The input product contains 3 IW bands, and 9 bursts. Mexico City is located on the IW3 sub-swath of the Sentinel-1 images.



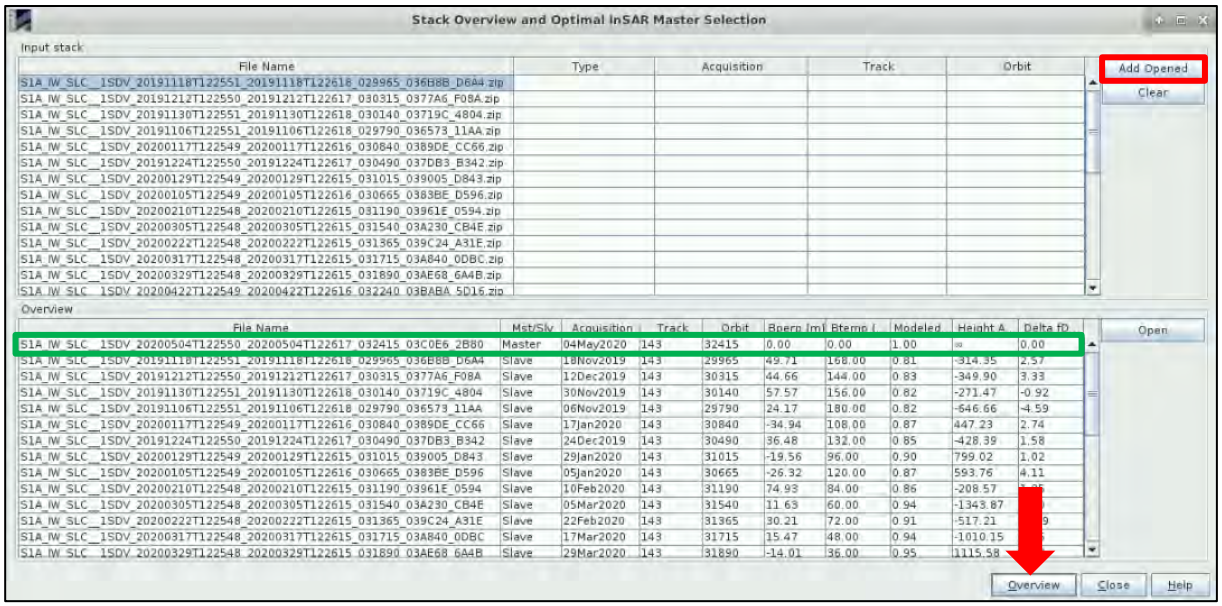

*Credits: ESA User Guides for Sentinel-1 SAR*

## 6.5 Master selection

Before we start with the pre-processing of the data, we need to select optimal master image. The master image is selected such that the distribution of the perpendicular baseline values is as low as possible as well as maximizing the (expected) stack coherence of the interferometric stack. Selection of the "optimal" master should lead to improved visual interpretation of the interferograms and assist quality assessment.

SNAP contains a tool to perform the optimal master selection for us while also providing the overview of the temporal and perpendicular baselines of all the product in respect to it.

Go to **Radar → Interferometric → InSAR Stack Overview** and click **Add Opened** to load all our 31 products (they will not be ordered by date, but it is not important). Then click **Overview**.




The following product has been identified as optimal master image:

***S1A_IW_SLC__1SDV_20200504T122550_20200504T122617_032415_03C0E6_2B80***

Now, close the Stack Overview window and also the SNAP window. In file explorer navigate to the original data folder:

***/shared/Training/HAZA09_SNAP2StaMPs_DataPrep_TutorialKit/Original/***

Move the identified optimal master product to the folder named ***"master"***:

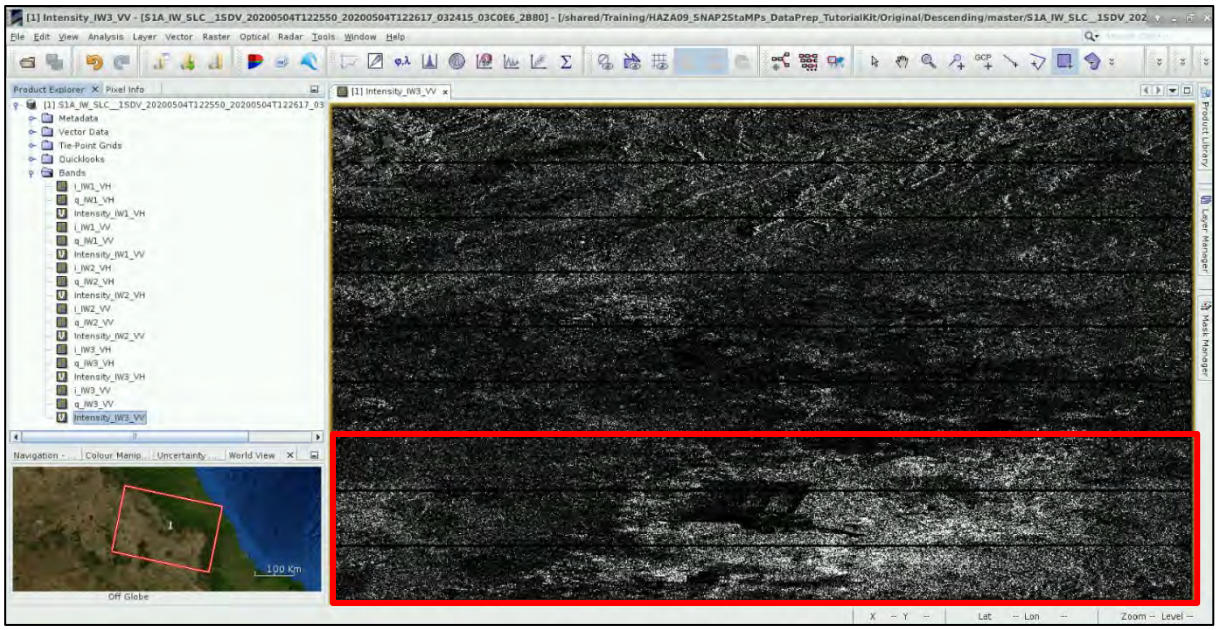
***/shared/Training/HAZA09_SNAP2StaMPs_DataPrep_TutorialKit/Original/master***

Move all other products to the folder named **"slaves"** located in the **Project** folder which will serve as our project folder:

***/shared/Training/HAZA09_SNAP2StaMPs_DataPrep_TutorialKit/Project/slaves***

Now let's relaunch SNAP again and load only the selected master product, go to the **Product Explorer** tab and open the ***Intensity_IW3_VV*** band in view.

The area of the city corresponds to the brighter pixels in the bottom-right part of the image and we can see that most of it is contained in the first three bursts from the bottom. Since the satellite was moving from North to South when acquiring the image these correspond to bursts with numbers 7 to 9 (See NOTE 1 & 2).



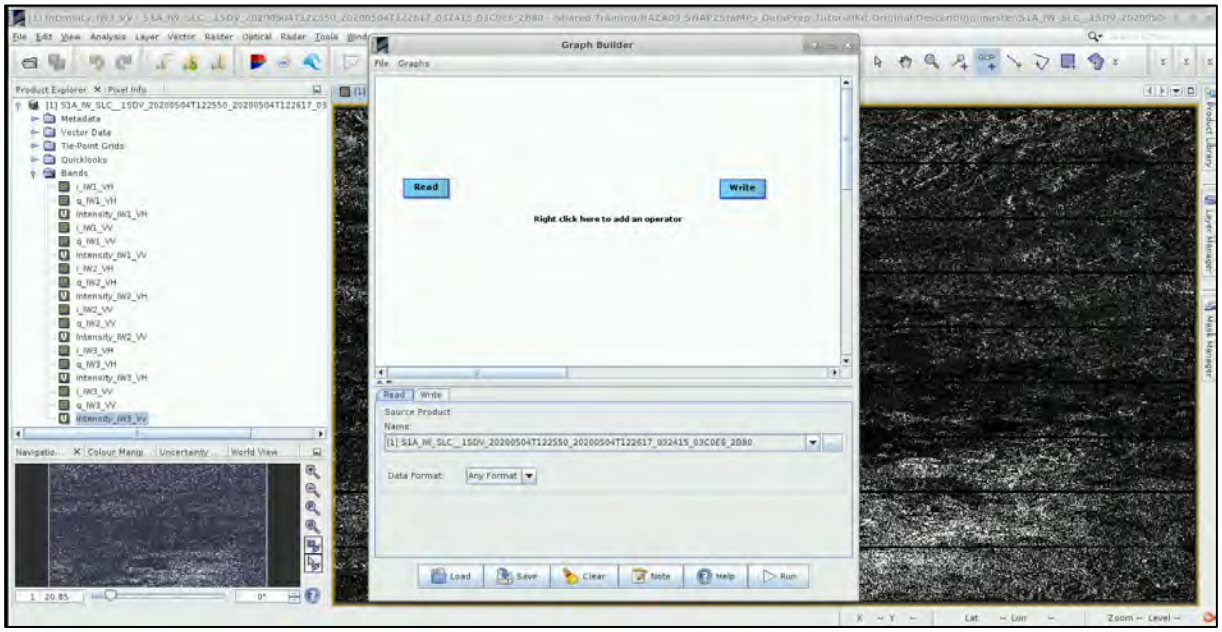
## 6.6 Master splitting and pre-processing

As mentioned in chapter ***6.1 SNAP2StaMPS download and installation***, the snap2stamps folder contains graphs and python scripts to automate the pre-processing of the Senitnel-1 image time series for the StaMPS ingestion.

The **Graph Builder** tool allows the user to assemble graphs from a list of available operators and connect operator nodes to their sources. Therefore, the processing chain we will follow, will be represented by a graph and saved as an XML file.

In this step we need to perform S-1 TOPS Split and Apply orbit file steps on the master product. While there is a SNAP2StaMPS graph corresponding to this step, we will define the graph again as loading predefined graphs to the graphical interface can sometimes cause errors (See NOTE 3).

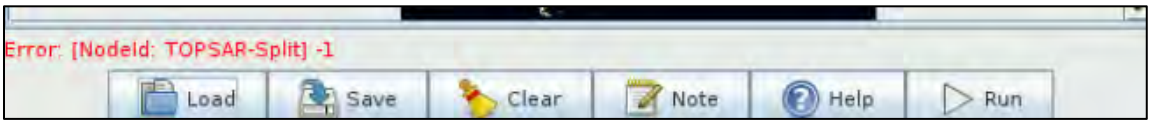
In order to setup the graph go to **Tools → GraphBuilder**.



NOTE 3: You can also attempt to open the predefined graph by clicking on ***Load*** in the **Graph Builder** menu (in the bottom) then navigating to:

***/shared/Training/HAZA09_SNAP2StaMPs_DataPrep_TutorialKit/Processing/snap2stamps/graphs*** and opening: ***master_split_applyorbit.xml***

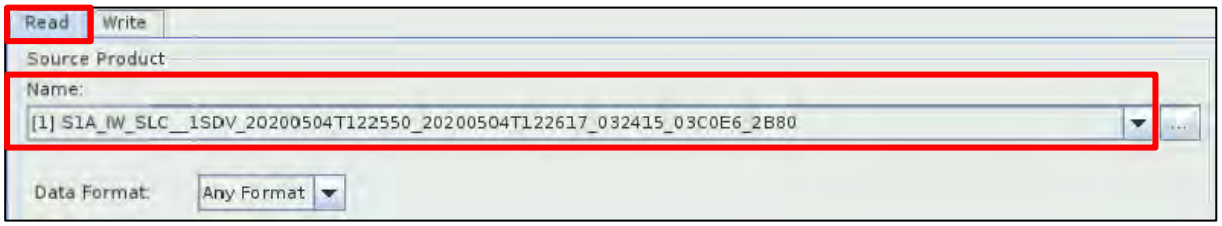
Unfortunately, sometimes opening graphs saved in previous SNAP versions in GUI causes issues. If you click on the TOPSAR-Split tab and see an error at the bottom of the window you will have to recreate the graph or use the python script to run it.




Initially, the graph has two operators: **Read** (to read the input) and **Write** (to write the output). With right-click on the top panel you can add an operator, while a corresponding tab is created and added on the bottom panel.

Then, in the **Read** tab select the name of the loaded master product from 4 May 2020:

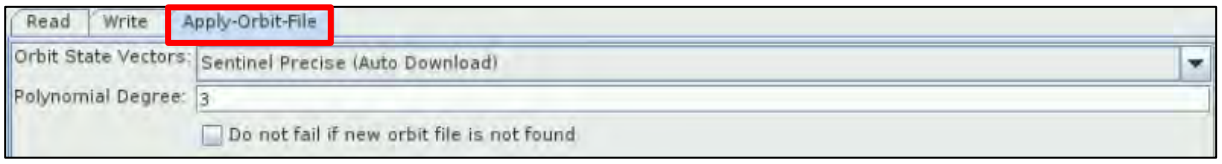
***S1A_IW_SLC__1SDV_20200504T122550_20200504T122617_032415_03C0E6_2B80***




The first processing step is to apply the orbit files in Sentinel-1 products in order to provide accurate satellite position and velocity information. To add the operator right-click on the white area to the right of the existing operator and go to **Add → Radar → Apply-Orbit-File**.

A new operator rectangle appears in our graph and a new tab appears below. Now connect the new **Apply-Orbit-File** operator with the **Read** operator by clicking to the right side of the **Read** operator and dragging the red arrow towards the **Apply-Orbit-File** operator.

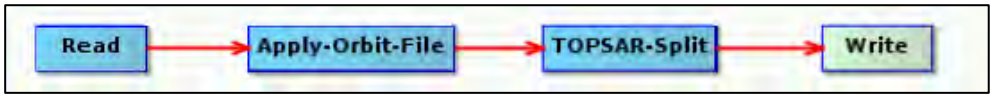

In the **Apply-Orbit File** tab keep the default parameters.




Since the area of interest is included in 3 bursts of the Sentinel-1 image there is no need to process the whole sub-swath with the 9 bursts (See NOTE 3). The extraction of Sentinel-1 TOPS bursts will be made per acquisition and per sub-swath. This process will reduce the processing time in the following processing steps and is necessary as further processing will only accept split products containing single full or partial sub-swath.

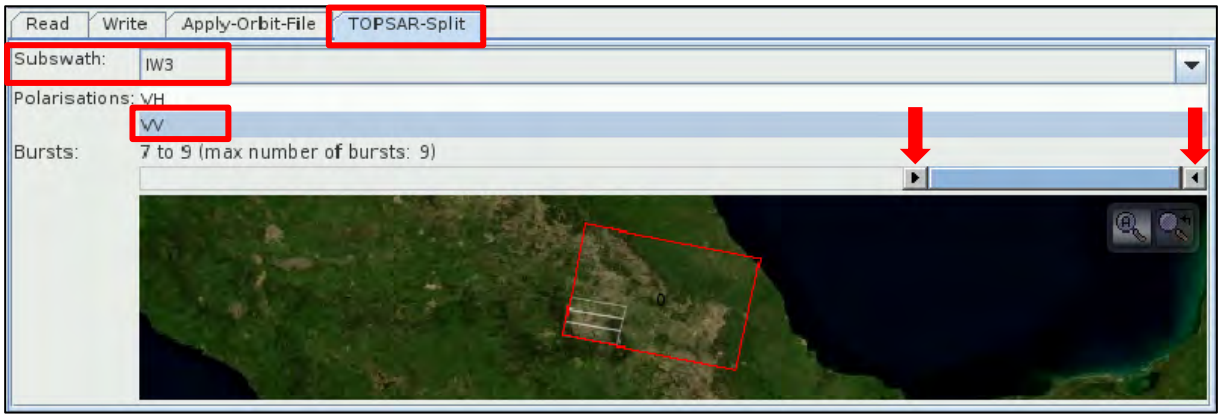
Right click the white space in the graph and go to **Add → Radar → Sentinel-1 TOPS → TOPSAR-Split.**

Connect the **TOPSAR-Split** operator and both **Apply-Orbit-File** and **Write** operators.




In the **TOPSAR Split** tab, set the parameters below:

**Subswath:** *IW3*
**Polarisations:** *VV*
**Bursts:** *7 to 9*

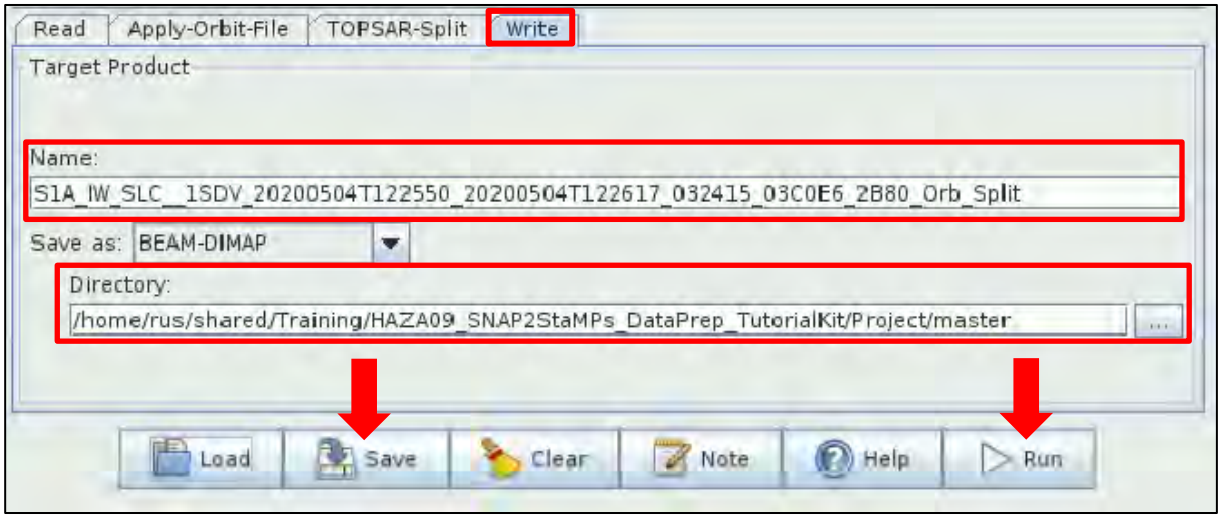



Finally, in the **Write** tab, define the output directory as:

***/shared/Training/HAZA09_SNAP2StaMPs_DataPrep_TutorialKit/Project/master***

Set the name of the output product as:

***S1A_IW_SLC__1SDV_20200504T122550_20200504T122617_032415_03C0E6_2B80_Orb_Split***

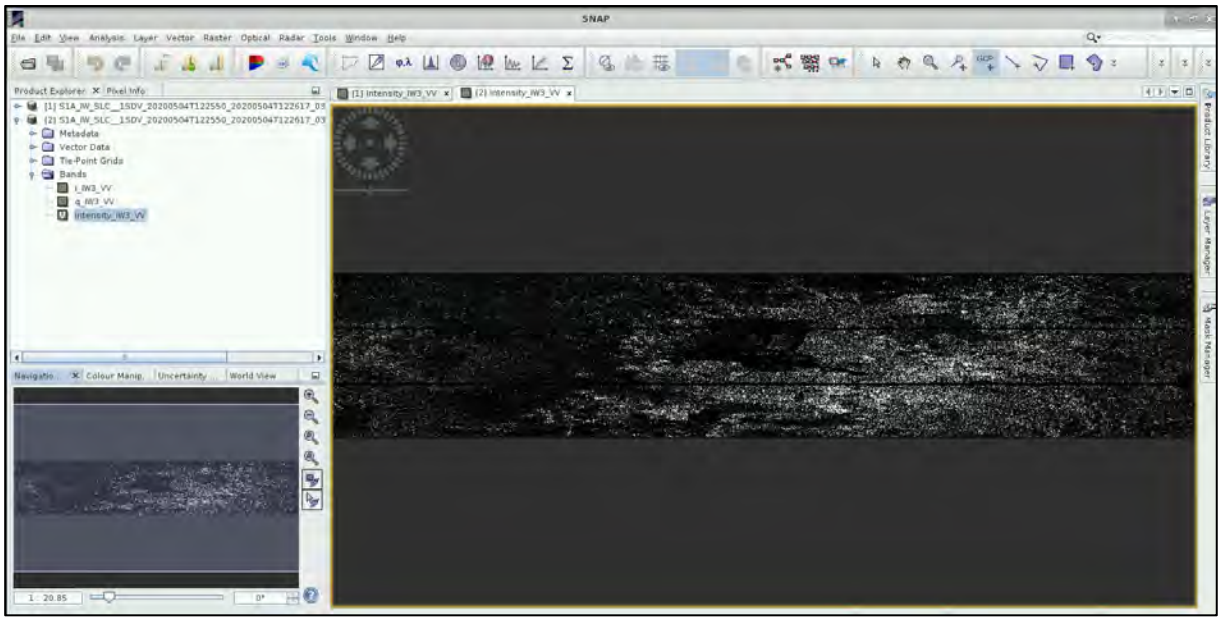



At this moment, click **Save** at the bottom of the window to save the graph as ***master_split_applyorbit_2.xml*** in:

***/shared/Training/HAZA09_SNAP2StaMPs_DataPrep_TutorialKit/Project/graphs***

Click **Run.**

A newproduct will appear in you **Product Explorer** tab. We can open the ***Intensity_IW3_VV*** band in view again to see the burst selection and check we have chosen the correct ones. Then **Close SNAP.**



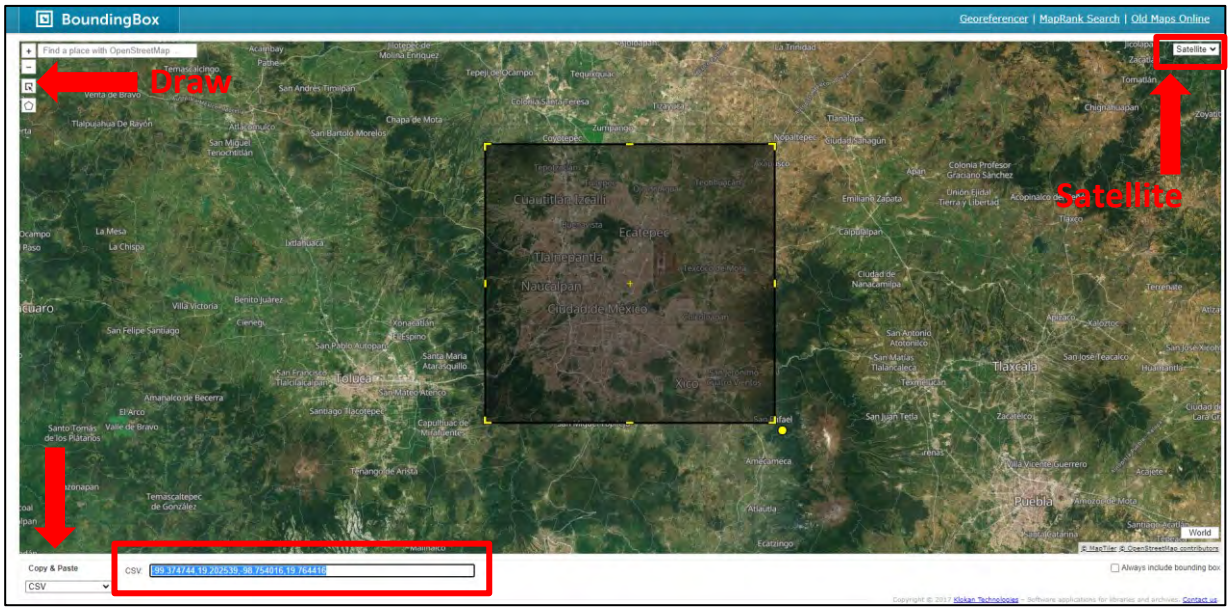

## 6.7 *Area of interest (AOI)*

To run the automated processing, we need to also define minimum and maximum latitude and longitude coordinates of the bounding box covering our area. These will be then used to further decrease the AOI size and in consequence the processing time. We can do this simply by going to an online tool such as **Bounding Box** - https://boundingbox.klokantech.com/

There we can simply navigate to Mexico City (easier if you change base map to satellite) click and draw a AOI as shown below.

Then at the bottom page, under the map window change the option **Copy & Paste** to ***CSV.*** You can then copy the bounding box coordinates in a simple format *< Min. longitude, Min. Latitude, Max. longitude, Max. Latitude >*

For polygon defined below these are*:* ***-99.374744, 19.202539, -98.754016, 19.764416***



You can copy-paste them into an empty Mousepad text file or leave the page open to use the coordinates in the next step.

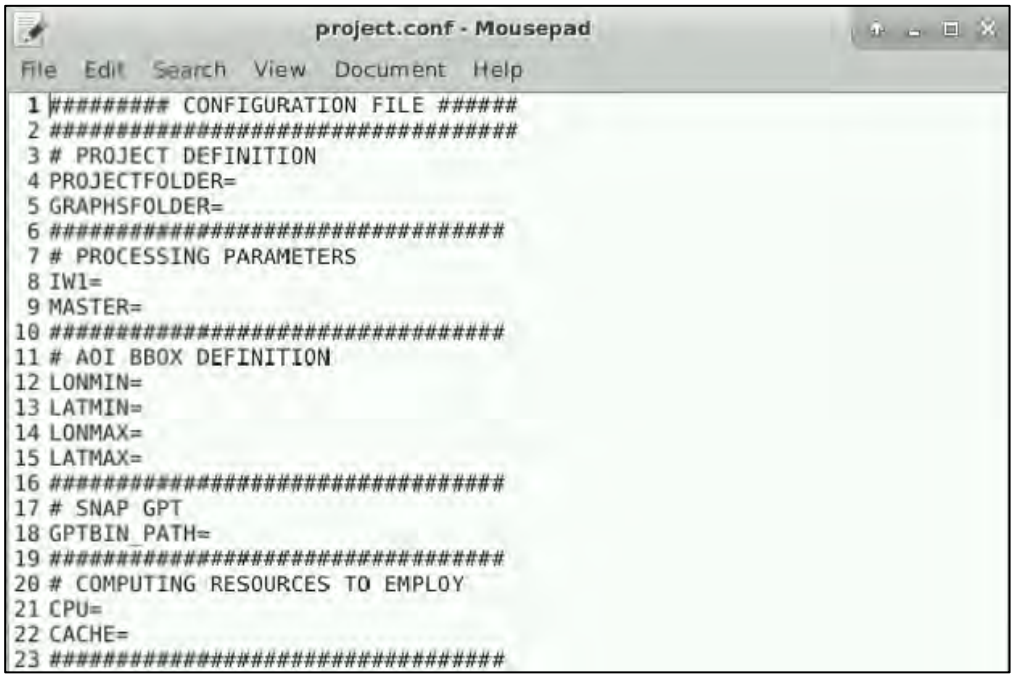
### 6.8 SNAP2StaMPS project configuration

To run the next steps in a more automated mode the snap2stamps includes ***project.conf*** file where all necessary user inputs are defined. Let's now set it up.

You can find the file in snap2stamps **"bin"** folder (we have copied it to the Processing folder) and have to edit it according to your data and paths:

***/shared/Training/HAZA09_SNAP2StaMPs_DataPrep_TutorialKit/Project/bin***

Right-click ***project.conf*** and select ***Open in Mousepad.*** At this moment all the settings are empty.




In the **PROJECT DEFINITION** section, we set default project folder to the ***Processing*** folder and set the graphs folder to the snap2stamps graph location.

PROJECTFOLDER=***/shared/Training/HAZA09_SNAP2StaMPs_DataPrep_TutorialKit/Project***
GRAPHSFOLDER=***/shared/Training/HAZA09_SNAP2StaMPs_DataPrep_TutorialKit/Project/graphs***

```
1 ######### CONFIGURATION FILE ######
2 ###################################
3 # PROJECT DEFINITION
4 PROJECTFOLDER=/shared/Training/HAZA09_SNAP2StaMPs_DataPrep_TutorialKit/Project
5 GRAPHSFOLDER=/shared/Training/HAZA09_SNAP2StaMPs_DataPrep_TutorialKit/Project/graphs
6 ###################################
```

TIP: Your paths must not contain any spaces if they do, enclose them into quote marks "path".

In the **PROCESSING PARAMETERS** section, we set the same sub-swath as used when splitting the master product and point to the full path where we saved it.

IW1=***IW3***
MASTER=***/shared/Training/HAZA09_SNAP2StaMPs_DataPrep_TutorialKit/Project/master***

```
 6 ###################################
 7 # PROCESSING PARAMETERS
 8 IW1=IW3
 9 MASTER=/shared/Training/HAZA09_SNAP2StaMPs_DataPrep_TutorialKit/Project/master/
   S1A_IW_SLC__1SDV_20200504T122550_20200504T122617_032415_03C0E6_2B80_Orb_Split.dim
10 ###################################
```

In the AOI BBOX DEFINITION we will use the coordinates retrieved in the online Bounding Box tool (6.7 Area of interest (AOI)).

***LONMIN=-99.374744*** ***LONMAX=-98.754016***

***LATMIN=19.202539*** ***LATMAX=19.764416***

```
10 ###################################
11 # AOI BBOX DEFINITION
12 LONMIN=-99.374744
13 LATMIN=19.202539
14 LONMAX=-98.754016
15 LATMAX=19.764416
16 ###################################
```

TIP: Be careful to copy minus signs correctly and remove all spaces.

In the SNAP GPT path, we need to point to the SNAP installation. On the RUS VM the SNAP gpt executable is located at:

GPTBIN_PATH=***/usr/local/snap/bin/gpt***

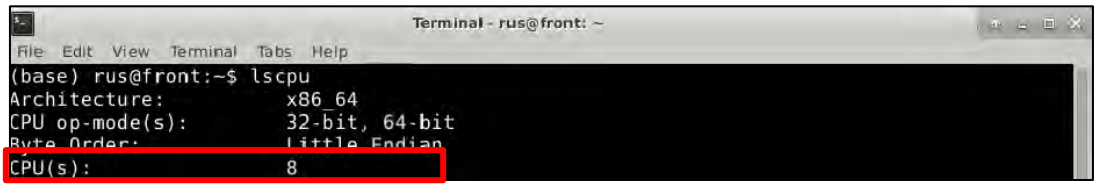
```
16 ###################################
17 # SNAP GPT
18 GPTBIN_PATH=/usr/local/snap/bin/gpt
19 ###################################
```

Finally, in the COMPUTING RESOURCES TO EMPLOY section, we can set the CPU number and Cache size to use (See NOTE 4). This needs to be selected based on your VM/PC. For standard RUS VM the settings could be as follows:

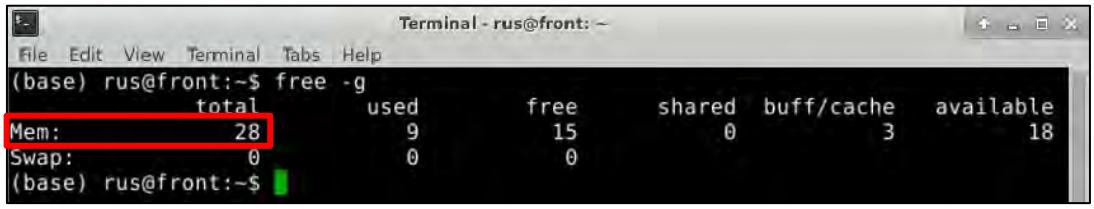
CPU=4
CACHE=12G

```
19 ###################################
20 # COMPUTING RESOURCES TO EMPLOY
21 CPU=4
22 CACHE=12G
23 ###################################
```

NOTE 4: you can check the system resources by going to command line terminal and typing:
**"lscpu"** to check the CPU number:




And **"free -g"** to check the total available RAM in GB (1 MB is equal to 1024 KB):




The cache size can be set to up to 70-80% of available memory. In this example (not standard RUS VM), it would be approx. 20 GB.

Now, go to **File → Save** and save the changes made to the ***project.conf*** file. We can now start with the automated processing using the python scripts.

## *6.9 Slaves preparation*

In the next step, we need to divide the slave images into folders with the name corresponding to the acquisition date in format ***<yyyymmdd>***. This is a necessary first step enabling the automated processing. Navigate to:

***/shared/Training/HAZA09_SNAP2StaMPs_DataPrep_TutorialKit/Project/bin***

Then right-click on the white space in the folder and select Open Terminal Here. A terminal window will open with the path to the bin folder. To call the first python script paste the following command in the terminal.

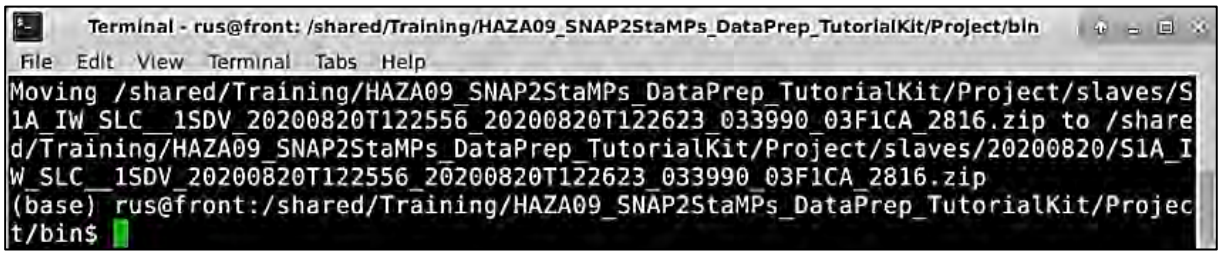
```
python2 slaves_prep.py project.conf
```

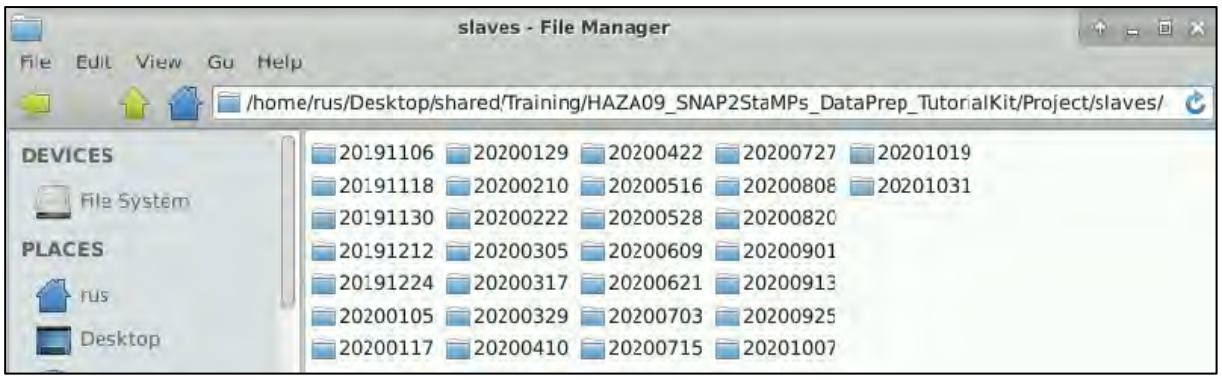

TIP: **The snap2stamps requires Python 2.7** – in RUS VM the default Python is 3.8 but 2.7 is installed as well – we can therefore use the command "python2" to call it. The default Python and the call for python 2.7 might be different in you machine. You can type "python -V" and "python2 -V" commands to your command line to find the associated version. **See chapter 6.1 for more information.**


Then press **ENTER** to run the command. *The processing will take few seconds depending on your VM.*

When the processing is completed (the bin path ending with $ will appear again). Leave the terminal window open and check the ***"slaves"*** folder (it should contain a folder for each slave image):

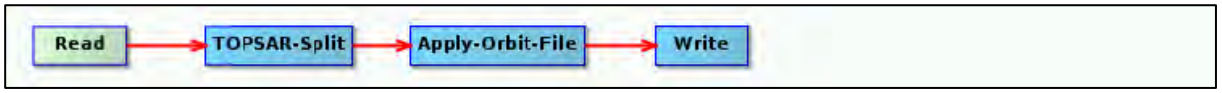
***/shared/Training/HAZA09_SNAP2StaMPs_DataPrep_TutorialKit/Project/slaves***




## 6.10 Slaves splitting

Next, we need to split and update the orbit information for each of the slave images. Again, this step is run in the terminal window. It uses the following SNAP graph (predefined in the "graph" folder).

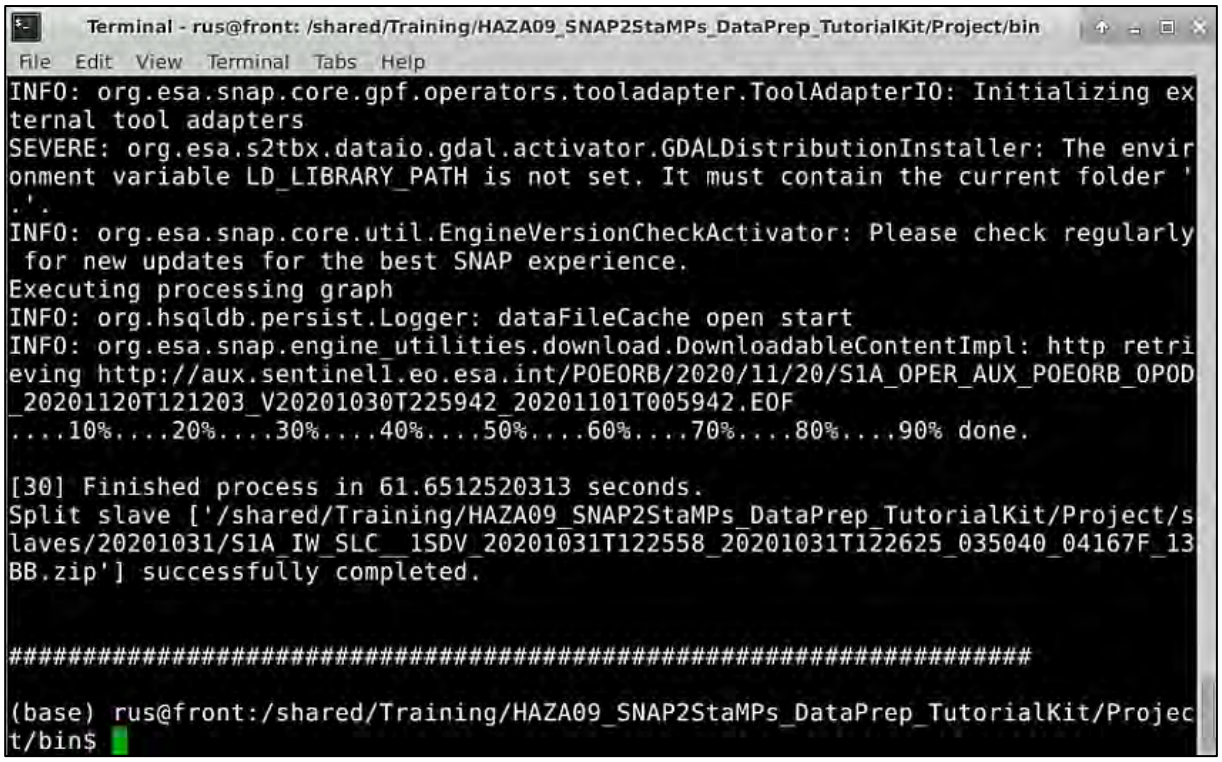



It includes the same operators as the graph we used to pre-process the master image. However, this time we do not need to open the graph. The python script will automate processing by looping over the slave images and updating the input and output accordingly and then running the graph for each slave image.

Go back to the **terminal** window and run the following command.

```
python2 splitting_slaves.py project.conf
```

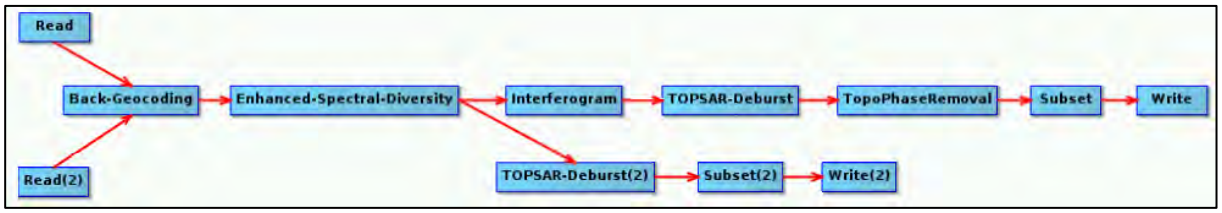

Then press **ENTER** to run the command. *The processing will take approximately a 1 minutes per slave image depending on your VM (30 minutes for 30 images on VM with 30GB RAM).*




When the processing is completed, a new folder ***"split"*** was created in the ***Project*** folder. It contains folder for each pre-processed slave product. Leave the terminal window open.

## *6.11 Master-slave coregistration and interferogram generation*

In the next step we need to co-register each slave image with the master image and create an interferogram and other steps. Below you will find the description of each of the steps, you however do not need to open the graph or change any settings.




### 6.11.1 Back-Geocoding

This operator co-registers two S-1 SLC split products (master and slave) of the same sub-swath using the orbits of the two products and a Digital Elevation Model (DEM). *(SNAP Help)*

### 6.11.2 Enhanced-Spectral-Diversity (ESD)

This operator follows the S-1 Back Geocoding operator in the TOPS InSAR processing chain. It first estimates a constant range offset for the whole sub-swath of the split S-1 SLC image using incoherent cross-correlation. The estimation is done for each burst using a small block of data in the center of the burst. The estimates from all bursts are then averaged to get the final constant range offset for the whole sub-swath. *(SNAP Help)*

### 6.11.3 Interferogram

This operator computes (complex) interferogram, with subtraction of the flat-earth (reference) phase (can also be run without). The flat-earth phase is the phase present in the interferometric signal due to the curvature of the reference surface (WGS84). *(SNAP Help)*

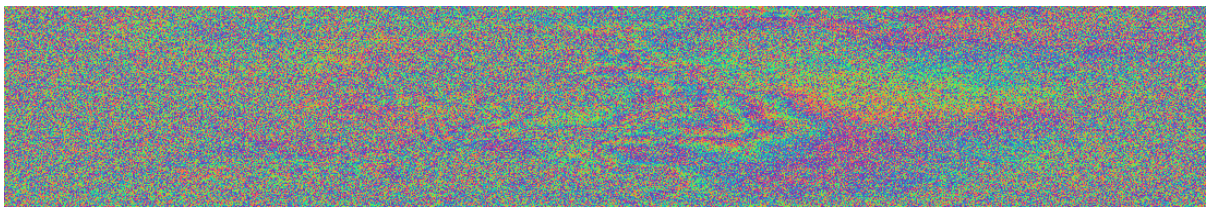
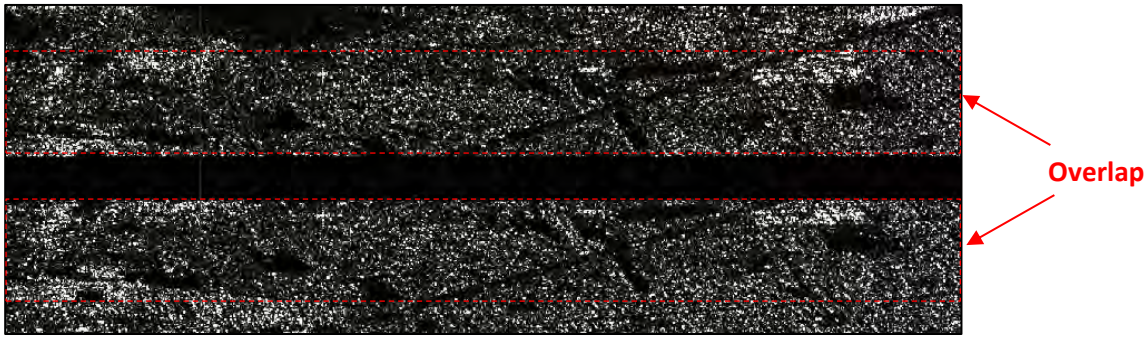


*Image: Mexico City interferogram (not multilooked)*

### 6.11.4 TOPSAR-Deburst

We have seen that each sub-swath image consists of a series of bursts, where each burst has been processed as a separate SLC image. The individually focused complex burst images are included, in azimuth-time order, in a single sub-swath image with black-fill demarcation in between. There is sufficient overlap between adjacent bursts and between sub-swaths to ensure the continuous coverage of the ground. Images for all bursts in all sub-swaths of an IW SLC product are re-sampled to a common pixel spacing grid in range and azimuth. This processor merges the bursts to a continuous image based on their zero Doppler time and removes the demarcation pixels. *(SNAP Help)*




### 6.11.5 Topographic Phase Removal

This operator estimates and subtracts topographic phase from the interferogram. More specifically, this operator first "radarcodes" the Digital Elevation Model (DEM) of the area of interferogram, and then subtracts it from the complex interferogram. *(SNAP Help)*

This operator must be performed after the interferogram generation. It also requires an input DEM, SRTM can be used, or any other supported DEM. The DEM handling for most of elevation models, selection and download from internet of tiles covering the area of interest, interpolation, accounting for geoid undulation, etc, is performed automatically by the operator itself. *(SNAP Help)*

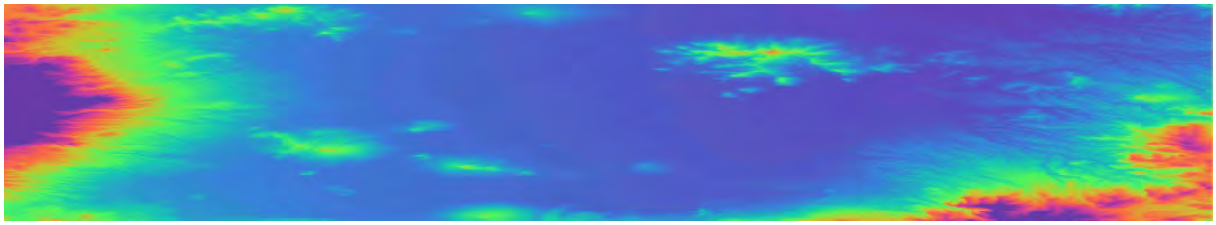


*Image: Mexico City Topographic phase based on SRTM 1Sec HGT DEM (not multilooked)*

### 6.11.6 Subset

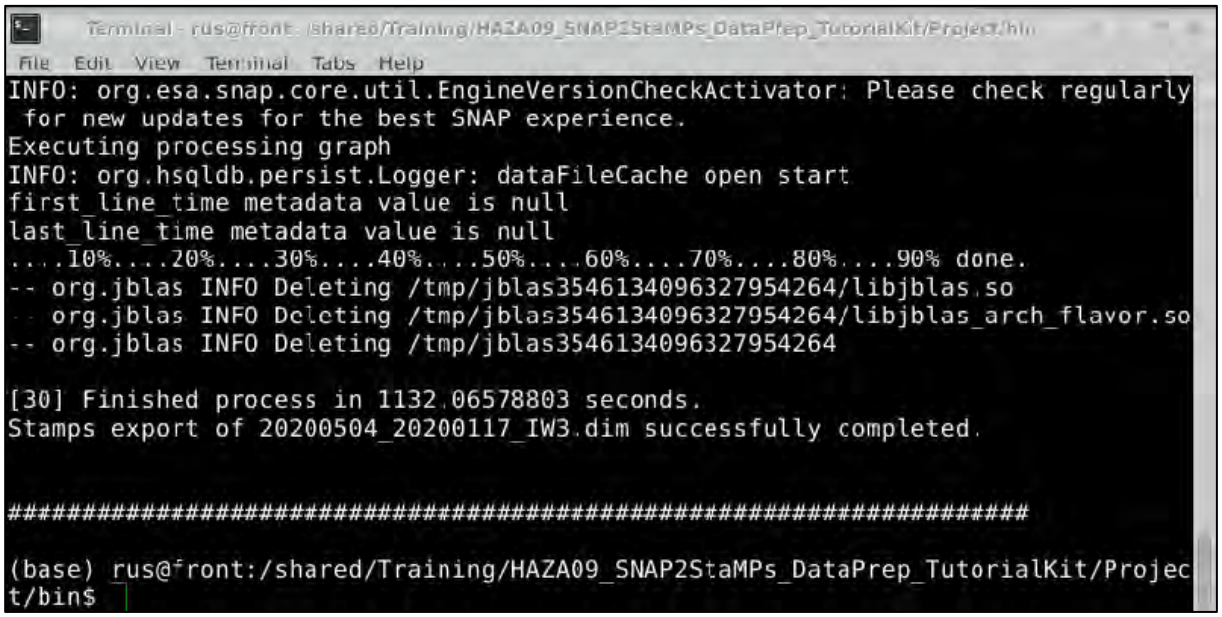
In this step a spatial subset is applied to reduce the product extent to the study area we have set in the ***"project.conf"*** and defined in ***6.7 Area of interest (AOI)***.

### 6.11.7 Run

Go back to the **terminal** window and run the following command. Note that this is the most time demanding step in the pre-processing chain.

```
python2 coreg_ifg_topsar.py project.conf
```


Then press **ENTER** to run the command. *The processing will take approximately a 15 minutes per slave image depending on your VM (7 hours and 30 minutes for 30 images on VM with 30GB RAM).*




When the processing is completed, two new folders ***"coreg"*** and ***"ifg"*** were created in the ***Project*** folder. The ***"coreg"*** folder contains coregistered and debursted product and the ***"ifg"*** folder contains the interferogram with topographic phase removed. The products in both folders are named identically as ***<masterDate_slaveDate_IW3>***. Leave the terminal window open.

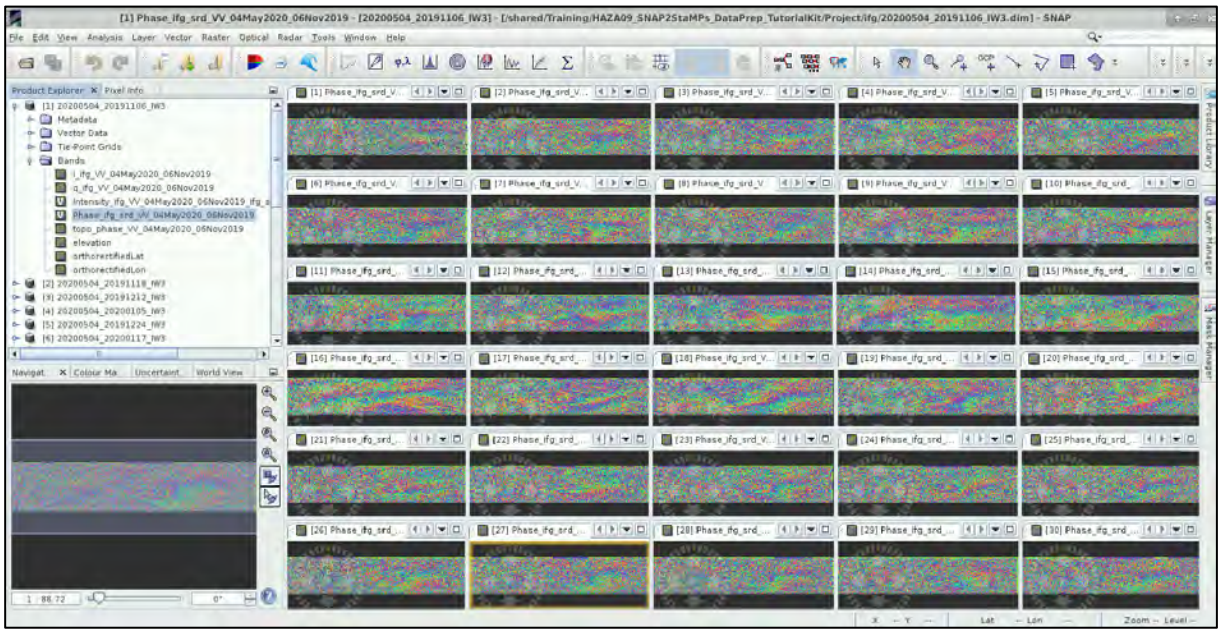
### 6.11.8 Check the interferograms

Now, before the export we need to test if all the interferograms have been correctly completed. Open SNAP graphical interface and load all the products in:

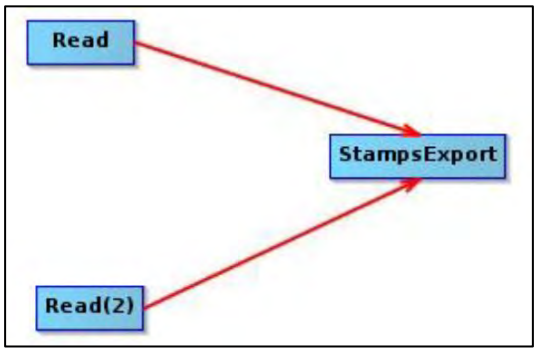
***/shared/Training/HAZA09_SNAP2StaMPs_DataPrep_TutorialKit/Project/ifg/***

Then open the ***"Phase_ifg_*****"*** band for each product. You do not need to open all at once as below, you merely need to check for empty interferograms.

In the image bellow you can see that all interferograms have been correctly created. If in your dataset you find an empty interferogram remove note the name and consequently remove the file from the ***"coreg"*** and ***"ifg"*** folders.



## *6.12 StaMPS export*




Now we need to prepare the data into a StaMPS compatible format. This is done by using a SNAP tool for StaMPS export.

The inputs are:

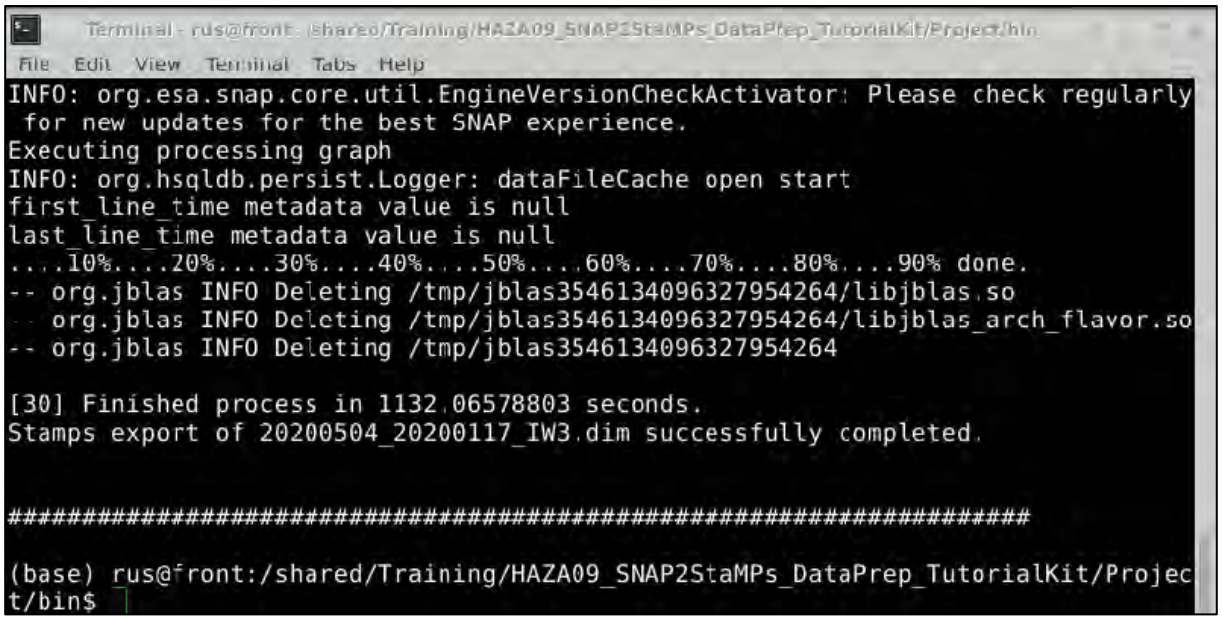
- the coregistered master-slave pair
- its corresponding interferogram with the elevation and orthorectified latitude and longitude bands

Go back to the **terminal** window and run the following command.

```
python2 stamps_export.py project.conf
```

```
(base) rus@front:/shared/Training/HAZA09_SNAP2StaMPs_DataPrep_TutorialKit/Projec
t/bin$ python2 stamps_export.py project.conf
```

Then press **ENTER** to run the command. *The processing will take again quite long time - approximately a 12 minutes per slave image depending on your VM (6 hours for 30 images on VM with 30GB RAM).*




When the processing is completed, a new folder named ***"INSAR_20200504"*** was created in the ***Project*** folder. It contains the final output structure - four folders: ***rslc***, ***diff0***, ***geo*** and ***dem***.

### *6.13 Amplitude dispersion (only possible with StaMPS installed)*

The final step before we can ingest the data into StaMPS for PSI processing is the estimation of the amplitude dispersion index. The $D_A$ is a value that describes the amplitude stability, which is used to preselect pixels and therefore reduces the number of pixels for the phase analysis. It is calculated as:

$$D_A = \frac{\sigma_A}{\mu_A}$$

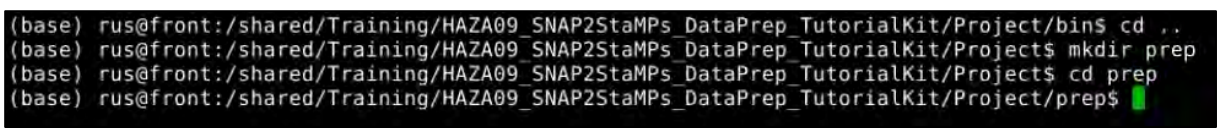
Where $\sigma_A$ is the standard deviation and $\mu_A$ is the mean of a series of amplitude values. The recommended range for $D_A$ is 0.40 - 0.42. The higher the threshold, more pixels will be selected for phase analysis. Note that surfaces like water and vegetation, where amplitude is instable, exhibit higher D_A values than bedrock outcroppings or man-made strucutres (i.e. most likeley PS pixels). See Ferretti et al. (2001) and Hooper et al. (2007) for further information. *(GIS-Blog - Matthias Schlögl)*

The script is included in the StaMPS installation but does not require MATLAB licence, it was provided by Andy Hooper (University of Leeds) and customised for SNAP interferograms generated.

Go back to the **terminal** window and run the following three commands.

```
cd ..
mkdir prep
cd prep
```

Now your terminal line should start with the path to the ***prep*** folder.




Then call the script without parameters: `mt_prep_snap`

```
(base) rus@front:/shared/Training/HAZA09_SNAP2StaMPs_DataPrep_TutorialKit/Project/prep$ mt_prep_snap
mt_prep_snap Andy Hooper, August 2017

usage: mt_prep_snap yyyymmdd datadir da_thresh [rg_patches az_patches rg_overlap az_overlap maskfile]
    yyyymmdd                = master date
    datadir                 = data directory (with expected structure)
    da_thresh               = (delta) amplitude dispersion threshold
                              typical values: 0.4 for PS, 0.6 for SB
    rg_patches (default 1)  = number of patches in range
    az_patches (default 1)  = number of patches in azimuth
    rg_overlap (default 50)  = overlapping pixels between patches in range
    az_overlap (default 50) = overlapping pixels between patches in azimuth
    maskfile (optional) char file, same dimensions as slcs, 0 to include, 1 otherwise

(base) rus@front:/shared/Training/HAZA09_SNAP2StaMPs_DataPrep_TutorialKit/Project/prep$
```

We choose the following parameters:

**da_thresh** = 0.4
**rg_patches** = 9
**az_patches** = 3
**rg_overlap** = 50
**rg_overlap** = 200

```
mt_prep_snap 20200504
/shared/Training/HAZA09_SNAP2StaMPs_DataPrep_TutorialKit/Project 0.4 9 3 50 200
```

```
(base) rus@front:/shared/Training/HAZA09_SNAP2StaMPs_DataPrep_TutorialKit/Project/prep$ mt_p
rep_snap 20200504 /shared/Training/HAZA09_SNAP2StaMPs_DataPrep_TutorialKit/Project/INSAR_202
00504 0.4 9 3 50 200
```

Then press **ENTER** to run the command. *The processing will take again quite long time - approximately a 12 minutes per slave image depending on your VM (6 hours for 30 images on VM with 30GB RAM).*

Once the calculations are finished, move the contents of the ***"prep"*** folder to the "***INSAR_20200504"*** folder. Now your data are ready to proceed with StaMPS processing in MATLAB.

TIP: Note that errors have been reported when running StaMPS on Windows – Linux is the preferred environment.

MATLAB should be run from command line (provided that StaMPS_CONFIG.bash is included in .bashrc)

Follow steps outlined in:
1) by Matthias Schlögl in his posts on GIS-Blog
2) in the stamps and snap2stamps manuals
3) StaMPS Persistent Scatterer Exercise by A. Hopper (2015, v3.1)

THANK YOU FOR FOLLOWING THE EXERCISE!

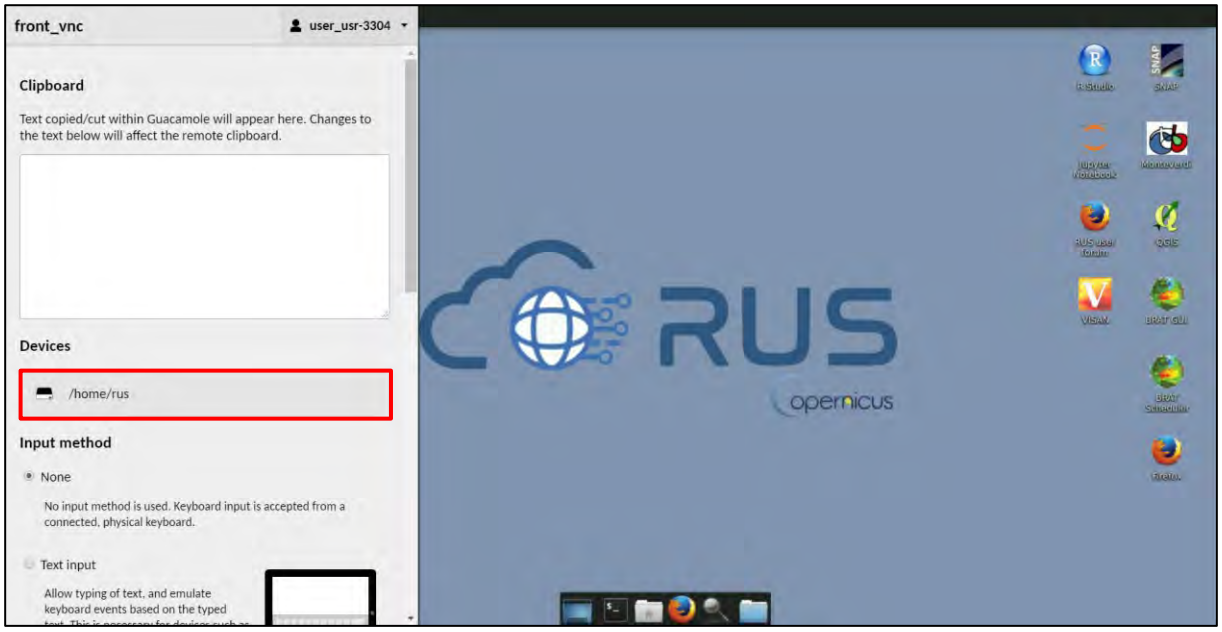
# 7 Extra steps

## *7.1 Downloading the outputs from the VM*

In your VM, press **Ctrl+Alt+Shift.**

A pop-up window will appear on the left side of the screen. Click on the bar below **Devices**, navigate to the folders you have saved the files you want to download and **double click** on them. The downloading process to your local computer will start automatically.

Once the KML files have been downloaded, you can load and visualize them in **Google Earth**.




# 8 References

## 8.1 Software

**SNAP2StaMPS Manual and download** - https://github.com/mdelgadoblasco/snap2stamps
**StaMPS Manual**
**StaMPS Download**

## 8.2 PSI resources

**Crosetto, M., Monserrat, O., Cuevas-González, M., Devanthéry, N., & Crippa, B. (2016). Persistent Scatterer Interferometry: A review**. *ISPRS Journal of Photogrammetry and Remote Sensing*, 115, 78–89. https://doi.org/10.1016/j.isprsjprs.2015.10.011

**Delgado Blasco, J. M., Foumelis, M., Stewart, C., & Hooper, A. (2019). Measuring Urban Subsidence in the Rome Metropolitan Area (Italy) with Sentinel-1 SNAP-StaMPS Persistent Scatterer Interferometry**. *Remote Sensing*, 11(2), 129. https://doi.org/10.3390/rs11020129

**Jia, H., & Liu, L. (2016). A technical review on persistent scatterer interferometry**. *Journal of Modern Transportation*, 24(2), 153–158. https://doi.org/10.1007/s40534-016-0108-4

## 8.3 Tutorials

**GIS-Blog** - Matthias Schlögl – Using StaMPS/MTI for PSI Analysis (post series) - https://www.gis-blog.com/stamps-1/
**A. Hooper** – StaMPS Persistent Scatterer Exercise (2015) – ESA Land Training Course
http://seom.esa.int/landtraining2015/files/Day_4/D4P2a_LTC2015_Hooper.pdf

## FOLLOW US!!!

 @RUS-Copernicus

 RUS-Copernicus

 RUS Copernicus Training

 RUS Copernicus

 RUS-Copernicus website

 RUS-Copernicus Training website