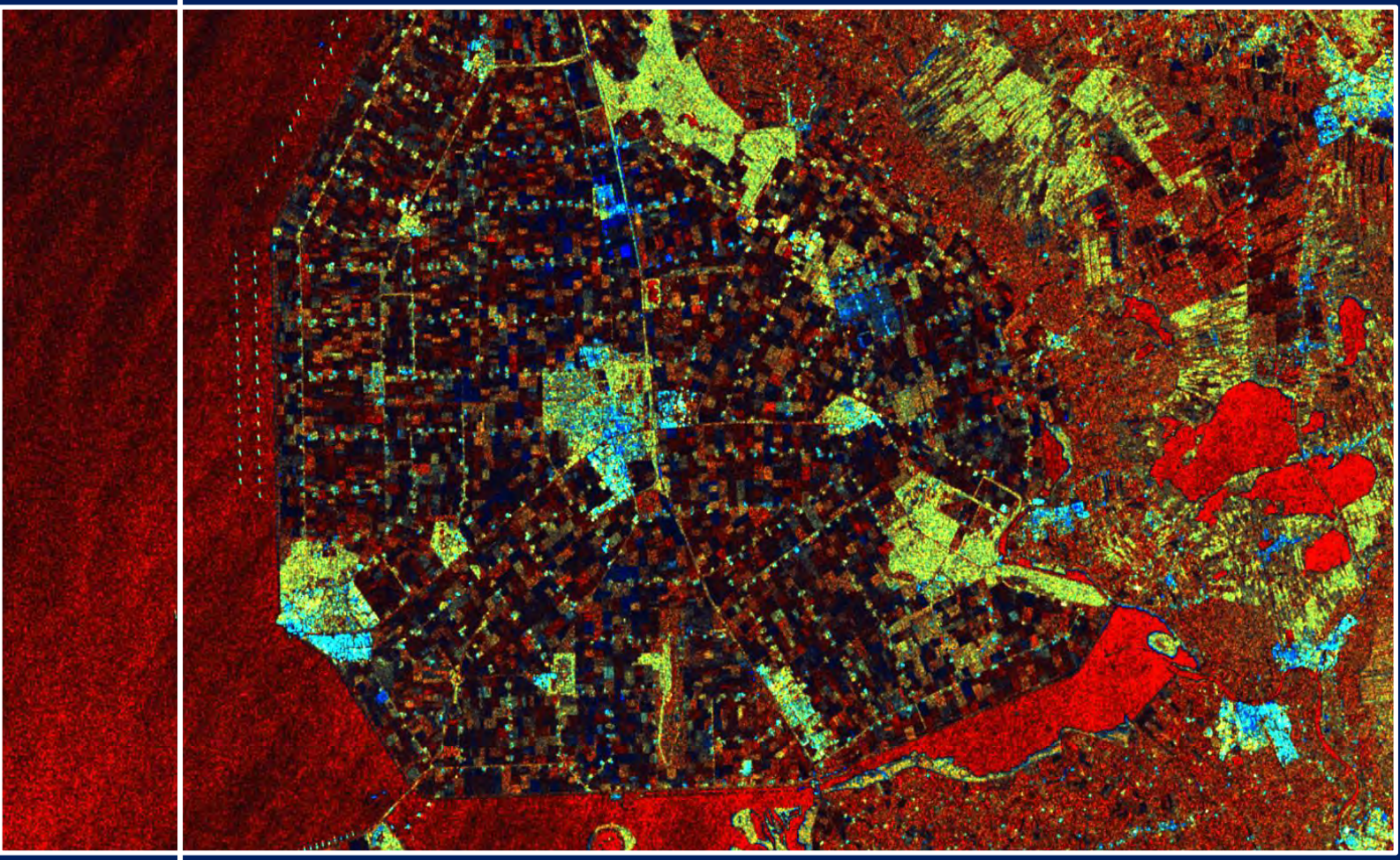
European Commission

esa

RUS

Copernicus



TRAINING KIT – PY01

SENTINEL-1 PROCESSING USING SNAPPY

serco

NOVELTIS

ALONG-TRACK

ROMANIA

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-copernicus.eu

Cover image credits: ESA

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

Date of publication: June 2020

Version: 1.1

Suggested citation:

Serco Italia SPA (2020). *Sentinel-1 processing using snappy (version 1.1).* Retrieved from RUS Lectures at https://rus-copernicus.eu/portal/the-rus-library/learn-by-yourself/

Table of Contents

1 Introduction to RUS

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 learn how to process Copernicus data (Senitnel-1) using the Python *snappy* module.

2 Sentinel-1 – background

A few years ago, the European Union (EU) started an ambitious program, Copernicus, which includes the launch of a new family of earth observation satellites known as Sentinels. The Sentinel-1 mission is the European Radar Observatory for the Copernicus joint initiative of the European Commission (EC) and the European Space Agency (ESA). The Sentinel-1 mission comprises a constellation of two polar-orbiting satellites, operating day and night and performing C-band Synthetic Aperture RADAR (SAR) imaging in four exclusive modes with different resolution (down to 5 m) and coverage (up to 400 km). It provides dual polarisation capability, very short revisit times and rapid product delivery. For each observation, precise measurements of spacecraft position and attitude are available.

In this exercise you will learn how to use *snappy* to apply a standard pre-processing chain and produce Sentinel-1 analysis-ready data. The analysis will be implemented using Python code that can be found in this Jupyter Notebook. Although highly recommended, the exercise DOES NOT require any Python programming skills and can be followed by any participant. You will not be asked to write any code but to follow the methodology and understand the processing steps.

3 Training

Approximate duration of this training session is **one** hour.

The Training Code for this tutorial is PY01. If you wish to practice the exercise described below within the RUS Virtual Environment, register on the RUS portal and open a User Service request from Your RUS service > Your dashboard.

3.1 Data used

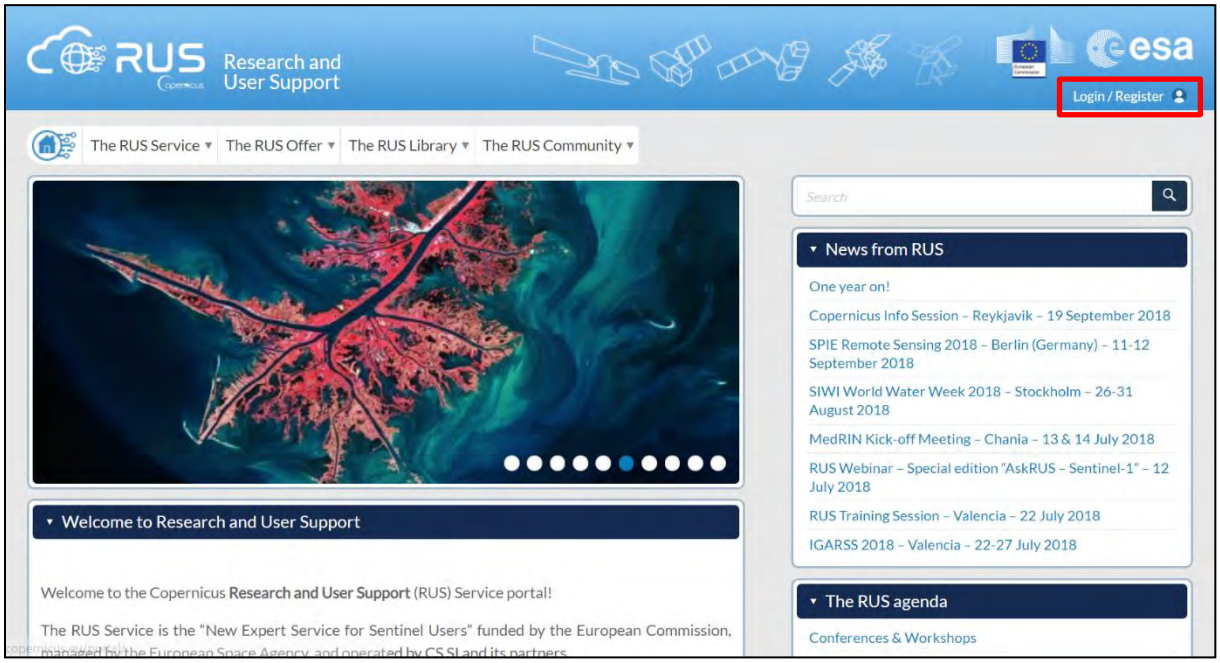
- 1 Sentinel-1 image acquired during April 2020
- Pre-processed data stored locally
  @/shared/Training/PY01_Sentinel1Processing_snappy/AuxData/

3.2 Software in RUS environment

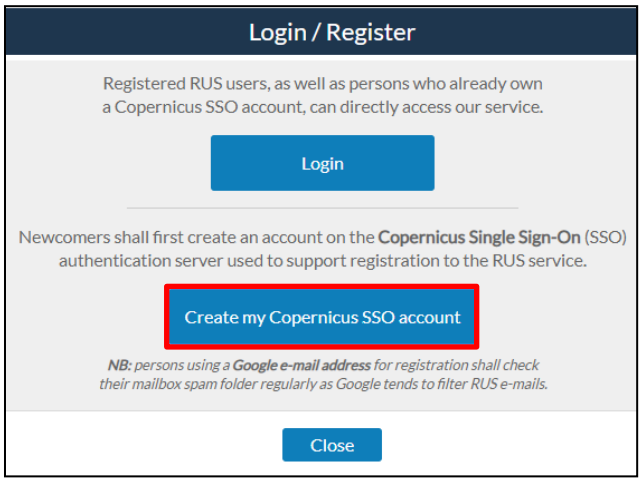
Internet browser, JupyterLab, Python, Anaconda, *snappy*

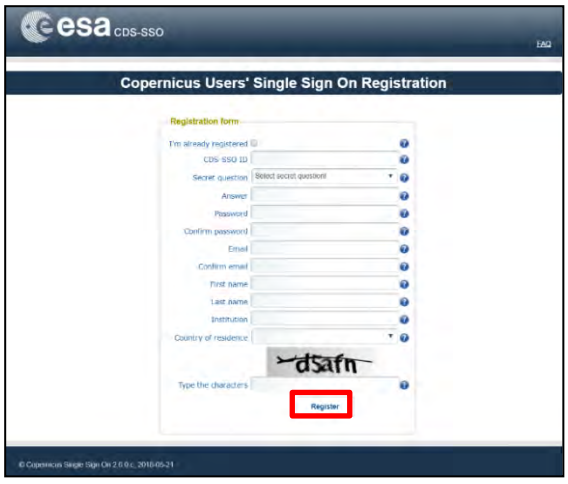
4 Register to RUS Copernicus

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.



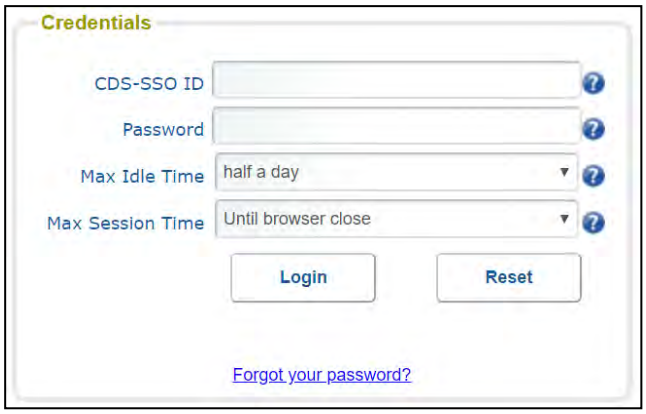
Select the option ***Create my Copernicus SSO account*** and then fill in ALL the fields on the **Copernicus Users' Single Sign On Registration**. Click ***Register***.



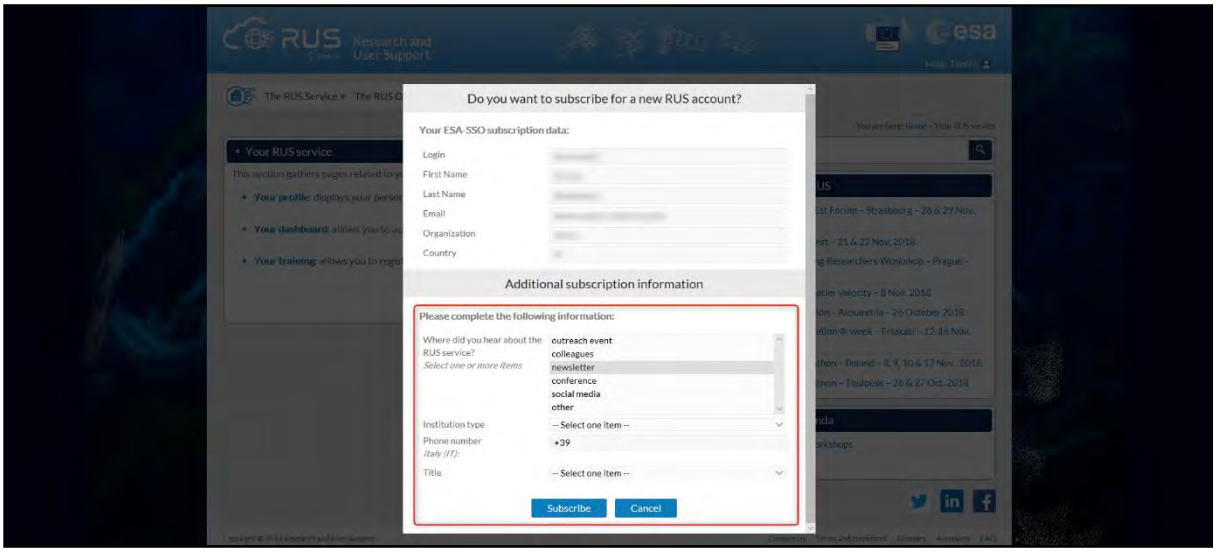


Within a few minutes you will receive an e-mail with activation link. Follow the instructions in the e-mail to activate your account.

You can now return to https://rus-copernicus.eu/, click on ***Login/Register***, choose ***Login*** and enter your chosen credentials.

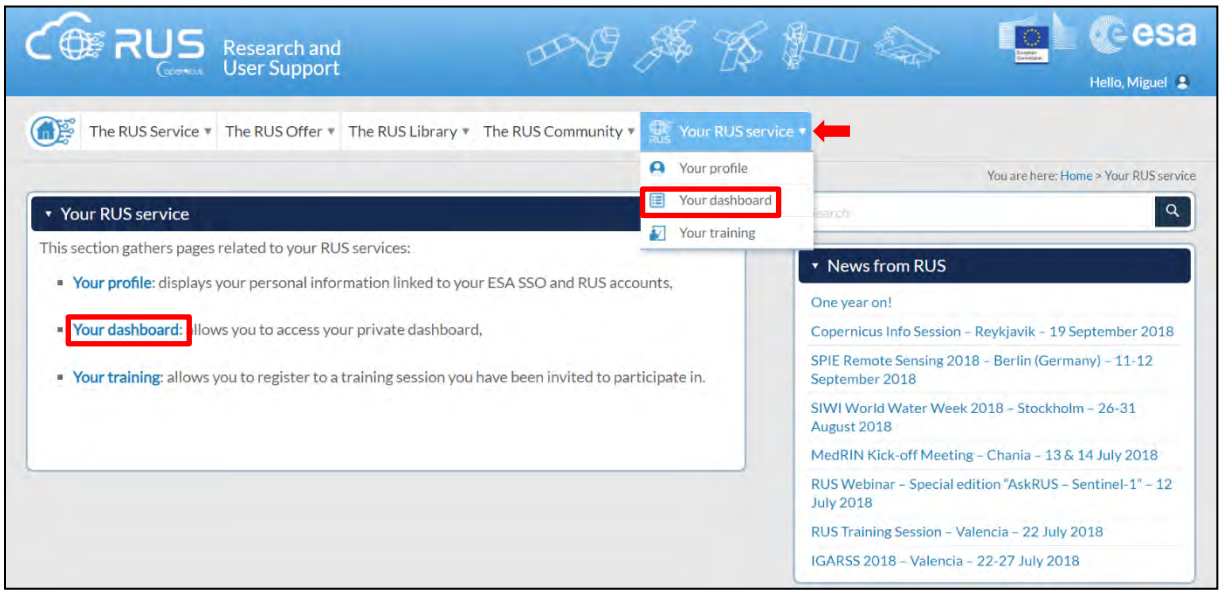


Upon your first login you will need to enter some details. You must fill all the fields.

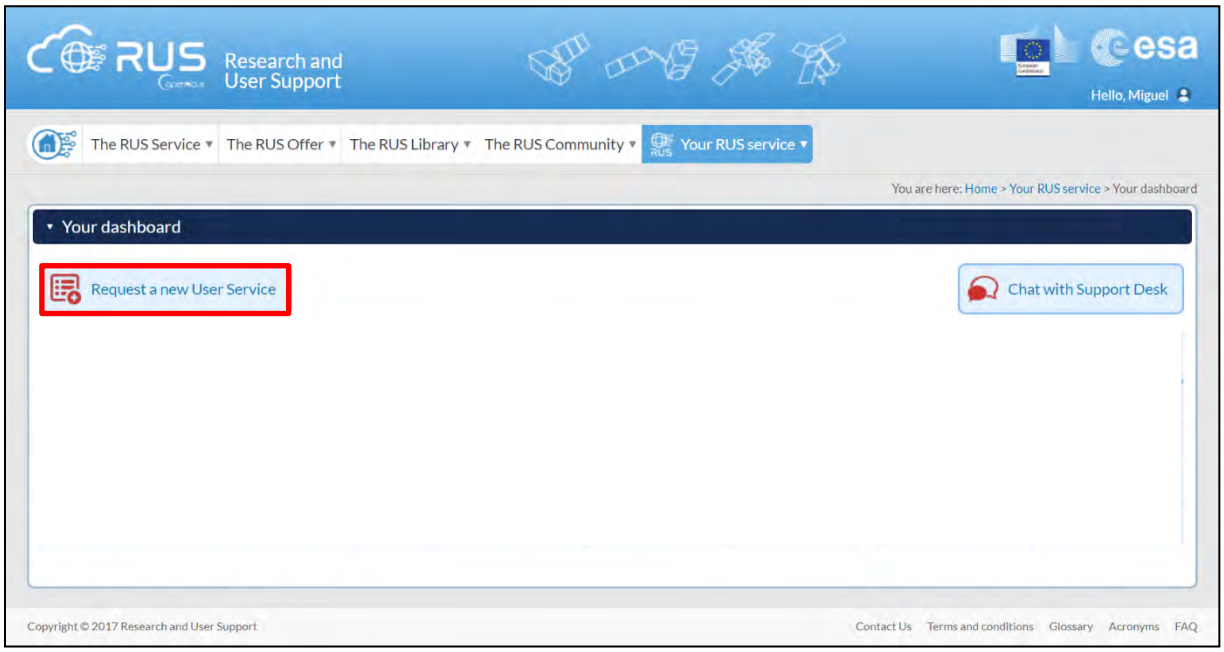


5 Request a RUS Copernicus Virtual Machine

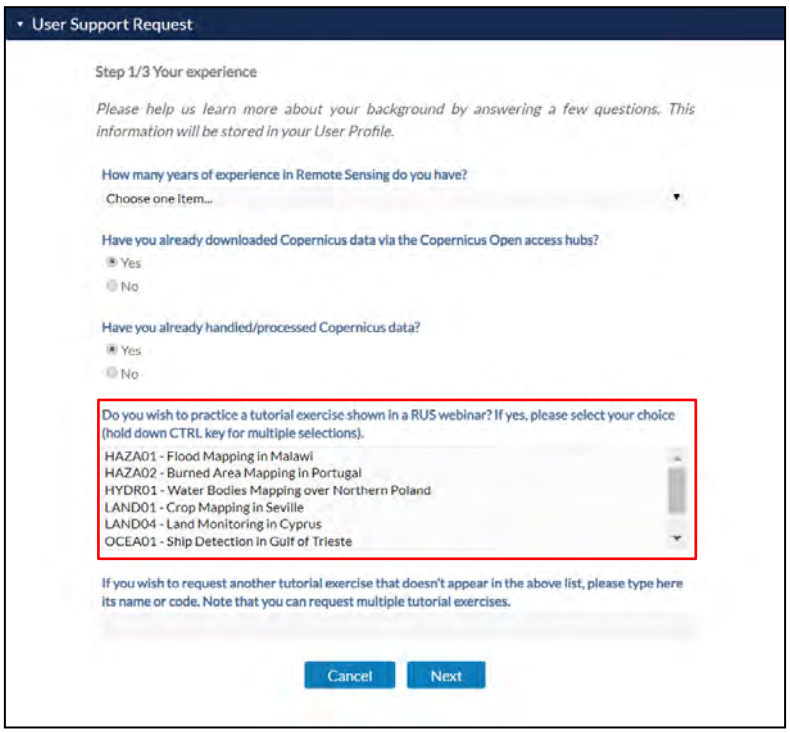
Once you are registered as a RUS user, you can request a RUS Virtual Machine to repeat this exercise or work on your own projects using Copernicus data. For that, log in and click on ***Your RUS Service*** → ***Your Dashboard***.



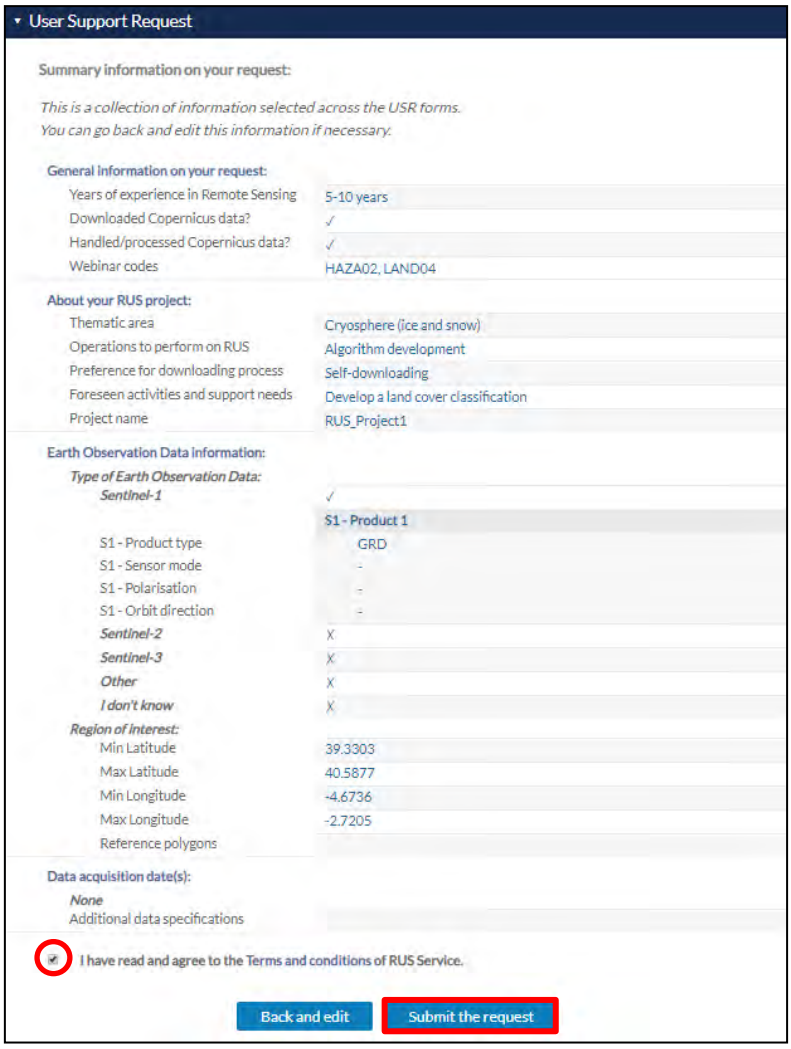
Click on ***Request a new User Service*** to request your RUS Virtual Machine. Complete the form so that the appropriate cloud environment can be assigned according to your needs.



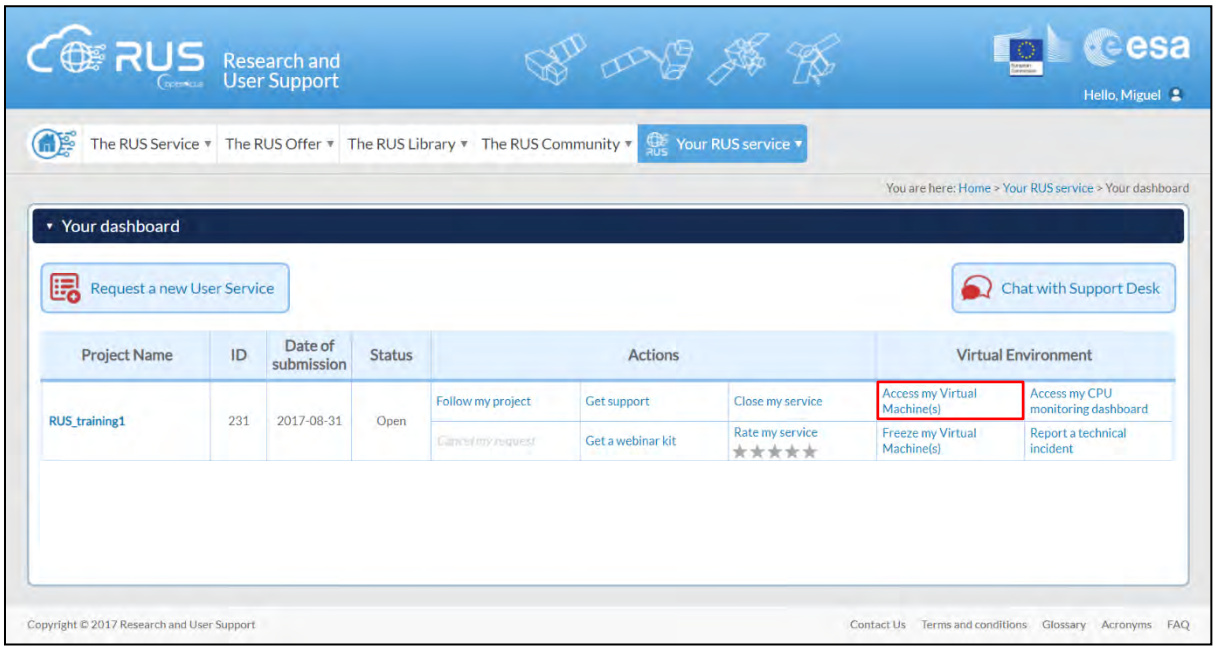
If you want to repeat this tutorial (or any previous one) select the one(s) of your interest in the appropriate field.



Complete the remaining steps, check the terms and conditions of the RUS Service and submit your request once you are finished.



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.

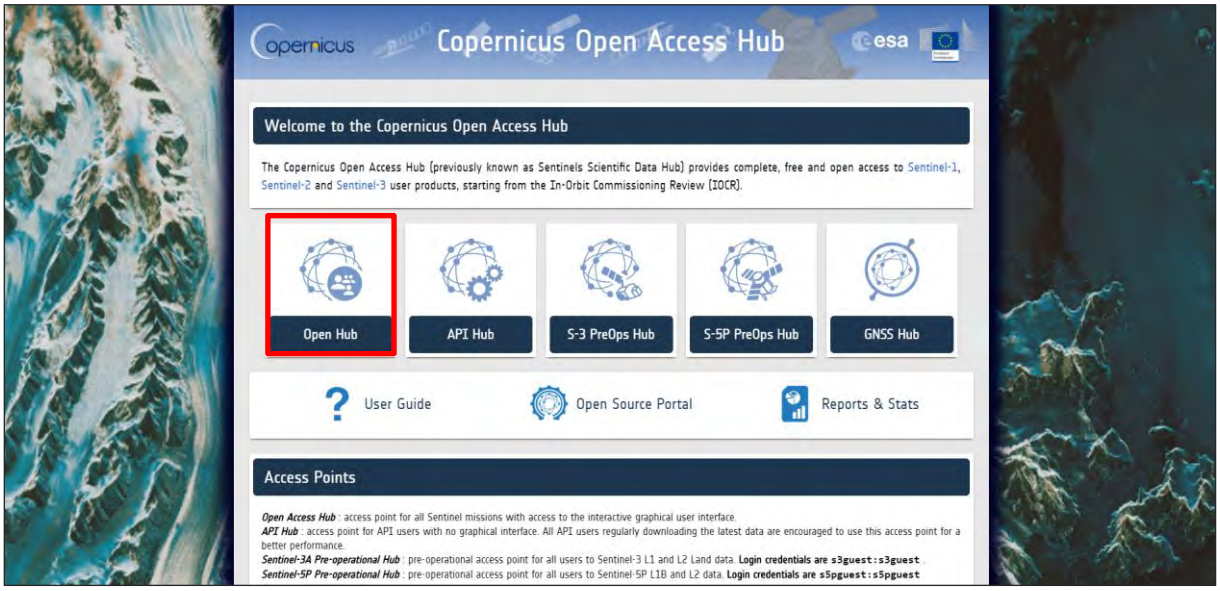
This is the remote desktop of your Virtual Machine.

6 Step by step

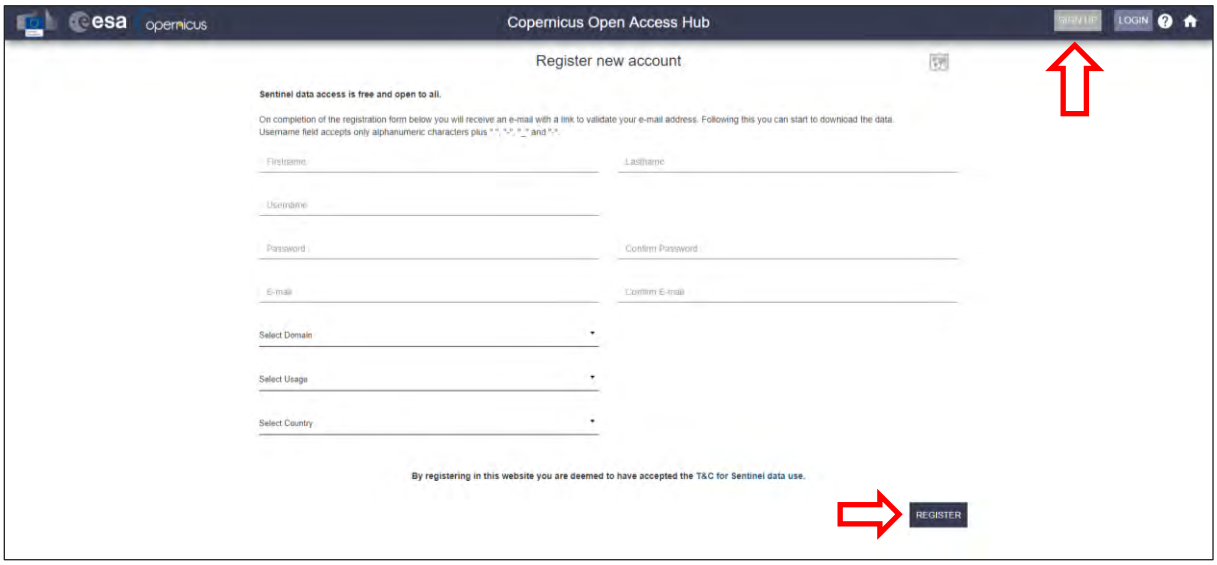
6.1 Data download – ESA SciHUB

Before starting the exercise, make sure you are registered in the Copernicus Open Access Hub so that you can access the free data provided by the Sentinel satellites.

Go to https://scihub.copernicus.eu/



Go to *Open Hub*. If you do not have an account, sign up in the upper right corner, fill in the details and click register.

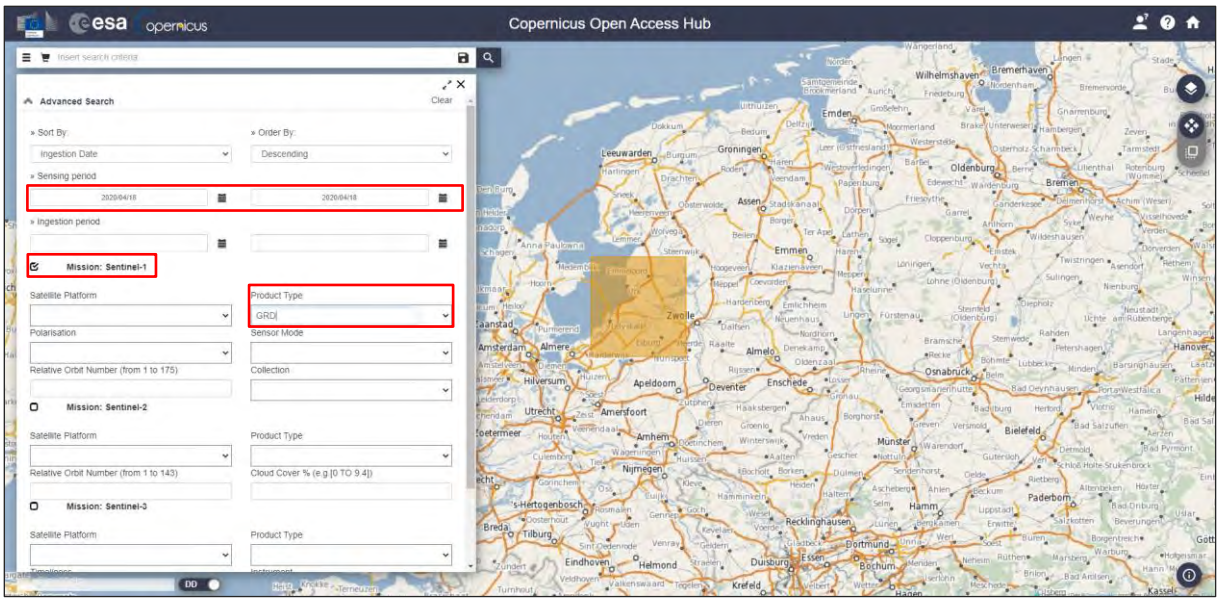


You will receive a confirmation email on the e-mail address you have specified: open the email and click on the link to finalize the registration.

Once your account is activated – or if you already have an account – log in.

In this guide, we describe the procedure to download the Sentinel-1 images. Make sure you repeat the same procedure and download the product for April 2020. Define the study area over The Netherlands. Then, open the search menu by clicking to the left part of the search bar (☰) and specify the parameters below. Press the search button (🔍) after that.

Sensing period: From 2020/04/18 to 2020/04/18
Check Mission: Sentinel-1
Product type: GRD



In this case the search returns 1 results. Download the product (it will be saved in */home/rus/Downloads)* and move it to the following path.

Path: */shared/Training/PY01_Sentinel1Processing_snappy/Original/*

6.2 Anaconda environment installation

In this exercise we will use *snappy* and python code in JupyterLab to process the Sentinel-1 product we have previously downloaded. However, before starting the analysis, we need to set up both the Anaconda environment and the *snappy* module to work (see NOTE 1 and NOTE 2).

> NOTE 1: Project Jupyter is a non-profit, open-source project, born out of the IPython Project in 2014 as it evolved to support interactive data science and scientific computing across all programming languages. Notebook documents (or "notebooks", all lower case) are documents produced by the Jupyter Notebook App, which contain both computer code (e.g. python) and rich text elements (paragraph, equations, figures, links, etc…). Notebook documents are both human-readable documents containing the analysis description and the results (figures, tables, etc..) as well as executable documents which can be run to perform data analysis. More info at: www.jupyter.org

> NOTE 2: Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability through use of significant whitespace. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects. Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including procedural, object-oriented, and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library. More info at: www.python.org
>
> Anaconda is a free and open-source distribution of the Python and R programming languages for scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, etc.), that aims to simplify package management and deployment. Package versions are managed by the package management system *conda*. More info at: https://www.anaconda.com/distribution/

To create the same conda environment as the one used for the creation of this training material open Terminal in your RUS Virtual Machine and copy-paste the following command. Then, press *Enter* to run it (this step may take several minutes, do not interrupt it). The conda environment will be called *snappyenv.*

```
conda env create -f /shared/Training/PY01_Sentinel1Processing_snappy/AuxData/condaenv_PY01.yml
```

Next, we will generate the *snappy* module in that environment so that it can be called and used later.

6.3 snappy module generation

SNAP provides the Python module *snappy* which allows you to access the SNAP Java API from Python. *snappy* requires either a SNAP installation or a SNAP build. The following instructions will guide you through the installation process to have *snappy* working in the RUS Copernicus Virtual Machine. Generic instructions on how to install *snappy* can be found in the following website:

https://senbox.atlassian.net/wiki/spaces/SNAP/pages/24051781/Using+SNAP+in+your+Python+programs

To start, open a Terminal window and navigate to the *bin* folder inside the SNAP installation directory (in RUS Copernicus Virtual Machines - */usr/local/snap/*):

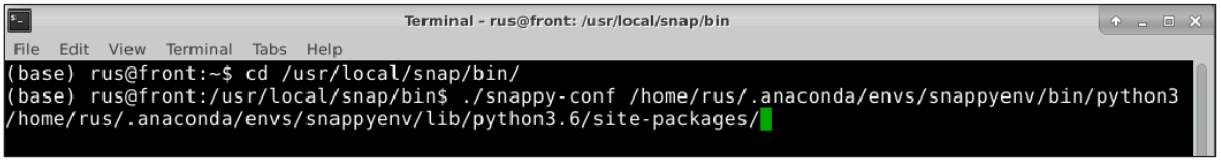
```
cd /usr/local/snap/bin/
```

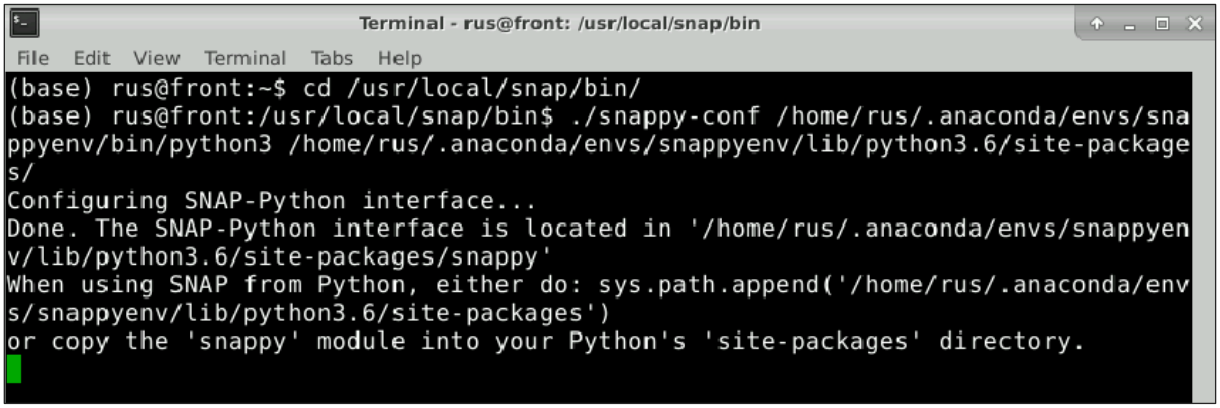
Next, we will generate the Python module *snappy* configured for the current SNAP installation and the Python interpreter of choice set in the *<python-exe>* parameter. In addition, instead of generating it in the default folder (*.snap/snap-python* in the home directory), we will place it in the site-package folder of our recently created conda environment. For that, we will make use of the *<snappy-dir>* parameter (see NOTE 3).

> NOTE 3: The parameter <python-exe> must be the full path to the Python interpreter executable which you want to use with SNAP (supported versions are 2.7, 3.3 to 3.6

Following the previous command in Terminal, copy-paste the next one and press *Enter*.

```
./snappy-conf /home/rus/.anaconda/envs/snappyenv/bin/python3 /home/rus/.anaconda/envs/snappyenv/lib/python3.6/site-packages/
```





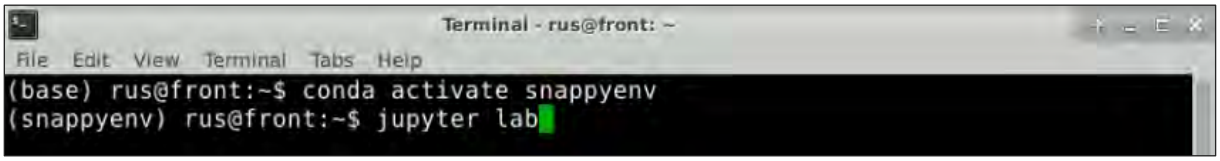
6.4 *Sentinel-1 Processing using snappy - JupyterLab*

Once the *snappy* module has been properly generated, we can start our exercise. Activate the *snappyenv* conda environment by running the following script in a new Terminal window.

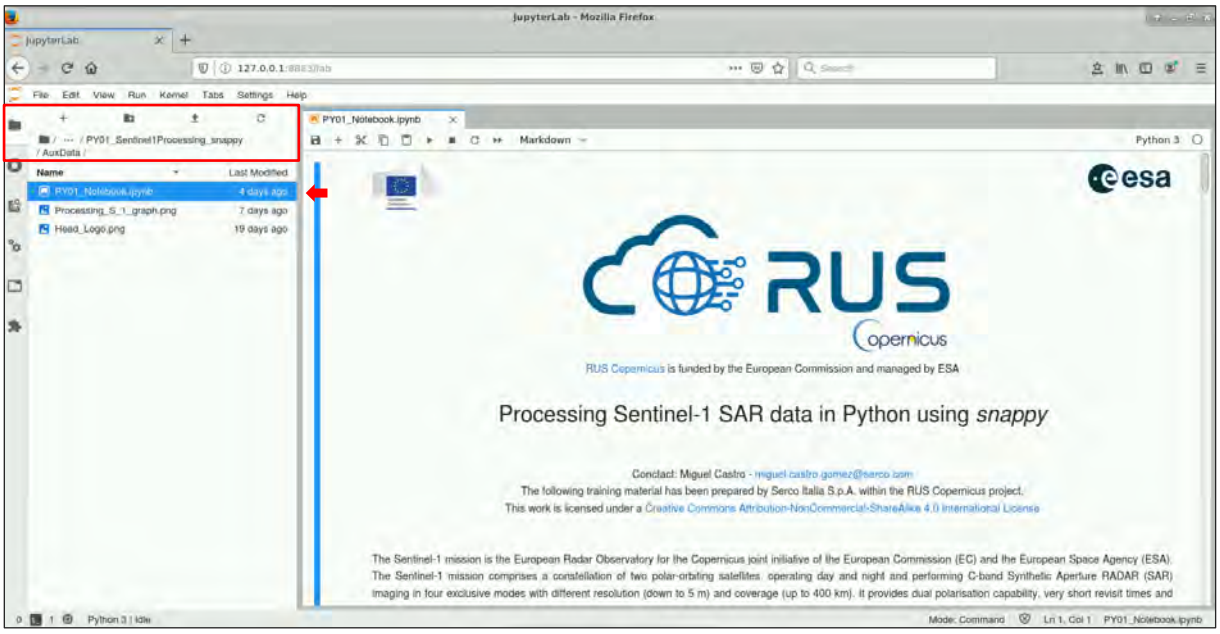
```
conda activate snappyenv
```

Next, write Jupyter Lab and press enter to launch the application. Once open, navigate to the following path inside the JupyterLab GUI and open the file *PY01_Notebook.ipynb*

```
jupyter lab
```



Path: */shared/Training/PY01_Sentinel1Processing_snappy/AuxData/*



Follow the instructions in the JupyterLab Notebook to continue the exercise.

Notes

THANK YOU FOR FOLLOWING THE EXERCISE!

7 Further reading and resources

Sentinel-1 User Guide

https://sentinel.esa.int/web/sentinel/user-guides/sentinel-1-sar

Sentinel-1 Technical Guide

https://sentinel.esa.int/web/sentinel/technical-guides/sentinel-1-sar

SNAP GPT Guide

https://senbox.atlassian.net/wiki/spaces/SNAP/overview

FOLLOW US!!!

@RUS-Copernicus

RUS-Copernicus

RUS-Copernicus

RUS Copernicus Training

RUS-Copernicus website

RUS-Copernicus Training website