SKYWATCH

esa

Sentinel-1 Toolbox

SAR-based landcover classification with Sentinel-1 GRD products

Issued October 2020

Andreas Braun

SAR-based land cover classification

The goal of this tutorial is to outline a possible approach to retrieve a land-use / land cover (LULC) classification based on Sentinel-1 image products. It provides notes on image selection and pre-processing, as well as different types of analysis for the detection of discrete land cover classes. The steps presented in this tutorial are suggestions on how to achieve this, but as always, there are other ways (e.g. polarimetric or coherence-based approaches) which can also be applied.

This tutorial requires basic understanding of radar data and its processing, for example as provided by the SAR Basics Tutorial or the S1TBX Introduction. Also, the S1TBX Graph Building Tutorial is recommended. A separate tutorial on Time-series analysis with Sentinel-1 focuses agricultural aspects in particular.

For an introduction into SAR-based classification approaches, the following materials and references are recommended.

- Caetano, M. (2018): **Land cover and land use theory.** ESA Land Training 2018 (PDF)
- NASA (2019): **SAR for landcover applications** (URL)
- NASA (2018): **SAR for mapping land cover** (URL)
- EO College (2016): **Classification** (URL), **Land cover classification** (URL)
- Abdikan et al. (2016): **Land cover mapping using Sentinel-1 SAR data** (URL)
- Banqué et al. (2015): **Polarimetry-based landcover classification with Sentinel-1 data** (PDF)

Background and preparation

Download the data

The data used in this tutorial can be downloaded from the Copernicus Open Access Hub at https://scihub.copernicus.eu/dhus (free registration requires). Search and download the following products:

S1A_IW_GRDH_1SDV_20180220T165225_20180220T165250_020693_023723_FB7D
S1A_IW_GRDH_1SDV_20180726T165231_20180726T165256_022968_027E14_0A09

Instead of entering the product IDs, you can also search for the following criteria:

Sensing dates	20.02.2018, 26.07.2018
Product type	GRD
Polarization	VV+VH
Sensor mode	IW
Orbit direction	ascending (enter in the text field, see Figure 1)
Longitude	14.40 to 14.70
Latitude	53.30 to 53.50 (draw polygon in this area, see Figure 1)

Please note: Since September 2018, the Copernicus Open Access Hub has transitioned into a long-term archive which means that products of older acquisition dates have to be requested first. Please find more information about this here: https://scihub.copernicus.eu/userguide/#LTA_Long_Term_Archive_Access and here https://scihub.copernicus.eu/userguide/LongTermArchive.

More information on Sentinel-1 IW products is provided here: https://sentinel.esa.int/web/sentinel/user-guides/sentinel-1-sar/acquisition-modes/interferometric-wide-swath

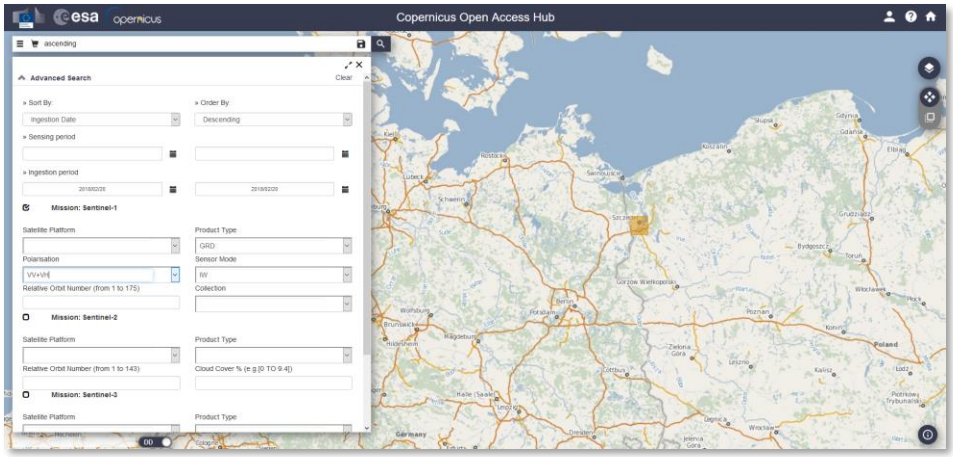


Figure 1: Retrieval of suitable images

Notes on image selection

The data used in this tutorial is a part of a Sentinel-1 IW product located at the border between Germany and Poland and covers an area of ca. 150 x 110 km (Figure 2, blue line). The subset which is selected contains the city of Szczecin and the Dąbie Lake (bottom center), surrounded by agricultural areas and forests, and the Szczecin lagoon and the Baltic Sea in the north.

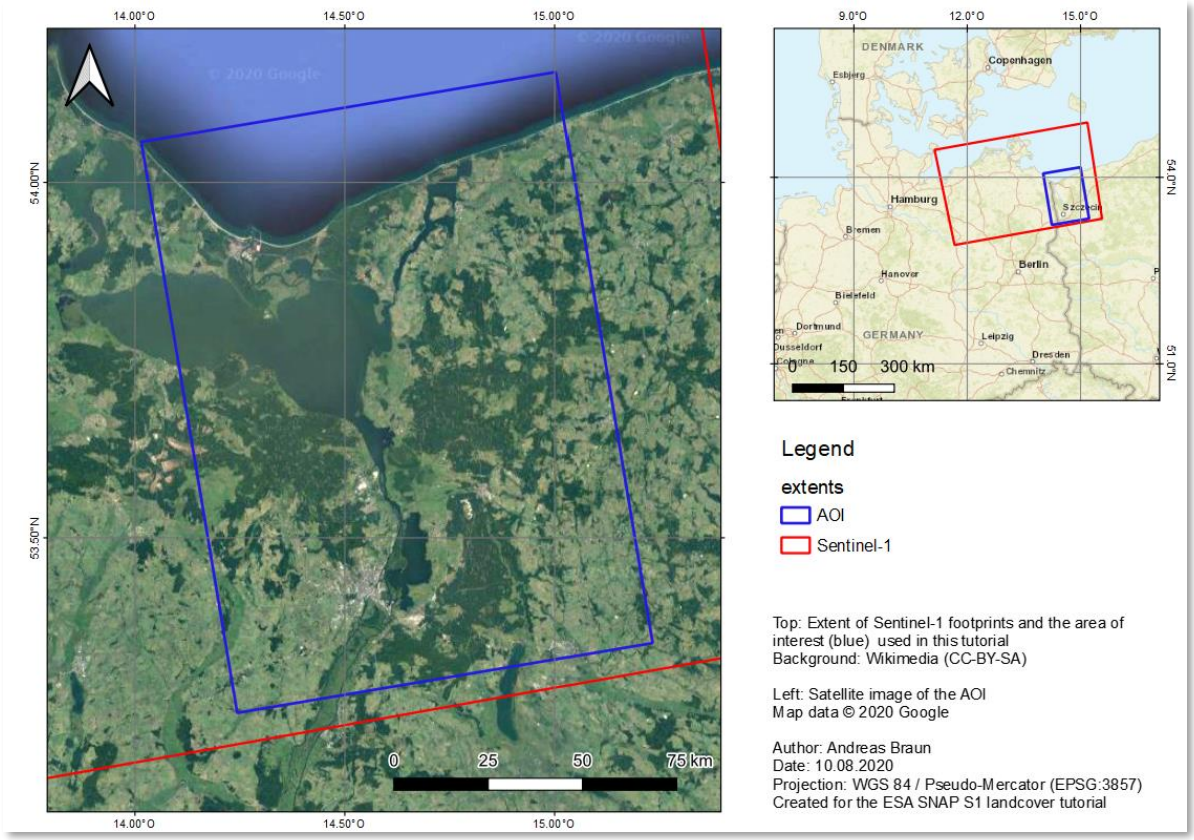


Figure 2: Extent and location of the data

Unlike multi-spectral missions, such as Sentinel-2 or Landsat-8, Sentinel-1 operates at a single wavelength only (5.6 cm which corresponds to a frequency of 5.4 GHz). However, Sentinel-1 mostly operates in dual-polarization mode which means that vertical waves are emitted from the sensor and both vertical and horizontal waves are measured when returning to the sensor, leading to backscatter intensity of VV and VH. As the proportion of vertically transmitted waves returning to the sensor horizontally is small, the intensity of the VH band is predominantly lower than the one of the VV band (Figure 9, middle and right).

Still, the feature space (the number of variables which can be used to predict target classes) of S1 data is limited compared to optical data. Therefore, two images are used in this tutorial which are taken in the same year, but at different seasons. This allows to describe surfaces based on their temporal characteristics to a certain degree. For this reason, image 1 was acquired in the month with the least precipitation (February) and image 2 was acquired at the end of July (most precipitation), as shown in Figure 3.

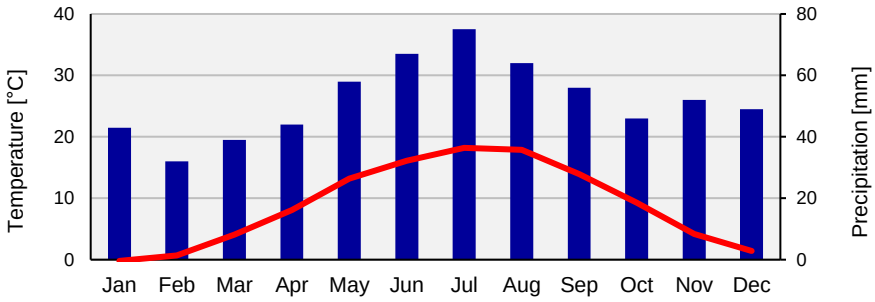


Figure 3: Climate chart of the study area

Pre-processing

Open the products

Use the *Open Product* button in the top toolbar and browse for the location of the downloaded data. Select the two downloaded zip files and press **Open Product**.

In the *Products View* you will see the opened products. Each Sentinel-1 product consists of *Metadata*, *Vector Data*, *Tie-Point Grids*, *Quicklooks* and *Bands* (which contains the actual raster data, represented by `Intensity` and `Amplitude`) as demonstrated in Figure 4).

Double-click on the `Intensity_VV` band to view the raster data (Figure 5). Depending on the capabilities of your computer, this might take a while. To reduce the amount of data, a subset is created in the next step which covers only a part of the dataset.

Note: The image is displayed “upside down” because it was acquired in an ascending pass.

The World View or World Map menus are also a good way to check the coverage of one or more products (indicated by a small black number). To reduce the effects of looking direction and angles, both images were acquired from the same track.

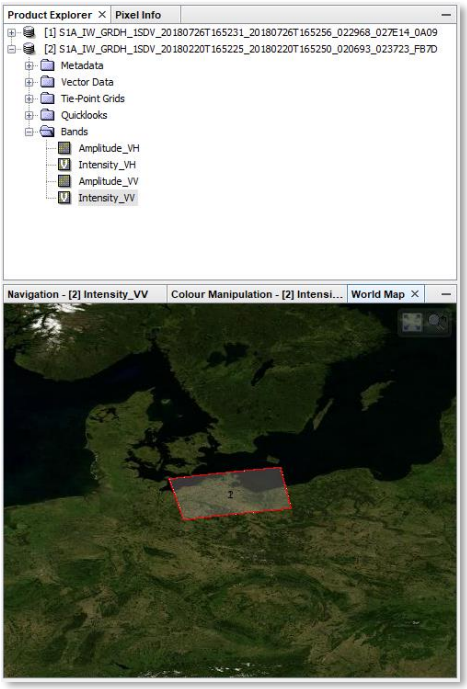


Figure 4: Product Explorer

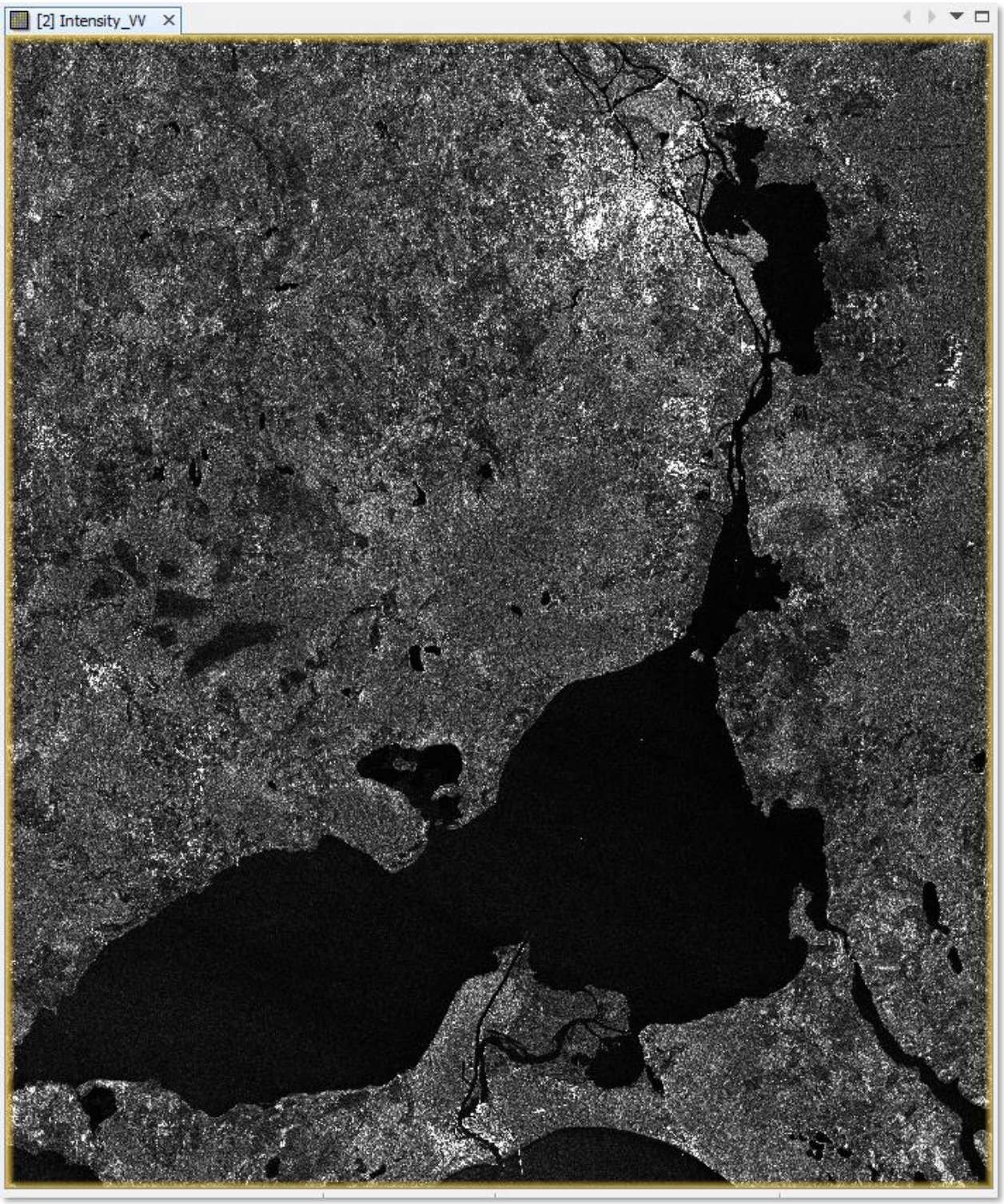


Figure 5: Preview of the S1 GRD product

Create a subset

To reduce the loaded data to the area of interest, select the first product and open the *Subset* operator (under *Raster*). A separate window might open ("Creating thumbnail image"), but this takes quite a long time, so it is ok to **Cancel** this window.

Switch to "Geo Coordinates" Enter the following numbers (Figure 6):

- North latitude bound: 53.25
- West longitude bound: 14.25
- South latitude bound: 54.15
- East longitude bound: 15.00

Confirm with **Run** and repeat the step for the second S1 image.

A new product is shown in the *Product Explorer*, starting with `subset_0_of_` for each subset. These products are only of temporary nature and lost after SNAP is closed. We use them as inputs for the next step.

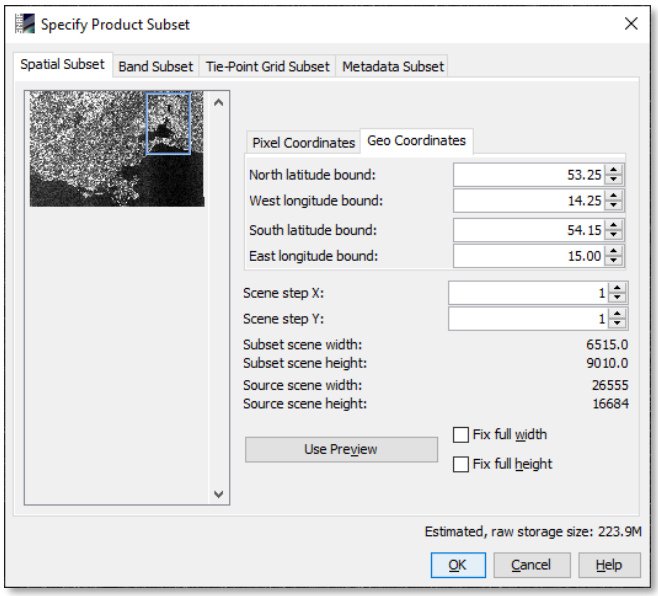


Figure 6: Creating a subset

Applying orbit information

Orbit auxiliary data contain information about the position of the satellite during the acquisition of SAR data. They are automatically downloaded for Sentinel-1 products by SNAP and added to its metadata with the *Apply Orbit File* (under the menu point *Radar*) operator.

The Precise Orbit Determination (POD) service for Sentinel-1 provides restituted orbit files and precise Orbit Ephemerides (POE) orbit files. POE files cover ca. 28 hours and contain orbit state vectors at intervals of 10 seconds intervals. Files are generated one file per day and are delivered within 20 days after data acquisition.

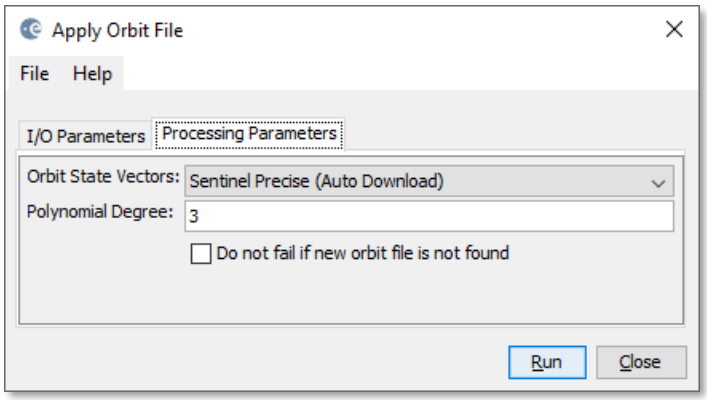


Figure 7: Apply Orbit File

If precise orbits are not yet available for your product, restituted orbits can be selected which may not be as accurate as the Precise orbits but will be better than the predicted orbits available within the product. Execute the operator for both subsets as generated in the previous step (Figure 7) and select the following target product names (also make sure to select suitable target directories):

- `20180220_Orb.dim`
- `20180726_Orb.dim`

Radiometric calibration

Radiometric calibration converts backscatter intensity as received by the sensor to the normalized radar cross section (Sigma0) as a calibrated measure taking into account the global incidence angle of the image and other sensor-specific characteristics. This makes radar images of different dates, sensors, or imaging geometries comparable.

Open the *Calibration* operator (under *Radar* > *Radiometric*) and use each product from the previous step (`20180220_Orb.dim` and `20180726_Orb.dim`) as a source product. For the target product name, add `_Cal` at the end of the name as suggested (Figure 7). All other settings can be left on default. SNAP uses the image metadata to calibrate the image as described in this documentation. After completion, the output products are added to the Product Explorer.

As shown in Figure 9, the data ranges between 0 and 1 after calibration to Sigma 0. Thermal Noise Removal is not applied in this tutorial but should be considered when working with larger subsets or entire images.

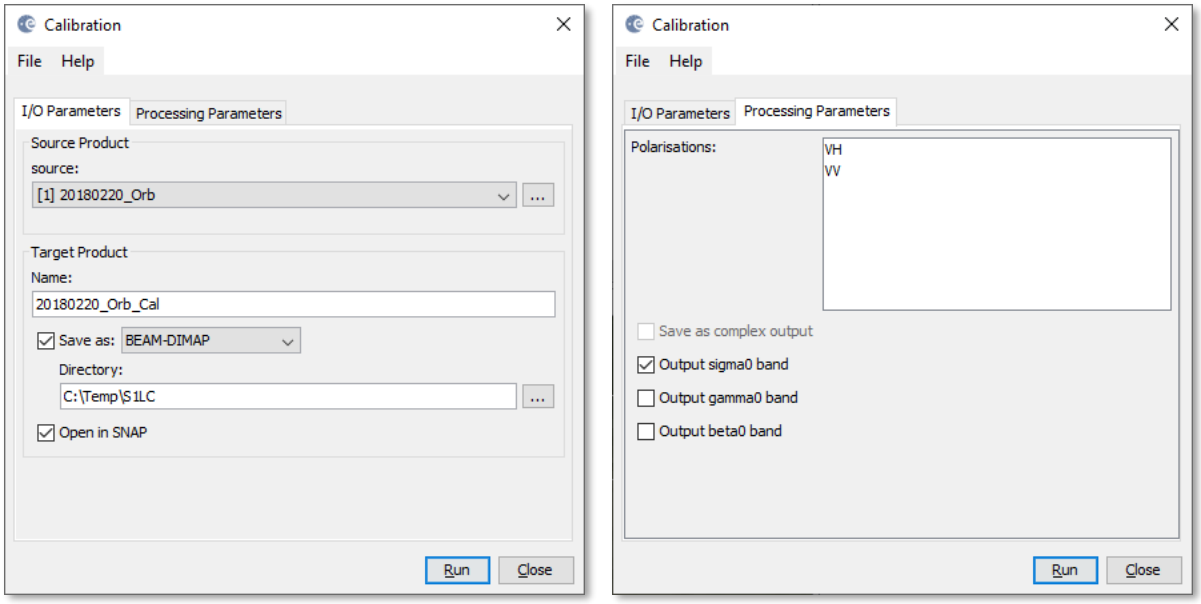


Figure 8: Radiometric Calibration

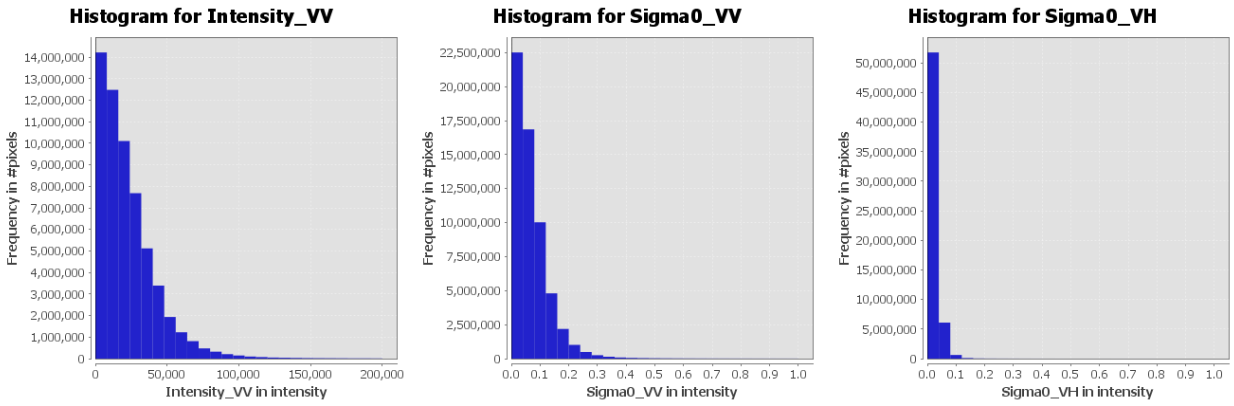


Figure 9: Histograms before (left), after (middle and right) after radiometric calibration

Coregistration

After both images were radiometrically calibrated in the previous step, the coregistration brings both into one stack. Even if both images were acquired from the same track (relative orbit), smaller differences in the incidence angle can cause inaccurate pixel positioning in parts of the image. These differences are identified and compensated by the *Coregistration* operator (under *Radar* > *Coregistration*) which produces one output product containing both images from February and July with best possible geometric overlay.

Load the two products (with orbit files applied) to the *ProductSet-Reader* tab (Figure 10, left) and leave all settings as predefined. Enter `S1_Orb_Cal_Stack` as target product name and click **Run** to start the coregistration. Depending on the size of the products and the number of GCPs, this process can take a while.

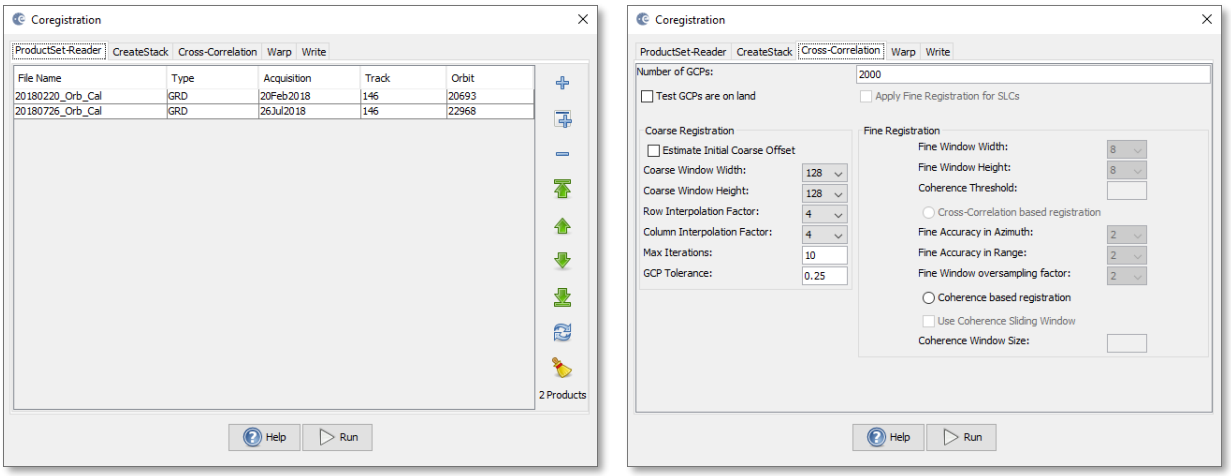


Figure 10: Coregistration of Sentinel-1 GRD products

Optional: To check the mathematical quality of the coregistration after its completion you can select the newly generated product and open the *InSAR Stack* tool from the toolbar . A new window will open which shows you (in the *Coregistration Residuals* tab) the position and root mean square (rms) error of the identified ground control points (GCPs).

In the example (Figure 11) 110 GCPs were left from the initial 2000 randomly placed points. These have a mean rms of 0.0214 which means that the images are now registered at sub-pixel accuracy.

If the coregistration fails it is advisable to increase the number of GCPs (Figure 10, right) and also to have the initial coarse offset estimated by the operator.

More information on error measures of image registration and transformation is given here.

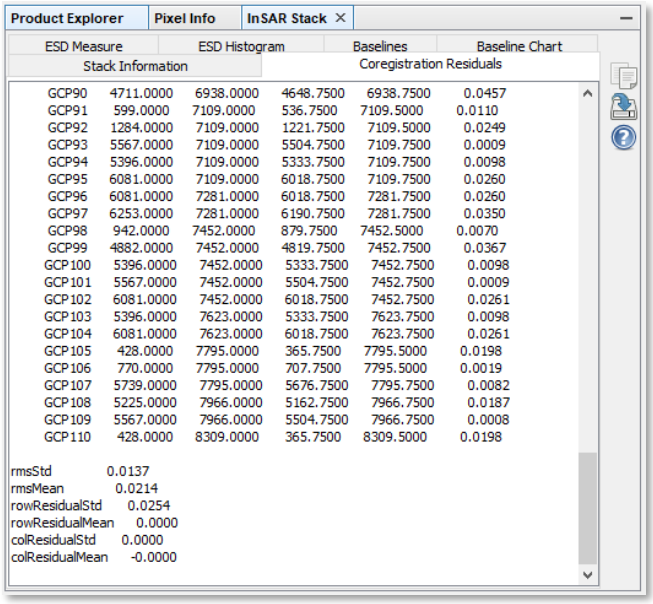


Figure 11: InSAR Stack window

Besides the accuracy of the residuals, it is recommended to visually check the quality of the stack. This can be done by an RGB representation of the master and slave product which shows if the images are correctly aligned. Select February VV for red, February VH for green and July VV for blue (Figure 12).

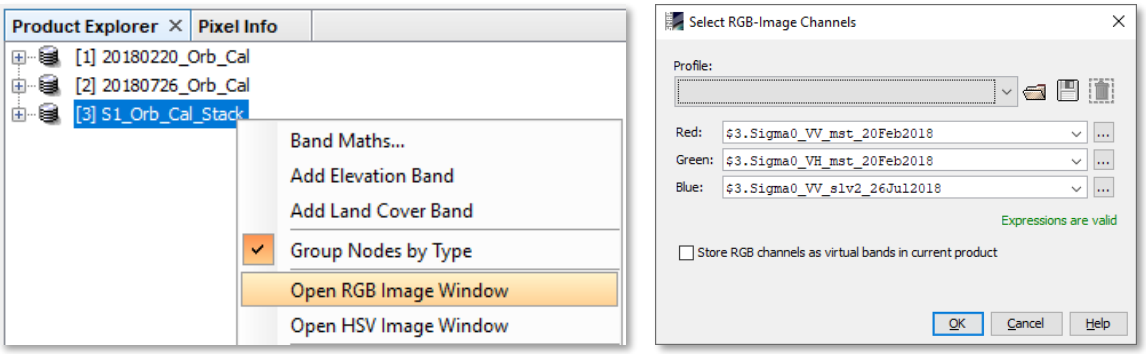


Figure 12: Generation of an RGB image

Zoom in to an area with distinct surfaces, for example the border between land and water. The RGB image should be clear and sharp. The only exception are changes in land cover or scattering mechanisms which occurred in the time between the first and the second image acquisition. In Figure 13, the rivers, roads and agricultural fields are sharply displayed in all colors. White pixels indicate high backscatter in all three bands (urban areas) and black pixels low backscatter values in all three bands (water). Blue pixels indicate a slightly different shoreline in July compared to February.

Figure 13: RGB image of VV1 (red), VH1 (green) and VV2 (blue)

Speckle filtering

As shown in Figure 12, many areas which are expected to have a homogenous backscatter intensity are characterized by granular patterns. These salt-and-pepper effects are speckle which is inherent in any SAR image and caused by different constructive and destructive contributions to backscatter intensity of different scattering mechanisms below the pixel resolution. Adaptive speckle filters were designed to enhance the image quality by suppressing random variations in the image while maintaining sharp edges.

Open *Radar* > *Speckle Filter* > *Single Product Speckle Filter* and select `S1_Orb_Cal_Stack` as input product. In the second tab, select the **Lee Sigma** filter from the drop-down menu and leave the window sizes and parameters as predefined. A new filtered product (`S1_Orb_Cal_Stack _Spk`) is created which has the same band names as the input, but now speckle has been reduced.

You can compare the image before and after filtering by using the **Split Window** function from the toolbar (Figure 14). As always, the different types of filters will produce slightly different results. Also, the impact of filter sizes and other parameters can strongly affect how the result looks. It is suggested to explorer and compare some filters and configurations to find the most suitable for the respective application.

As shown in the example, the Lee Sigma filter smoothed larger patches of similar backscatter while preserving linear features and sharp edges between different surfaces. This is an important prerequisite for the later land cover classification because speckle introduce unwanted patterns and granular effects in the classified outputs. Especially for SAR-based land cover classifications a strong filtering is advised.

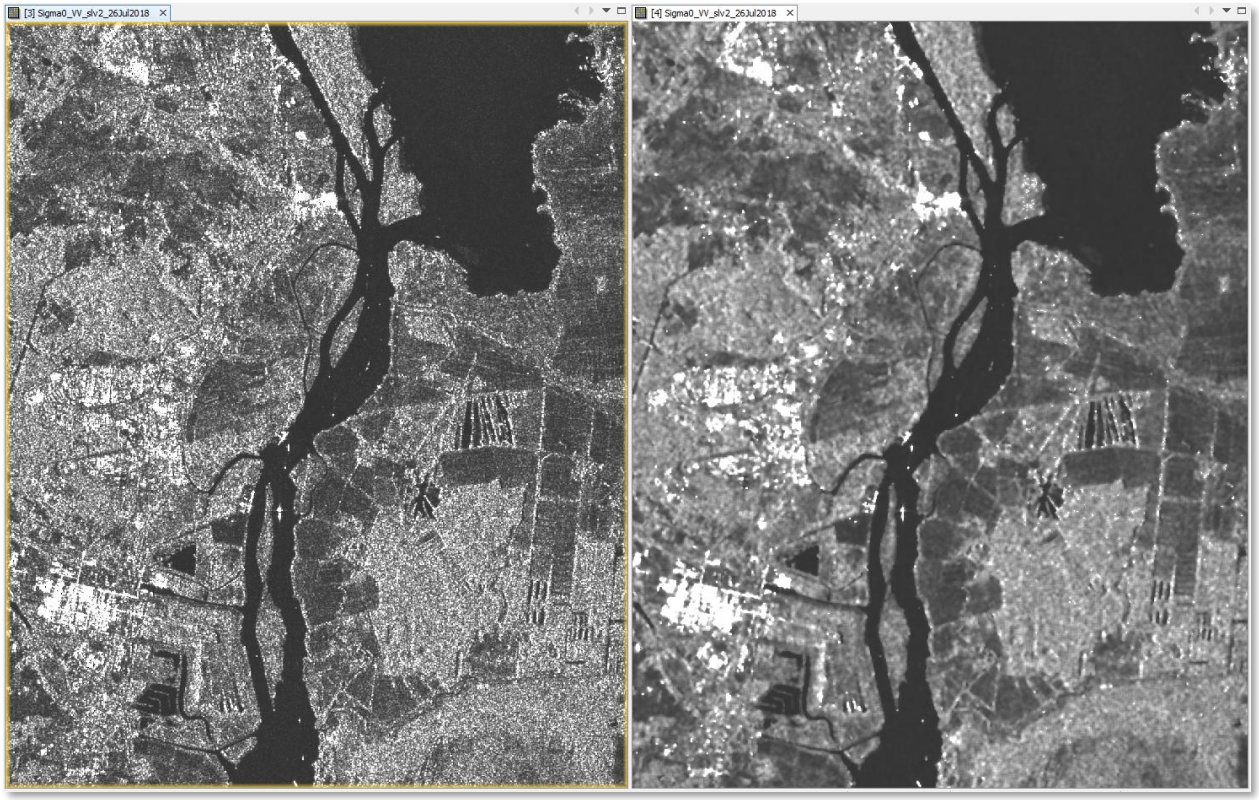


Figure 14: Sigma0 VV before (left) and after (right) application of a speckle filter

Terrain correction

Terrain Correction will geocode the image by correcting SAR geometric distortions using a digital elevation model (DEM) and producing a map projected product.

Geocoding converts an image from slant range or ground range geometry into a map coordinate system. Terrain geocoding involves using a Digital Elevation Model (DEM) to correct for inherent geometric distortions, such as foreshortening, layover and shadow (Figure 15). More information on these effects is given in the ESA radar course materials.

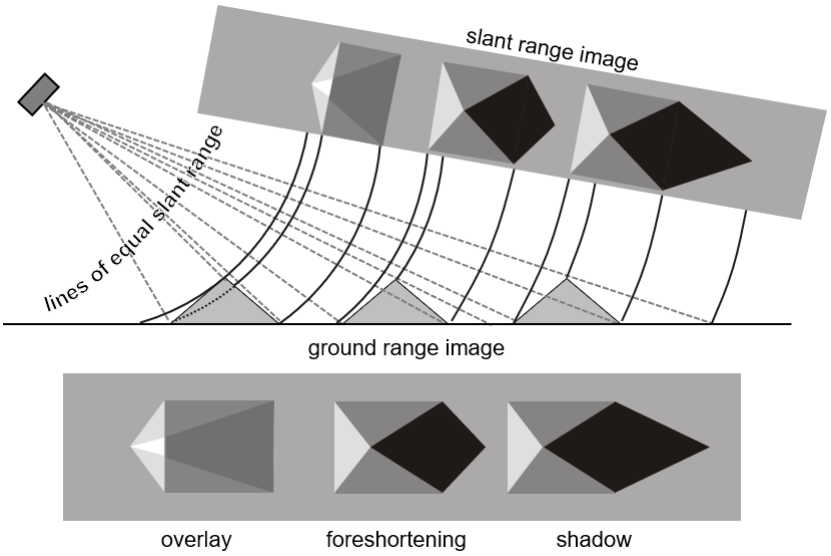


Figure 15: Geometric distortions in radar images (Braun 2019)

Open the *Range Doppler Terrain Correction* operator (under *Radar* > *Geometric* > *Terrain Correction*). Select the coregistered product as an input in the first tab.

In the Processing Parameters tab, select **SRTM 1Sec HGT (AutoDownload)** as input DEM.

Please be aware that SRTM is only available between 60° North and 54° South. If your area lies outside this coverage (https://www2.jpl.nasa.gov/srtm/coverage.html), you can use one of the other DEMs with AutoDownload option or use an external DEM (just make sure it stored as a GeoTiff and projected in geographic coordinates / WGS84).

In this tutorial we select **WGS84** as Map Projection. It is based on geographic coordinates (latitude and longitude). For the later use in a geographic information system (GIS) projected coordinate systems, such as UTM (Automatic) could be selected as well.

If no **Source Band** is selected, all bands of the input product are geometrically corrected.

Click on **Run** to start the terrain correction. SNAP now establishes a connection to an external elevation database to download all SRTM tiles required to fully cover the input dataset.

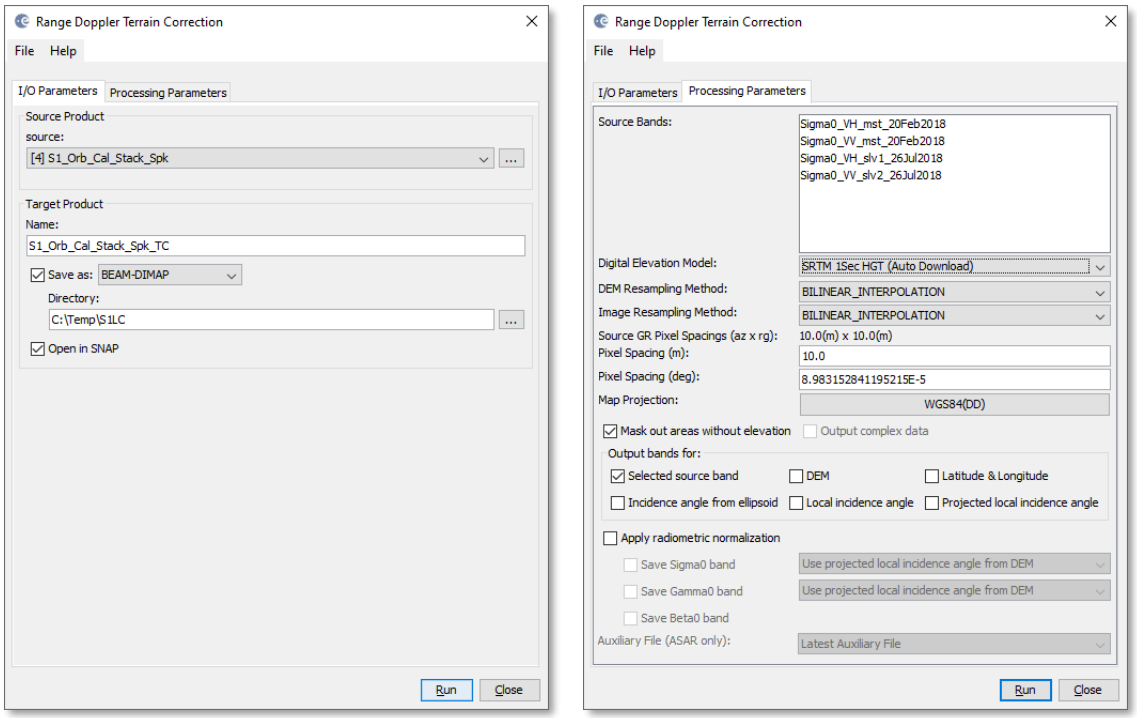


Figure 16: Range Doppler Terrain Correction

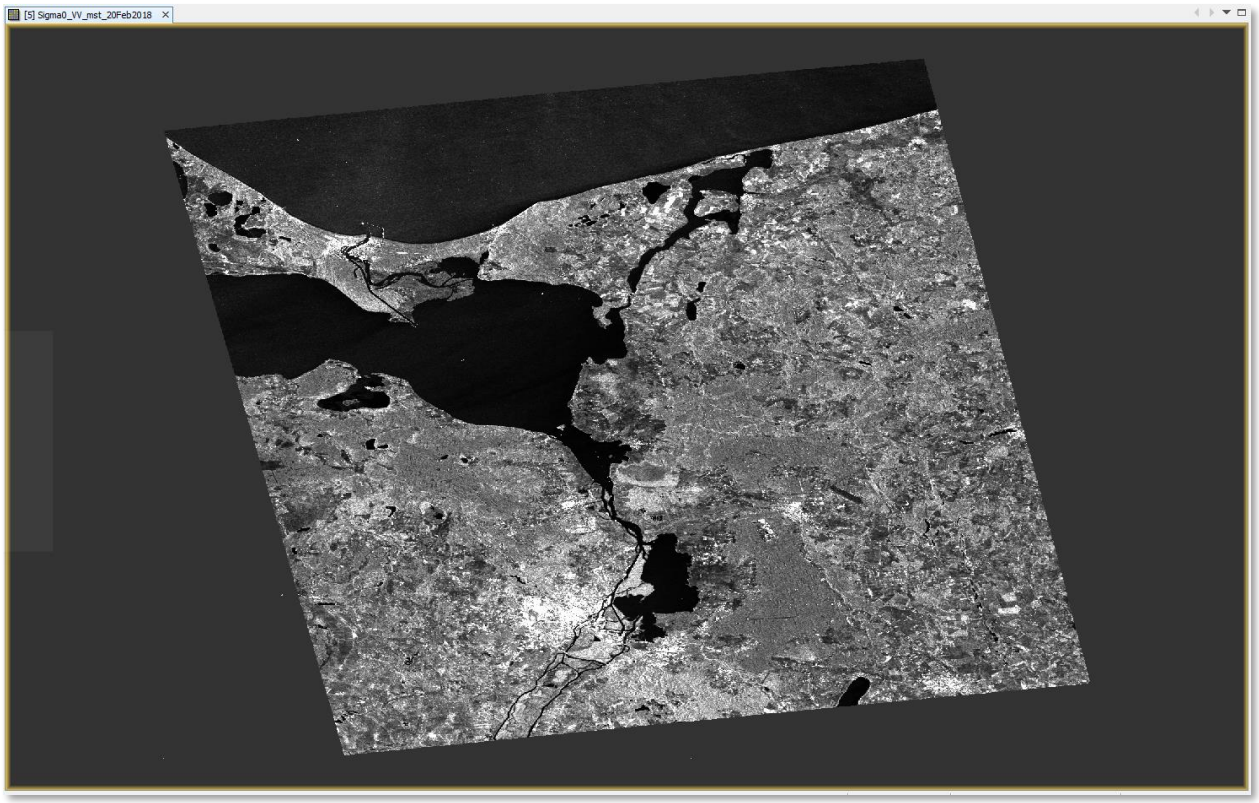


Figure 17: Geocoded image product after Terrain Correction

Conversion to dB scale

As highlighted by the histograms in Figure 9, the values of Sigma0_VV roughly range between 0 and 1, with average backscatter values of 0.07 (VV) and 0.02 (VH). This means that there are many dark pixels and only very few bright pixels with large values. This is not ideal in a statistical sense and leads to low visual contrasts.

To achieve a more normal distribution of values, the log function is applied to the radar image. It translates the pixel values into a logarithmic scale and yields in higher contrasts, because the bright values are shifted towards the mean while dark values become stretched over a wider range (Figure 18, right). The values of calibrated dB data roughly range between -35 to +10 dB.

Right-click on each of the four terrain corrected bands and select **Linear to/from dB**. Confirm with Yes to create a virtual band (indicated by the  symbol and the `_db` suffix). These virtual bands are not physically stored on the hard drive but can still be displayed based on the underlying mathematical expression.

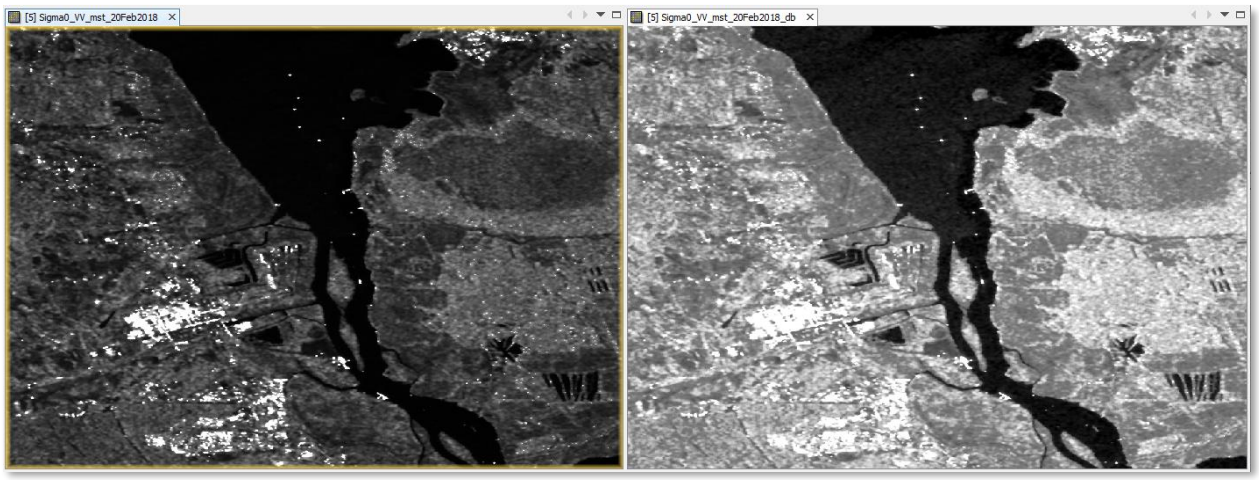


Figure 18: Sigma0 VV before (left) and after (right) conversion to dB scale

Unsupervised classification

An unsupervised classification is a good way to identify and aggregate pixels with similar features.

Open the *K-Means Cluster Analysis* under *Raster > Classification > Unsupervised Classification* and select the terrain corrected stack as input product. In the second tab, set the **number of clusters** to 10, select all dB bands as **source bands** and confirm with **Run**. This can take some minutes because the classifier now undertakes 30 iterations with random cluster centers of the multi-dimensional feature space to find pixels of similar properties. More information.

The output is a product with a single band (class_indices) where each pixel is assigned to one of 10 classes (clusters). These clusters do not necessarily represent semantic classes of landcover, but they are homogenous regarding their backscatter characteristics.

As shown in Figure 20, you can use the *Color Manipulation* tab to assign colors and even names to these clusters to see how well they coincide with the general land use and land cover classes.

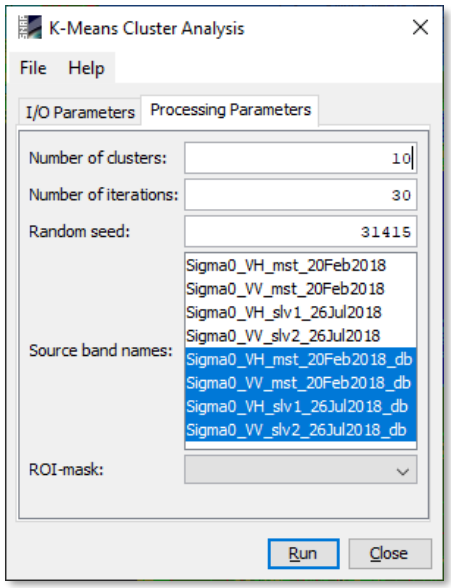


Figure 19: Unsupervised clustering

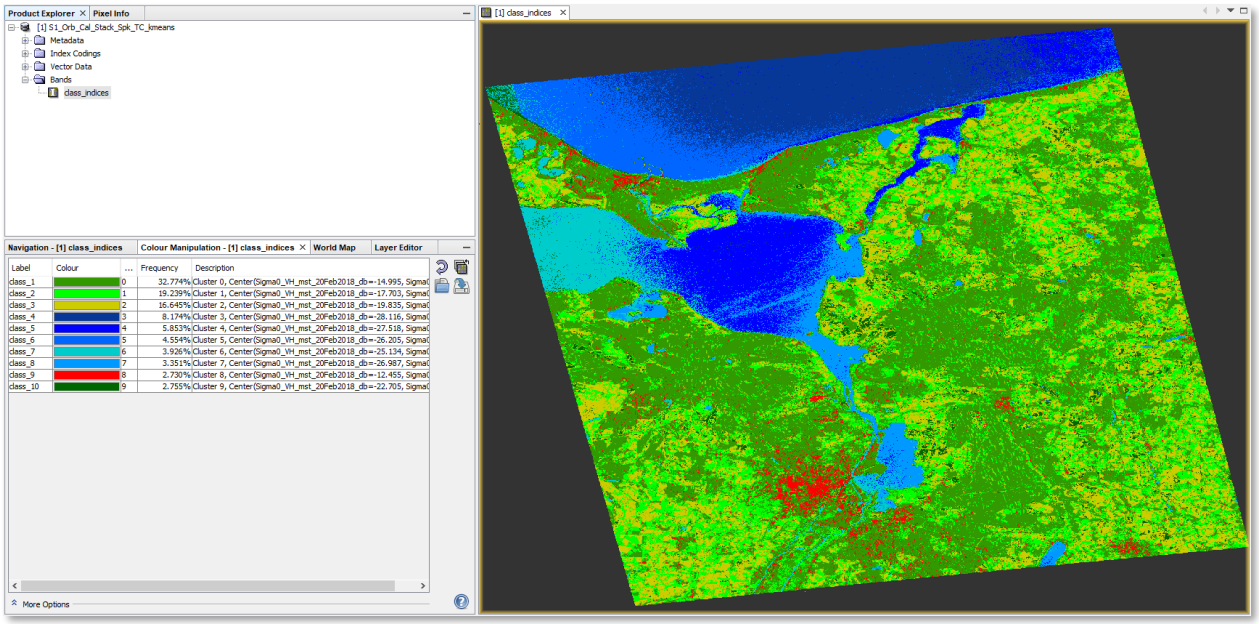


Figure 20: Possible result of the unsupervised k-means clustering

Accordingly, the advantage of an unsupervised classification is that it does not require any a-priori knowledge on the study area or the data and still groups pixels of similar characteristics. The downside is that the number of classes strongly determines if the result is useful or not. As shown in the example, water consists of five different classes (3-7) while urban areas are represented by only one class (8). In case of class overlap, a higher number of clusters should be selected and merged subsequently. On the other hand, too many classes make it hard to find patterns in the data.

Rule-based classification

Although there are automated ways of assigning classes to pixels, especially radar images are suitable for rule-based classification approaches, because the number of input features is limited (four polarizations). This can be done in the following order:

1. **Definition of classes**: In contrast to the unsupervised classification, rule-based classification requires the a-priori definition of target classes. This means, a list of expected land cover types has to be defined in advance. In this case, these are: 1=water, 2=urban, 3=forest, 4=agriculture.
2. Inspection of **pixel values**: Find value ranges which represent thee using the four defined classes based on the available bands (VV and VH for February and July). You can use the *Pixel Information* tab, the *Histogram* (of defined ROIs), the *Profile Plot* view (across two different classes) to understand which values represent the different surface types. You can even use the Pin Manager to label different surfaces and export a table with the class names and the corresponding values, for example for the Tree classification module in the free and open data mining software Orange to systematically search for thresholds in the data to separate the classes.
3. **Implement the thresholds** in the Mask Manager. In this case, a systematic thresholding is applied to the four bands in order to separate the four target classes in a hierarchical way. An example is given in Figure 12

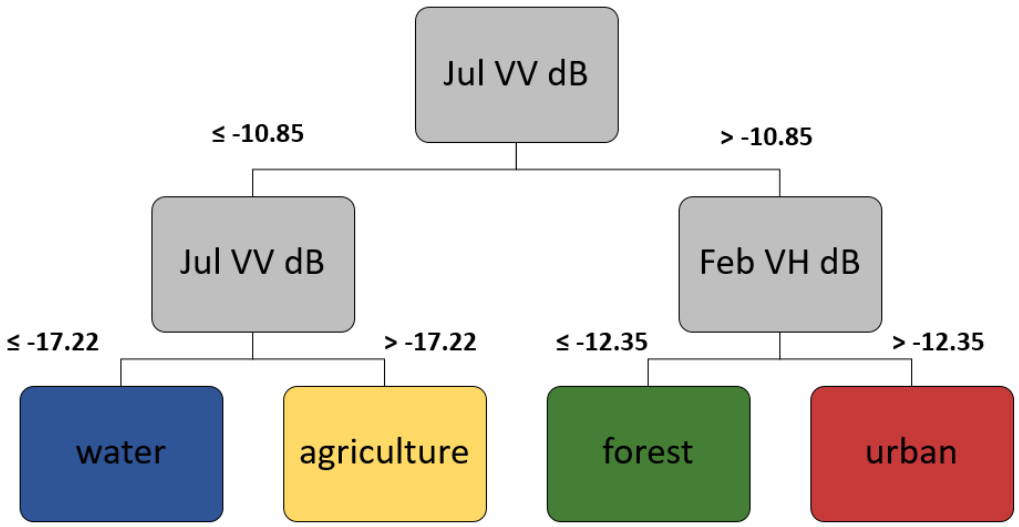


Figure 21: Rules for class thresholding

Now we use the Mask Manager to create the expressions shown in Table 1 which represent the predicted spatial occurrence of the four target classes. Note: A detailed explanation on the use of the Mask Manager is given in the tutorial “Synergetic use of S1 (SAR) and S2 (optical) data Tutorial”.

Table 1: Expressions for the Mask Manager

name	expression
water	`Sigma0_VV_slv2_26Jul2018_db <= -10.85 AND Sigma0_VV_slv2_26Jul2018_db <= -17.22`
agriculture	`Sigma0_VV_slv2_26Jul2018_db <= -10.85 AND Sigma0_VV_slv2_26Jul2018_db > -17.22`
forest	`Sigma0_VV_slv2_26Jul2018_db > -10.85 AND Sigma0_VH_mst_20Feb2018_db <= -12.35`
urban	`Sigma0_VV_slv2_26Jul2018_db > -10.85 AND Sigma0_VH_mst_20Feb2018_db > -12.35`

Note: The band names depend on the order in the stack and might differ accordingly. Also, the thresholds might be different if you used another filter technique.

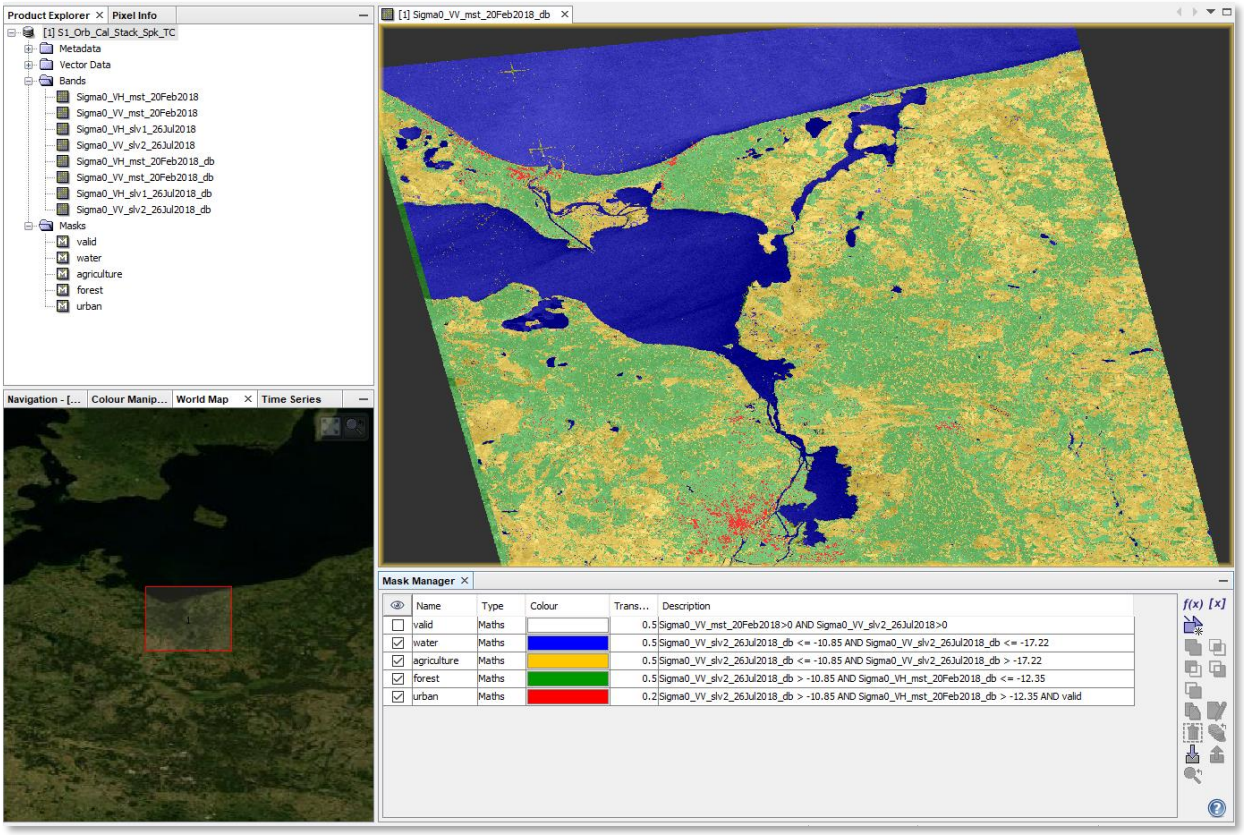


Figure 22: Implementation and result of the rule-based classification

To export the masks, for example to work with them in an external software packages such as QGIS, they must be converted into a single band. This can be done with the *Band Maths*. Deactivate the option “Virtual” and activate “Show masks” as shown to build the following statement:

```
IF water THEN 1 ELSE IF urban THEN 2 ELSE IF agriculture THEN 3 ELSE IF forest THEN 4 ELSE 0
```

After its creation, confirm with *File* > *Save product.* You can now open the created .img file in any GIS. Please also check the last chapter “

Creation of image features

Principal component analysis

To illustrate the information content of the newly generated stack, a principal component analysis (PCA) can be performed. It is a technique which identifies redundancies and differences in a stack of bands and transforms them into new uncorrelated bands (principal components) with have a maximized information content. Accordingly, the first principal component (PC1) represents the maximum proportion of variance from the original stack. The remaining variation is expressed by PC2, PC3 and so on. More information can be found here.

Open the Principal Component Analysis operator (under *Raster* > *Image Analysis*). Select the dB image bands as **Source bands** (Figure 23). Leave the other settings and start the processing with **Run**. Note that a PCA can take a considerable amount of time, especially for large datasets or stacks with many bands.

Figure 24 is an RGB image of the first three components and shows how they contain most of the variation of all images, including differences resulting from the two polarizations (VV and VH) and the backscatter differences between both acquisition dates (February and July). While urban areas are largely red (PC1=red, values which differ only little between all four input bands), because they are constantly high in VV and VH for both dates, water areas can be of various colors because they are different throughout the four input bands and therefore represented by different principal components (additive mixture of PC2=green and PC3=blue). All other land use or landcover classes have mixtures of red, green and blue: For example, croplands are purple because of a strong red component (volume scattering in VV and VH) and a strong blue component (strong difference between February and July). Generally, a PCA reveals interesting patterns in the data, but interpreting the result can be difficult.

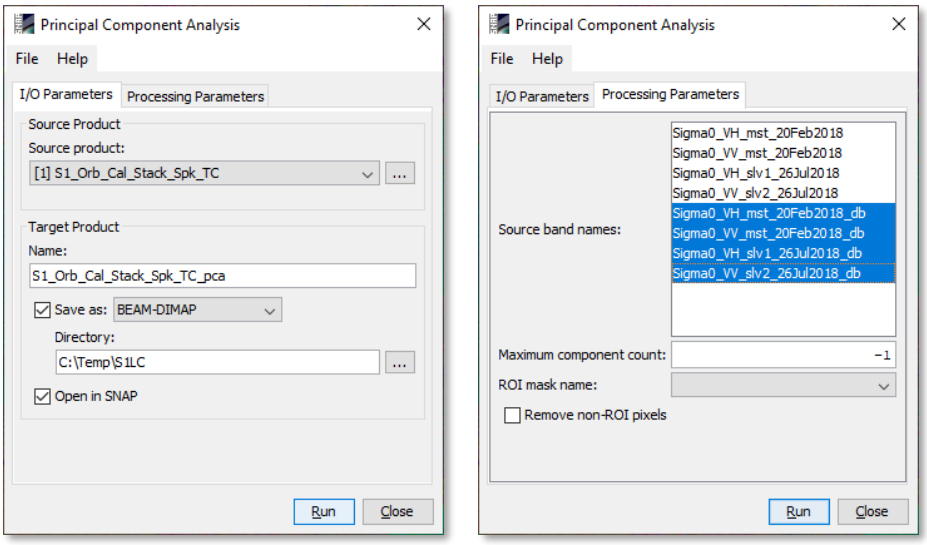


Figure 23: Principal component analysis

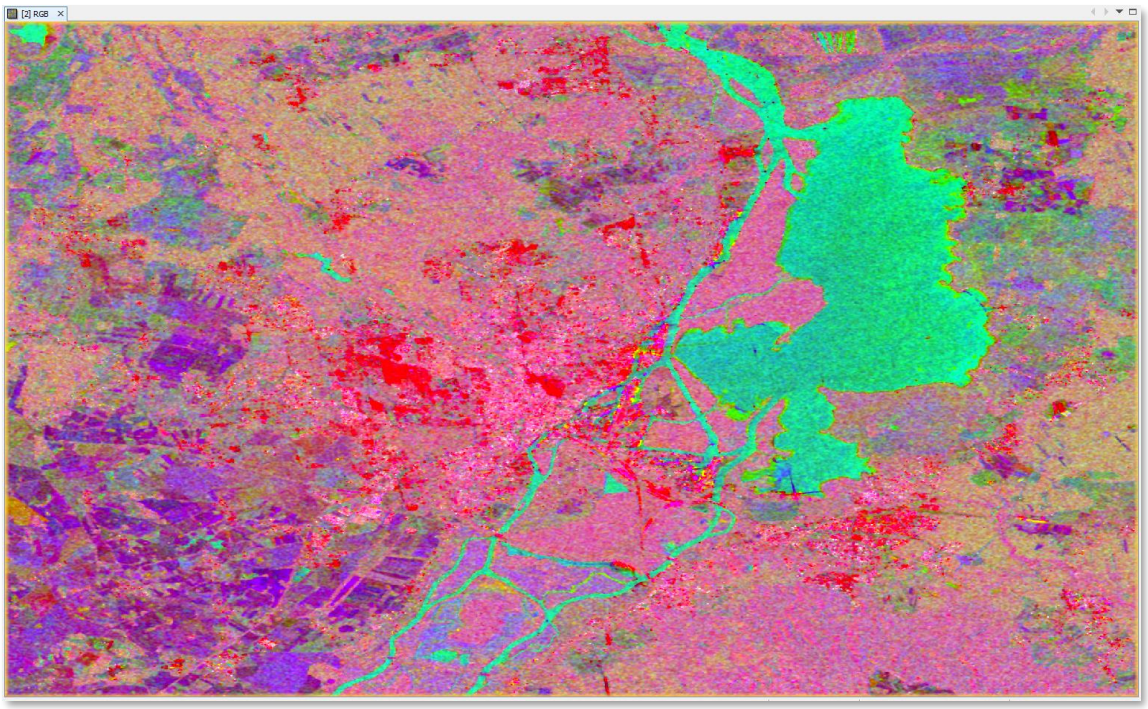


Figure 24: Principal Component Analysis of red=PC1, green=PC3, blue=PC5

Generation of textures

Image textures are metric derivatives which describe local spatial patterns of a greyscale image in a quantitative way. They are especially popular for SAR products, because most of them consist of only a limited number of bands (single or dual polarization). As image classifications based on a one or two-dimensional feature space are often not bringing the desired accuracies, image textures are a way to increase the number of input bands (Ulaby et al. 1986). Additionally, they are capable of describing the degree of speckle in different parts of the image and are therefore also contributing to a better separation of surface types. Image textures are implemented in SNAP in two ways:

- Right-click on a raster band > **Filtered band**: Allows to apply edge detection filters, statistical measures, non-linear and morphological filters and directly create the output as a virtual band.
- Menu > Raster > Image Analysis > Texture > **Grey Level Co-occurrence Matrix**: Allows to compute a set of image features which describe contrast, orderliness and local statistics. These were initially introduced by Haralick et al. (1973). A detailed introduction on the concepts and the mathematical derivation of the different texture measures is given in Hall-Beyer (2007): GLCM Texture: A Tutorial.

In this tutorial, GLCM textures are used as additional features for the later supervised classification. However, as the information content of the four polarizations is redundant to a larger degree and also many textural features produce similar results, the module is only applied to the VV_dB and VH_dB bands from July, and only the following textures are used: *Homogeneity*, *Energy*, *Maximum Probability*, *Entropy* and *GLCM Mean*. Since SNAP Version 8 it is possible to determine a **No Data value** outside the range of possible outcomes, so -9999 is a legit choice.

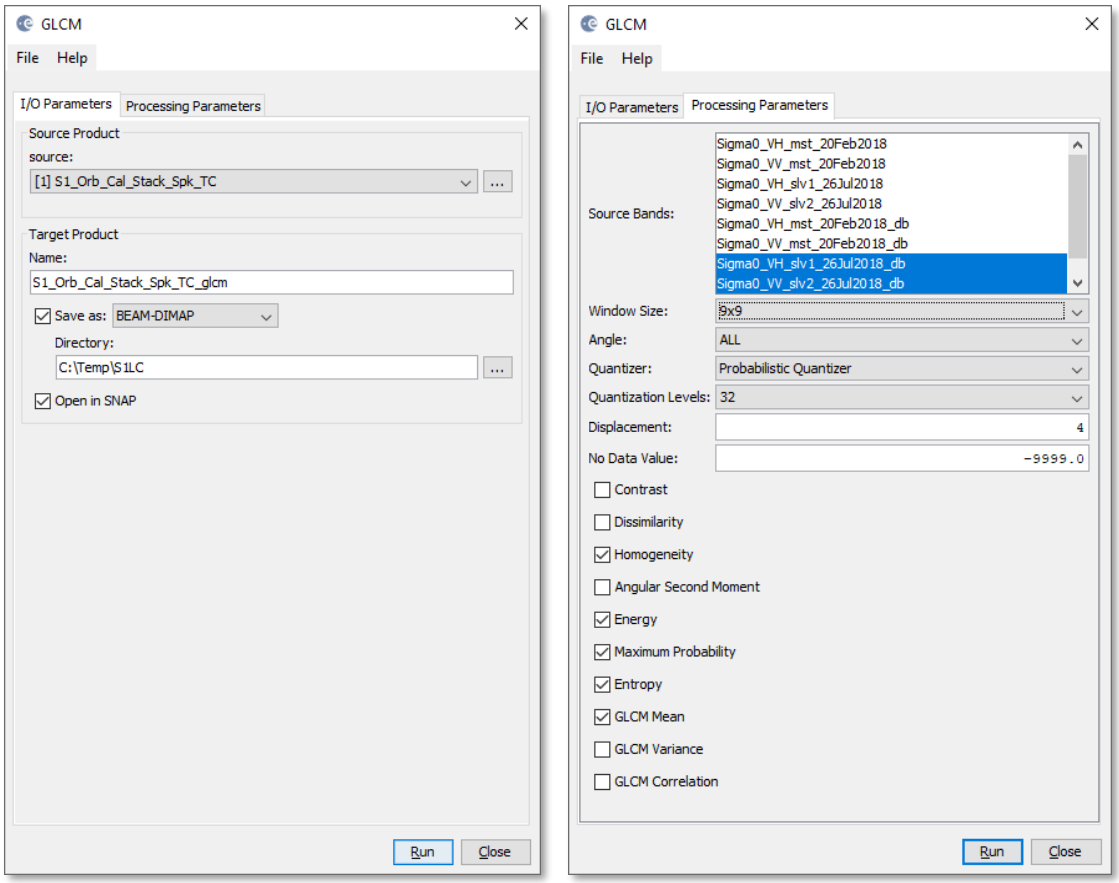


Figure 25: Calculation of image textures

The map in Figure 26 illustrates the information content of the generated textures by the use of RGB composites (right-click on the product > *Open RGB Image Window*) using the bands Homogeneity VV (red), Energy VH (green) and Entropy VV (blue).

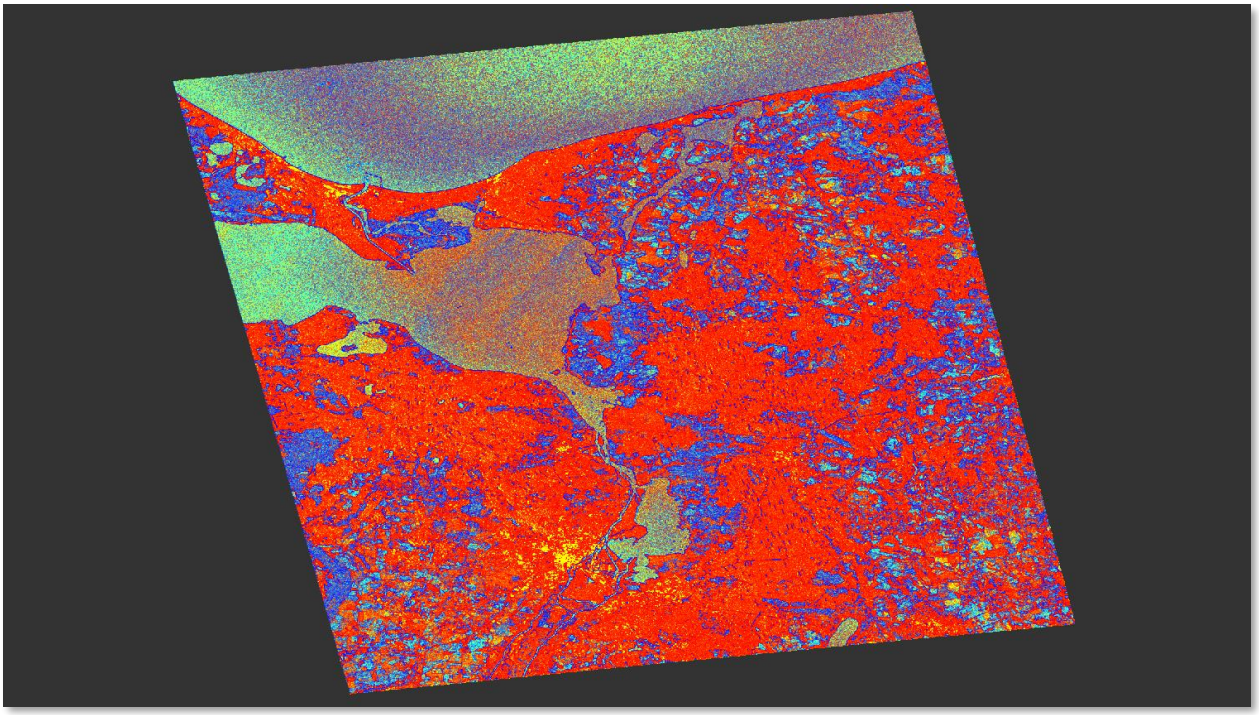


Figure 26: RGB image of different textures

Supervised Classification

Renaming of the bands

The classifiers in SNAP support the input of multiple products as input features for the classification. However, especially for SAR data there can be errors during the saving of the classifier as a text file which determines which bands were used to predict the input. This is because all raster bands created in this tutorial originate from either Sigma0_VV or Sigma0_VH.

As the input products were partially used for other classification modes in the previous chapters (e.g. rule-based classification), we recommend to copy all input products (.dim file and .data folder) into a new directory so you can modify them without losing previous results.

As shown in Figure 27 the input for the supervised classification should be the following products:

1. **Terrain Corrected stack**
   rename as suggested and only use dB raster
   also remove *rule-based_LULC* and the
   created masks from the previous chapter
2. **Principal component analysis**
   delete error band and rename as suggested
3. **Image textures**
   rename as suggested

Of course, the classification also works on any input data, but the later used Random Forest classifier requires a larger number of input bands, because it repeatedly uses random subsets of both the input bands and the training pixels to find the most robust thresholds based on the input features with the highest prediction importance (Breiman 1999).

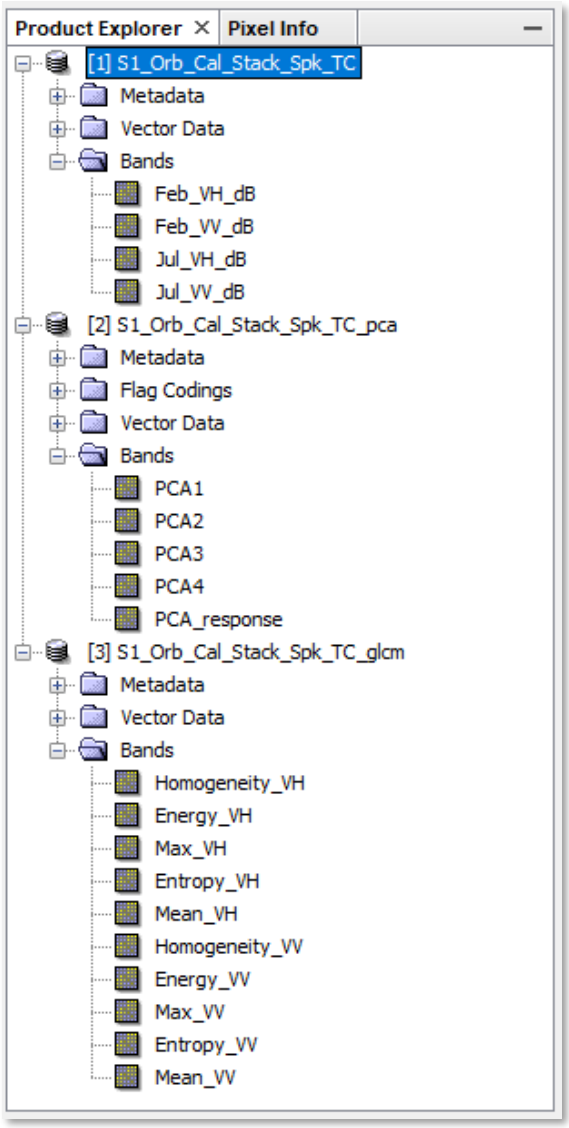


Figure 27: Renamed input bands for the classification

Very important:

- To make the changes permanent, select each of the input products where bands were renamed, and select *File* > *Save product*.
- Please visually control each of the input bands for correct contents before starting the classification.
- Any pixel which is defined as *no data* in the input features will not be classified.

Digitization of training areas

Supervised classification requires training data which defines the landcover classes you want to have in your later result and where they are located. This can be done in different ways:

- manually digitizing vectors which belong to a defined class
- importing a shapefile which has an attribute column storing the class name
- importing CSV files which define the geometries of a class

Examples for the import options (shp and csv) are attached to this document. They can be imported by selecting the main product (S1_Orb_Cal_Stack_Spk_TC) in the *Product Explorer* and then selecting from the Menu > *Vector* > *Import* > *ESRI Shapefile* or *Import from CSV*.

To manually digitize training areas for the different classes, click on **New Vector Data Container**  . A new image pops up which asks for a name of the class (Figure 28, left side). We enter “urban” and confirm with **OK**. The new vector container will appear in the *Layer Manager*  and also in the folder “Vector Data” in the corresponding product. Repeat this step for the following classes so that the *Layer Manager* will look like displayed in Figure 28 (right side). Note: *pins* and *ground_control_points* are automatically set as vectors, but they can be ignored for this task.

1. urban
2. agriculture
3. forest_broadleaf
4. forest_deciduous
5. grassland
6. water

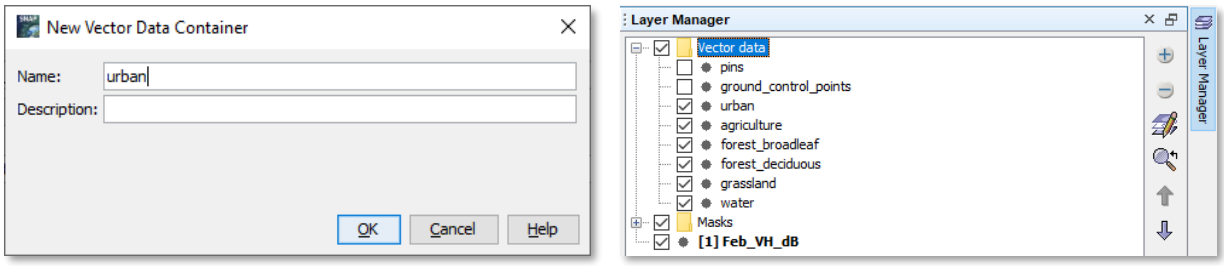


Figure 28: Creation and management of vector containers

If you want to digitize geometries of multiple classes, it is important that you first define them as vector data containers as previously explained and then select the folder “Vector data” in the the *Layer Manager*  , as displayed in Figure 28.

Once you select the **Polygon Drawing Tool**  and click in the map, you will be asked which of the vector containers you want to edit or fill with geometries (Figure 29). This allows to digitize multiple polygons which belong to the same class (container) and to effectively manage vectors of multiple classes.

Select a container and proceed with OK. You can then digitize polygons and finish with a double-click. You can modify or delete existing polygons using the Selection tool  (Figure 29). In this tutorial, a polygon is created covering a homogenous crop area.

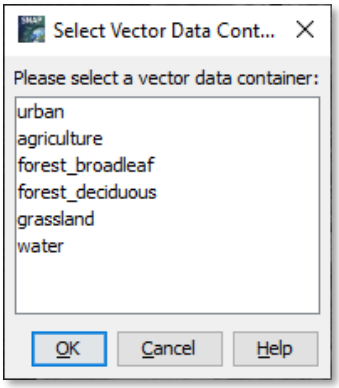


Figure 29: Class selection

To supply the Random Forest classifier with a sufficient number of training samples (5000 randomly selected pixels are used for each class per iteration) the result of the training data collection could look like this:

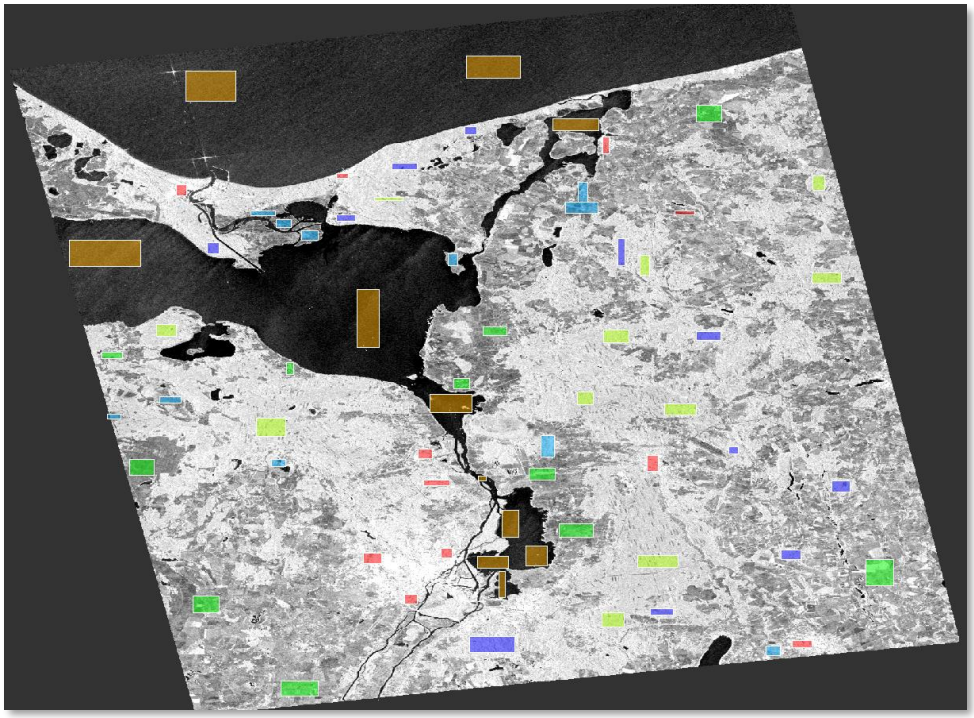


Figure 30: Digitized training areas

Conducting the classification

The Random Forest classifier (under *Raster* > *Classification* > *Supervised Classification*) is suitable for SAR data, because it is based on thresholding and can be applied to input features of different scaling (Sigma0 dB, but also PCA and texture bands). Load the three input products into the list as shown in Figure 31

In the second tab, enter `RF25` under **Train and apply classifier**.

If the option **Evaluate classifier** is selected, you will receive a summary on the training accuracy of the classifier. This is discussed in the next chapter.
If **Evaluate Feature Power** set is selected, you will get an additional summary on the contribution of each input raster for the classification. However, this makes the processing considerably longer.

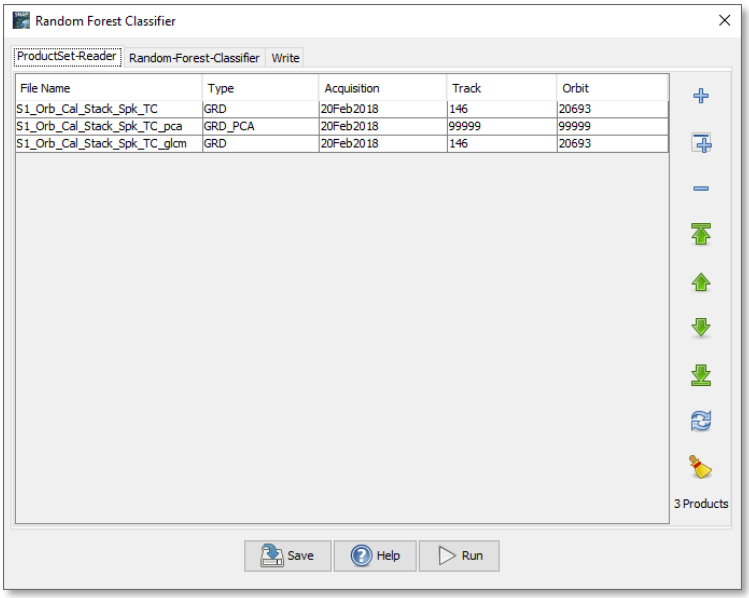


Figure 31: Random Forest - input data

Raise the **Number of trees** to 25. If you select no **Feature bands**, all bands in the stack will be used for the training of the classifier. This is important when you have bands that cannot be used for training (for example masks or the rule-based classification in case it was not deleted from the stack).

Click on **Run** to start the training and classification of the product. As there are many polygons and input bands, this can also take some time.

The Random Forest classifier now extracts all pixels under the training polygons and tries to find thresholds which allow to separate the different input types as best as possible.

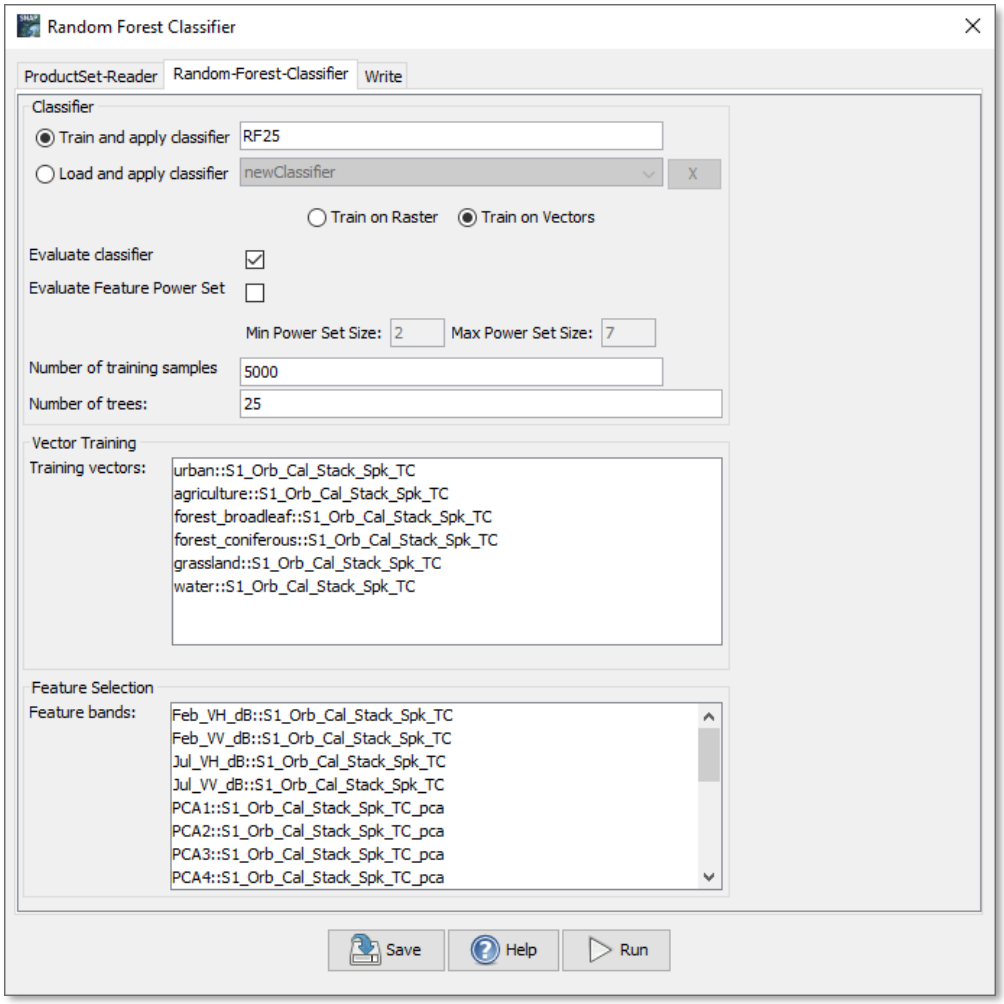


Figure 32: Random Forest - parameters

Once the processing is finished, a new product is created containing a band called `LabeledClasses`, which is the result of the classification and `Confidence` band as a side product of the Random Forest classification. To view the classification, double-click the bands. You will see that the pixels are now assigned to one of the training classes. You see the legend in the *Color Manipulation* tab, which also displays the relative frequency of each class (Figure 33 left).

However, the Random Forest classifier does not automatically assign a class to each pixel – some of the pixels remain transparent. This is for all pixels which could not be classified with a confidence higher than 0.5. This can have various reasons:

1. The pixel has statistical values which do not match any of the training data
2. The pixel was assigned to many different classes during the 25 iterations without a clear minimum
3. The training area representing this pixel was inhomogeneous and not representing the majority its actual areas.

Accordingly, the quality of the training data plays a significant role for the final result. Check the confidence band (Figure 33, right side, scaled from red [0] to green [1]) to identify those areas which have low confidence and try to modify the training data accordingly. Another option is to raise the confidence threshold in the band properties of `LabeledClasses`.

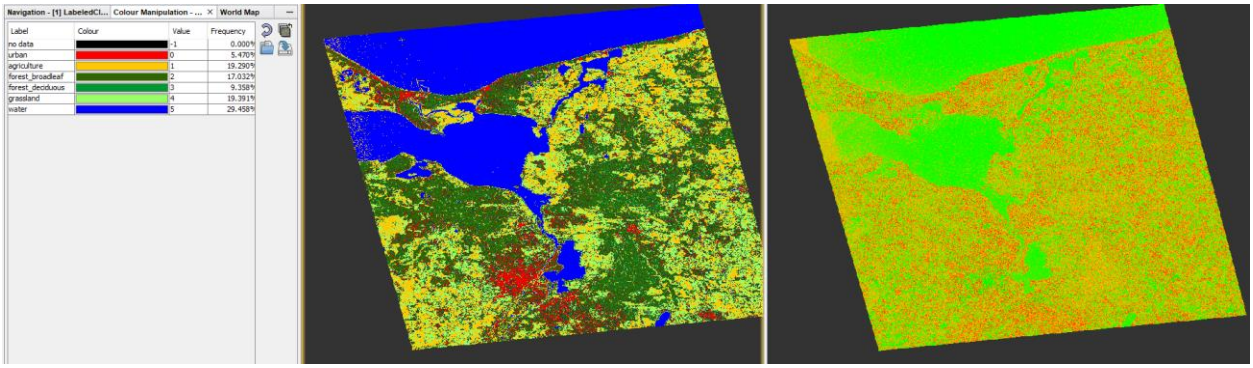


Figure 33: Result of the Random Forest classification (left) and confidence (right)

To display classes with low confidence values, open the **Band Properties** of the *LabeledClasses* dataset and change the Valid-Pixel Expression from `Confidence >= 0.5` to `Confidence >=0.2`. Pixels with low confidence are now no longer masked out. An example is given in Figure 34.

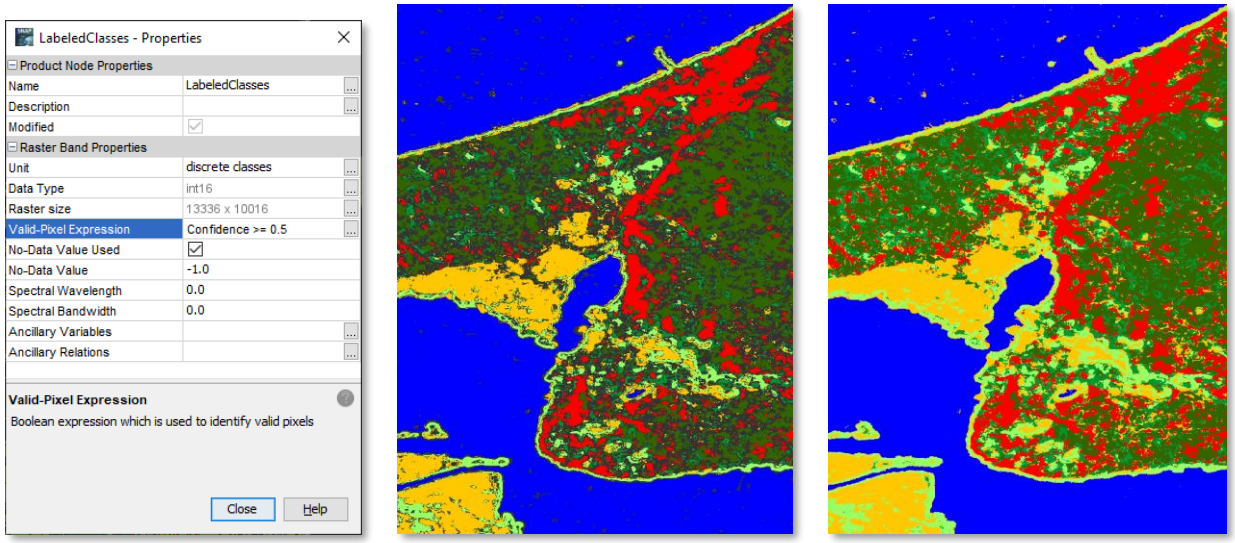


Figure 34: Classification with confidence threshold 0.5 (middle) and 0.2 (right)

Validating the training accuracy

If **Evaluate classifier** was selected in the Random Forest operator (Figure 32), a text file `RF_25.txt` will open during the classification process which gives information on how well the rule-set of the classifier was able to describe the training data based on the input feature bands (Sigma0 dB, PCA and textures). It does not validate the accuracy of the entire prediction, but only how well the training data was classified based on the hierarchical thresholding of the Random Forest classifier.

Table 2 presents the different accuracy metrics for the 6 trained classes

Table 2: Training accuracy

	urban	agriculture	forest_broadleaf	forest_deciduous	grassland	water
accuracy	0,971	0,967	0,970	0,954	0,984	1,000
precision	0,904	0,927	0,897	0,853	0,965	1,000
correlation	0,901	0,882	0,897	0,844	0,944	1,000
errorRate	0,029	0,033	0,030	0,046	0,016	0,000
TruePositives	772	725	773	730	782	833
FalsePositives	82	57	89	126	28	0
TrueNegatives	4082	4107	4075	4038	4137	4164
FalseNegatives	61	108	60	103	50	0

While accuracy, precision, correlation and errorRate are relative measures (1 = 100%). The numbers show that the Random Forest classifier was able to predict 95.4 % (deciduous forest) to 100 % (water) of the training pixels correctly using the SAR input raster features. For a more detailed interpretation of these metrics, please see Accuracy and precision and Sensitivity and specificity

The last four metrics refer to the statistical distribution of the training pixels per class (x).

A. **True Positive**: Number of pixels which are class x and have been assigned to class x.
High values indicate that most of the pixels of this class were **reliably** assigned.

B. **False Positive**: Number of pixels which are class x, but have not been assigned to class x.
High values indicate that this class is **overestimated** in the prediction.

C. **False Negative**: Number of pixels which are not class x, but have been assigned to class x.
High values indicate that this class is **underestimated** in the prediction.

D. **True Negative**: Number of pixels which are not class x and have not been assigned to class x.
High values indicate that only few pixels of this class were **missed** in the prediction.

		reference	
		+	-
prediction	+	**True Positive**	**False Positive**
	-	**False Negative**	**True Negative**

Lastly, the file reports: `Using Testing dataset, % correct predictions = 92.35` which means that the training accuracy is 92.3 % in total. Accordingly, the Random Forest was not able to predict 7.7 of the training areas correctly. This should be considered when applying this classifier to non-trained pixels (next chapter).

Validating the prediction accuracy

A high training accuracy, as observed in the previous chapter, does not grant a good result. It simply indicates how well the classification method (Random Forest) was able to replicate the classes from the training areas based on the available features (Sigma0 dB, PCA and textures). Accordingly, a low training accuracy a result from bad configuration of the classifier (not enough trees) an insufficient number of features or unsuitable training areas. In turn, some classifiers tend to over-fitting, which means that the training data was predicted to a very high accuracy, but the application to non-trained pixels can still be low. Therefore, accuracy assessment of the actual result has to be undertaken which is not based on the training areas, but on independently collected reference areas.

At the moment, SNAP has no automated accuracy assessment, but the number of True Positive, False Positive, False Negative and True Negative pixels can be easily assessed with the *Mask Manager*.

In this study, we use landcover information from the **Corine Landcover dataset of the year 2018** (CLC 2018) provided by the Copernicus Programme: It is available for large parts of Europe at a spatial resolution of around 100 meters with 45 classes of land use and landcover and can be freely downloaded under this address: https://land.copernicus.eu/pan-european/corine-land-cover/clc2018

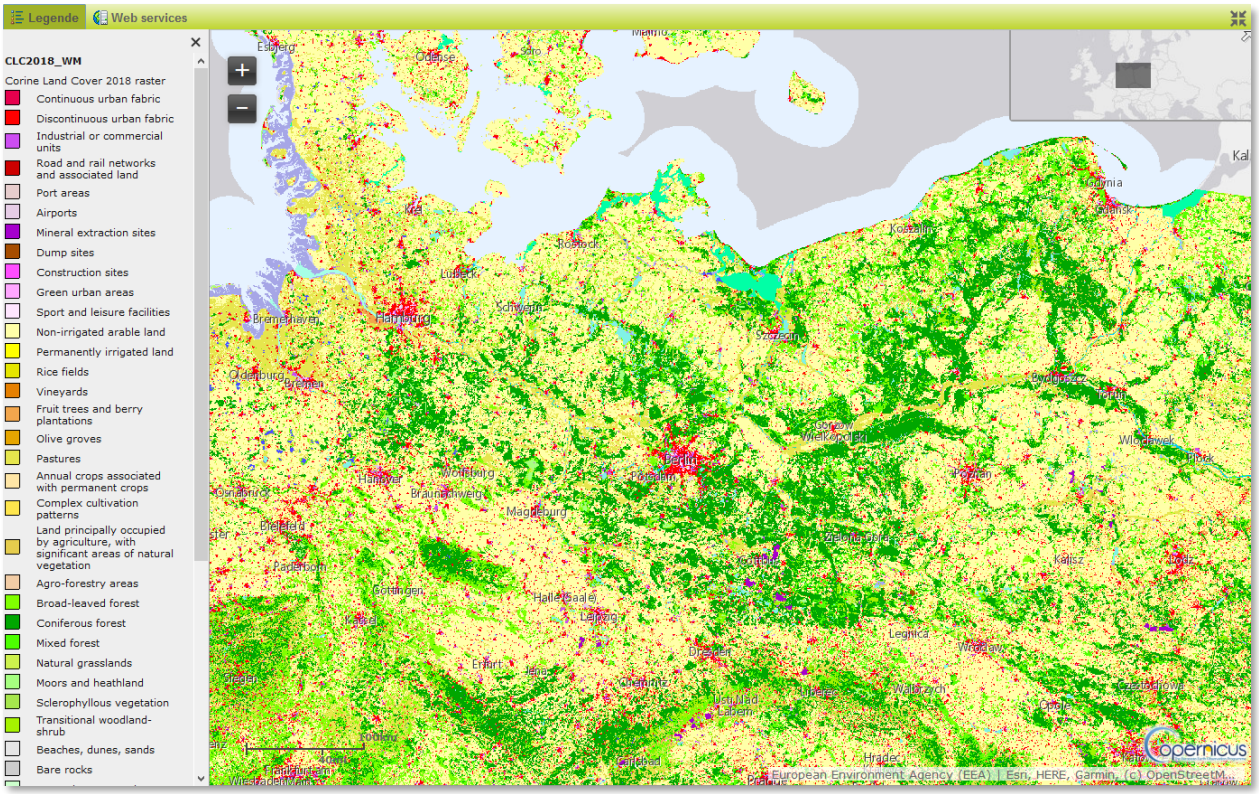


Figure 35: Reference dataset

Because of its high number of classes, it cannot not directly be used as a reference, because many of the defined classes which exist in the study area are not part of the classification.

For this reason, the 45 classes of the CLC2018 were re-classified to match the 6 classes of the predictions produced by the Random Forest supervised classification as shown in Figure 36.

In this tutorial, we use the class of **urban areas** to demonstrate how an accuracy assessment can be conducted with SNAP.

In a first step, the classes 1-11 were extracted from the CLC2018 dataset in the study area and converted to a polygon.

To compare them against the classification, this polygon is imported into SNAP via the menu > *Vector* > *Import* > ESRI *Shapefile*. The upcoming dialogue asks if the polygons shall be imported separately can be declined with **No**, because all polygons belong to the same class (urban).

The result is a mask which can be now used in the Mask manager (here as `glc_urban`).

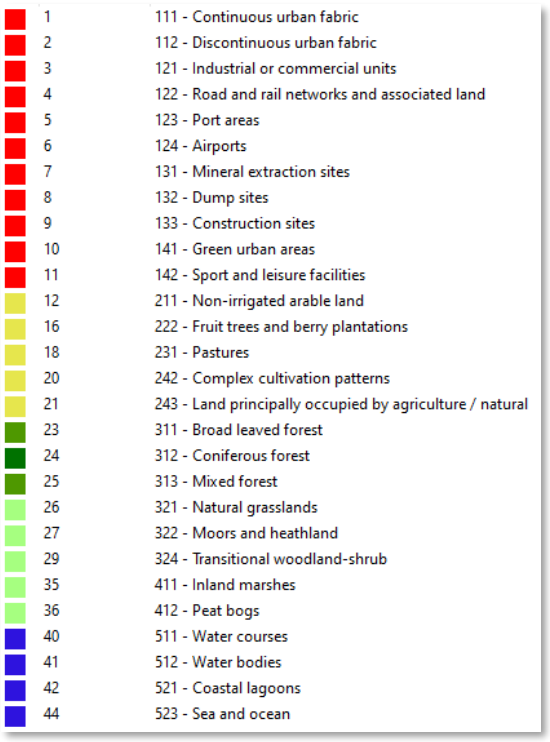


Figure 36: Re-classification of CLC2018 data

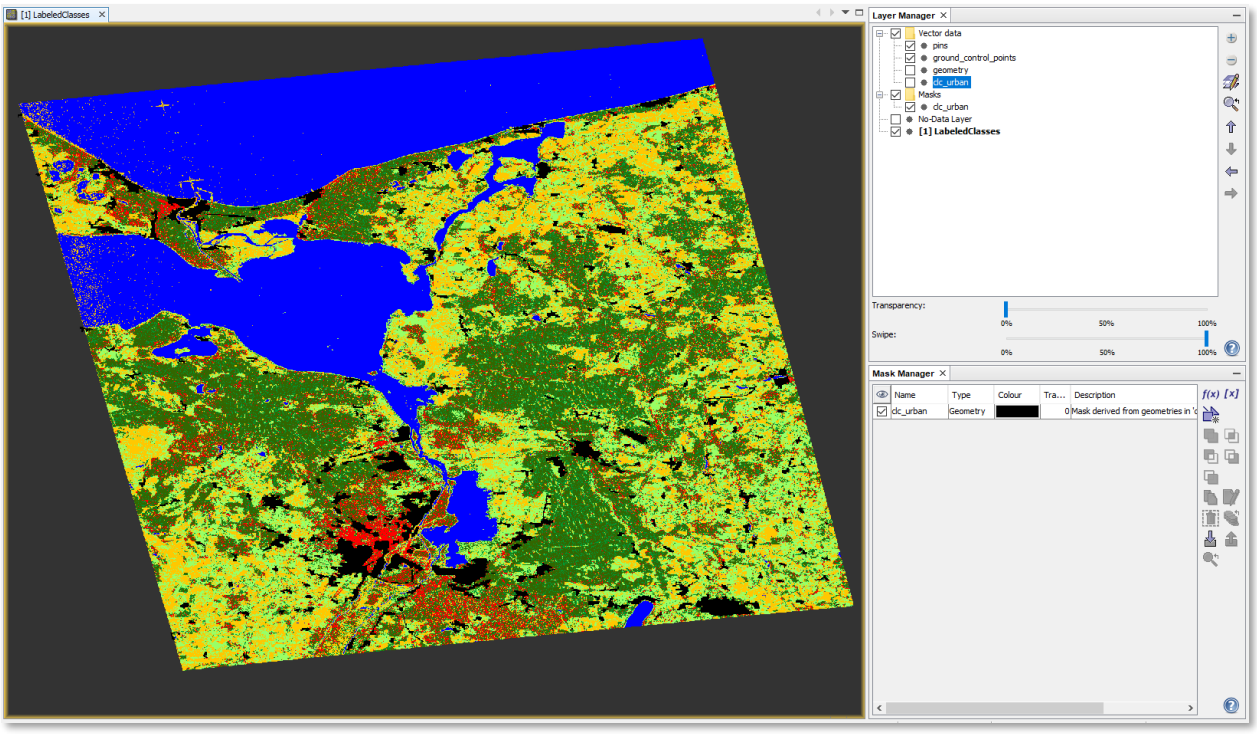


Figure 37: Classification result with imported 'urban' reference areas (black) from CLC2018

To get the prediction accuracy of the urban class, we need the following numbers:

A. Total number of pixels
B. **True Positive**: Pixels which are classified as urban and also are urban in the reference mask
C. **True Negative**: Pixels which are not classified as urban and are also not urban in the reference mask
D. **False Positive**: Pixels which are classified as urban, but are not urban in the reference mask
E. **False Negative**: Pixels which are not classified as urban, but are urban in the reference mask

We use the *Mask Manager* to identify these pixels in a first step. The *Mask Manager* allows to define logic or mathematic expressions to see which pixels fulfill certain conditions. A raster product (or RGB composite) must be opened in the current view for the tools of the Mask Manager to become active.

A new mask can be created with f(x). A new window (Figure 38, left) opens where a statement can be created in the *Expression* field. Make sure to activate **Show masks** to see the imported reference mask. Given that the imported mask of urban areas is called `clc_urban` and the urban class in the classified raster has the pixel value of `0` (Figure 33) the expression of the True Positive pixels is

```
LabeledClasses == 0 AND clc_urban
```

Click **OK** to confirm. A new mask occurs in the *Mask Manager* (Figure 38, right), you can name it all_pixels by double-clicking in the first column (*Name*). If you want to change the expression, double-click *Maths* to open the expression editor again.

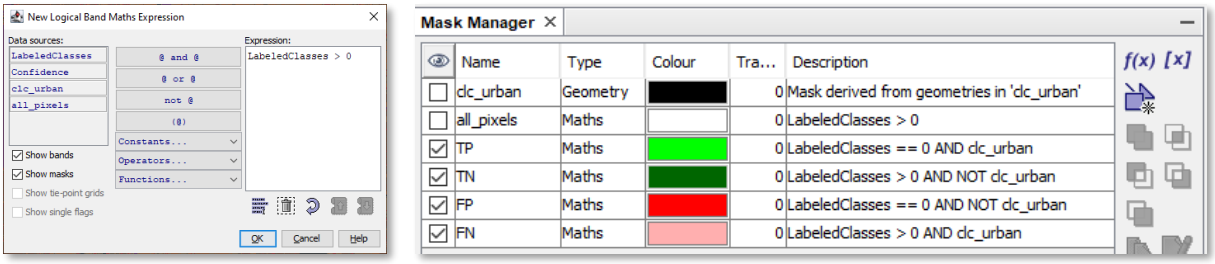


Figure 38: Using the Mask Manager to identify pixels

Table 3 lists the expressions for the four accuracy metrics.

Table 3: Assessment of accuracy metrics with the Mask Manager

Metric	Expression	Area [km²]
True Positive	`LabeledClasses == 0 AND clc_urban`	80
True Negative	`LabeledClasses > 0 AND NOT clc_urban`	5250
False Positive	`LabeledClasses == 0 AND NOT clc_urban`	356
False Negative	`LabeledClasses > 0 AND clc_urban`	175
		total: 5861

Once all masks are created, we can use the menu > *Raster* > *Mask* > *Mask Area* to get the number of pixels of each mask.

A new window opens asking for the mask to be analyzed (Figure 39, left). Select a mask and confirm with OK. After some seconds of computation, a report is shown listing the number of pixels and the area of the selected mask (Figure 39, right).

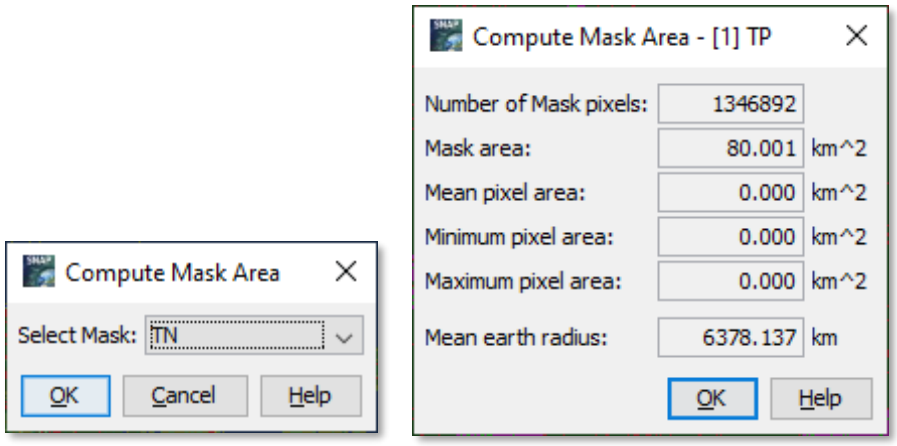


Figure 39: Retrieve mask areas

Based on these values, the prediction accuracies of the urban class can be calculated. A visual form of **True Positive**, **True Negative**, **False Positive** and **False Negative** is shown in Figure 40. The proportions of these four values indicate how accurate a defined landcover type was classified.

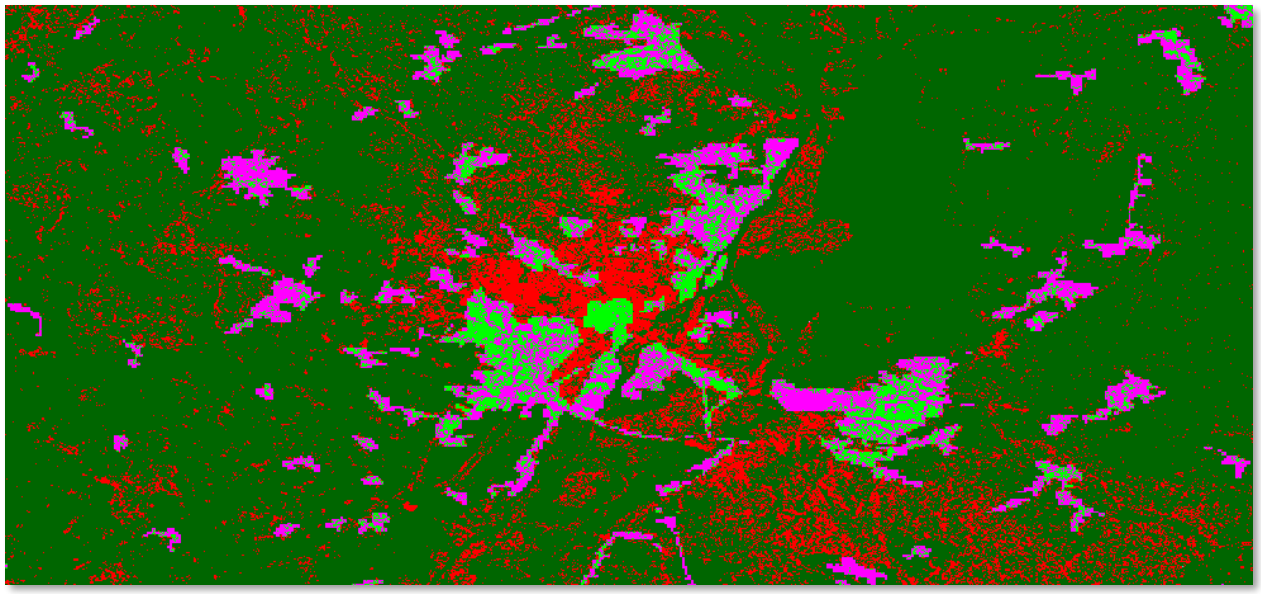


Figure 40: Pixel masks based on the expressions

$$\text{Overall classification accuracy} = \frac{\text{True Positive} + \text{True Negative}}{\textbf{all classified pixels}} = \frac{80 + 5250}{5861} = 0.9094 = \mathbf{90.94}\,\%$$

The **overall accuracy** determines how much the urban classification is correct in general. 90.94 % seems a good result at first sight, but as you can see in Figure 40 and the equation, this high accuracy is mostly caused by the large amount of True Negative pixels (correctly classified non-urban landcover). The actual urban areas are partly overestimated and underestimated. Therefore, the producer and user accuracies are calculated as more specific measures per class.

$$\text{producer accuracy} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}} = \frac{80}{80 + 175} = 0.3137 = \mathbf{31.37}\,\%$$

The **producer accuracy** is a measure for the probability that urban landcover in the study area is classified correctly. It means that only 31 % of the urban areas were identified by the classifier. The large number of False Negative pixels leads to a high **error of omission**, that means many urban areas were missed by the classifier (pink pixels). Accordingly, these areas were strongly underestimated.

$$\text{user accuracy} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}} = \frac{80}{80 + 356} = 0.1834 = \mathbf{18.34}\,\%$$

In turn, the **user accuracy** is a measure for the reliability of the classified pixels. It means that only 18.34 % of all pixels which were classified as urban are correct. The large number of False Positive pixels leads to a high **error of commission**, that means that many pixels were also classified as urban while they are of different landcover in the reference dataset (red pixels). In these regions, urban areas were strongly over-estimated.

To conclude, this example underlines the necessity for independent validation datasets and shows that

- high training accuracies can still result in bad results (prediction accuracies)
- high overall prediction accuracies can be misleading, because they do not really reflect overestimation and underestimation of classes which hold a small share of the study area.

Finally, it is also the quality and processing of the reference data which can have an effect on the estimated accuracies. For example, the aggregation of classes as shown in Figure 36 can contain false assumptions and therefore have a negative impact on the accuracy.

Notes on exporting results

If you want to use the classified land use produced in SNAP in other programs, such as QGIS or ArcMap, please consider the following points. Usually, there is no need to export it to another file format, because the BEAM DIMAP product already contains a fully compatible raster inside the .data folder of the classified product. As shown in Figure 41, each raster in SNAP consists of two files: img and hdr. The first is the actual raster (geocoded as specified in the Range Doppler Terrain Correction) and the hdr file contains metadata. That means you can directly open LabeledClasses.img in QGIS or ArcMap. If you need a special file format you can still use File > Export but some of the formats require a change in the data type (for example from16 to 8 bit) which potentially alters the pixel values.

Please also be aware that the class names and colors assigned by SNAP are only stored in the .dim file, so these are not part of the raster. Accordingly, you have to give the colors in any software again based on the pixel coding. For example, in QGIS, this can be done by switching from a “singleband grayscale” to a “paletted/unique values” symbology (Figure 41). An example is given in Figure 42.

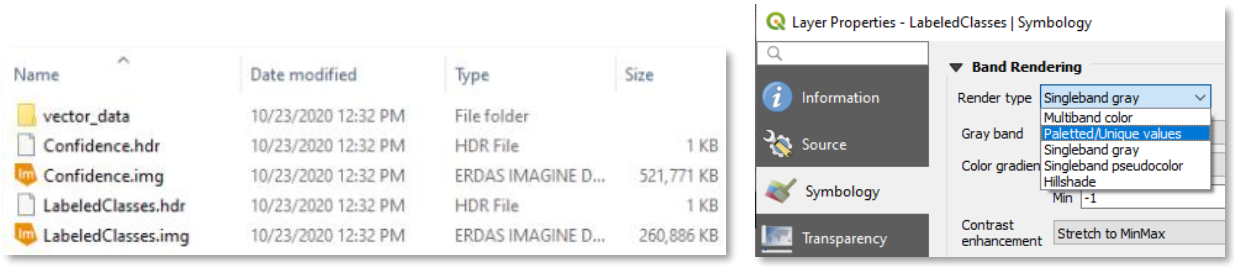


Figure 41: Files inside the data folder of the BEAM DIMAP format (left) and selection of unique values representation in QGIS (right)

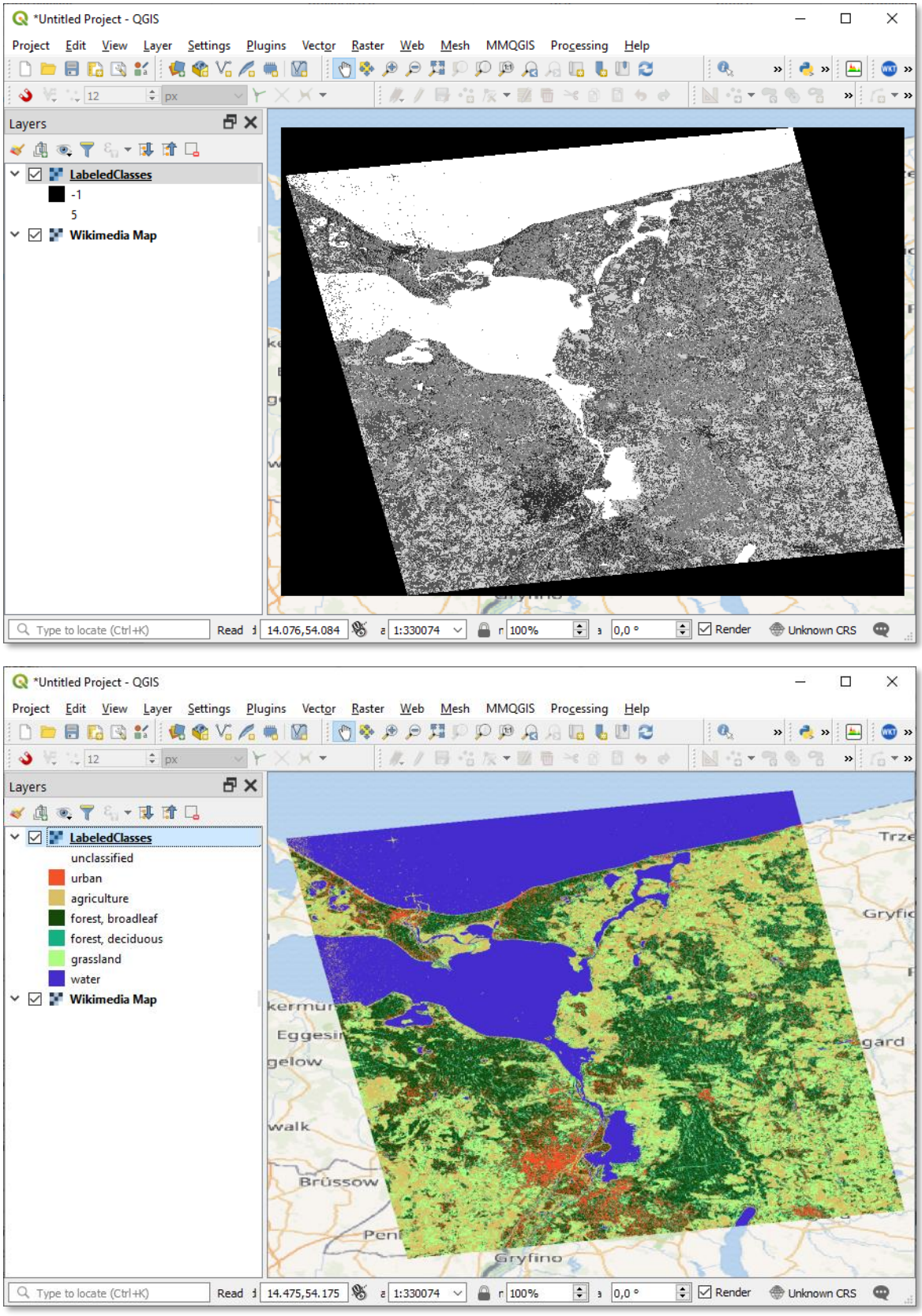
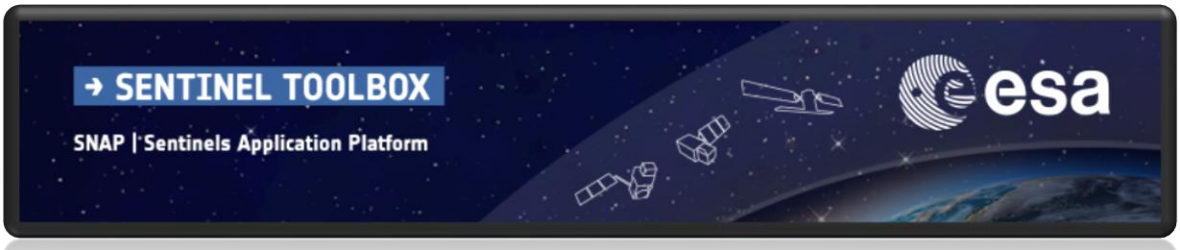


Figure 42: Imported img file of the classification before (top) and after (bottom) assignment of colors



For more tutorials visit the Sentinel Toolboxes website

http://step.esa.int/main/doc/tutorials/

Send comments to the SNAP Forum

http://forum.step.esa.int/