ADVANCED HYPERSPECTRAL DATA ANALYSIS PLUGIN [AVHYAS]

TECHNICAL REPORT 1.0

Hyperspectral Techniques Development Division
AMHTDG | EPSA | Space Applications Centre | ISRO

TABLE OF CONTENTS

Contents

Contents 1

Acknowledgement 1

Introduction 2

Installation 3

1 ATMOSPHERIC CORRECTION MODULE 6

2.1 BASIC TOOLS MODULE 9

2.1 Sensor Utility 10

2.1.1 ASD UTILITY 10

2.1.2 Generate SRF From FWHM 12

2.1.3 Hx Spatial Spectral Resample 13

2.1.4 AVIRIS-NG Spatial Spectral Resample 16

2.2 Data Subsetting 19

2.3 Spectral Plot 20

2.4 Scatter Plot 22

2.5 Scaling 23

2.6 Spectral Library 25

2.7 ROI Seperability 26

3 PREPROCESSING 34

3.1 Dimensionality Reduction 35

3.1.1 PCA-Forward 35

3.1.2 PCA-Inverse 38

3.1.3 MNF- Shift Difference 40

3.1.4 MNF- NAPC 42

3.1.5 Kernel-PCA 43

3.2. General Purpose Utility 45

3.2.1. Sparse Coding Based Cloud Removal 45

3.3 Feature Extraction 48

3.3.2 Spectral Gradient Filter 49

TABLE OF CONTENTS

3.3.3 Extended-Morphological Filters ____ 50
3.4 Feature Transformation ____ 52
3.4.1 Fit Transform ____ 53
3.4.2 Apply Transform ____ 54
3.4.3 Continuum Removal ____ 55
3.5 Savitzky Golay Filter ____ 56
4 DATA QUALITY ANALYSIS ____ 59
4.1 Bad Band detection ____ 60
4.2 Whitening ____ 60
4.3 Covariance or Correlation matrix ____ 61
4.4 Noise Estimation ____ 63
4.5 De-noising ____ 66
4.6 De-striping ____ 66
5 UNMIXING MODULE ____ 68
5.1 Material Count ____ 71
5.1.1 HFC-VD ____ 71
5.1.2 HySIME ____ 72
5.2 End Member Extraction ____ 75
5.2.1 Pixel Purity Index (PPI) ____ 75
5.2.2 NFINDR ____ 77
5.2.3 ATGP ____ 80
5.2.4 Vertex Component Analysis ____ 83
5.3 Abundance Estimation ____ 85
5.3.1 Linear Abundance Estimation ____ 85
5.5 Iterative Scatter Ploy Mixing Visualization ____ 91
5.6 Unmixing Result Viewer ____ 92
6 CLASSIFICATION MODULE ____ 94
6.1 Supervised Classification ____ 95
6.1.1 Spectral mapper ____ 95
6.1.2 Random Forest Classification TOOL ____ 100

TABLE OF CONTENTS

6.1.3 K Nearest Neighbour Classification TOOL 109
6.1.4 Support Vector Machine Classification TOOL 118
6.1.5 Classification Wokflow 130
6.1.6 Predict Classification 135
6.1.7 Cross Validation Classification 138
6.2 Unsupervised 141
6.2.1 K-Means Clustering TOOL 142
6.2.2 Aggloemtetric Clustering TOOL 148
6.2.3 DBSCAN Clustering TOOL 155
6.3 Segmentation 158
6.3.1 Spatio-Spectral Segment TOOL 158
6.3.2 Split and merge Segmentation TOOL 162
7 DEEP LEARNING MODULE 167
7.1 Deep Learning 168
7.2 Inference Deep Learning 181
8 REGRESSION MODULE 183
8.1 Regression workflow 184
8.2 Predict Regression 190
8.3 Cross Validation Classification 192
9 FUSION MODULE 194
9.1 Coregistration 195
9.2 Coupled non negative matrix Factorization 199
10 INDICES 203
11 INTERACTIVE VISUALIZATION OF VEGETATION 204

ACKNOWLEDGEMENT

Acknowledgement

We would like to express our sincere gratitude and thanks to Director SAC, Shri D K Das, and Deputy Director EPSA, Dr. Raj Kumar, who has given us the opportunity to work in this particular toolbox. Our deepest and heartfelt thanks also goes to AVIRIS-NG team and all the collaboration agencies for their immense support. We would also like to extend our sincere gratitude to SIPG, SEDA and entire EPSA for providing us all the required data, resources and help.

INTRODUCTION

Introduction

AVHYAS stands for Advanced Hyperspectral Data Analysis plugin. It is a free and open source plugin developed in QGIS environment. It is primarily developed to analyze Hyperspectral Imaging data (HSI) as well as Field Spectroradiometer data. However, other than that the other main focus of AVHYAS was to have a platform ready for handling future imaging spectrometer sensors data of ISRO. The designed guidelines basically have three major goals (i) to provide state-of-the-art processing techniques and algorithms for analyzing high dimensional spectral and temporal remote sensing data (ii) to provide advanced techniques and algorithms currently not available in any COTS packages either developed in-house or implemented through literature survey and (iii) to provide a graphical user interface (GUI) so that any users can used it.

The purpose of integrating AVHYAS as a plugin in QGIS is to utilize the existing GIS visualization capabilities of QGIS and further enhances its capabilities to advance techniques for HSI data processing. Apart from readily available QGIS GDAL capabilities for reading and writing satellite/airborne data, AVHYAS enriches it more with other modules like eg. Tensorflow for deep learning, SPAMS for sparse based modeling, Scikit-Learn for machine learning, AROSICS for coregistration etc. AVHYAS bridges the gap between existing QGIS capabilities with other advanced algorithm available in literature for different types of applications using hyperspectral data. AVHYAS is written purely in Python and as python is a portable language, hence it has cross platform support.

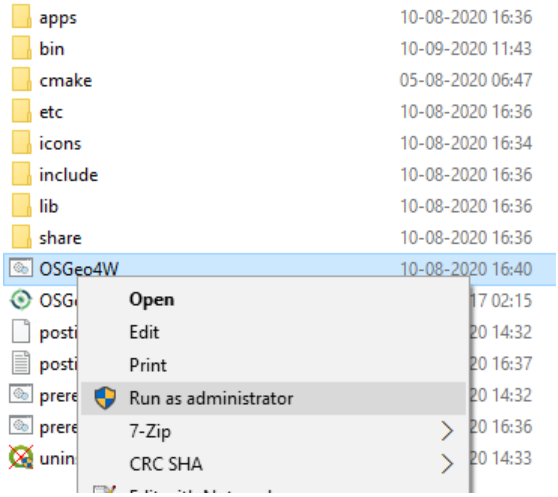
Installation

Installing AVHYAS V0.1 (Beta)

AVHYAS requires QGIS Version 3.0 or higher and it also has dependency on some additional python packages which needs to be installed manually before proceeding with installation process.

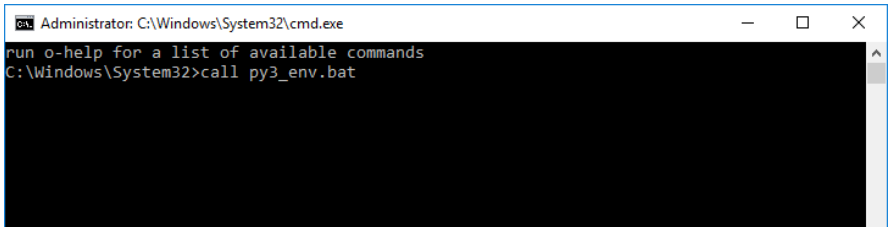
Pre-Required Packages installation:

In case QGIS is not installed you can get QGIS from this link **qgis**. Kindly go to the install directory of QGIS 3.xx in Program Files, the folder structure should look similar to the one given below.

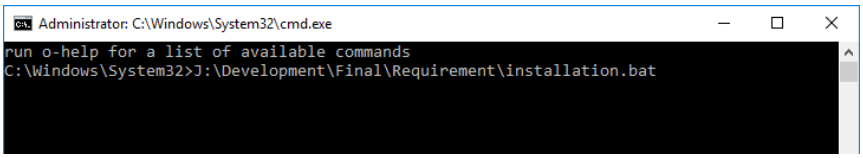


C:\ →Program Files → QGIS 3.xx → OSGeo4W Shell → Right-Click → Run as administrator

Activate the Python 3 Environment calling:



Kindly enter the full path of the Requirement directory where the installation.bat file is located.

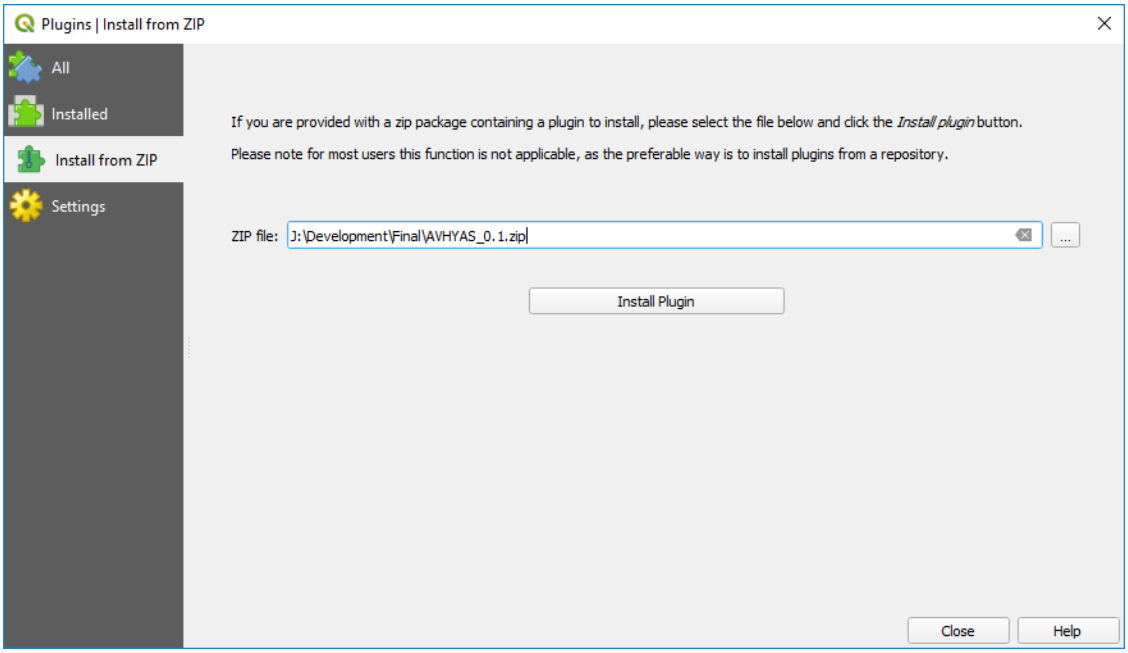


Press enter and wait till the entire installation process is complete. It will take a bit of time to complete.

Install AVHYAS Plugin

Start QGIS 3 and open Plugins > Manage and Install Plugins > Install from ZIP.

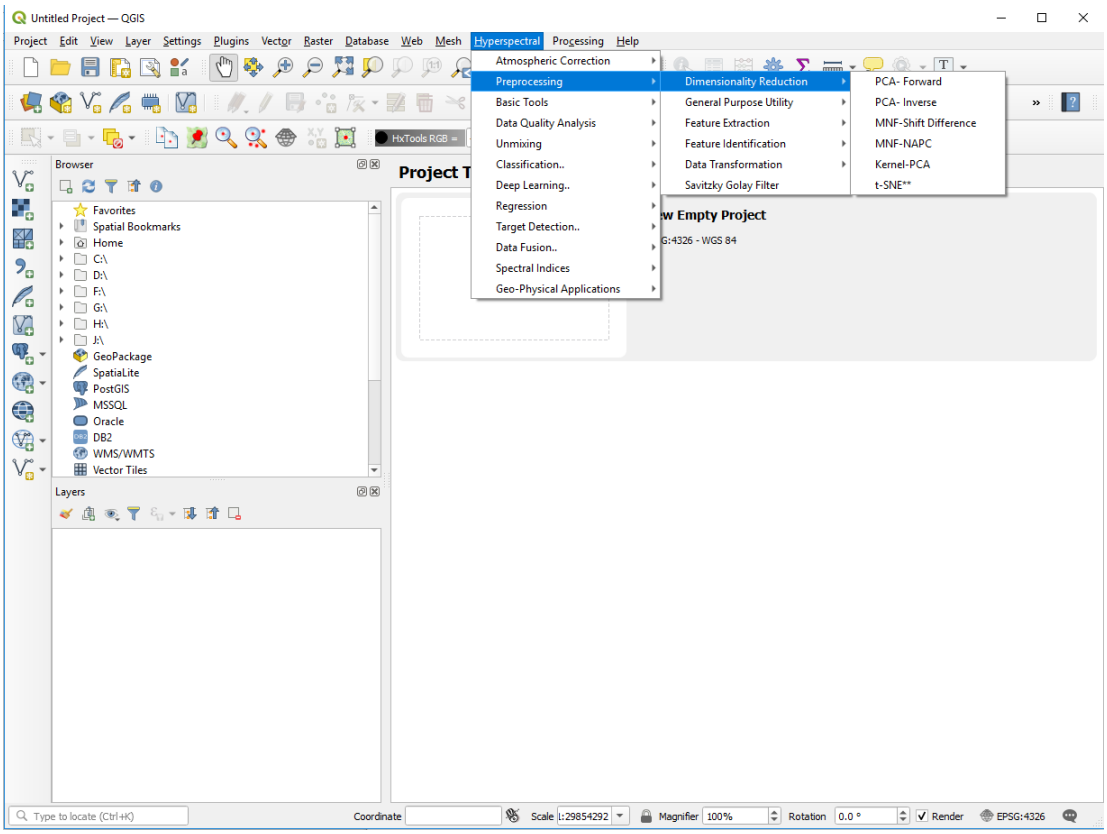
Select the AVHYAS_0.x.zip and start *Install plugin*



On finishing installation process, a new menu will be added in QGIS with the name Hyperspectral. In case if the menu does not appear or some error happen kindly check the log file by Right-Clicking and select log-message from QGIS and report the same accordingly to the development team for rectification of the problem.

If the installation is successfully done. A menu as shown below will appear

INSTALLATION



1 ATMOSPHERIC CORRECTION MODULE

QGIS 3.XX → Hyperspectral→ Atmospheric Correction → AVIRIS-NG Atmospheric Correction

User Interface

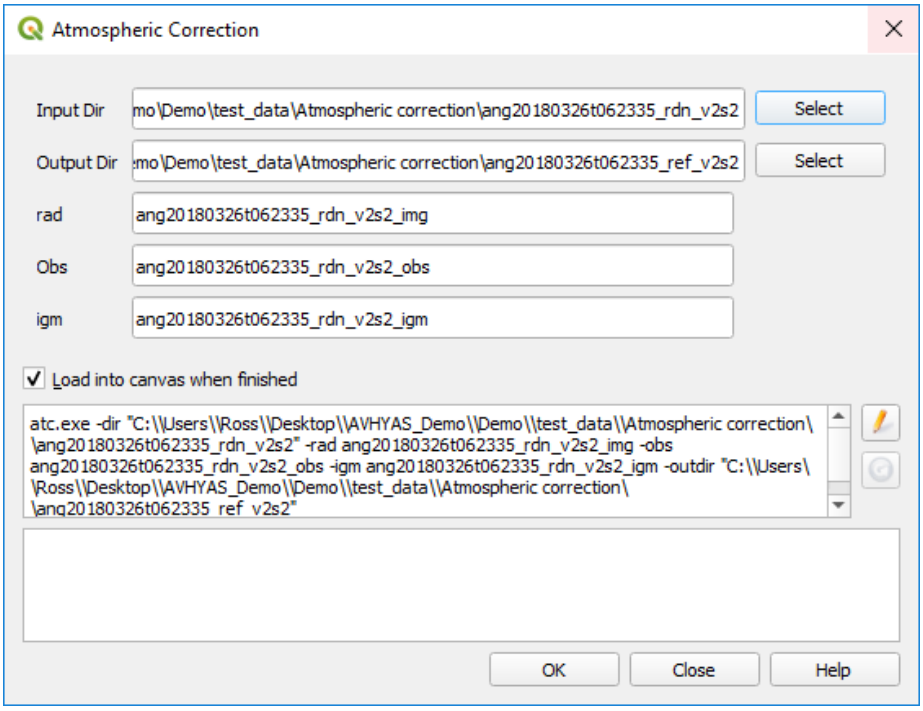


Figure. User interface for Atmospheric Correction

Input Arguments

Input Directory: Specify the input directory of the radiance Level 1 data. The input folder should have three data in it with extension obs, igm, img. Other parameters will be automatically populated once a valid input directory is specified.

Output Directory: Specify the output directory to save the reflectance directory.

rad: This field is non updatable field. File containing at-sensor radiance data. Should be in the same directory/folder as input directory.

obs: This field is also non updatable. This file contains observation geometries like azimuth, zenith for sensor and solar and also the path length, Earth-Sun distance etc. Should be in the same directory/folder as input directory.

igm: This field is also non updatable. This file contains surface elevation data at each pixel.

I/O PARAMETERS

TYPE	DATA FORMAT	DATA TYPE	VALUE RANGE	REMARK
Input				
Input directory	Folder path			Input Radiance Folder
Rad	ENVI			Radiance File
Obs	ENVI			Observation File
Igm	ENVI			Input file geometry
Output				
Output Folder	Folder Path			

Overview

Applications of high-spatial resolution imaging spectrometer data acquired from the AVIRIS-NG require a thorough compensation for atmospheric absorption and scattering. Retrieval of critically important atmospheric parameters i.e. water vapor and aerosol optical depth (AOD) over land is done before atmospheric correction. The dark dense vegetation method and radiative transfer modelling is used to derive AOD. Estimation of perceptible water vapor is carried out using short-wave hyperspectral measurements for each pixel. A differential absorption technique (continuum interpolated band ratio) has been used for this purpose. Further, these parameters were used to derive 'atmospherically corrected surface reflectance for land pixels, assuming horizontal surfaces having Lambertian reflectance.

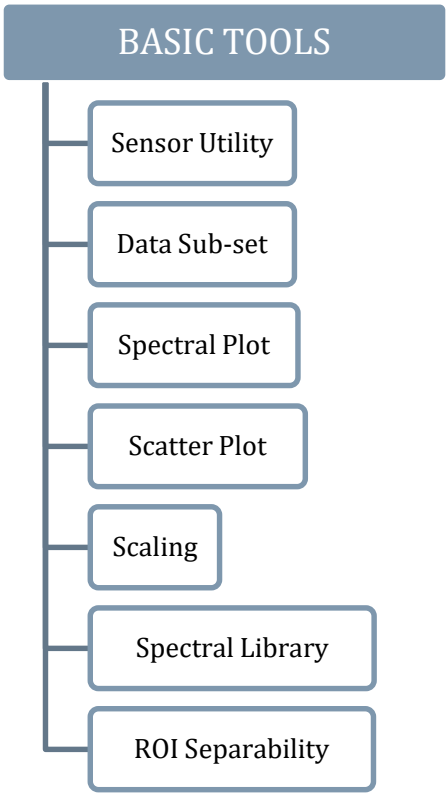
This utility is for converting Level-1 AVIRIS-NG data to Level-2 (Reflectance data). This module contains a complete standalone version of the operational model use to convert radiance data to reflectance data. All the necessary auxiliary looks up tables has been bundled along with AVHYAS. The module requires a valid radiance data path before proceeding with atmospheric correction. If valid path has been provided the other parameters will be loaded automatically. Two auxiliary files are required for the modules to work. One is the obs file which is an observation file containing geometry information and illumination parameters. The other is igm file which stands for input geometry file that has parameter information of elevation, North-South projections etc. If valid files are provided and clicking on Ok button will call the

1 ATMOSPHERIC CORRECTION MODULE

atmospheric correction module and process the data. It is generally a lengthy operation so user is advice to be patient.

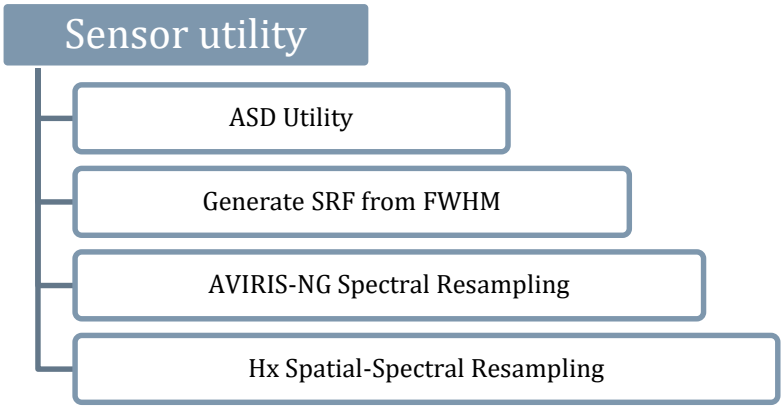
2.1 BASIC TOOLS MODULE

2.1 BASIC TOOLS MODULE



2.1 BASIC TOOLS MODULE

2.1 SENSOR UTILITY



2.1.1 ASD UTILITY

QGIS 3.XX → Hyperspectral→ Basic Tools → Sensor Utility → ASD Utility

User Interface

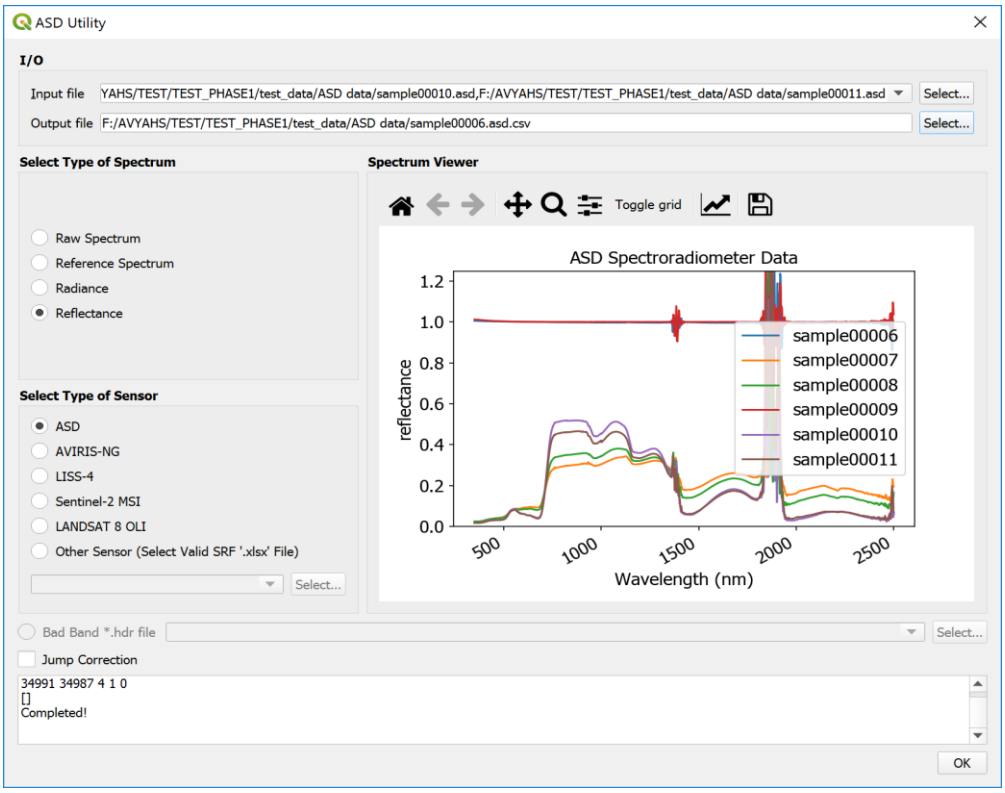


Figure. User-Interface for Spatio-Spectral Segment

2.1 BASIC TOOLS MODULE

Input Arguments

Type of Spectrum: Select type of spectral data stored in .asd file.

Type of Sensor: ASD data will be saved based on the selected sensor types (algorithm employs Spectral-Response Function of the selected Sensor)

Bad-band: Remove the bad band from the resampled version of AVIRIS-NG data using the bad band list available with ".hdr" file.

Jump Correction: Offset error of SWIR1 and SWIR2 sensor arrays of ASD sensor can be corrected by selecting this option.

I/O PARAMETERS

TYPE	DATA FORMAT	DATA TYPE	VALUE RANGE	REMARK
Input				
Input File	*.asd			ASD spectroradiometer files
Bad band file	*.hdr file associated with AVIRIS-NG data			
Output				
Output File	*.csv			Save input files to single *.csv file format

Overview

ASD utility tool is capable of reading the ASD spectroradiometer data (*.asd) and display the associated data types (raw data, Radiance, reflectance and white-reference). It can be used to resample the ASD spectrum based on the spectral-response-function of different sensors (AVIRIS-NG, LISS-4, Senstinel-2 MSI, LANDSAT8-OLI). Bad-band list associated with the AVIRIS-NG data can be used for removing the bad-bands from the resampled-ASD-spectrum.

Jump/Splice Correction: The so-called splice/Jump correction eliminates the gaps in the signal between the domains of the different detector arrays. Critical transitions are located at λ=1000nm and λ=1800nm. The objective of the splice correction is to compensate the difference between the reflectance R1000nm and R1001nm by adapting all values from 1001nm upwards to the level of those to 1000nm.

$$R_{i,splice} = \begin{cases} R_i\,, & i \in [1;1000] \\ R_i - f_{1000}\,, & i \in [1001;1800] \\ R_i - f_{1800}\,, & i \in [1801;2500] \end{cases}$$

$$f_{1000} = R_{1001} - (2 \cdot R_{1000} - R_{999})$$
$$f_{1800} = R_{1801} - (2 \cdot R_{1800} - R_{1799})$$

f_{1000} and f_{1800} represent biases which are added to the original values and – depending on their algebraic sign – either increase or decrease all further reflectances. $R_{i,splice}$ are the corrected new values with smooth splices at the critical transition wavelengths.

2.1.2 GENERATE SRF FROM FWHM

QGIS 3.XX → Hyperspectral→ Basic Tools → Sensor Utility → GENERATE SRF FROM FWHM

User Interface

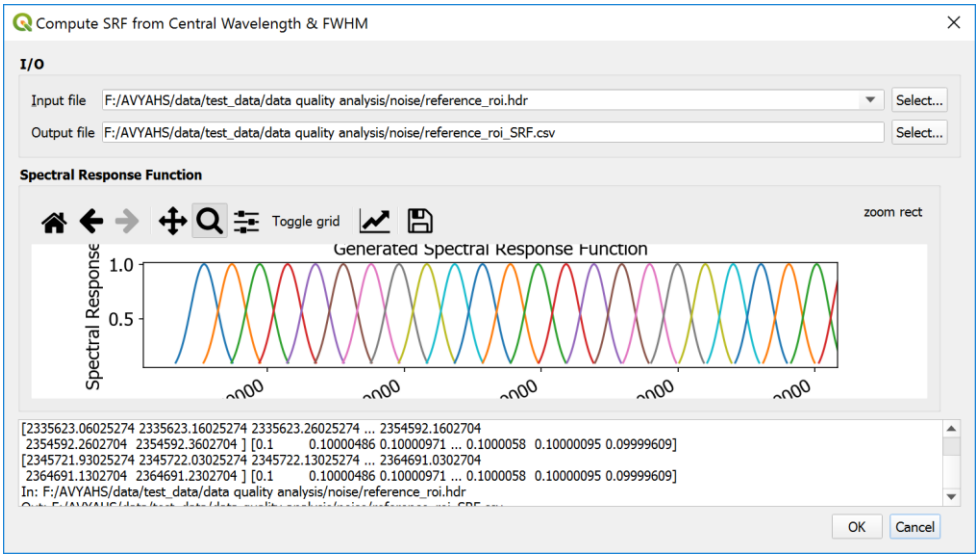


Figure. User-Interface for Generating SRF from FWHM

Input Arguments

I/O PARAMETERS				
TYPE	DATA FORMAT	DATA TYPE	VALUE RANGE	REMARK

2.1 BASIC TOOLS MODULE

Input				
Input File	*.hdr file associated with Hx data			*.hdr file should contain fwhm and Centre wavelength
Output				
Output File	*.csv			Save SRF of the sensor to a single *.csv file

Overview

This tool generates SRF from given Central wavelength and FWHM of Hx sensor.

Python code for generating the Gaussian Function for specific mean(mu, centre wavelength) and FWHM.

```
def gauss(mu,fwhm,step):
    c2=2*np.sqrt(2*np.log(2))
    stop=(np.sqrt(-np.log(step)*2*(fwhm/c2)**2))+mu
    start=mu-(np.sqrt(-np.log(step)*2*(fwhm/c2)**2))
    x=np.arange(start,stop+step,step)
    sigma=fwhm/c2
    c=-(np.square((x-mu)/sigma)*0.5)
    return x,np.exp(c)
```

2.1.3 HX SPATIAL SPECTRAL RESAMPLE

QGIS 3.XX → Hyperspectral→ Basic Tools → Sensor Utility → Hx Spatial Spectral Resample

User Interface

2.1 BASIC TOOLS MODULE

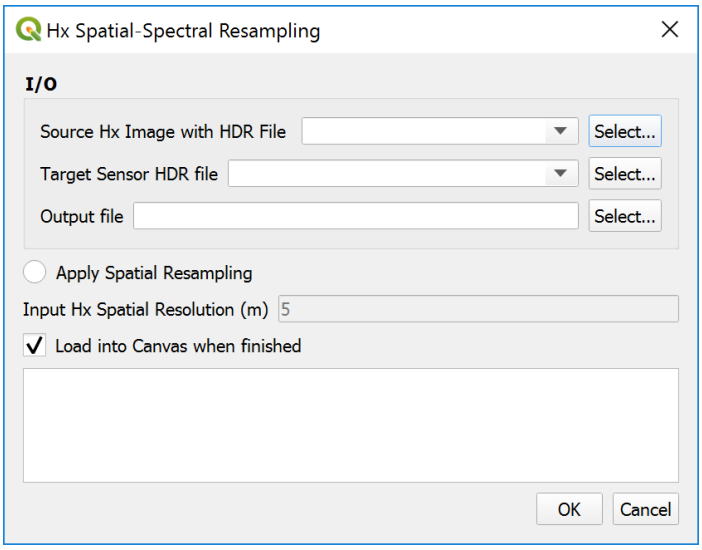


Figure. User-Interface for Hx Spatial-Spectral Resampling

Input Arguments

Source Hx Image with HDR File: Select the input Hx image (Hx image file should be associated with a HDR File).

HDR file should contain
Header titles 'wavelength' and 'fwhm'
wavelength = {443,490,560,665,705,740,783,842,865,945,1610,2190}
fwhm = {20,65,35,30,15,15,20,115,20,20,90,180}

Target Sensor HDR File: Select the target sensor HDR file
HDR file should contain Header titles 'wavelength'

If 'fwhm' is not available, fwhm will be computed by assuming FWHM is midway between adjacent bands. If the header file has 'spatial_res' header, then the spatial resampling will be performed based on the spatial resolution available in the header provided the 'Apply spatial resampling 'option is active.
wavelength = {443,490,560,665,705,740,783,842,865,945,1610,2190}
fwhm = {20,65,35,30,15,15,20,115,20,20,90,180}

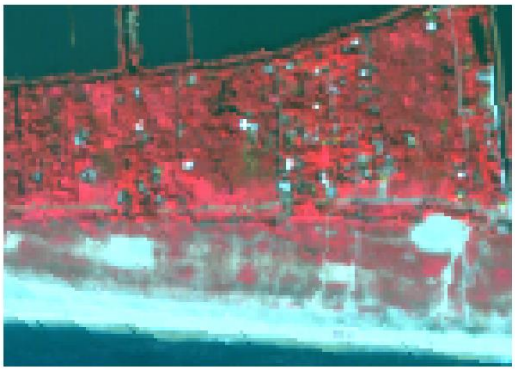
Apply Spatial Resampling: If this option is active, then the spatial resampling is performed using the Gaussian Point Spread Function. If this option is active, it will take entries under the 'spatial_res' header available in the target HDR file.

spatial_res = {60,10,10,10,20,20,20,10,20,60,20,20}

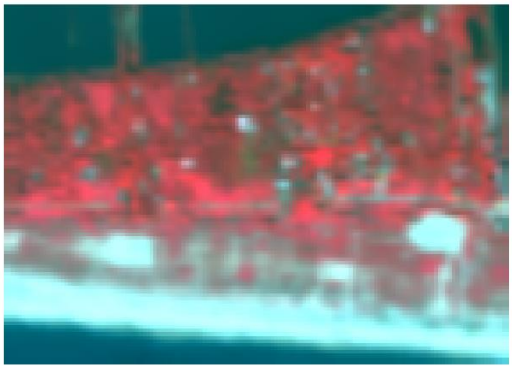
I/O PARAMETERS

TYPE	DATA FORMAT	DATA TYPE	VALUE RANGE	REMARK
Input				
Source Hx Image with HDR File	Geotiff or ENVI With HDR metadata file			Should contain central wavelength (fwhm is optional)
Target Sensor HDR File	HDR metadata file (.txt file with extension .hdr)			Should contain central wavelength (fwhm, spatial-res are optional)
Input Hx Spatial Resolution		integer	1-999	Spatial resolution in meters
Output				
Output File	ENVI file With HDR metadata file		Depends on input data range	

Results



Input AVIRIS-NG IMAGE (750nm, 650nm, 550nm Composite)



Output Sentinel Image (NIR, Red, Green Composite) with spatial resampling

REFERENCE

- Burt, Peter, and Edward Adelson. "The Laplacian pyramid as a compact image code." IEEE Transactions on communications 31.4 (1983): 532-540.
- https://scikit-image.org/docs/dev/api/skimage.transform.html

2.1 BASIC TOOLS MODULE

2.1.4 AVIRIS-NG SPATIAL SPECTRAL RESAMPLE

QGIS 3.XX → Hyperspectral→ Basic Tools → Sensor Utility → AVIRIS-NG Spatial-Spectral down sample

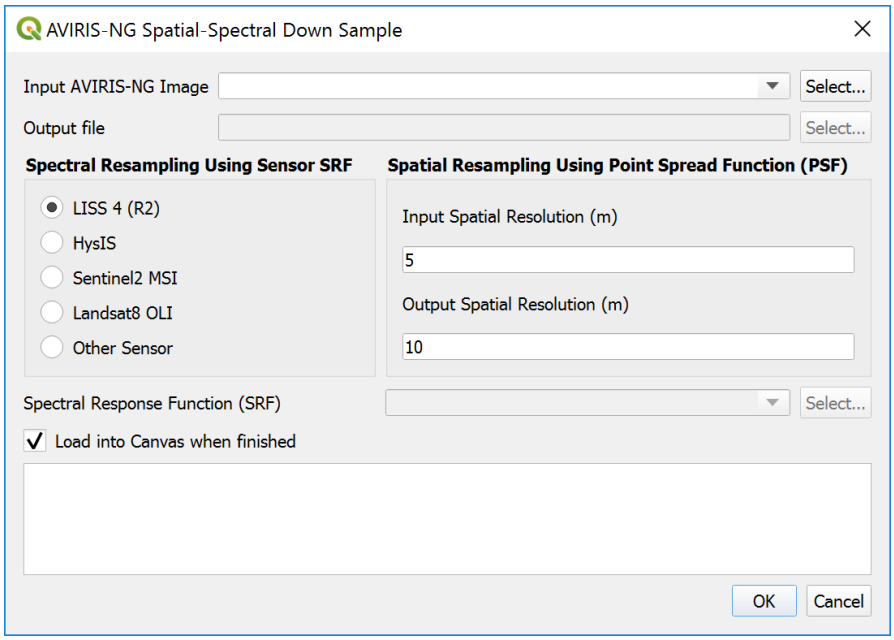


Figure. User-Interface for Hx Resampling

Input Arguments

Spectral Resampling using Sensor SRF: Select the Sensor Type (which uses sensor-specific Spectral Response Function (SRF)) to resample AVIRIS-NG data to the corresponding Hx/Mx data.
LISS 4: Resourcesat-2 Mx sensor - ISRO
HySIS: Hyperspectral Imaging Sensor - ISRO
Sentinel-2 MSI: European Space Agency (ESA)
Landsat-8 OLI: NASA

Spatial Resampling using Point Spread Function (PSF):

Input Spatial Resolution (m): Enter the spatial resolution of AVIRIS-NG data

Output Spatial Resolution (m): Enter the spatial resolution of the target sensor

2.1 BASIC TOOLS MODULE

I/O PARAMETERS

TYPE	DATA FORMAT	DATA TYPE	VALUE RANGE	REMARK
Input				
Input AVIRIS-NG image	Geotiff or ENVI			
Input Spatial Resolution		Integer	Depends on Input data	
Output Spatial Resolution		Integer	Depends on output data	Output spatial resolution > Input Spatial resolution
Output				
Output File	ENVI			Output Mx or Hx image with spatial resolution resampled to input image

Overview

Spectral Resampling is performed to resample the input AVIRIS-NG data to match either the response of a known instrument (for example, Landsat-8, Sentinel-2, HysIS, LISS-4 etc.) or user-defined spectral response function (as .xlsx file, format is shown below) with central wavelength and Spectral width. AVHYAS uses inbuilt SRF function provided by the manufacturers. If you provide a SRF, AVHYAS uses that for the resampling. Resampled spectra is generated based on the weighted sum of the SRF and the input spectra.

$$\hat{f}_{\lambda_j} = \frac{\sum_{i=1}^{n} S_i f_{\lambda_i}}{\sum_{i=1}^{n} S_i}$$

Where, $f_{\lambda i}$ is the pixel value (spectral value of the input spectrum) at wavelength λ_i. S_i is the Spectral response function weight at λ_i.

Format of the Excel File (Sheet0)

2.1 BASIC TOOLS MODULE

	A	B	C	D	E	F
1	Band		Centre	Spectral width	Spatial	
2	B	1	443	16	30	Band 1 - coastal aerosol
3	B	2	482	60	30	Band 2 - blue
4	B	3	561	57	30	Band 3 - green
5	B	4	655	37	30	Band 4 - red
6	B	5	865	28	20	Band 5 - Near Infrared (NIR)
7	B	6	1609	85	20	Band 6 - Short-wave Infrared (SWIR) 1
8	B	7	2201	187	20	Band 7 - Short-wave Infrared (SWIR) 2
9						

Sheet0 Sheet1 Sheet2

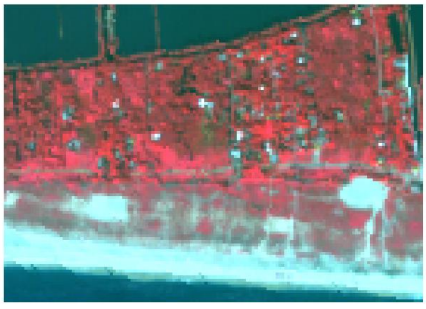
Format of the Excel File (Sheet1)

	A	B	C	D	E	F	G	H
1	SR_WL	CA	Blue	Green	Red	NIR	SWIR1	SWIR2
2	300	0	0	0	0	0	0	0
3	301	0	0	0	0	0	0	0
4	302	0	0	0	0	0	0	0
5	303	0	0	0	0	0	0	0
6	304	0	0	0	0	0	0	0
7	305	0	0	0	0	0	0	0
8	306	0	0	0	0	0	0	0
2295	2593	0	0	0	0	0	0	0
2296	2594	0	0	0	0	0	0	0
2297	2595	0	0	0	0	0	0	0
2298	2596	0	0	0	0	0	0	0
2299	2597	0	0	0	0	0	0	0
2300	2598	0	0	0	0	0	0	0
2301	2599	0	0	0	0	0	0	0
2302	2600	0	0	0	0	0	0	0

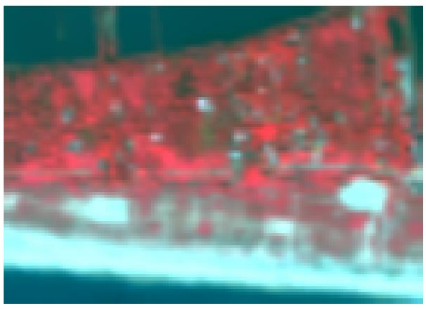
Sheet0 Sheet1 Sheet2

Spatial resampling is performed as a twofold process. At first, spatial smoothing is performed using Gaussian Point Spread Function with user defined scaling factor. Then Image resizing is performed and resampled to the original image dimension using Bi-Linear interpolation method. For more details, refer the scikit-image functions: 'pyramid_reduce' and 'resize'. Note that when down-sampling an image, Gaussian smoothing is performed to avoid aliasing artifacts. Sigma for Gaussian smoothing = (2 * downscale / 6.0), which corresponds to a filter mask twice the size of the scale factor that covers more than 99% of the Gaussian distribution.

Results



Input AVIRIS-NG IMAGE (750nm, 650nm, 550nm Composite)



Output Sentinel Image (NIR, Red, Green Composite) with spatial resampling

2.1 BASIC TOOLS MODULE

REFERENCE

- Burt, Peter, and Edward Adelson. "The Laplacian pyramid as a compact image code." IEEE Transactions on communications 31.4 (1983): 532-540.
- https://scikit-image.org/docs/dev/api/skimage.transform.html

2.2 DATA SUBSETTING

QGIS 3.XX → Hyperspectral→ Basic Tools → Data Subsetting

User Interface

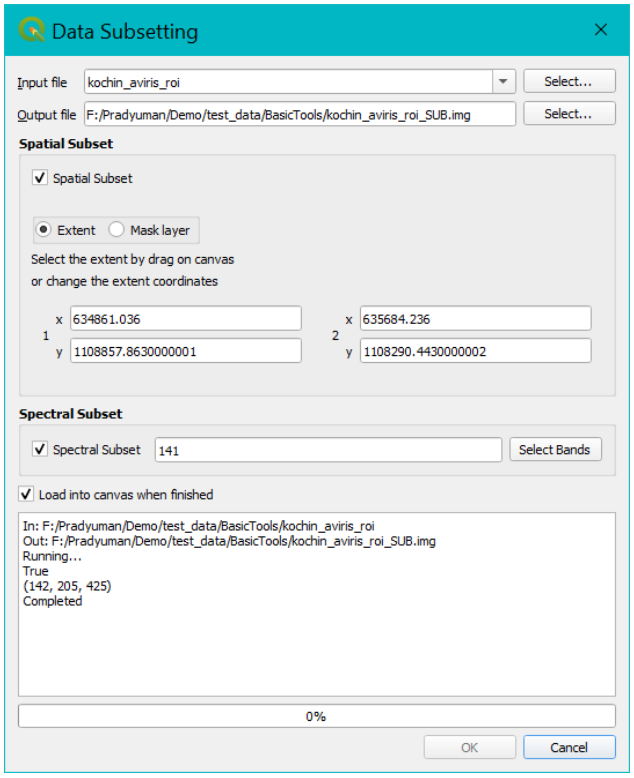


Figure. User interface for data subsetting

Input Arguments

Input File: Specify the input raster file for performing the subsetting.

2.1 BASIC TOOLS MODULE

Output File: Specify the output raster path to save to disk

Spatial Subset: ☐ Activate this setting for performing spatial subsetting. Spatial subsetting can either be done using a separate mask layer or by selecting x,y coordinates by dragging a window over the input data from the map canvas of QGIS.

Spectral Subset: ☐ Activate the setting for performing spectral subset. User has to select the spectral bands for which user desires to keep.

I/O PARAMETERS

TYPE	DATA FORMAT	DATA TYPE	VALUE RANGE	REMARK
Input				
Input File	GeoTiff / ENVI	Raster		Data from which a subset is required.
Output				
Output File	ENVI	Raster		Subset of the original data.

Overview

Data Subsetting tool is capable in providing subset of a data both spatially as well as spectrally. For spatially a user can either a mask file or can drag over the canvas to create a mask or can provide x,y coordinates for the subset. For spectral subset, the required bands by the user can be selected for the user interface.

2.3 SPECTRAL PLOT

QGIS 3.XX → Hyperspectral→ Basic Tools → Spectral Plot

User Interface

2.1 BASIC TOOLS MODULE

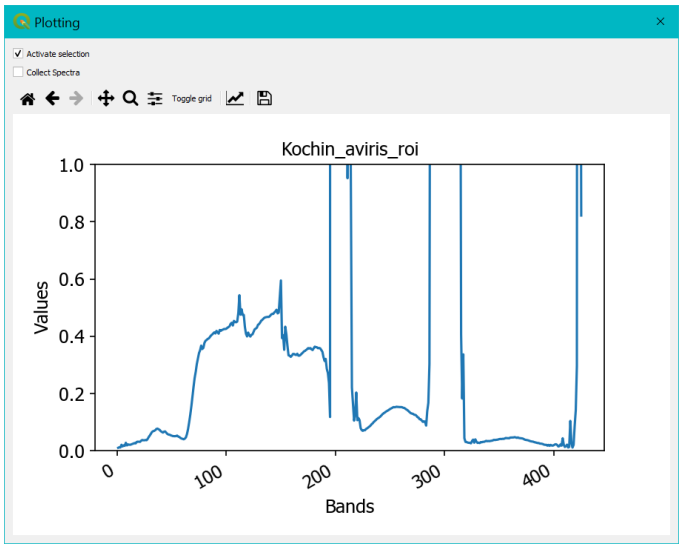


Figure. User interface for spectral plotting

Input Arguments

Activate selection: Activate this check box allows you to plot spectral signature from the data by clicking on the pixel in QGIS canvas.

Collect Spectra: This check box allows user to collect spectra from multiple targets/pixel in a single plot.

I/O PARAMETERS

TYPE	DATA FORMAT	DATA TYPE	VALUE RANGE	REMARK
Input				
Input File	GeoTiff / ENVI	Raster		Data from which spectral plots needs to be generated.
Output				
Output File	ENVI	Raster		Plot images

Overview

Spectral Plot tool provides a utility to plot spectral information from a pixel in a data. Multiple spectra from different pixels can also be plotted in the same plot. Several

options to customize plot like x and y axis range, spectra colour, spectra line width, scaling etc. are also present in it.

Note : By default, it will select the file which is already open in the QGIS canvas.

2.4 SCATTER PLOT

QGIS 3.XX → Hyperspectral→ Basic Tools → Scatter Plot

User Interface

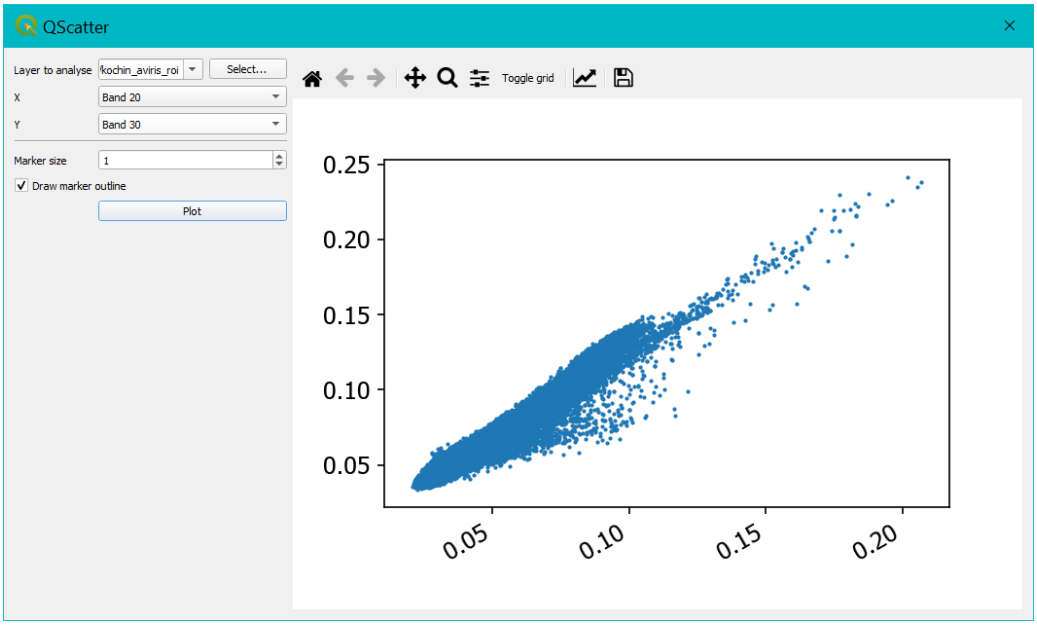


Figure. User interface for scatter plotting

Input Arguments

X: Feature along the x-axis for which scatter plot is required against Y.

Y: Feature along the y-axis for which scatter plot is required against X.

Marker Size: Size of the scatter plots markers.

I/O PARAMETERS

TYPE	DATA FORMAT	DATA TYPE	VALUE RANGE	REMARK
Input				

2.1 BASIC TOOLS MODULE

Input File	GeoTiff / ENVI			Data from which scatter plot needs to be generated.
Marker Size		Integer	> 0	Default = 1
Output				
Output File	ENVI			Plot images

Overview

Scatter Plot tool provides a utility to display values for two variables for a set of data to provide a better understanding between the correlation between the two variables. Several options to customize plot like x and y axis range, spectra colour, spectra line width, scaling etc. are also present in it.

By default, it will select the file which is already open in the QGIS canvas.

2.5 SCALING

QGIS 3.XX → Hyperspectral→ Basic Tools → Scaling

User Interface

2.1 BASIC TOOLS MODULE

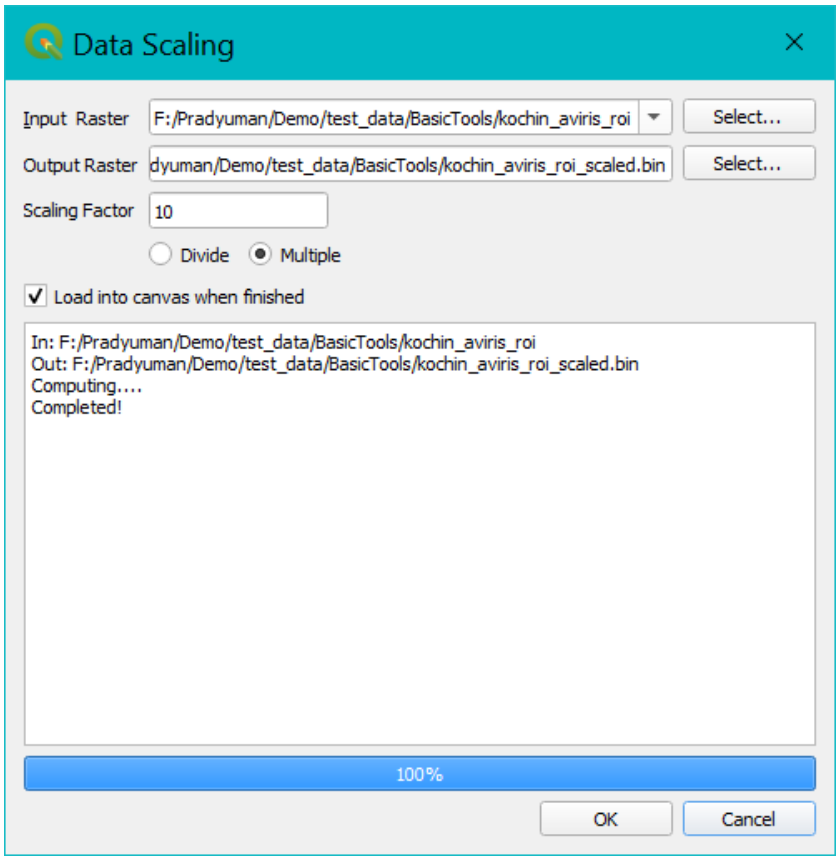


Figure. User interface for scaling

Input Arguments

Scaling Factor: Amount by which the user wants to increase or decrease the scaling of data.

Divide or Multiple: Depending on whether user wants to upscale or downscale

I/O PARAMETERS

TYPE	DATA FORMAT	DATA TYPE	VALUE RANGE	REMARK
Input				
Input File	GeoTiff / ENVI			Data required to be scaled
Scaling Factor		Real numbers	>= 0	Default = 10000
Output				
Output File	*.bin			

2.1 BASIC TOOLS MODULE

Overview

This tool provides a utility to increase or decrease the values in a data by using a scaling factor.

User needs to select the file since there is no default file selection in it.

2.6 Spectral Library

QGIS 3.XX → Hyperspectral→ Basic Tools → Spectral Library

User Interface

Figure. User interface for Spectral library

I/O PARAMETERS

TYPE	DATA FORMAT	DATA TYPE	VALUE RANGE	REMARK
Input				
Input File	GeoTiff / ENVI			Data from which spectral information is required
Output				
Output File	*.sli			Spectral information of a pixel over the full range

2.1 BASIC TOOLS MODULE

Overview

Spectral library tool is used to save the spectral information of any pixel over the full range of wavelengths available in the data. The user interface allows to generate multiple spectral information for various pixels which can be saved in the form of spectral library. The data from which spectral information is taken should be open in the QGIS canvas to select the pixels.

2.7 ROI SEPERABILITY

QGIS 3.XX → Hyperspectral→ Basic Tools → ROI Separability Analysis

User Interface

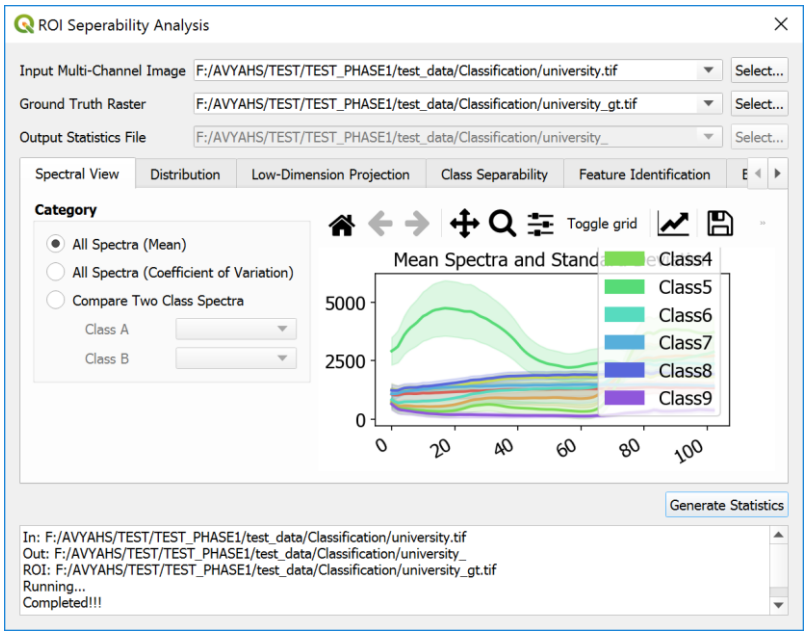


Figure. User-Interface for ROI Separability (Spectral View)

Input Arguments

Spectral View

- **All Spectra (Mean):**
  Select **All Spectra (Mean)** → Click **Generate Statistics**
  This function generates a graph with mean spectra (thick lines) and its standard deviations (fill-areas) with different color codes for each class.
- **All Spectra (Coefficient of variation)**

Select **All Spectra (Mean)** → Click **Generate Statistics**
This function generates a graph with coefficient of variation (CV) spectra (thick lines) with different color codes for each class.

- **Compare Two Class Spectra**
  Select **Compare Classes**→ Click **Generate Statistics** → Select **ClassA** → Select **ClassB**
  This function generates a graph with mean spectrum and corresponding spectral standard deviations (different color codes for selected class).

Unclassified → ROI file raster value = 0

Class 1 → ROI file pixel value = 1

Class 2 → ROI file pixel value = 2

I/O PARAMETERS FOR DATA

TYPE	DATA FORMAT	DATA TYPE	VALUE RANGE	REMARK
Input				
Input Image	Geotiff or ENVI			Hx or Mx image
ROI File	Geotiff or ENVI			Single band image with values represent corresponding class Value 0 = unclassified
Output				
Output Statistics File	*.csv file			Save class mean spectra and standard deviations

2.1 BASIC TOOLS MODULE

Distribution

User Interface

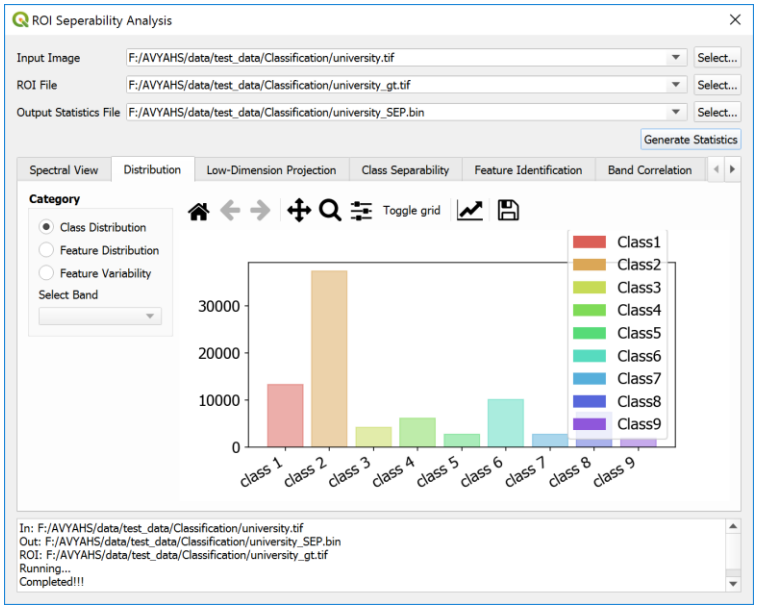


Figure. User-Interface for ROI Separability (Distribution)

- **Class Distribution:** This functionality shows the number of spectral samples available in each region of interest (or classes)
- **Feature Distribution:** This functionality shows the kernel density estimate of the histogram of samples belong to each class
- **Feature Variability:** Generates the box plot of each class corresponds to a particular band/feature

I/O PARAMETERS

TYPE	DATA FORMAT	DATA TYPE	VALUE RANGE	REMARK
Select Band			No. of bands in the image	

2.1 BASIC TOOLS MODULE

Low-dimension projection:

User Interface

Figure. User-Interface for Low-Dimension projection

- **PCA:** Generates the scatter plot of 1st and 2nd PCs of the data with different color codes for each class (PCA is performed on the mean centered data). (ideal for linear data)
- **T-SNE:** (t-stochastic Nearest Neighbor Embedding) generates the 2D scatterplot of 1st and 2nd components of the data projected on t-SNE space. (ideal for display the non-linear data). This function performs t-SNE on the PCs computed over the normalized input data. T-SNE operation is applied on the sampled (select every 10 samples) input data to reduce the computation time.

I/O PARAMETERS

TYPE	DATA FORMAT	DATA TYPE	VALUE RANGE	REMARK
Perplexity		Integer	[1,999]	Best result for 30-100 range

2.1 BASIC TOOLS MODULE

Class Separability

User Interface

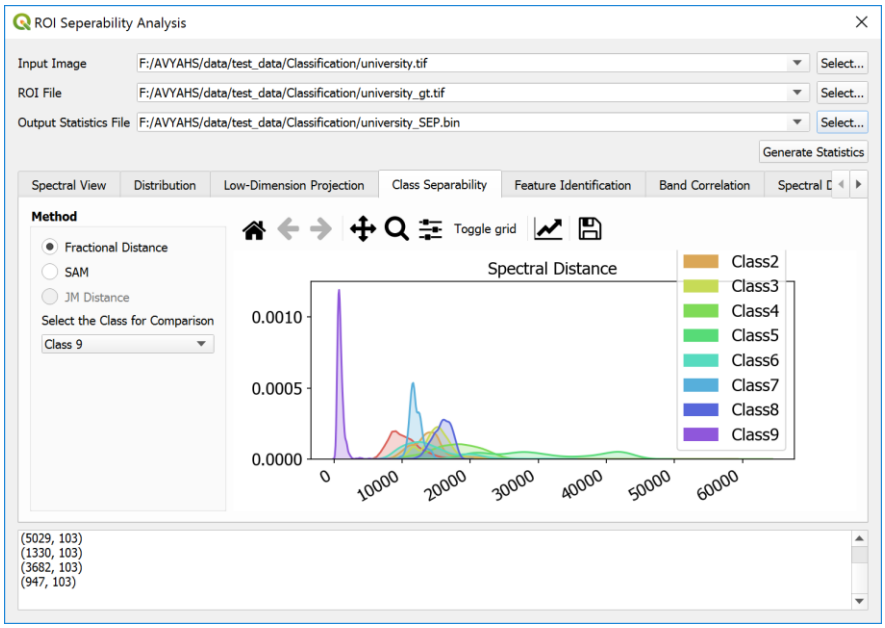


Figure. User-Interface for Class Separability

Fractional Distance: Computes the fractional spectral distance between mean spectra of the selected class and the individual spectrum of all other classes including the selected class

$$d_k(g_p, g_c) = \sum_{i=1}^{N}((\left\|g_p^i - g_c^i\right\|^k)^{1/k})$$

SAM: (Spectral Angle Mapper): Computes cosine distance between mean spectra of the selected class and the spectra of all other classes including the selected class

2.1 BASIC TOOLS MODULE

Feature Identification

User Interface

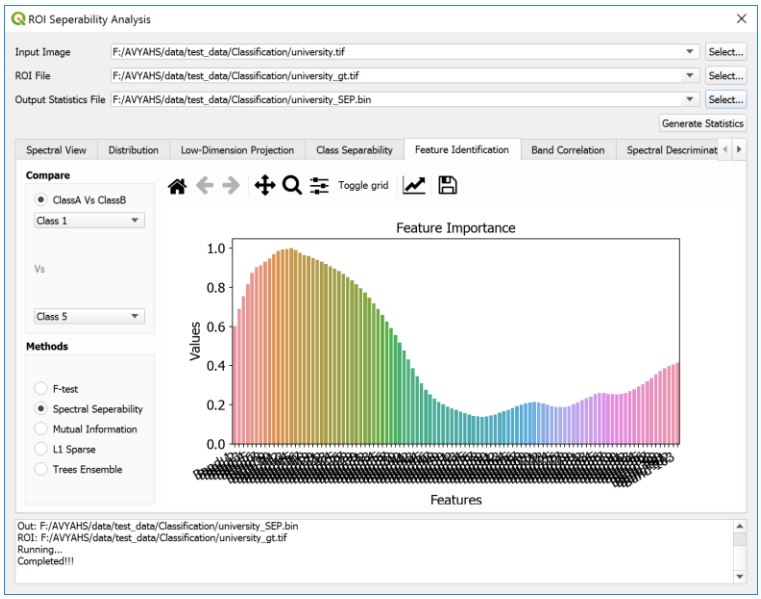


Figure. User-Interface for Feature Identification

Methods:

Class A is assigned with label '0' and Class B is assigned with label '1'. If 'X' is the feature-set with size=MxL (M=samples, L=features/bands), Then the feature importance is computed by finding the relation between (X, y) using the methods given below.

F-test: Compute the ANOVA F-value between each feature and the target vector. The F-value scores examine if, when we group the numerical feature by the target vector, the means for each group are significantly different. The methods based on F-test estimate the degree of linear dependency between two random variables.

Spectral Separability: Separability metric is defined by $\frac{|m1-m2|}{s1+s2}$

Where, **m1** is the mean of the feature belong to class-1, **m2** is the mean of the feature belong to class-2, **s1** is the standard deviation of the feature belong to class-1, **s2** is the standard deviation of the feature belong to class-2.

2.1 BASIC TOOLS MODULE

Mutual Information: Estimate mutual information for a discrete target variable. Mutual information (MI) between two random variables is a non-negative value, which measures the dependency between the variables. It is equal to zero if and only if two random variables are independent, and higher values mean higher dependency. The function relies on nonparametric methods based on entropy estimation from k-nearest neighbors distances as described in Kraskov et. al. 2004 and Ross, 2014. In AVHYAS, normalized MI is computed by dividing the MI value of each feature by the maximum value of MI across all the features (For more information, refer Scikit-Learn Feature Selection implementation).

Tree Ensemble: Extra-trees classifier can be used to compute impurity-based feature importance, which in turn can be used to discard irrelevant features (Geurts et. al., 2006). This method implements a meta estimator that fits a number of randomized decision trees (a.k.a. extra-trees).

REFERENCE

P. Geurts, D. Ernst., and L. Wehenkel, "Extremely randomized trees", Machine Learning, 63(1), 3-42, 2006.

2.1 BASIC TOOLS MODULE

Band Correlation

User Interface

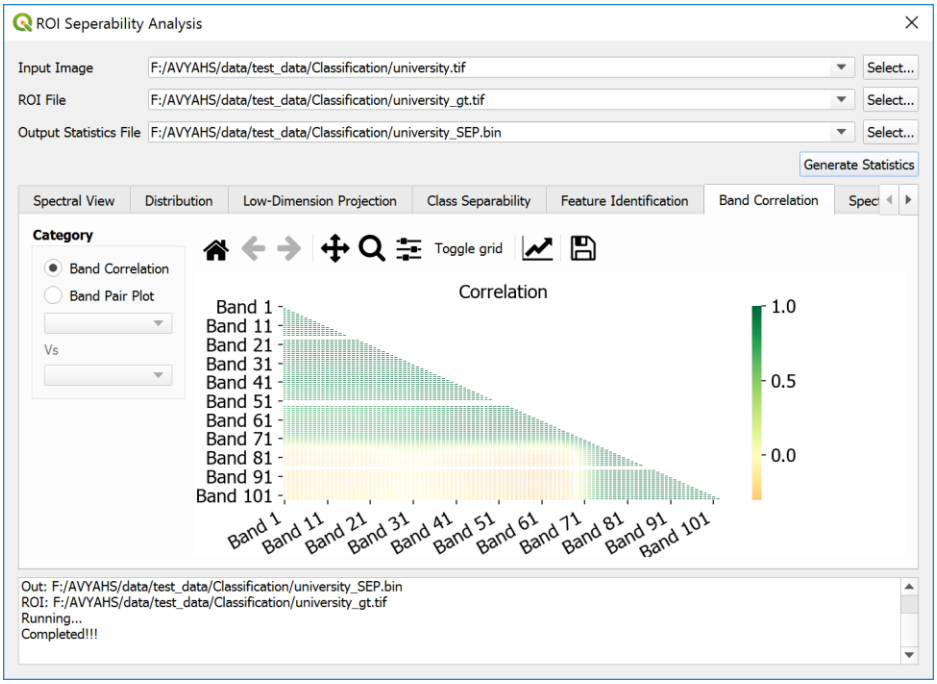


Figure. User-Interface for Band Correlations

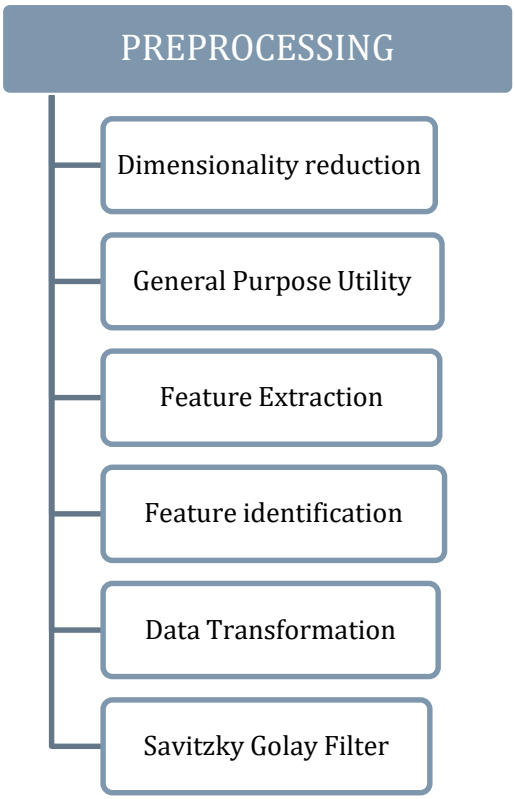
Band Correlation: This function computes the Pearson-correlation-coefficient between two bands (or features) in the multi-variate data and forms a correlation matrix $C_{p,q}$.
'p' and 'q' varies from Band-1 to Band-L (if the size of the data is MxNxL)
Correlation Coefficient is computed as,

$$r_{xy} = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{\sqrt{n\sum x_i^2 - (\sum x_i)^2}\sqrt{n\sum y_i^2 - (\sum y_i)^2}}$$

Where, **x** and **y** are two bands or features and '***i'*** denotes the sample.

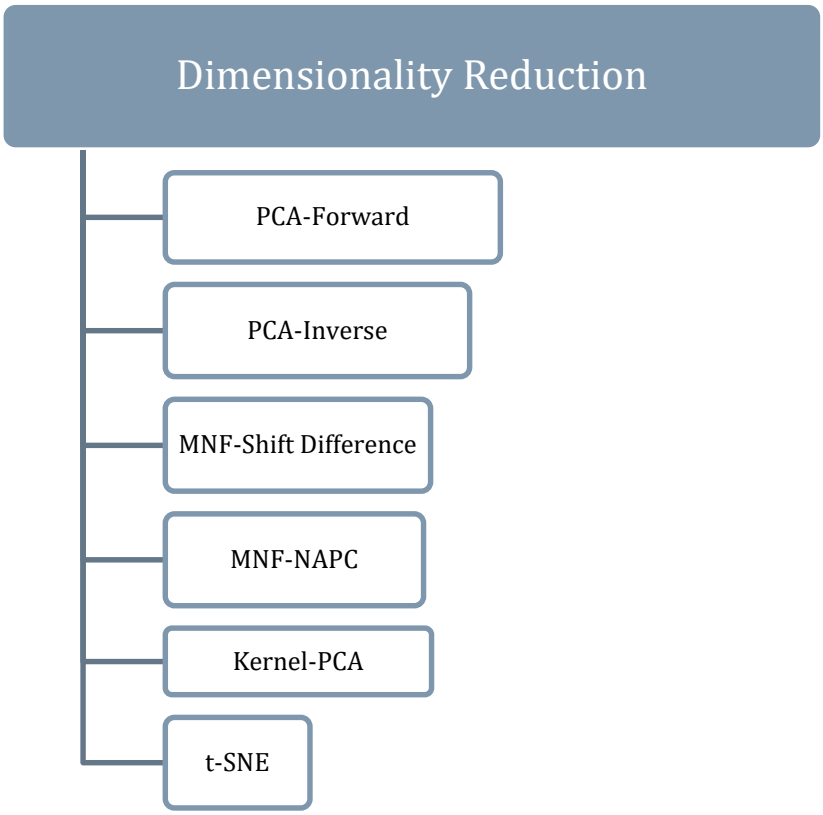
Band Pair Plot: It computes the correlation between two bands or two features selected from the drop down list.

3 PREPROCESSING



3 PREPROCESSING

3.1 DIMENSIONALITY REDUCTION



3.1.1 PCA-FORWARD

QGIS 3.XX → Hyperspectral→ Preprocessing → Dimensionality Reduction → PCA-Forward

User Interface

3 PREPROCESSING

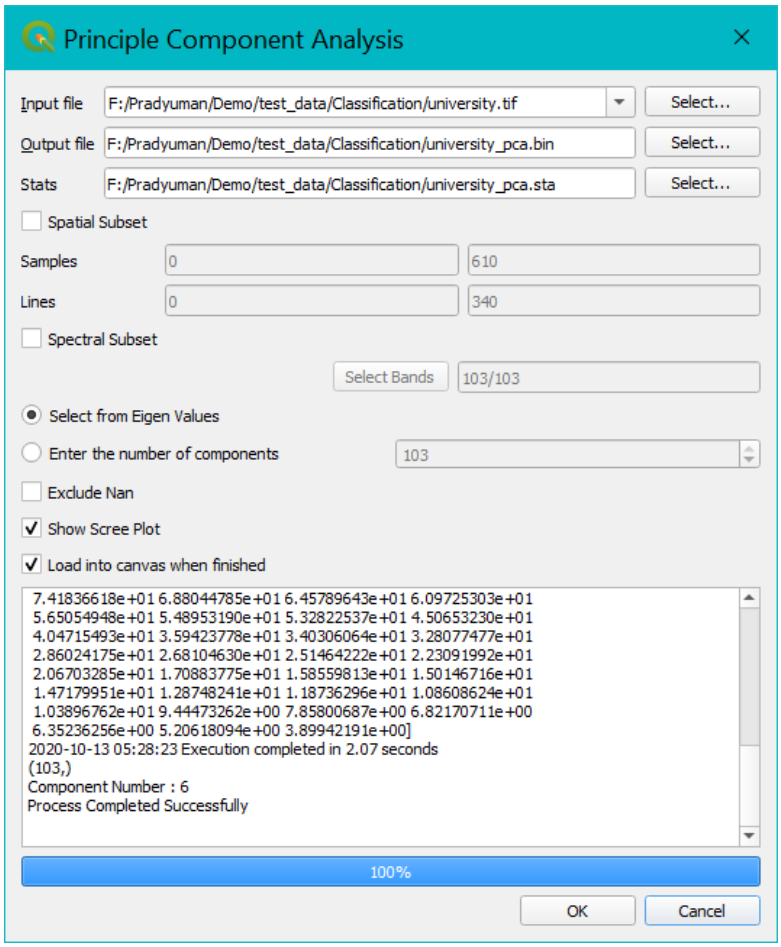


Figure. User interface for PCA

Input Arguments

Spatial Subset: Allows to create a spatial subset before processing PCA-forward.

Spectral Subset: Allows to create a spectral subset of data.

Select from Eigen Values: Allows users to select components after calculating eigen values.

Enter the number of components: Number of components required by the user can be put directly.

3 PREPROCESSING

I/O PARAMETERS

TYPE	DATA FORMAT	DATA TYPE	VALUE RANGE	REMARK
Input				
Input File	GeoTiff / ENVI			
Output				
Output File	ENVI			By default saves in the same folder as input file with '*_pca.bin'
	pickle			Statistics file

Overview

This module provides an interface for performing principal component analysis. The principal components transformation, also called principal components analysis (PCA), generates linear combinations of hyper-spectral pixel intensities which are mutually uncorrelated and which have maximum variance. Specifically, consider a hyper-spectral image represented by the random vector $\mathbf{G}$ (for vector of gray-scale values) and assume that $\langle \mathbf{G} \rangle = \mathbf{0}$, so that the covariance matrix is given by $\sum = \langle \mathbf{G}\mathbf{G}^T \rangle$. PCA tends to seek a linear combination $\mathbf{Y} = \boldsymbol{w}^T\mathbf{G}$ whose variance $\boldsymbol{w}^T\sum\boldsymbol{w}$ is maximum. This quantity can trivially be made as large as we like by choosing $\boldsymbol{w}$ sufficiently large, so that the maximization only makes sense if we restrict $\boldsymbol{w}$ in some way. A convenient constraint is $\boldsymbol{w}^T\boldsymbol{w} = \boldsymbol{1}$, which can be solved by maximizing the unconstrained Lagrange function.

$$\mathbf{L}=\mathbf{w}^T\sum\mathbf{w} - \lambda(\mathbf{w}^T\mathbf{w}-\mathbf{1})$$

This leads directly, to the eigenvalue problem

$$\sum\mathbf{w}= \lambda\mathbf{w}$$

Where denotes the orthogonal and normalized eigenvectors of $\sum$ obtained by solving the above problem by $\boldsymbol{w_1} \ldots \boldsymbol{w_N}$, sorted according to decreasing eigenvalue $\lambda_1 \geq \ldots \geq \lambda_N$. These eigenvectors are the principal axes and the corresponding linear combinations $\boldsymbol{w_i}^T\boldsymbol{G}$ are projections along the principal axes, called The principal components of $\mathbf{G}$. The individual principal components

$$\mathbf{Y_1} = \mathbf{w_1}^T\mathbf{G}, \mathbf{Y_2} = \mathbf{w_2}^T\mathbf{G}, \ldots, \mathbf{Y_N} = \mathbf{w_N}^T\mathbf{G}$$

can be expressed more compactly as a random vector Y by writing

$$\mathbf{Y} = \mathbf{W}^T\mathbf{G}$$

where $\mathbf{W}$ is the matrix whose columns comprise the eigenvectors, that is,

$$\mathbf{W} = (\mathbf{w_1} \ldots \mathbf{w_N}).$$

Since the eigenvectors are orthogonal and normalized, W is an orthonormal matrix:

$$\mathbf{W}^T\mathbf{W} = \mathbf{I}.$$

3 PREPROCESSING

PCA is a commonly used dimensionality reduction technique by projecting each data point onto the first few principal components to obtain lower-dimensional data while preserving as much of the data's variation as possible. The principal components are the eigenvectors of the data's covariance matrix.

3.1.2 PCA-INVERSE

QGIS 3.XX → Hyperspectral→ Preprocessing → Dimensionality Reduction → PCA-Inverse

User Interface

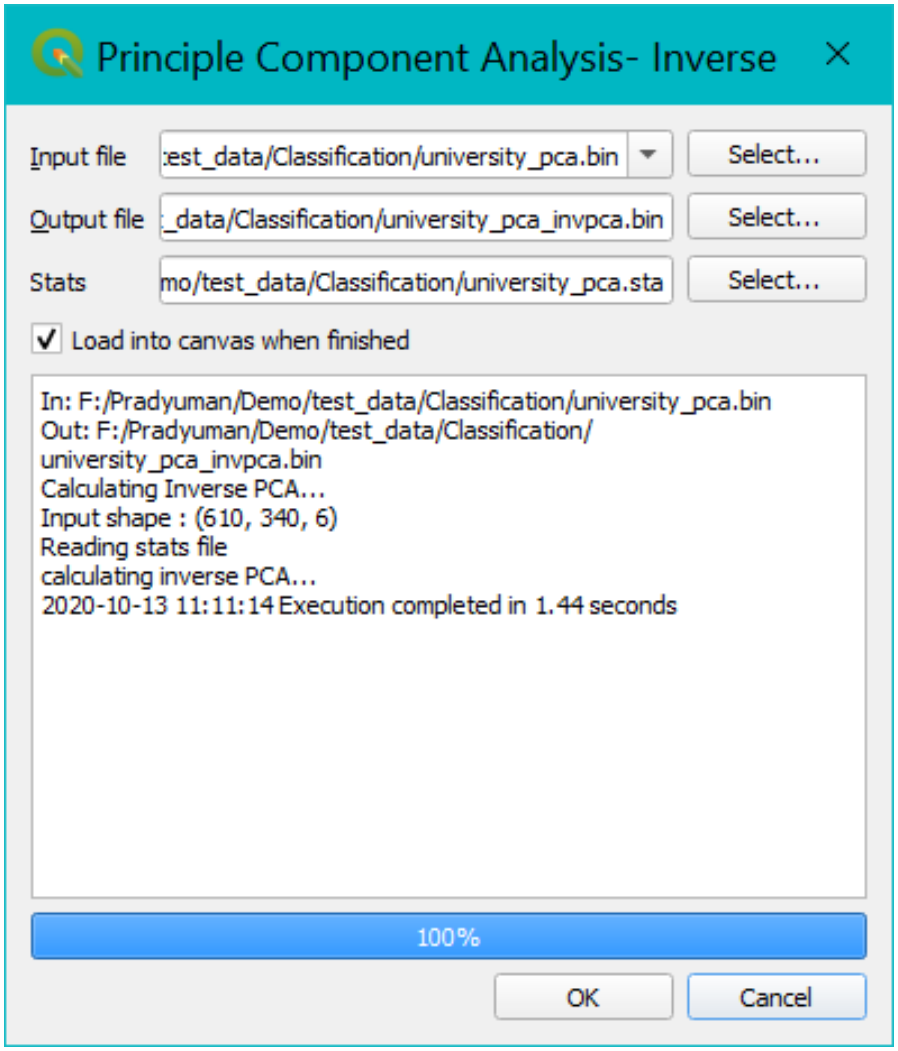


Figure. User interface for PCA-Inverse

Input Arguments

Stats: Statistics file generated during PCA-Forward has to be provided here.

I/O PARAMETERS

TYPE	DATA FORMAT	DATA TYPE	VALUE RANGE	REMARK
Input				
Input File	GeoTiff / ENVI/ *.bin			PCA forward output file
	*.sta			Statistics file
Output				
Output File	ENVI			By default saves in the same folder as input file with '*_invpca.bin'

Overview

This module provides an interface to reconstruct the data back and inverse the step done in forward PCA. As inputs it takes the resulting scores plus the statistics file generated in the Forward PCA program.

3.1.3 MNF- SHIFT DIFFERENCE

QGIS 3.XX → Hyperspectral→ Preprocessing → Dimensionality Reduction → MNF-Shift Difference

User Interface

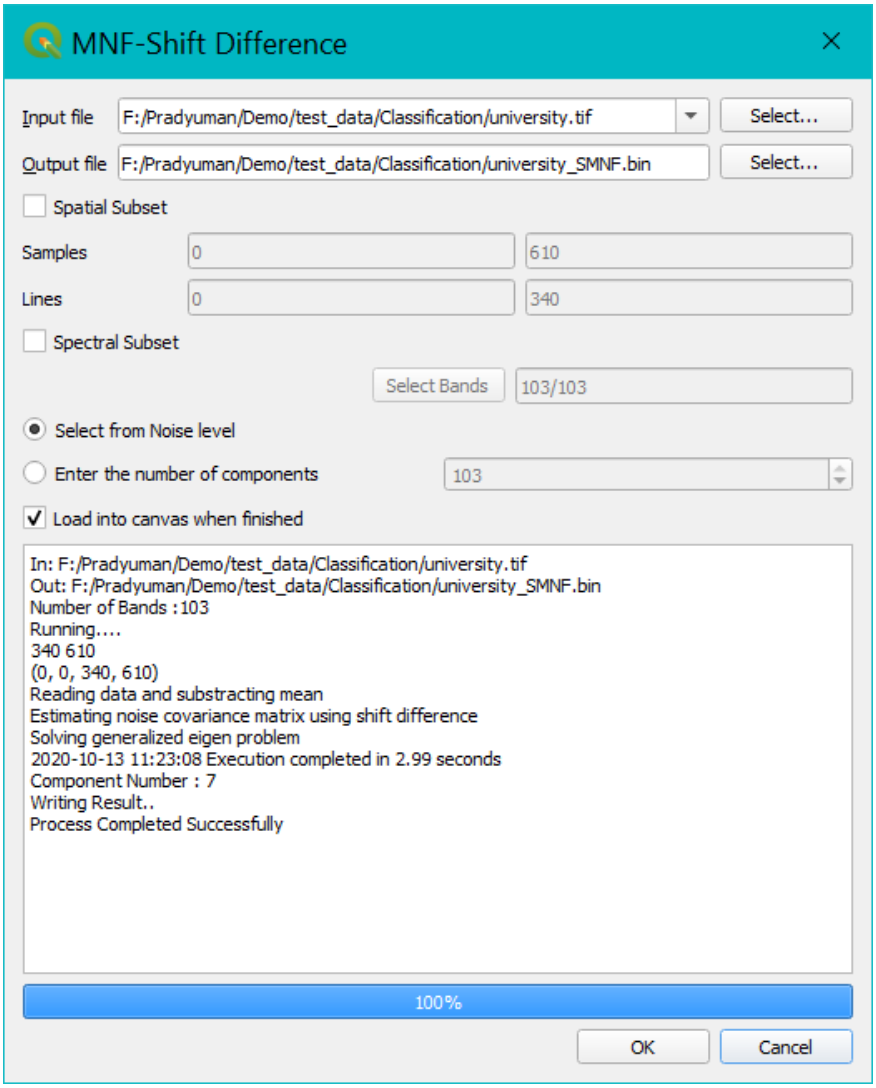


Figure. User interface for MNF-Shift Difference

Input Arguments

Select from Noise Level: Components selection based on the SNR values can be done here.

Enter the number of components: Instead of selecting components based on SNR, user can directly put the number of components required.

I/O PARAMETERS

TYPE	DATA FORMAT	DATA TYPE	VALUE RANGE	REMARK
Input				
Input File	GeoTiff / ENVI			
Spatial Subset(Sample/Line)		Integer	0 to Max no. of sample/line	
Output				
Output File	ENVI			By default saves in the same folder as input file with '*_SMNF..bin'

Overview

Minimum noise fraction (MNF) (*Green et al,* 1988) is another dimension reduction technique consisting of two consecutive data reductions operations. The first is based on an estimation of noise in data as represented by a correlation matrix. This transformation decorrelates and rescales the noise in the data, by variance. At this stage, the information about between band noise has not been considered. The second operation accounts for the original correlations, and creates a set of components that contain weighted information about the variance across all bands in the raw data set. Often, most of the surface reflectance variation in a data set can be explained in the first few components, with the rest of the components containing variance as contributed primarily by noise. Here, noise covariance matrix is estimated using shift difference method i.e. each pixel in each band is replaced by the average difference between it and one horizontal and one vertical neighbor and then the sample covariance matrix of these average differences is used (*Berman et al,* 2011).

3.1.4 MNF- NAPC

QGIS 3.XX → Hyperspectral→ Preprocessing → Dimensionality Reduction → MNF-NAPC

User Interface

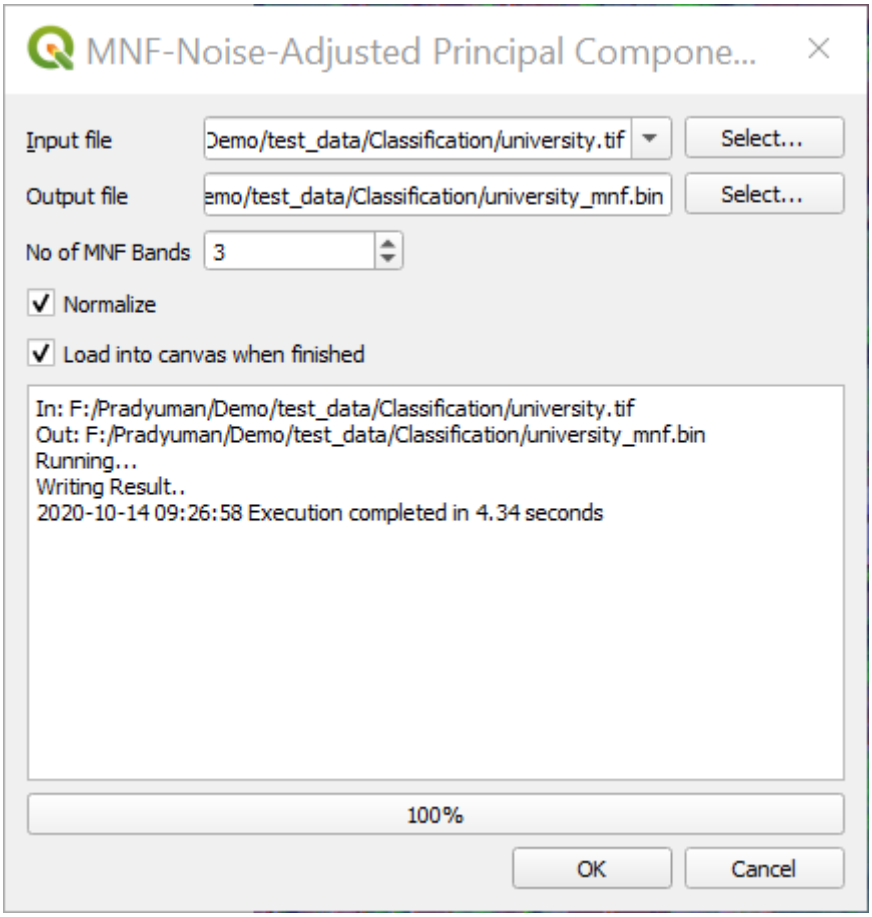


Figure. User interface for MNF-NAPC

Input Arguments

Input File: Specify the input HIS data cube

No. of MNF Bands: Define the number of components to save to disk

I/O PARAMETERS

TYPE	DATA FORMAT	DATA TYPE	VALUE RANGE	REMARK
Input				

Input File	GeoTiff / ENVI			
Output				
Output File	ENVI			By default saves in the same folder as input file with '*_mnf..bin'

Overview

MNF-NAPC basically abbreviated for Minimum Noise Fraction Using Noise Adjusted Principal Component. Like MNF defined in the earlier section it is a linear transformation method that consists of a noise whitening step before the PCA rotation. The noise whitening step uses the noise covariance matrix to decorrelate and rescale the noise in the data (noise whitening). Results in transformed data in which the noise has unit variance and no band-to-band correlations. The process of MNF-NAPC is designed to determine the inherent dimensionality of image data, segregate noise in the data, allow efficient elimination and/or reduction of noise and reduce the computational requirement for subsequent processing.

Reference

C-I Change and Q Du, "Interference and Noise-Adjusted Principal Components Analysis," IEEE TGRS, Vol 36, No 5, September 1999

3.1.5 KERNEL-PCA

QGIS 3.XX → Hyperspectral→ Preprocessing → Dimensionality Reduction → Kernel-PCA

User Interface

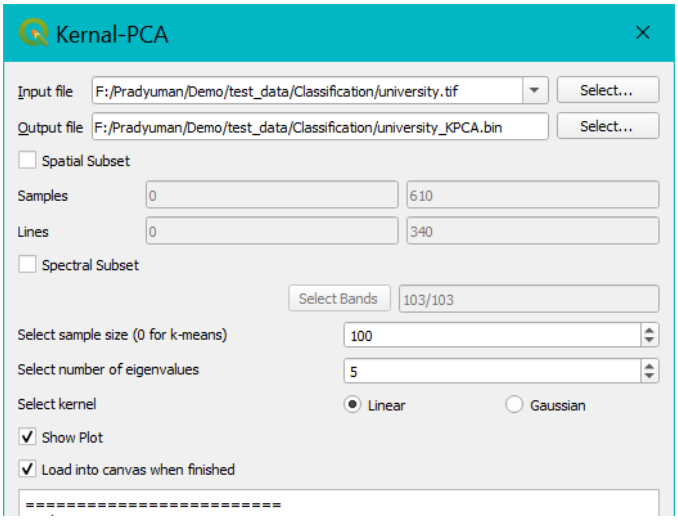


Figure. User interface for Kernal-PCA

3 PREPROCESSING

Input Arguments

Select Sample size: Sample size for estimation of kernel matrix. Zero (0) for kmeans to determine 100 cluster centers (default)

Select number of eigenvalues: Number of eigenvectors to keep.

Select Kernel: User can select either linear or Gaussian kernel.

I/O PARAMETERS

TYPE	DATA FORMAT	DATA TYPE	VALUE RANGE	REMARK
Input				
Input File	GeoTiff / ENVI			
Select sample size		Integer	> =0	0 for selecting sample size using k-means clustering
Select number of eigenvalues		Integer	> = 0	
Output				
Output File	ENVI			

Overview

Kernel PCA is generally used when the dataset is non-linear. It uses a kernel function to project dataset into a higher dimensional feature space where it is linearly separable. Spatial and spectral subsets generation can be done in the GUI itself before processing if required by the user. Sample size selection should be kept 0 if K-means clustering for selection is required. Number of eigenvalues selection as desired by user. Linear or Gaussian kernel function can be selected by the user for generating eigenvalues.

3.2. GENERAL PURPOSE UTILITY

3.2.1. SPARSE CODING BASED CLOUD REMOVAL

QGIS 3.XX → Hyperspectral→ Basic Tools → Sensor Utility → SPARSE CODING BASED CLOUD REMOVAL

User Interface

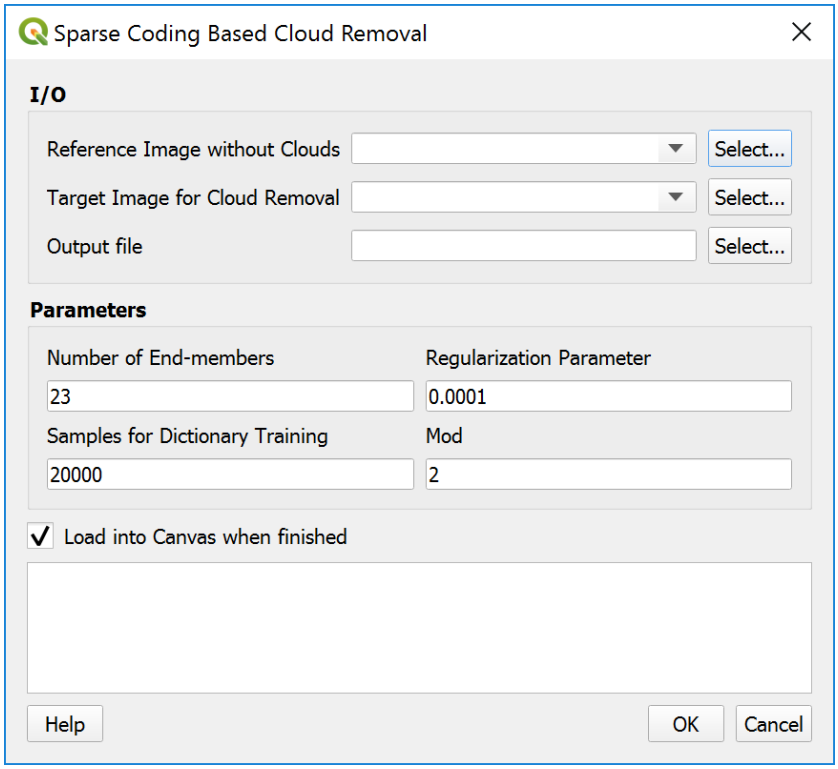


Figure: User-Interface for Cloud removal method

Reference Image: M×N×L sized image, where M=samples, N=rows, L=bands. A cloud free image with acquisition time close to target image.

Target Image: M×N×L sized image, where M=samples, N=rows, L=bands. Image with clouds and its shadows. Target and reference images should be captured with same sensor.

No. of samples (p): Integer Value. More samples may produce better result.

No. of End-members (E_n): Integer Value. Use HySIME or HSF-VC to find the number of end-members.

Mod: mod=1 for sum to one constraint (regularization parameter=1). mod=2 for regularisation parameter 0.0001.
Regularisation parameter: sum to one constraint (regularisation parameter=1), for minimization of L1 Error (regularisation parameter< 0.001)

I/O PARAMETERS

TYPE	DATA FORMAT	DATA TYPE	VALUE RANGE	REMARK
Input				
Refernce Image	Geotiff or ENVI			Dimension: MxNxL
Target Image	Geotiff or ENVI			Dimension: MxNxL
No. of end-members		Integer	$1< E_n <L$	
Regularization Parameter		Float	1 or <0.001	
Samples for Training		Integer	$E_n<p<M*N$	
Mod		Integer		mod=1 (sum to one constraint); mod=2 (Minimizing the L1 error)
Output				
Output File	ENVI		Depends on output data	Dimension: MxNxL

Overview

In this method, a new cloud removal approach, which is called multi-temporal dictionary learning (MDL), is implemented. Dictionaries of the cloudy areas (target data) and the cloud-free areas (reference data) are learned separately in the spectral domain. The removal process is conducted by combining coefficients from the reference image and the dictionary learned from the target image. This method could well recover the data contaminated by thin and thick clouds or cloud shadows.

As HSI can be sparsely represented by a linear combination of a set of end-members, sparse representation (or un-mixing) has been successfully applied for various HIS based applications [1]. This is a joint optimization problem with respect to the dictionary D and the coefficients. The online dictionary learning method can be applied to solve this problem.

3 PREPROCESSING

Let $X = [x_1, ..., x_n] \in \mathbb{R}^{b \times n}$ represent a spectral image signal with b bands and $n = r \times c$ pixels. Then, let x_i satisfy

$$\min_{D \in \mathcal{C}, \alpha \in \mathbb{R}^{b \times n}} \sum_{i=1}^{n} \frac{1}{2} \left\| x_i - D\alpha_i \right\|_2^2 \quad s.t. \quad D \geqslant 0, \forall \alpha_i \geqslant 0 \tag{1}$$

where $D \in \mathbb{R}^{b \times k}$ is the dictionary matrix, and typically $k > n$. The columns of D are referred to as the atoms, which represent dominant spectra (end-members) in HSI. The elements of k represent the sparse coefficients which satisfy the condition that number of non-zero coefficients are less than k. Nonnegative constraints are enforced in the decomposition process to make the values of α positive. $\mathcal{C}$ is the convex set of matrices with the following constraint,

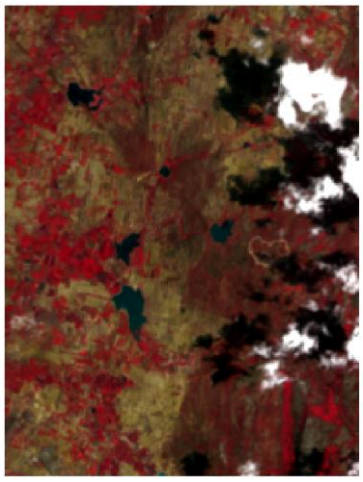
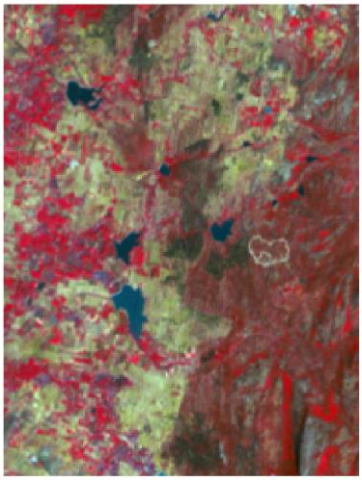
$$\mathcal{C} \triangleq \{D \in \mathbb{R}^{b \times k} \quad s.t. \quad \forall j = 1, \ldots, k, \left\| d_j \right\|_2 \leqslant 1 \; and \; d_j \geqslant 0\} \tag{2}$$

When there is a cloud cover, we can expect that the fundamental components that a pixel contains remain unchanged. Therefore, the same number of atoms for the two dictionaries is selected. However, the two dictionaries are not identical due to two reasons.

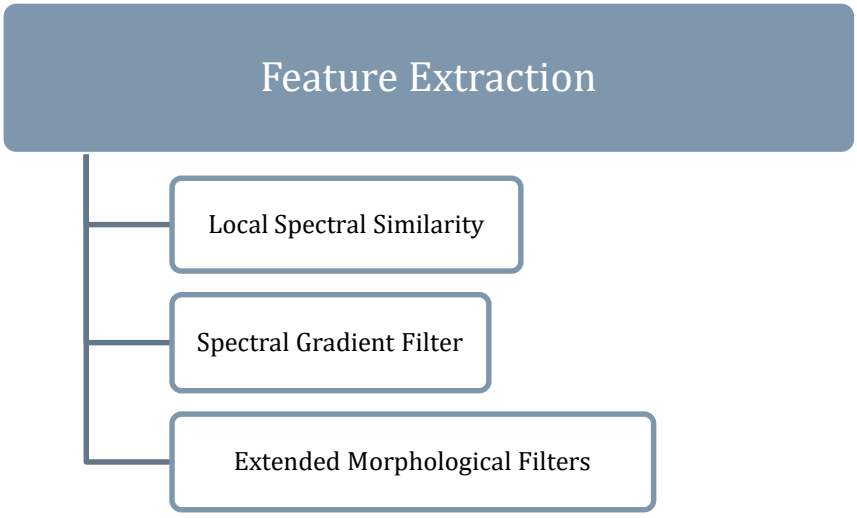
First, the order of the atoms may not be the same since they are generated separately during each dictionary training. Second, the atoms corresponding to the same spectral components (subclasses) change with imaging conditions. Nevertheless, the corresponding atoms from the cloudy/shadowy image should have the same pattern as those from the clear image and are highly correlated. It is important that the two dictionaries are in the same order for performing the next step of the proposed method. Hence, reordering is performed using the correlation matrix (CM) between the atoms in the two dictionaries. The CM is determined by calculating the correlation coefficient (CC) between each atom in Dt and Dr. Every element in the CM is the CC of the two dictionaries. The reordered matrix is generated by searching the highest CC value of each column in the CM. Each column is reordered by moving the highest CC value to the nth row in the nth column. Then, each column in Dt will change according to the movement in the CM. These steps are used to make sure that the two dictionaries have the same order. Reordered dictionary is denoted by Dt. The CM is first calculated to present the correlation coefficients between each column in Dt and Dr. The purpose of reordering is to make the atom in Dt have the highest correlation with the corresponding atom in Dr. The reordered dictionary Dt is adjusted this way. Reordering procedure is a necessary process to make sure that the sparse coefficients represent the associated components as accurately as possible.

3 PREPROCESSING

Results

 Input Hyperion cloudy image (750nm, 650nm, 550nm Composite)	 Output Hyperion cloud-removed image (750nm, 650nm, 550nm Composite)

3.3 FEATURE EXTRACTION



3.3.2 SPECTRAL GRADIENT FILTER

QGIS 3.XX → Hyperspectral→ Preprocessing → Feature Extraction → Spectral Gradient Filter

User Interface

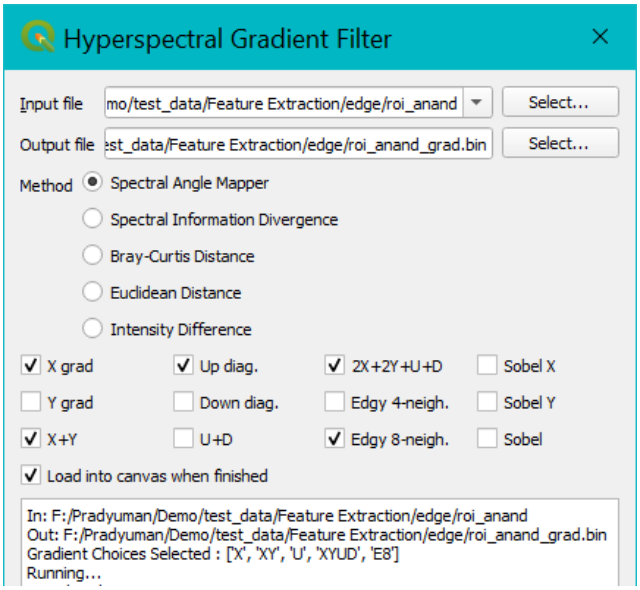


Figure. User interface for Gradient Filter

Input Arguments

Method: Distance/difference measuring methods between pixel vectors to generate filters. It can be spectral or intensity driven method.

I/O PARAMETERS

TYPE	DATA FORMAT	DATA TYPE	VALUE RANGE	REMARK
Input				
Input File	GeoTiff / ENVI			
Output				
Output File	ENVI			Creates a layer for each type of gradient selected.

3 PREPROCESSING

Overview

This is an extension of the Hyperspectral edge detection module. It can calculate up to 9 different hyperspectral gradients using five different spectral distance measurs. Supported spectral distance measures are Spectral Angle Mapper, Spectral Information Divergence, Bray-Curtis Distance, Euclidean Distance, Intensity Distance. 9 gradient directions are listed below

X grad: gradient along x axis,

Y grad: gradient along y axis

X+Y: sum of X and Y gradient

Up diag.: Gradient up

Down diag.: gradient down

U+D: sum of up and down gradients

2X+2Y+U+D: weighted average of the x,y,up and down gradient. X and Y gradients are weighted heavier than the diagonals

Edgy 4-neigh: sum of distance between the central pixels and its 4 closest neighbors (left, right, up and down)

Edgy 8-neigh: weighted sum of distance between central pixel and all its 8 neighbors. The four closest pixels are weighted double.

Sobel X

Sobel Y

Sobel XY

It is to be noted that edge map generated using 2X+2Y+U+D gradient tends to give a very good input for hyperspectral image segmentation.

3.3.3 EXTENDED-MORPHOLOGICAL FILTERS

QGIS 3.XX → Hyperspectral→ Preprocessing → Feature Extraction → Extended-Morphological Filters

3 PREPROCESSING

User Interface

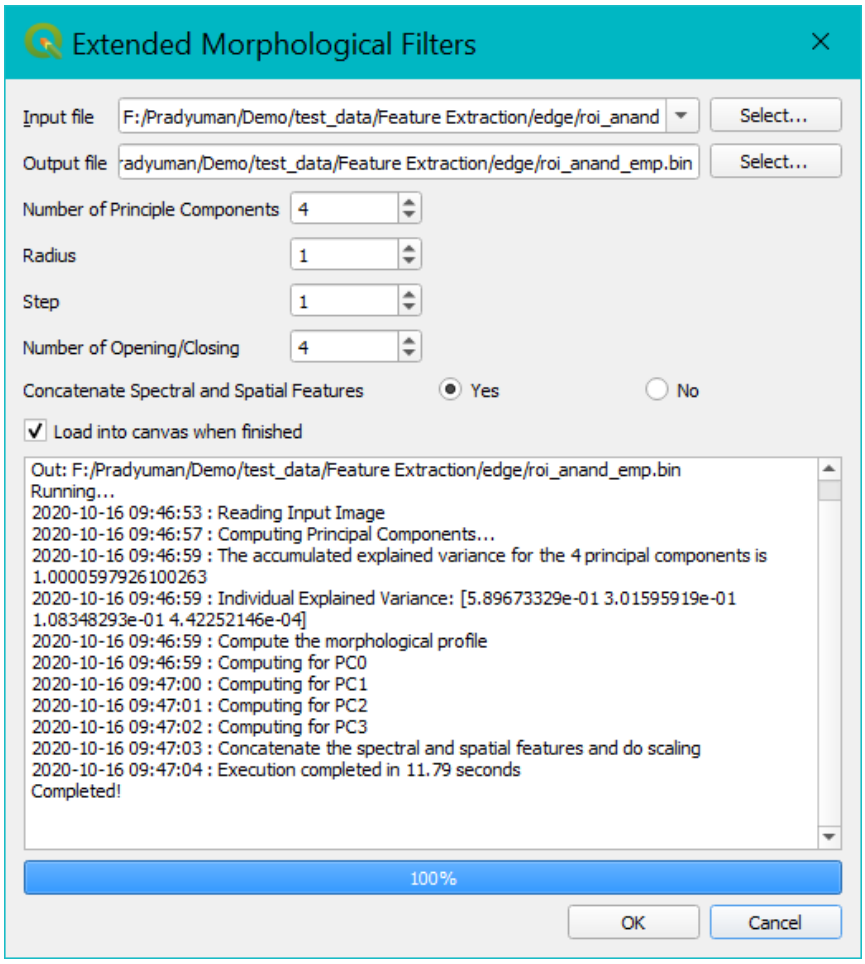


Figure. User interface for Extended morphological filters

Input Arguments

Number of Principal Components: Define the number of principal component to use for the process of extracting features. The larger the number of principal component the longer it will take for the process to complete. Default value should be sufficient for majority of the problems as such.

Radius: Radius of the circular structuring element in the filter.

Step: Step of the moving window after operation on a pixel.

Number of Opening/Closing: Number of times the process of dilation/erosion applied on data.

Concatenate Spectral and Spatial Features: Activate whether to stack the result in one file or save the extracted features as separate file.

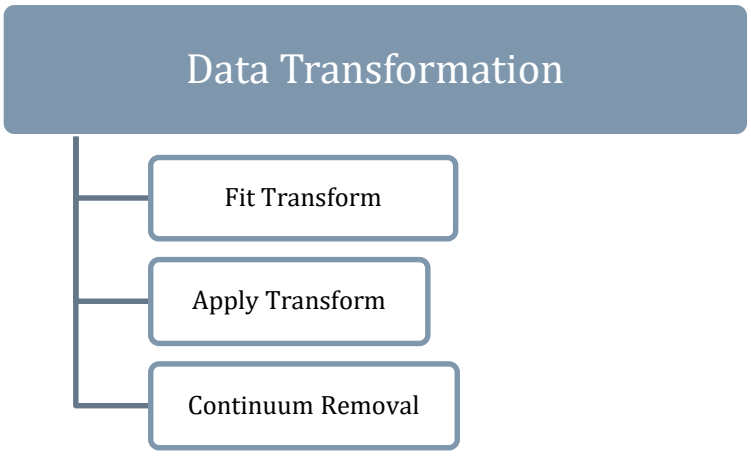
I/O PARAMETERS

TYPE	DATA FORMAT	DATA TYPE	VALUE RANGE	REMARK
Input				
Input File	GeoTiff / ENVI			
Number of principal components		Integer	> 0	
Radius		Integer	> 0	
Step		Integer	> 0	
Number of Opening/Closing		Integer	> 0	
Output				
Output File	ENVI			

Overview

Morphological operators like dilate, erode, open, close can be applied through image filtering to grow or shrink image regions, as well as to remove or fill-in image region boundary pixels. These basic operators process objects in the input image based on the characteristics encoded in the selected structuring element. Structuring element used here is circular.

3.4 FEATURE TRANSFORMATION



3.4.1 FIT TRANSFORM

QGIS 3.XX → Hyperspectral→ Preprocessing → Feature Extraction → Fit Transform

User Interface

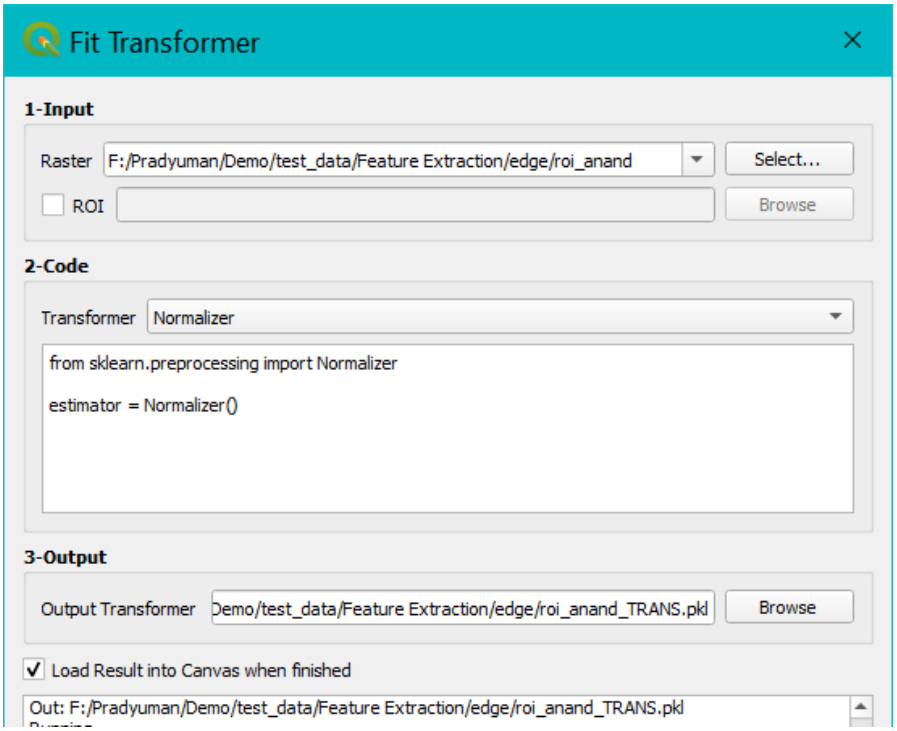


Figure. User interface for fit transformation

Input Arguments

Transformer: Select from the five different transformations to generate parameters for the model from the training dataset.

I/O PARAMETERS

TYPE	DATA FORMAT	DATA TYPE	VALUE RANGE	REMARK
Input				
Input File	GeoTiff / ENVI			Training dataset

Output				
Output File	*.pkl			Contains model parameters

Overview

Transformation of data is one among the most commonly used steps in machine learning and other optimization techniques. AVHYAS has several transformation methods available. It provides an interface to scikit-learn algorithms on transformation. The usual way to apply these methods is to use a Fit… algorithm first and then apply it to an image. So two modules are provided one for the fit and the other for the apply.

This module allows user to generate transformation parameters over a training dataset. Region of interest can be given on the input dataset. Five transformers are provided to select from, namely, StandardScaler (standardize features by removing the mean and scaling to unit variance), MinMaxScaler (transform features by scaling each feature to a given range), MaxAbsScaler (scale each feature by its maximum absolute value), Normalizer (normalize samples individually to unit norm), RobustScaler (scale features using statistics that are robust to outliers).

3.4.2 APPLY TRANSFORM

QGIS 3.XX → Hyperspectral→ Preprocessing → Feature Extraction → Apply Transform

User Interface

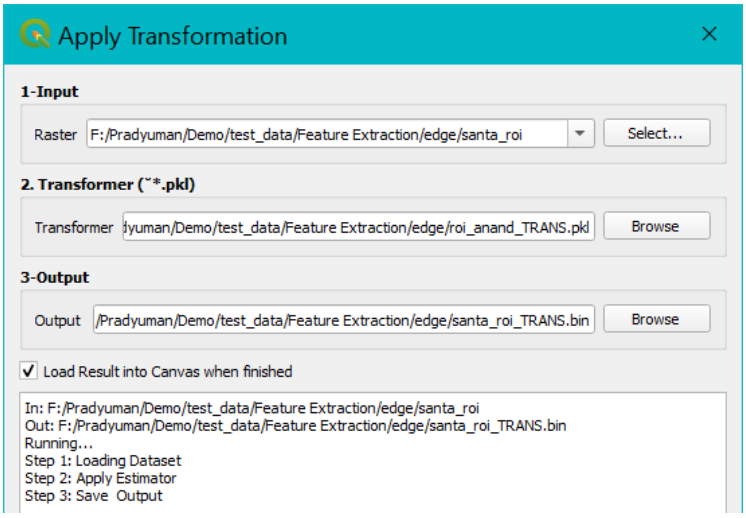


Figure. User interface for Apply Transformation

3 PREPROCESSING

I/O PARAMETERS

TYPE	DATA FORMAT	DATA TYPE	VALUE RANGE	REMARK
Input				
Input File	GeoTiff / ENVI			
Transformer	*.pkl			Having transformation parameters
Output				
Output File	ENVI			

Overview

See also

You can find all the available information on transformation algorithm here

3.4.3 CONTINUUM REMOVAL

QGIS 3.XX → Hyperspectral→ Preprocessing → Feature Extraction → Continuum Removal

User Interface

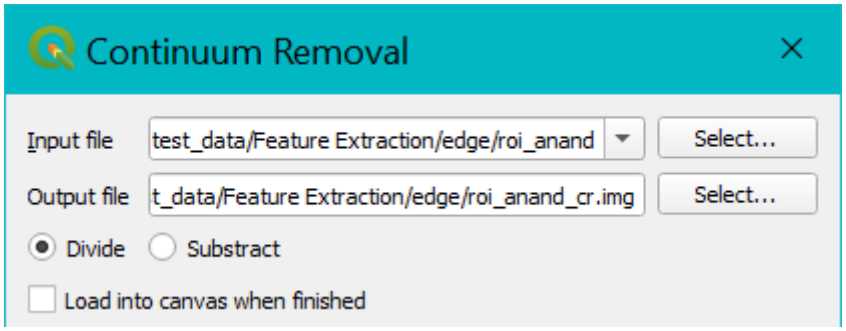


Figure. User interface for Continuum Removal

Input Arguments

Divide/Subtract: Two methods are available to normalize spectra using a convex hull.

3 PREPROCESSING

I/O PARAMETERS				
TYPE	DATA FORMAT	DATA TYPE	VALUE RANGE	REMARK
Input				
Input File	GeoTiff / ENVI			
Output				
Output File	ENVI			Continuum removed data at each pixel.

Overview

For quantification of the absorption features in the spectra the overall concave shape of a spectra should be removed. This normalization procedure is referred to as 'continuum removal' or 'convex-hull' transform and allows comparison of spectra that are acquired by different instruments or under different light conditions. A convex hull over the spectra is created using straight line-segments that connect the local spectra maxima which is used to normalize it by either subtraction or division.

3.5 Savitzky Golay Filter

QGIS 3.XX → Hyperspectral→ Preprocessing → Feature Extraction → Savitzky Golay Filter

User Interface

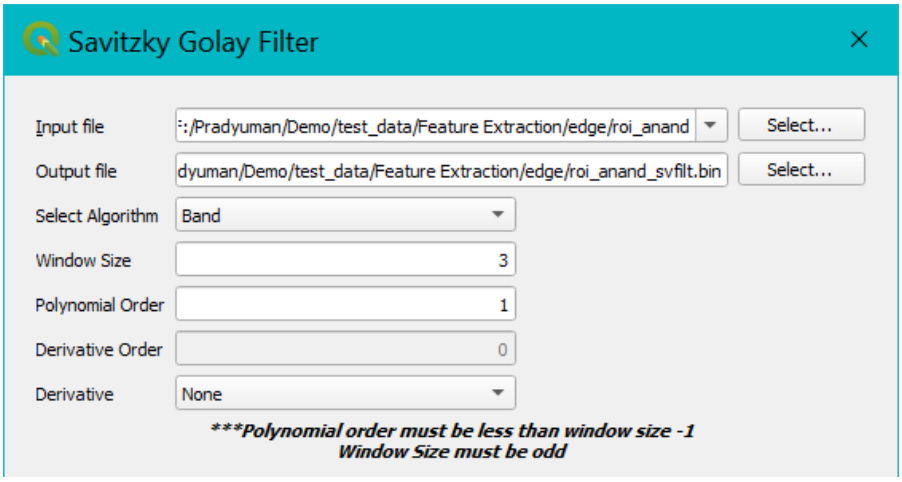


Figure. User interface for Savitzky Golay Filter

Input Arguments

Select Algorithm: Can be applied on a single band or over spectrally over all the bands in a pixel.

Window Size: The length of the window. Must be an odd integer.

Polynomial Order: The order of the polynomial used in the filtering. Must be less than window size – 1.

Derivative Order: The order of the derivative to compute. Active in spectral algorithm only.

Derivative: Direction of the derivative to compute. Active only in band algorithm. It can be *None, col, row, both*.

I/O PARAMETERS

TYPE	DATA FORMAT	DATA TYPE	VALUE RANGE	REMARK
Input				
Input File	GeoTiff / ENVI			
Window Size		Integer	≥ 1	Must be odd integer
Polynomial Order		Integer	> 0	Must be less than Window size - 1
Derivative Order		Integer		
Output				
Output File	ENVI			Continuum removed data at each pixel.

Overview

Savitzky Golay filter can be applied either on a band or over a spectrum for the purpose of smoothing the data. This filter locally smooths the data by convoluting a specified order polynomial to samples in the window in least square sense. The window size should be an odd integer and the order of the fitting polynomial must be less than window size-1. The derivative direction while applying over a band can be provided either in column direction or row direction or even in both directions.

3 PREPROCESSING

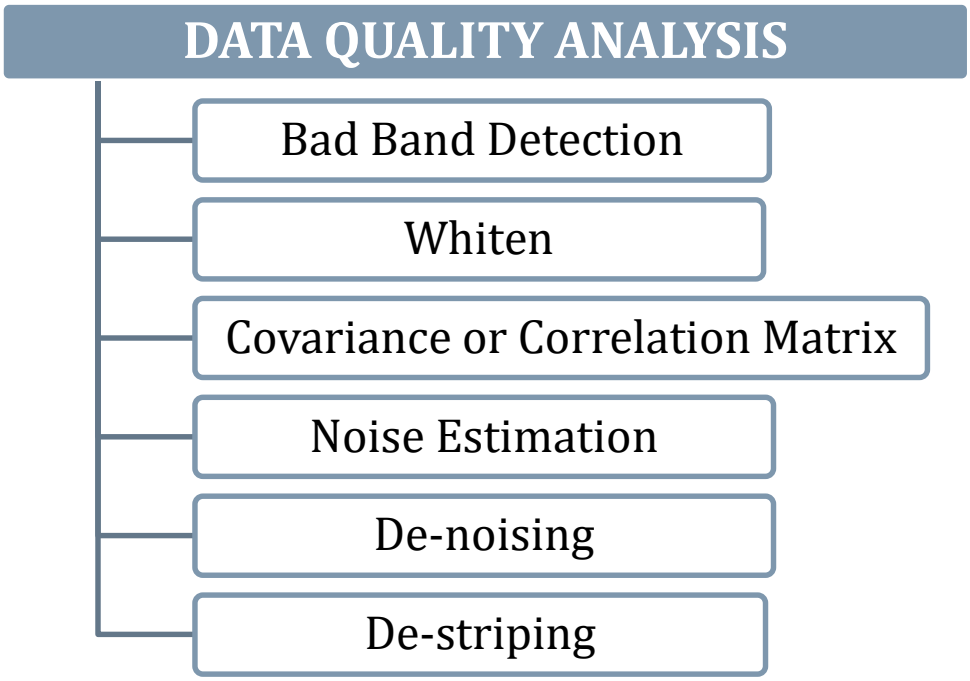
Anand roi RGB

Anand roi RGB after applying Savitzky Golay filter

Reference:

R. W. Schafer, "What Is a Savitzky-Golay Filter? [Lecture Notes]," in *IEEE Signal Processing Magazine*, vol. 28, no. 4, pp. 111-117, July 2011, doi: 10.1109/MSP.2011.941097.

4 DATA QUALITY ANALYSIS



4 DATA QUALITY ANALYSIS

4.1 BAD BAND DETECTION

QGIS 3.XX → Hyperspectral→ DATA QUALITY ANALYSIS → BAD BAND DETECTION

User Interface

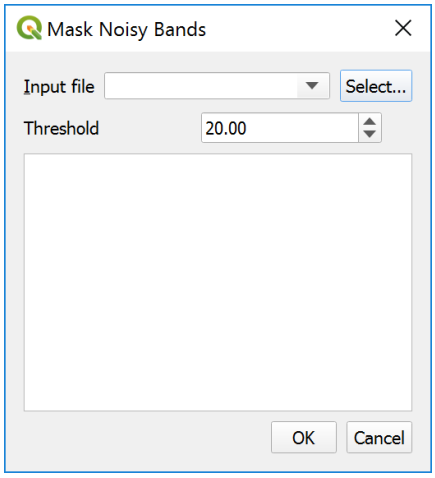


Figure. User-Interface for Spatio-Spectral Segment

Overview

This module user signal to local busyness ratio (SLBR) for automatically detecting noisy bands. User needs to define the threshold for masking and the bad bands are recorded in the bad band list (bbl) in the output metadata. For this module only the header information is changed. A graph will be plotted at the end of the result so as user has a fair bit of an idea if they want to change the threshold value and re run the module. A higher threshold will result in more bands being marked as bad. The individual value of SLBR will also be displayed in the log window for user reference.

Reference

- Hyppy3 Mask noisy band module by Wim Bakker

4.2 WHITENING

QGIS 3.XX → Hyperspectral→ DATA QUALITY ANALYSIS → WHITEN

User Interface

4 DATA QUALITY ANALYSIS

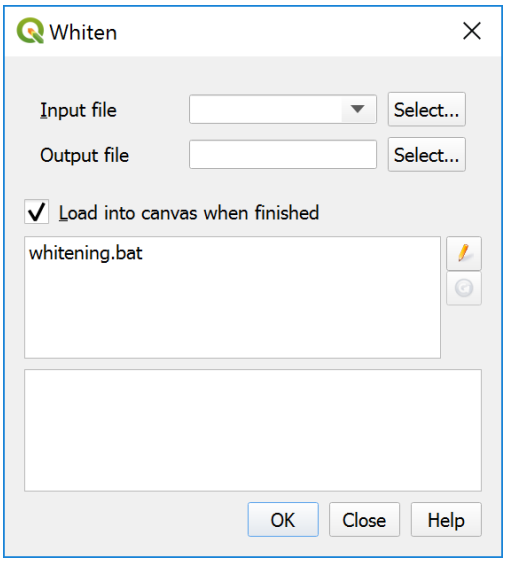


Figure. User-Interface for Whitening Transform

Input Arguments

OVERVIEW

Compute sample covariance matrix ($\mathbf{\Sigma}_N$) of the Normalized data, then calculate orthonormalized eigenvector matrix **E** from the covariance matrix, and diagonal matrix of its eigenvalues, **D*n***. Then Whitening can be defined as,

$$\mathbf{F} = \mathbf{E}\mathbf{D}_n^{-1/2}$$

4.3 COVARIANCE OR CORRELATION MATRIX

QGIS 3.XX → Hyperspectral→ DATA QUALITY ANALYSIS → COVARIANCE OR CORRELATION MATRIX

4 DATA QUALITY ANALYSIS

User Interface

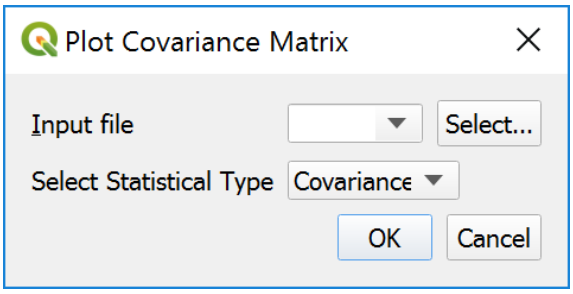


Figure. User-Interface for Covariance/Correlation Matrix Plot

Input Arguments

OVERVIEW

Assume that $\boldsymbol{x}$ is a set of L-dimensional image pixel vectors and **u** is the mean of the sample pool obtained by,

$$\boldsymbol{u} = \left(\frac{1}{N}\right)\sum_{i=1}^{N}\boldsymbol{x}_i$$

We set **X** to be the sample data matrix formed by $\boldsymbol{X} = (\boldsymbol{x}1, \boldsymbol{x}2, \ldots, \boldsymbol{x}_N)$. Then a sample covariance matrix

$$\sum = \left(\frac{1}{N}\right)[XX^T] = \left(\frac{1}{N}\right)\left[\sum_{i=1}^{N}(\boldsymbol{x}_i - \boldsymbol{u})(\boldsymbol{x}_i - \boldsymbol{u})^T\right]$$

The correlation coefficient matrix **R** can be easily converted from the covariance matrix **Σ**:

$$\boldsymbol{R} = \boldsymbol{D}_{\boldsymbol{\sigma}}^{-\frac{1}{2}}\sum\boldsymbol{D}_{\boldsymbol{\sigma}}^{-\frac{1}{2}}$$

Where, Dσ is the diagonal matrix with variance of each band as the diagonal element.

4.4 NOISE ESTIMATION

QGIS 3.XX → Hyperspectral→ DATA QUALITY ANALYSIS → NOISE ESTIMATION

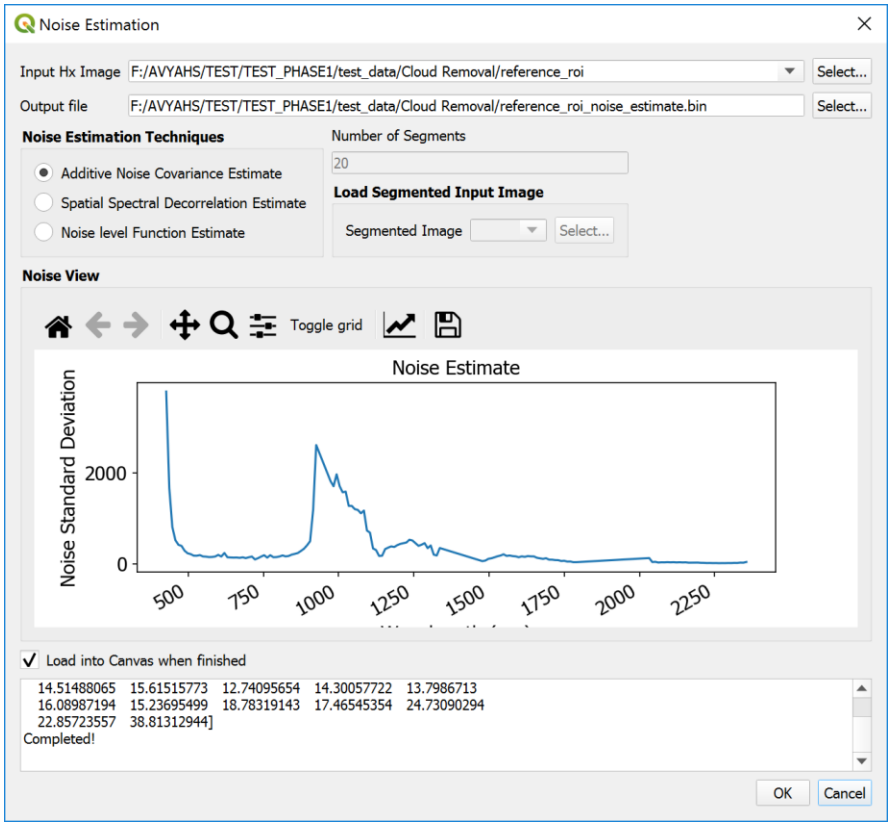


Figure. User-Interface for noise estimation

Input Arguments

Noise Estimation Technique: Select the required Noise Estimation Algorithm.

Spatial-Spectral Decorrelation Estimate and ***Noise Level Function Estimate*** require Segmented raster image of the Input Hx image. (Segmentation result can be generated from ***QGIS 3.XX → Hyperspectral→ Classification → Segmentation → Spatio-Spectral-Segment,*** where small segments should be removed using the segment area threshold option).

No. of segments: This parameter selects the required number of segments for computing the noise from the image.

Segmented input image: this options takes the segmented input image obtained from *QGIS 3.XX → Hyperspectral → Classification → Segmentation → Spatio-Spectral-Segment*

I/O PARAMETERS

TYPE	DATA FORMAT	DATA TYPE	VALUE RANGE	REMARK
Input				
Input File	GeoTiff / ENVI			Radiance Image
Number of Segments		Integer	one to total number of segments in the segmented image	
Segmented Image	Geotiff/ENVI			Segmented image (single channel) obtained from the segmentation algorithm
Output				
Output File	ENVI			Raster file with integer values for each segment

OVERVIEW

Additive Noise Covariance Estimate:

Let Y be an L×N matrix, where N is the number of observed spectral vectors and L is the number of bands. Then define a matrix $Z=Y^T$, which is an N×L matrix, a N×1 vector, Z_i contains the data read by the hyperspectral sensor at the i^{th} band for all image pixels.
If Zi is given by the linear regression equation,

$$Z_i = Zdi\beta_i + \varepsilon_i$$

where, Zdi - data matrix of dimensions N×(L - 1)

β-regression vector of size (L-1)×1

ϵ- noise vector of size N×1

The least square estimate for the regression vector is,

$$\beta = (Z^T\ Zdi)^{-1}\ Z_{di}\ Z_i$$

4 DATA QUALITY ANALYSIS

The noise estimates, , are given by the equation,

$$Rn = [\varepsilon 1, \varepsilon 2, \ldots . \varepsilon N]^T [\varepsilon 1, \varepsilon 2, \ldots ., \varepsilon N]/N$$

Spatial-Spectral Decorrelation (SSDC) estimate:

In SSDC (Gao and Lianru, 2013) the high between-band (spectral) and within-band (spatial) correlations are used to de-correlate an image data via linear regression, and the remaining unexplained residuals are the estimates of noise. In this method, the image also is divided into spatial-spectral homogeneous segments. The pixel value at in band can be predicted by its spatial and spectral neighbors and the residual is calculated as,

$$r_{i,j,k} = x_{i,j,k} - \hat{x}_{i,j,k}$$

$$\hat{x}_{i,j,k} = a + bx_{i,j,k-1} + cx_{i,j,k+1} + dx_{p,k}$$

Where, $\hat{x}$ expresses estimated center-pixel value computed from the neighborhood pixel values. **x** represents the observed pixel value at the center pixel. The local standard deviation (LSD) of all the residuals in a block is computed using,

$$LSD = \left[\frac{1}{w^2 - 4} \sum_{i=1}^{w} \sum_{j=1}^{w} r_{i,j,k}^2 \right]^{\frac{1}{2}}$$

Noise Level Function Estimate:

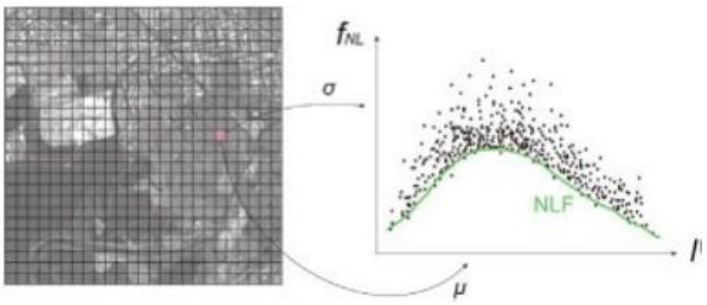
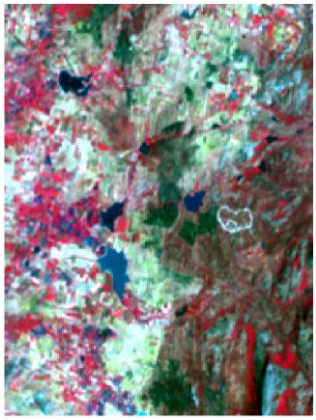
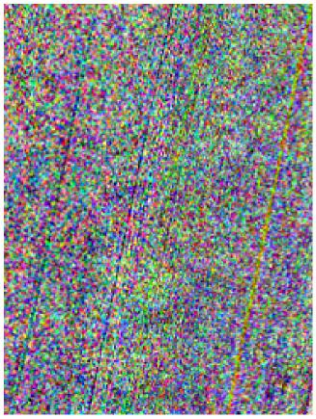


Figure. Noise level function

4 DATA QUALITY ANALYSIS

The noise level function (NLF) is a continuous function describing the noise level as a function of image brightness (Yokoya, Naoto, and Akira Iwasaki, 2010). NL is defined as the standard deviation with respect to image intensity. We estimate the NLFs in all bands by the segmentation based noise level estimation as follows. In all bands, each image component is divided into segments. Then, for all segments, the mean and standard deviation are calculated.

Results

 Input Hyperion Image (750nm, 650nm, 550nm Composite)	 Hyperion Noise Estimated (NIR, Red, Green Composite)

REFERENCE

Gao, Lianru, et al. "A comparative study on linear regression-based noise estimation for hyperspectral imagery." IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing 6.2 (2013): 488-498.

Yokoya, Naoto, and Akira Iwasaki. "A maximum noise fraction transform based on a sensor noise model for hyperspectral data." Proc. 31th Asian Conf. Remote Sens., 2010.

4.5 DE-NOISING

QGIS 3.XX → Hyperspectral→ DATA QUALITY ANALYSIS → DE-NOISING

4.6 DE-STRIPING

QGIS 3.XX → Hyperspectral→ DATA QUALITY ANALYSIS → DE-STRIPING

4 DATA QUALITY ANALYSIS

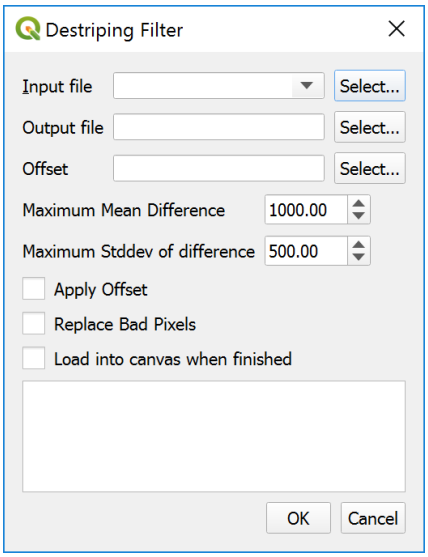
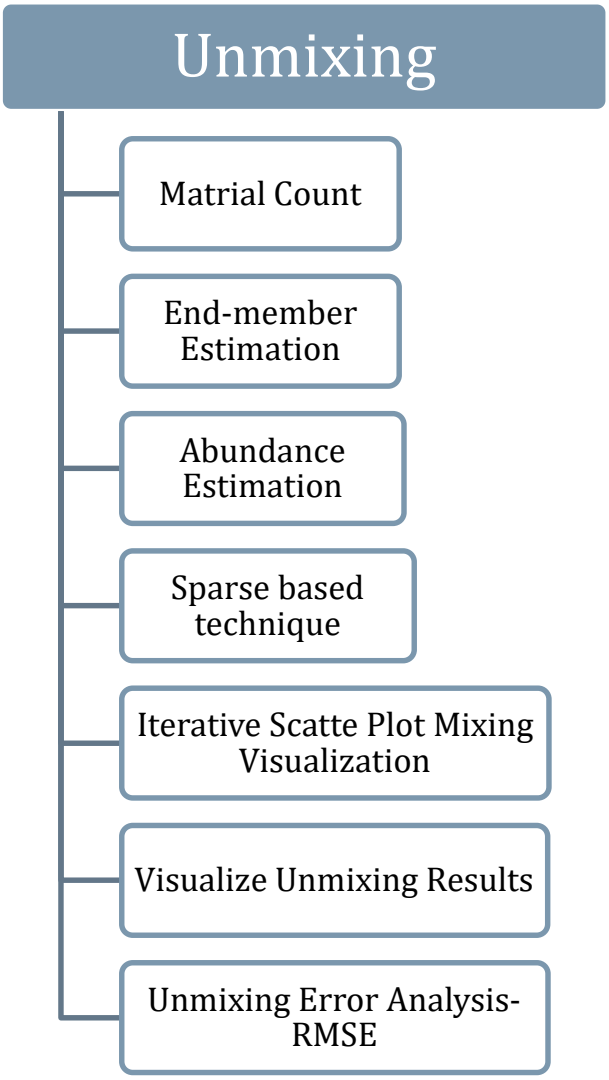


Figure. User-Interface for Spatio-Spectral Segment

5 UNMIXING MODULE



5 UNMIXING MODULE

Overview

Hyperspectral sensors acquire the electromagnetic energy in a high number of spectral channels under an instantaneous field of view. These rich datacubes allow us identifying materials in the scenes with high detail. However, high spectral resolution is achieved at the price of spatial resolution due to technical constraints, and the resulting pixels are spatially coarse. A direct consequence is that the spectral vectors acquired are no longer pure but rather mixtures of the spectral signatures of the materials present in the scene. In addition, one could argue that mixed pixels always exist as they are fundamentally due to the heterogeneity of the landscape and not only because of the characteristics of the sensor. In this scenario, only a small fraction of the available pixels can be considered as pure, i.e., composed by a single material and thus representative of its true spectral signature. The field of spectral mixture analysis (or spectral unmixing for short) is devoted to both identifying the most probable set of pure pixels (called endmembers) and estimating their proportions (called abundances) in each of the image pixels. In fact, when the endmembers have been identified, every single pixel in the image can be synthesized as a linear (or nonlinear) combination of them. The process of unmixing allows many interesting applications which are mainly related to subpixel target detection [Chang, 2003, Shaw and Manolakis, 2002], crop and mineral mapping [Keshava and Mustard, 2002], and multitemporal monitoring [Zurita-Milla et al., 2011] etc.

SPECTRAL MIXTURE ANALYSIS

A. **Dimensionality reduction:** Spectral unmixing intrinsically assumes that the dimensionality of hyperspectral data is lower and can be expressed in terms of the endmembers (a kind of basis). Some methods require a previous dimensionality reduction, either feature selection or extraction. The most common approaches use a physically-based selection of the most information bands or rely on the application of principal component analysis (PCA) or the minimum noise fraction (MNF) transformation.
B. **Endmember extraction:** The second step deals with the search for a proper vector basis to describe all the materials in the image. Many approaches exist in the literature but, roughly speaking, they can be divided into two groups: the first one tries to find the most extreme spectra, which are the purest and those better describing the vertices of the simplex; the second looks for the spectra that are the most statistically different.
C. **Abundance estimation:** The third and last step is a model inversion that typically exploits linear or nonlinear regression techniques for estimating the mixture of materials, called abundance, in each image pixel. A wide variety of methods, from linear regression

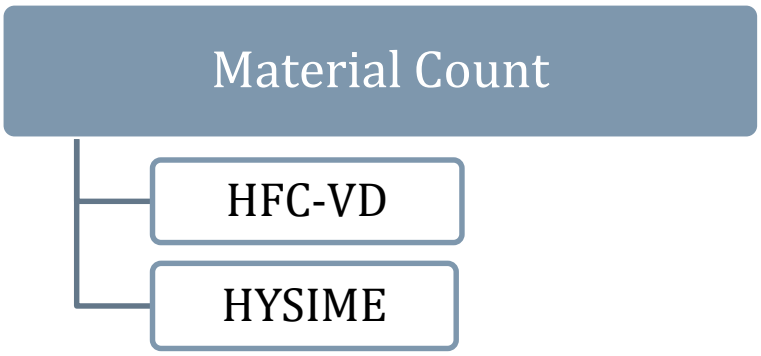
to neural networks and support vector regression, is commonly used for this purpose. The inversion step consists in solving a constrained least-squares problem which minimizes the residual between the observed spectral vectors and the linear space defined by the endmembers.

References:

- C.-I. Chang. *Hyperspectral Imaging: Techniques for Spectral Detection and Classification*. Plenum Publishing Co., 2003.
- G. Shaw and D. Manolakis. Signal processing for hyperspectral image exploitation. *IEEE Signal Proc. Magazine*, 50: 12–16, Jan 2002.
- N. Keshava and J. F. Mustard. Spectral unmixing. *IEEE Sign. Proc. Mag.*, 19 (1): 44–57, 2002.
- R. Zurita-Milla, L. Gómez Chova, L. Guanter, J. G. P. W. Clevers, and G. Camps Valls. Multitemporal unmixing of medium-spatial-resolution satellite images: A case study using MERIS images for land-cover mapping. *IEEE Trans. Geosc. Rem. Sens.*, 46 (4), 2011.

5 UNMIXING MODULE

5.1 MATERIAL COUNT



5.1.1 HFC-VD

QGIS 3.XX → Hyperspectral→ Unmixing → Material Count → HFC-VD

User Interface

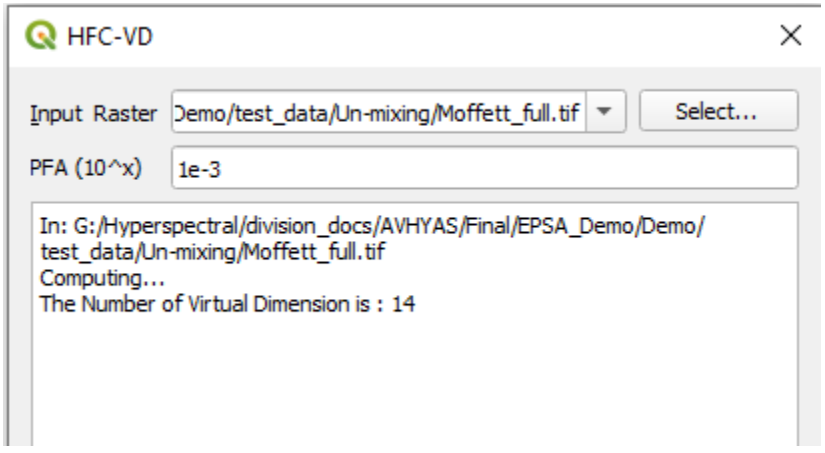


Figure. User-Interface for material count (HFC-VD)

5.1.2 HYSIME

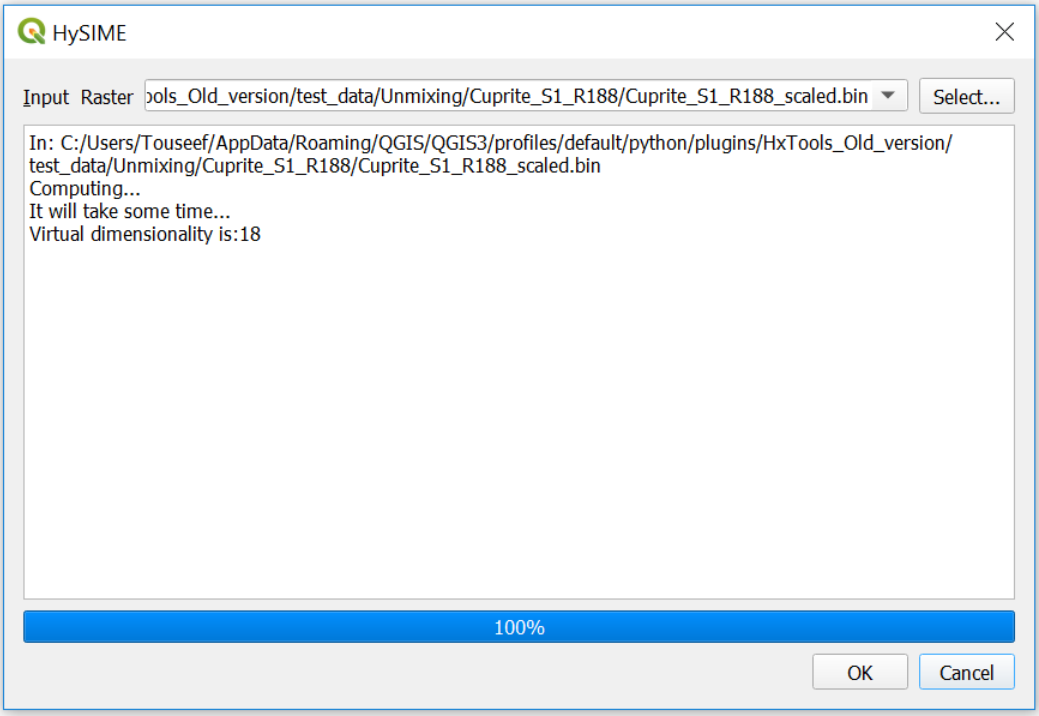


Figure. User-Interface for material count (HySime)

I/O PARAMETERS

Type	Data format	DATA type	value range	Remark
Input				
Input File	GeoTiff / ENVI			Bad band Removed Data, Radiance/Reflectance
Output				
Output File				Virtual dimensionality of data is displayed on the user interface

Overview

5 UNMIXING MODULE

Material count or number of endmember estimation is the first step in the spectral unmixing analysis of hyperspectral datacube. It is commonly accepted that such a number is lower than the number of bands. This mathematical requirement means that the spectral vectors lie in a low-dimensional linear subspace. The identification of such dimensionality would not only reveal the intrinsic dimensionality of the data but also would reduce the computational complexity of the unmixing algorithms because data could be projected onto this low-dimensional subspace. Many methods have been applied to estimate the intrinsic dimensionality of the subspace.

Most of the methods involve solving eigen problems. While principal component analysis (PCA) [Jollife, 1986] looks for projection of data that summarize the variance of the data, the minimum noise fraction (MNF) [Green et al., 1988a, Lee et al., 1990] seeks for the projection that optimizes the signal-to-noise ratio. Recently, the hyperspectral signal identification by minimum error (HySime) method [Bioucas-Dias and Nascimento, 2005] is a very efficient method to estimate the signal subspace in hyperspectral images. The method also solves an eigen decomposition problem and no parameters must be adjusted, which makes it very convenient for the user. Essentially, the method estimates the best signal and noise eigenvectors that represent the subspace in mean-square-error (MSE) terms.

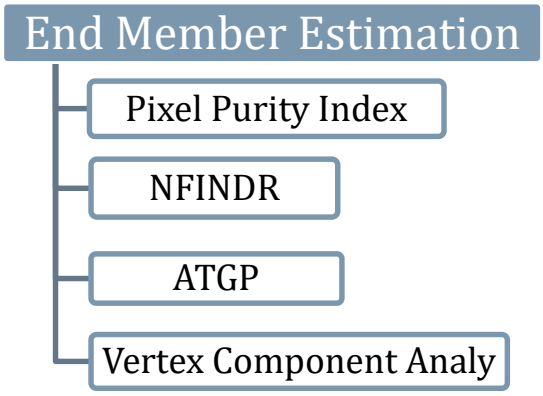
It is worth noting the work of Harsanyi et al. [1993], in which a Neyman-Pearson detection method (called HFC) was successfully introduced and evaluated. The method determines the number of spectral endmembers in hyperspectral data and implements the so-called virtual dimensionality (VD) [Chang and Du, 2004], which is defined as the minimum number of spectrally distinct signal sources that characterize the hyperspectral data. Essentially, the estimated dimensionality by HFC reduces to the highest number for which the correlation matrix has smaller eigenvalues than the covariance matrix. A modified version of HFC is the noise-whitened HFC (NWHFC), which includes a noise-whitening process as a preprocessing step to remove the second-order noise statistical correlation [Chang and Du, 2004].

References

- I.T. Jolliffe. Principal Component Analysis. Springer-Verlag, 1986.
- A. Green, M. Berman, P. Switzer, and M. D. Craig. A transformation for ordering multispectral data in terms of image quality with implications for noise removal. IEEE Trans. Geosc. Rem. Sens., 26 (1): 65–74, 1988a.
- J. B. Lee, S. Woodyatt, and M. Berman. Enhancement of high spectral resolution remote-sensing data by noise-adjusted principal components transform. IEEE Trans. Geosc. Rem. Sens., 28 (3): 295–304, 1990.

- J. M. Bioucas-Dias and J. Nascimento. Hyperspectral subspace identification. IEEE Trans. Geosc. Rem. Sens., 46 (8): 2435–2445, 2005.
- J. Harsanyi, W. Farrand, and C.-I.Chang. Determining the number and identity of spectral endmembers: An integrated approach using Neyman-Pearson eigen thresholding and iterative constrained RMS error minimization. In Proc. 9th Thematic Conf. Geologic Remote Sensing, 1993.
- C.-I Chang and Q. Du. Estimation of number of spectrally distinct signal sources in hyperspectral imagery. IEEE Trans. Geosc. Rem. Sens., 42 (3): 608–619, 2004.

5.2 END MEMBER EXTRACTION



5.2.1 PIXEL PURITY INDEX (PPI)

QGIS 3.XX → Hyperspectral→ Unmixing → Endmember Extraction → PPI

User Interface

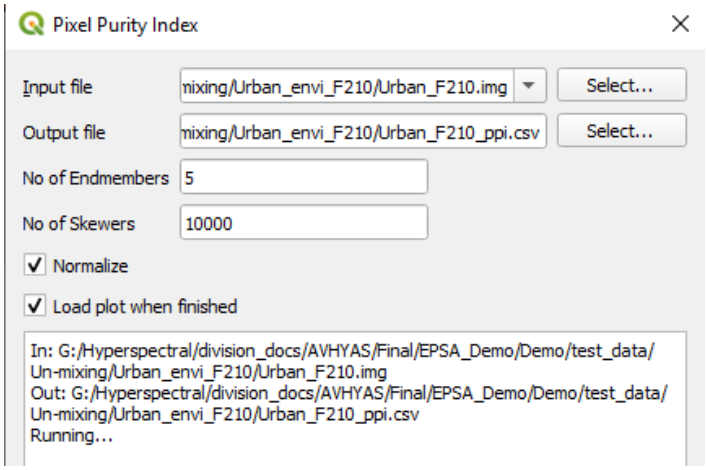


Figure. User interface for PPI

Input Arguments

No. of Endmembers: define number of endmember for estimation.

No. of Skewers: By default 10000.

Normalize: Normalize hx data between 0-1.

5 UNMIXING MODULE

I/O PARAMETERS

Type	Data format	DATA type	value range	Remark
Input				
Input File	GeoTiff / ENVI			Bad band Removed Data, Radiance/Reflectance
Output				
Output File				Virtual dimensionality of data is displayed on the user interface

Overview:

One of the most successful approaches has been the Pixel Purity Index or PPI (Boardmanet al., 1995), which is based on the geometry of convex sets (Ifarraguerri and Chang, 1999). PPI considers spectral pixels as vectors in an N-dimensional space. A dimensionality reduction is first applied to the original data cube by using the Minimum Noise Fraction (MNF). The algorithm proceeds by generating a large number of random N-dimensional vectors, also called (skewers) (Theiler et al.,2000), through the dataset. Every data point is projected onto each skewer, along which position it is pointed out. The data points which correspond to extreme values in the direction of a skewer are identified and placed on a list. As more skewers are generated, the list grows, and the number of times a given pixel is placed on this list is also tallied.The pixels with the highest tallies are considered the purest ones, since a pixel count provides a «pixel purity index». It is important to emphasize that the PPI algorithm does not identify a final list of endmembers. PPI was conceived not as a solution, but as a guide; in fact, the author proposed comparing the pure pixels with target spectra from a library and successively projecting the data to lower dimensional spaces while endmembers were identified. There are several interactive software tools oriented to perform this task, but the supervised nature of such tools leads to the fact that the analysis of a scene usually requires the intervention of a highly trained analyst. Then, interactive solutions may not be effective in situations where large scenes must be analyzed quickly as a matter of routine. In addition, randomness in the selection of skewers has been identified by some authors as a shortcoming of the algorithm. The original implementation of PPI proposes the use of unitary vectors as skewers in random directions of the N-dimensional space. This implementation may be improved by a careful selection of existing vectors to skew the

dataset. In-telligent selection of skewers may result in amore efficient behavior of the algorithm. Some Tools based on variations of PPI concepts have been proposed (Theiler et al., 2000).

Result

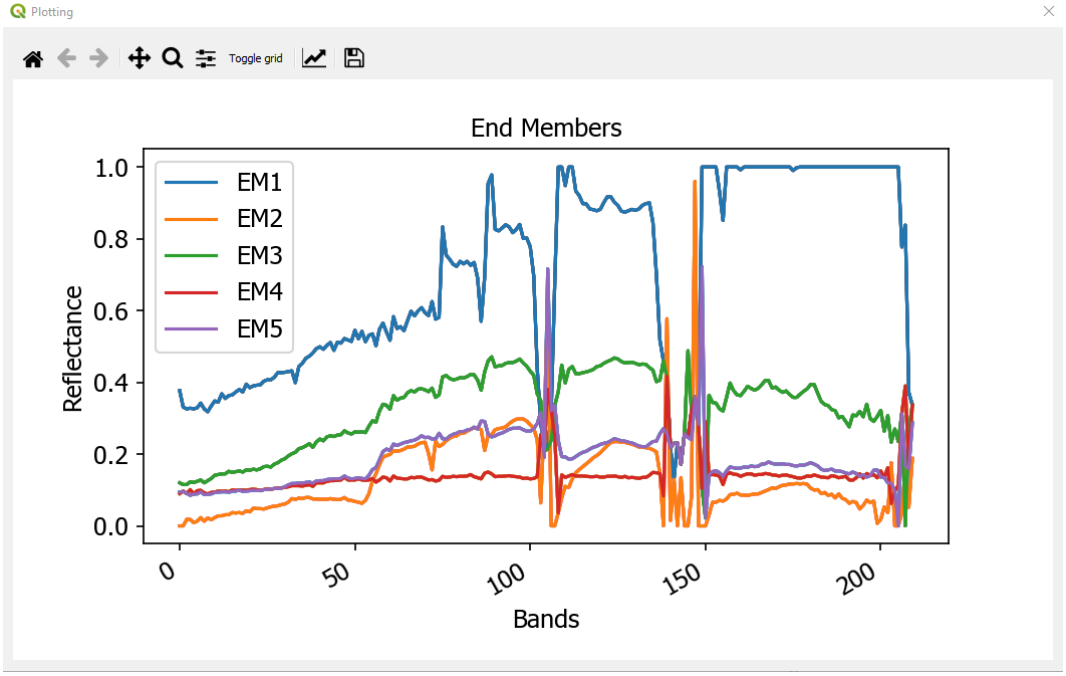


Figure:Five endmembers extracted using PPI algorithm.

References:

- BOARDMAN, J.W., F.A. KRUSE and R.O. GREEN (1995): Mapping Target Signatures via Partial Unmixing of AVIRIS Data, in Summaries of the V JPL Airborne Earth Science Workshop.
- IFARRAGUERRI, A. and C.-I. CHANG (1999): Multispectral and hyperspectral image analysis with convex cones, IEEE Trans. Geosci. Remote Sensing, 37 (2), 756- 770.
- THEILER, J., D.D. LAVENIER, N.R. HARVEY, S.J. PERKINS and J.J. SZYMANSKI (2000): Using blocks of skewers for faster computation of Pixel Purity Index, SPIE Proc., 4132, 61-71

5.2.2 NFINDR

QGIS 3.XX → Hyperspectral→ Unmixing → End Member Extraction → NFINDR

5 UNMIXING MODULE

User Interface

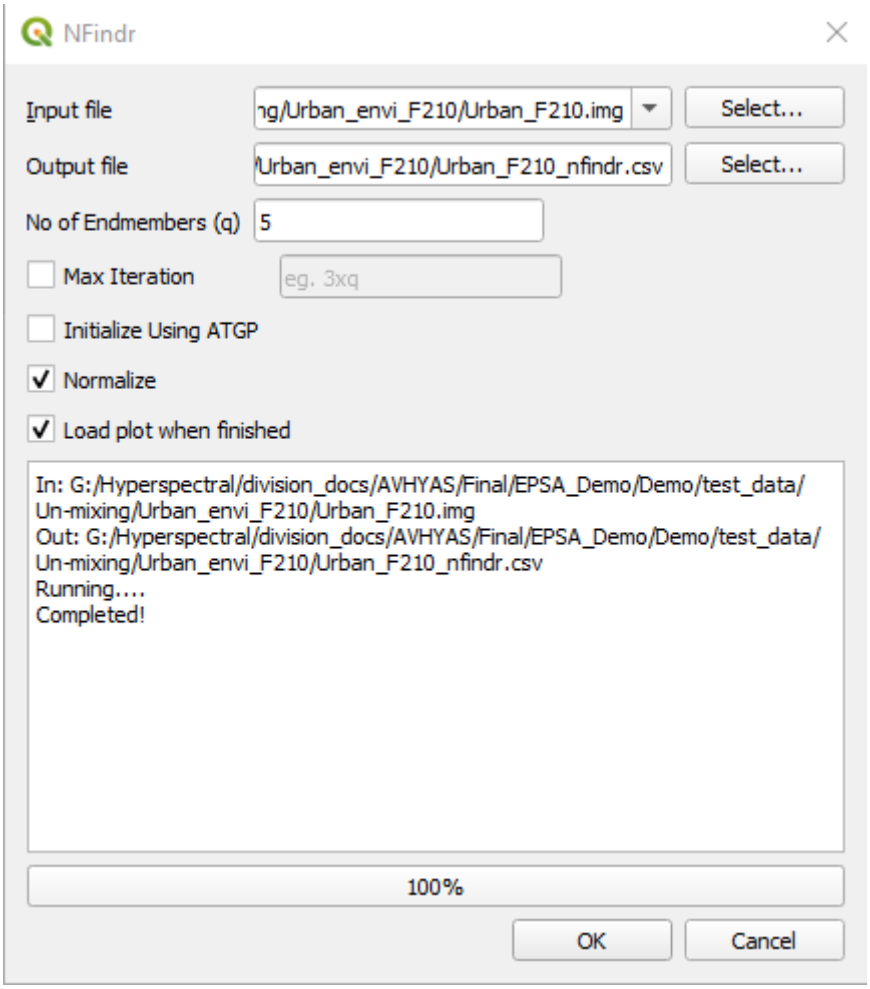


Figure. User-Interface for N-Findr algorithm

Input Arguments

No. of Endmembers: define number of endmember for estimation.

Max Iteration: define maximum iteration

Initialize using ATGP:

Normalize: Normalize hx data between 0-1.

5 UNMIXING MODULE

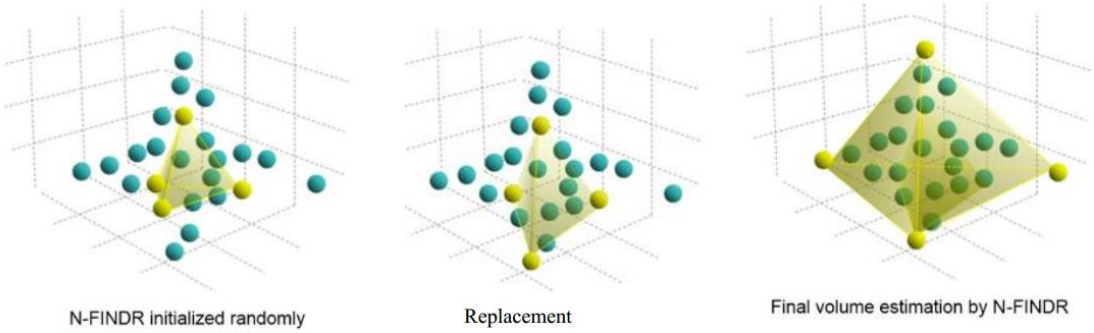
I/O PARAMETERS

Type	Data format	DATA type	value range	Remark
Input				
Input File	GeoTiff / ENVI			Bad band Removed Data, Radiance/Reflectance
Output				
Output File				Virtual dimensionality of data is displayed on the user interface

Overview:

The N-FINDR method (Winter, 1999) is an automated approach that finds the set of pixels which define the simplex with the maximum volume, potentially inscribed within the data-set. First, a dimensionality reduction of the original image is accomplished by MNF. Next, randomly selected pixels qualify as endmembers, and a trial volume is calculated. In order to refine the initial volume estimation, a trial volume is calculated for every pixel in each endmember position by replacing that end-member and recalculating the volume (see in Fig.). If the replacement results in a volume increase, the pixel replaces the endmember. This procedure, which does not require any input parameters, is repeated until there are no replacements of end-members left. It should be noted that both PPI and N-FINDR rely on spectral properties of the data alone, neglecting the information related to the spatial arrangement of pixels in the scene. The N-FINDR method is sensitive to the random selection of an initial set of endmembers. If the initial estimation is appropriate, the algorithm will not require to do many iterations until it reaches the optimum solution. On the contrary, an erroneous initial estimation may lead to a high computational complexity of the algorithm. On the other hand, the algorithm recalculates the volume of the simplex every time a new pixel is incorporated to the endmember set. This property makes the algorithm quite sensitive to noise.

Result



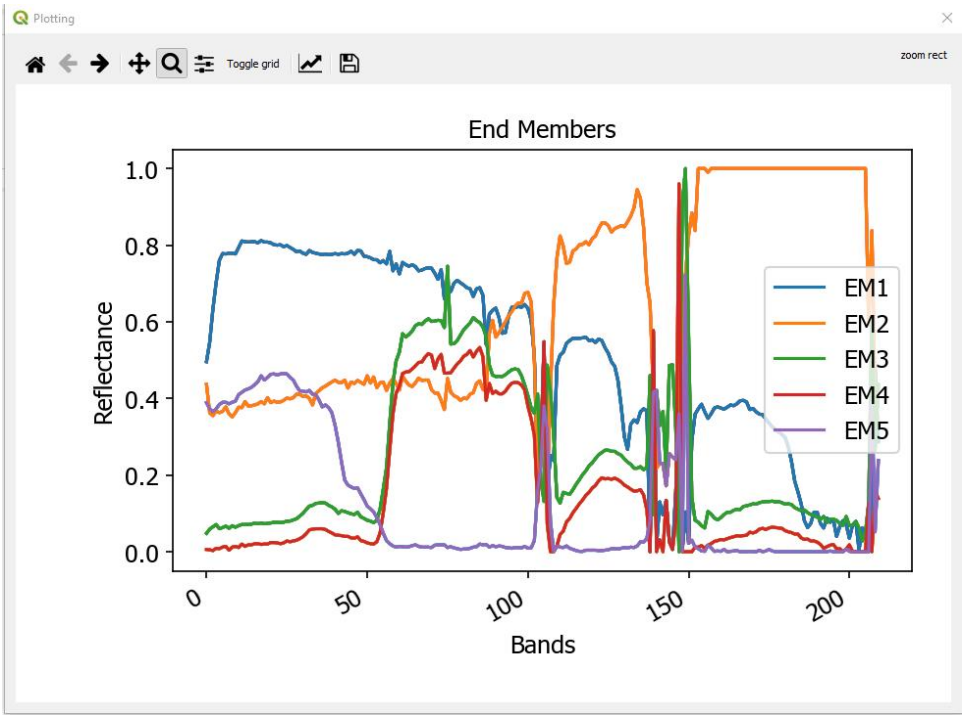


Figure: Five endmembers extracted by N-Findr algorithm

References:

Winter, Michael E. “N-FINDR: An Algorithm for Fast Autonomous Spectral End-Member Determination in Hyperspectral Data.” Proc. SPIE Imaging Spectrometry V 3753, (October 1999): 266–75.

5.2.3 ATGP

QGIS 3.XX →Hyperspectral→ Unmixing → Endmember Extraction → ATGP

5 UNMIXING MODULE

User Interface

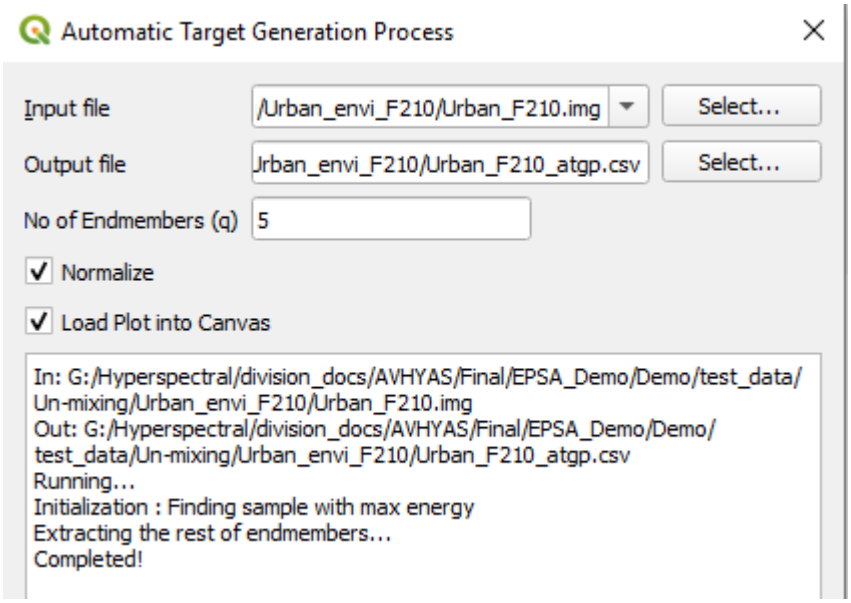


Figure. User-Interface for ATGP algorithm.

Input Arguments

No. of Endmembers: define number of endmember for estimation.

Normalize: Normalize hx data between 0-1.

I/O PARAMETERS

Type	Data format	DATA type	value range	Remark
Input				
Input File	GeoTiff / ENVI			Bad band Removed Data, Radiance/Reflectance
Output				
Output File				Virtual dimensionality of data is displayed on the user interface

Overview:

The development of ATGP primarily came from a need of finding targets of interest in hyperspectral image data when no prior knowledge is available. It implements a nested sequence

of decreasing orthogonal complement subspaces from which a succession of targets of interest can be extracted via finding maximal OPs. Interestingly, as shown in [C-I, Chang, 2013], most of such ATGP generated target pixels turned out to be also endmembers. This is certainly not a coincidence because the concept behind ATGP is actually the same as PPI except for two key differences. PPI requires a very large number of random unit vectors, referred to as skewers, to find maximal/minimal OP compared to ATGP that finds targets of interest from a sequence of OP subspaces with maximal OP. As a result, PPI simultaneously extracts all endmembers, whereas ATGP extracts targets sequentially one at a time. Additionally, PPI takes advantage of random nature in skewers to uncover all possible endmembers as opposed to ATGP which searches for specific target candidates in particular directions by finding the maximal OP in a sequence of OP subspaces. Nonetheless, both PPI and ATGP use the same principle, that is, OP in two different ways as shown in [C-I, Chang, 2013] and [Chang & Wen et.al.,2013]

Result

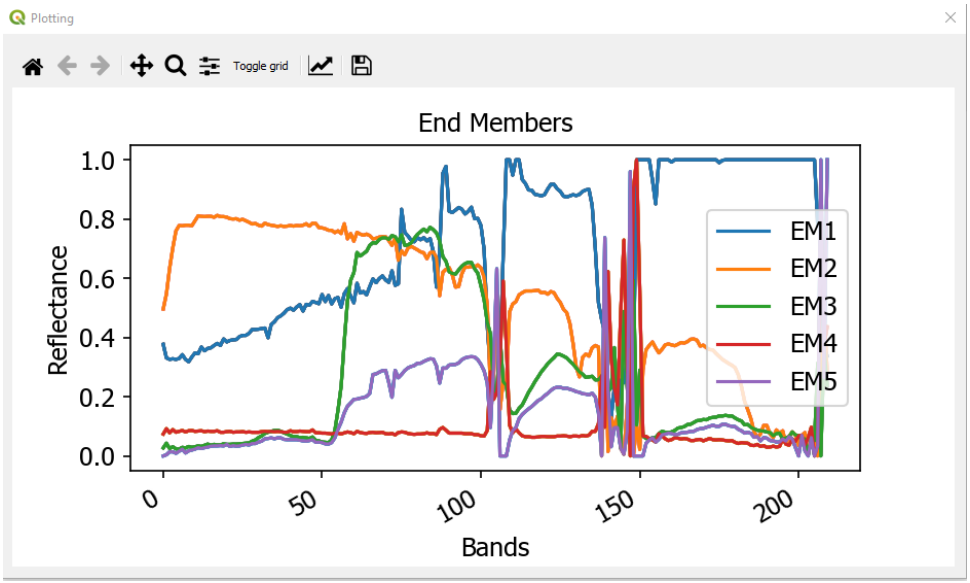


Figure: Five endmembers extracted by ATGP algorithm.

References:

- C.-I Chang, Hyperspectral Data Processing: Algorithm Design and Analysis. New York, NY, USA: Wiley, 2013.
- C.-I Chang, C. H. Wen, and C. C. Wu, "Relationship exploration among PPI, ATGP and VCA via theoretical analysis," Int. J. Comput. Sci. Eng., vol. 8, no. 4, pp. 361–367, 2013.

5.2.4 VERTEX COMPONENT ANALYSIS

QGIS 3.XX → Hyperspectral→ Unmixing → Endmember Extraction → VCA

User Interface

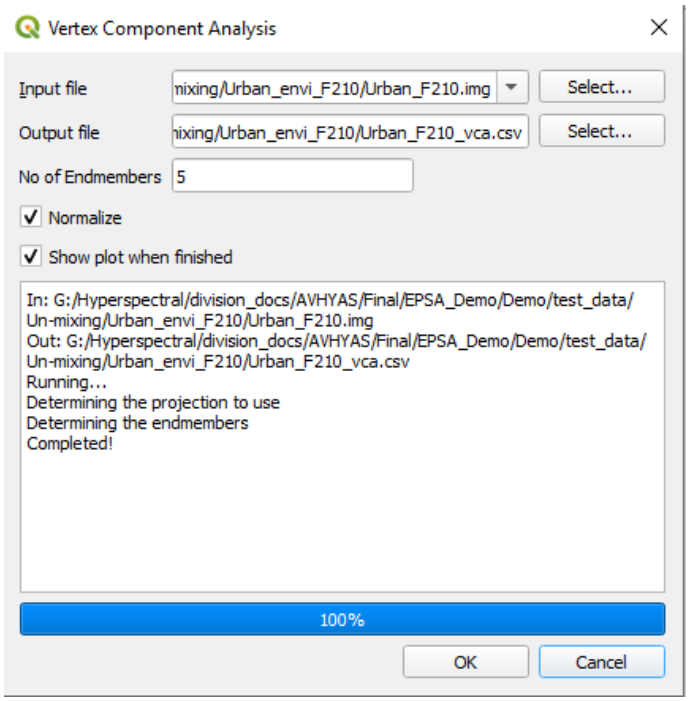


Figure. User-Interface for VCA algorithm.

Input Arguments

No. of Endmembers: define number of endmember for estimation.

Normalize: Normalize hx data between 0-1.

I/O PARAMETERS

Type	Data format	DATA type	value range	Remark
Input				
Input File	GeoTiff / ENVI			Bad band Removed Data, Radiance/Reflectance

Output				
Output File				Virtual dimensionality of data is displayed on the user interface

Overview:

The VCA algorithm [Nascimento and Dias,2005] is very similar to the ATGP, although it originates from the concept of the simplex. It uses the OSP criterion to find pixels that can be used as the vertices of a simplex, but it includes a dimensionality reduction process before end-member extraction, while the ATGP does not.

Result

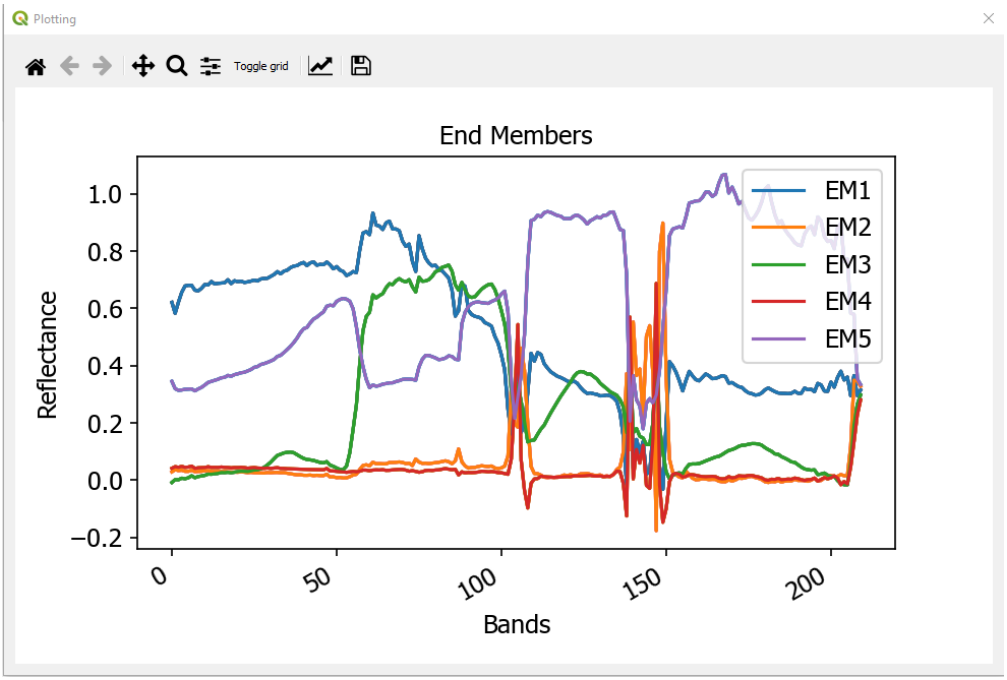
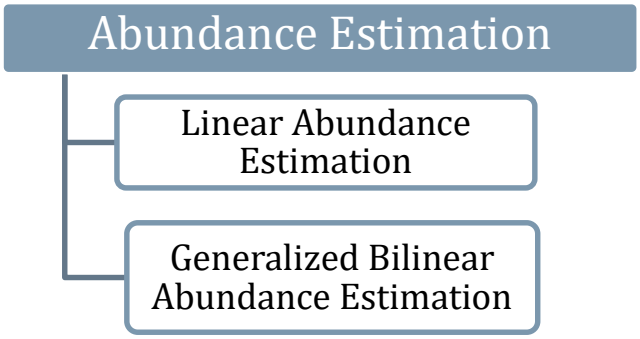


Figure: Five endmembers extracted by VCA algorithm

References:

- J. M. P. Nascimento and J. M. B. Dias, Vertex component analysis: a fast algorithm to unmix hyperspectral data, in IEEE Transactions on Geoscience and Remote Sensing, vol. 43, no. 4, pp. 898-910, April 2005.

5.3 ABUNDANCE ESTIMATION



5.3.1 LINEAR ABUNDANCE ESTIMATION

QGIS 3.XX → Hyperspectral→ Unmixing → Abundance Estimation → Linear

User Interface

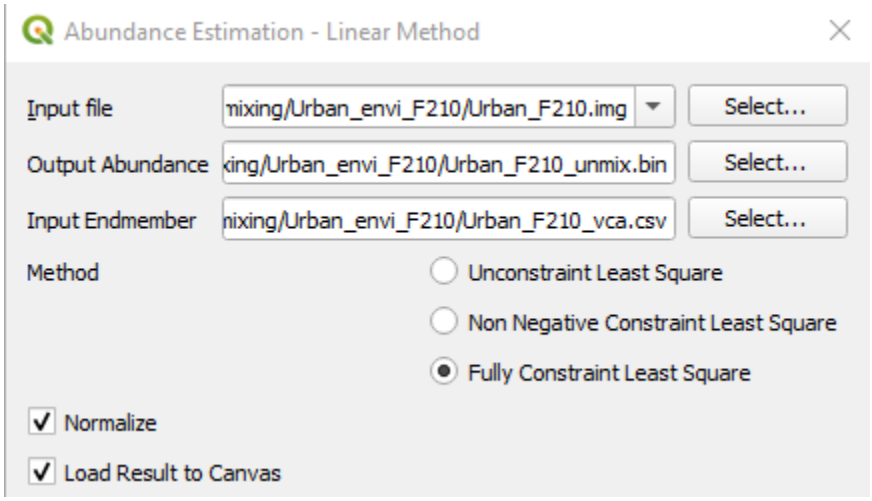


Figure. User-Interface for abundance estimation by linear method

Input Arguments

Input Endmembers: define file having endmembers estimated.

Method: select one method (1. Unconstrained Least Square 2. Non-Negative Constraint Least Square 3. Fully Constrained Least Square).

Normalize: Normalize hx data between 0-1.

5 UNMIXING MODULE

I/O PARAMETERS

Type	Data format	DATA type	value range	Remark
Input				
Input File	GeoTiff / ENVI			Bad band Removed Data, Radiance/Reflectance
Output				
Output File				Virtual dimensionality of data is displayed on the user interface

Overview:

Linear Mixing Model:

When multiple scattering can be reasonably disregarded, the spectral reflectance of each pixel can be approximated by a linear mixture of endmember reflectances weighted by their corresponding fractional abundances [Keshava and Mustard, 2002]. Notationally, let **y** be a $L \times 1$ reflectance vector, where L is the total number of bands, and $\mathbf{m}i$ is the signature of the ith endmember, $i = 1, \ldots, R$ (R is total number of endmember). The reflectance vector can then be expressed as

$$\mathbf{y} = \mathbf{M}\alpha + \mathbf{n}, \qquad (1)$$

where $\mathbf{M} = [\mathbf{m}1, \mathbf{m}2, \ldots, \mathbf{m}_R]$ is the *mixing matrix* and contains the signatures of the endmembers present in the observed area, $\alpha = [\alpha 1, \alpha 2, \ldots, \alpha_R]^T$ is the fractional abundance vector, and $\mathbf{n} = [n1, \ldots, n_L]^T$ models additive noise in each spectral channel. Given a set of reflectances **y**, the problem basically reduces to estimate appropriate values for both **M** and α.This estimation problem is usually completed with two physically reasonable constraints: 1) all abundances must be positive, $\alpha i \geq 0$, and 2) they have to sum one, $\sum_{i=1}$ □ □ = 1, since we want a plausible description of the mixture components for each pixel in the following image.

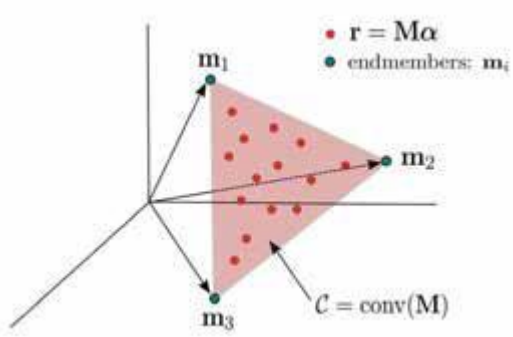


Figure: Illustration of the simplex set C for $p = 3$. Points in red denote the available spectral vectors **r** that can be expressed as a linear combination of the *endmembers* **m***i* , $i = 1, \ldots, 3$, (vertices circled in green). The subspace formed by these endmembers is the convex hull C of the columns of **M**. [Bioucas-Dias and Plaza, 2010].

Now, assuming that the columns of **M** are independent, the set of reflectances fulfilling these conditions form a $(R - 1)$-simplex in □^{L} See an illustrative example in above figure, showing the simplex set C for an hypothetical mixing matrix **M** containing three endmembers (C is the *convex hull* of the columns of **M**). Points in red denote spectral vectors, whereas the vertices of the simplex (in green) correspond to the endmembers. Inferring the mixing matrix **M** may be seen as a purely geometrical problem that reduces to identify the vertices of the simplex C.

Bilinear Mixing Model:

Bilinear mixture models account for the presence of multiple photon interactions by introducing additional "interaction" terms in the LMM. This section presents bilinear mixing models previously introduced in the literature. These bilinear models mainly differ by the constraints associated with the mixing parameters.

Nascimento's bilinear mixing model: Nascimento's model (**NM**) [Nascimento & Bioucas, 2009] considers 2nd order interactions between ith and jth endmembers (□□□ □, □ = *1*, . . . , □ □□□ □ ≠ □)such that the observed mixed pixel y can be written

$$\mathbf{y} = \sum_{r=1}^{R} a_r \mathbf{m}_r + \sum_{i=1}^{R-1} \sum_{j=i+1}^{R} \beta_{i,j} \mathbf{m}_i \odot \mathbf{m}_j + \mathbf{n} \tag{2}$$

$$\mathbf{m}_i \odot \mathbf{m}_j = \begin{pmatrix} m_{1,i} m_{1,j} \\ \vdots \\ m_{L,i} m_{L,j} \end{pmatrix}.$$

where $\mathbf{m}_i \odot \mathbf{m}_j$ denotes the Hadamard (term-by-term) product of ith and jth endmember spectra.

Note that the parameter $\beta_{i,j}$ in (2) is the amplitude of the interaction term due to the ith and jth components. Unknown parameters $(a, a_{1,2}, \ldots, a_{R-1,R})$ have to satisfy the following constraints

$$\begin{aligned} a_r &\geq 0 \quad r = 1, \ldots, R, \\ \beta_{i,j} &\geq 0 \quad i, j = 1, \ldots, R, \quad i \neq j \\ \sum_{r=1}^{R} a_r + \sum_{i=1}^{R-1} \sum_{j=i+1}^{R} \beta_{i,j} &= 1. \end{aligned} \tag{3}$$

It is clear that, NM can be expressed as a linear mixture model with $R = R(R+1)/2$ correlated endmembers. By considering the interaction spectra $\mathbf{m}_i \odot \mathbf{m}_j$ as new spectral components with fractions $\beta_{i,j}$, then NM model (2) can be rewritten as

$$\mathbf{y} = \sum_{p=1}^{R^*} a_p^* \mathbf{m}_p^* + \mathbf{n}$$

where

$$\begin{aligned} a_p^* &= a_r \quad , \quad \mathbf{m}_p^* = \mathbf{m}_r \quad p = 1, \ldots, R, \\ a_p^* &= \beta_{i,j} \quad , \quad \mathbf{m}_p^* = \mathbf{m}_i \odot \mathbf{m}_j \quad R+1 \leq p \leq R^*. \end{aligned} \tag{4}$$

Consequently, the new abundance vector $\mathbf{a}^* = [a^*_1, \ldots, a^*_{R^*}]^T$ can be estimated using existing algorithms for linear spectral unmixing, such as least square methods (Unconstrained, Non-negative Constrained, Fully Constrained) [Heinz & Change, 2001]. NM can also reduce ti the LMM when $a^*_r = 0$ for $r = R+1, \ldots, R^*$.

5 UNMIXING MODULE

Fan's bilinear mixing model : Fan model (FM) [Fan et.al., 2009] is similar to NM, the FM assumes that the interaction terms □□ ⊙ □□ are additional spectra, resulting from the Hadamard products of pure spectral components. However, the FM assumes that the amplitudes of these interactions depend on the component fractions involved in the mixture. Thus observed pixel of the hyperspectral image can be written

$$\mathbf{y} = \sum_{r=1}^{R} a_r \mathbf{m}_r + \sum_{i=1}^{R-1} \sum_{j=i+1}^{R} a_i a_j \mathbf{m}_i \odot \mathbf{m}_j + \mathbf{n}, \tag{5}$$

subject to the constraints (non-negativity & sum-to-one). An unmixing procedure based on a Taylor series expansion and least-squares method was proposed in [Fan et.al, 2009].

Generalized bilinear mixing model: This model is introduced by Halimi et.al, 2011, for nonlinear spectral unmixing and referred as "generalized bilinear model (GBM)". As in the previous methods NM and FM, the interaction term □□ ⊙ □□ is included in the GBM as an additional spectrum. However, the GBM assumes that contribution of the interaction term □□ ⊙ □□ is proportional to the fractions of the involved components with an amplitude $\gamma_{□,□}$ □□ □□ (it makes sense to assume that the "quantity" of interaction between two materials is related to the "quantity" of each material present in a given pixel). where $\gamma_{□,□} \in (\boldsymbol{0,1})$,leads to the following formulation

$$\mathbf{y} = \sum_{r=1}^{R} a_r \mathbf{m}_r + \sum_{i=1}^{R-1} \sum_{j=i+1}^{R} \gamma_{i,j} a_i a_j \mathbf{m}_i \odot \mathbf{m}_j + \mathbf{n}. \tag{6}$$

Note that $\gamma = [\gamma_{1,2}, \ldots, \gamma_{R-1,R}]^T$, is a real parameter vector that quantifies the interactions terms between the different spectral components. Constraints associated with the GBM can be expressed as

$$a_r \geq 0 \quad r = 1, \ldots, R, \quad \sum_{r=1}^{R} a_r = 1,$$
$$1 \geq \gamma_{i,j} \geq 0 \quad i, j = 1, \ldots, R, \quad i \neq j. \tag{7}$$

The main motivation for introducing the additional parameters $\gamma_{i,j}$ in the mixing model is to obtain a more flexible model than NM or FM. GBM reduces to LMM for $\gamma = 0$ (where $\mathbf{0}$ is an $(L^* - L) \times 1$ vector of zeros). similarly, the GBM reduces to the FM for $\gamma = 1$ (where $\mathbf{1}$ is an $(L^* - L) \times 1$ vector of ones. The unmixing procedure based on the GBM can be performed by using least squares estimator [Fan, et.al,2009] or by Bayesian algorithm [Halimi et.al, 2011].

Reference:

- J. M. P. Nascimento and J. M. Bioucas-Dias, "Nonlinear mixture model for hyperspectral unmixing," in *Proc. SPIE Image and Signal Processing for Remote Sensing XV*, L. Bruzzone, C. Notarnicola, and F. Posa, Eds., vol. 7477. Berlin, Germany: SPIE, Sept. 2009, pp. 74 770I–1–74 770I–8
- D. C. Heinz and C.-I Chang, "Fully constrained least-squares linear spectral mixture analysis method for material quantification in hyperspectral imagery," *IEEE Trans. Geosci. and Remote Sensing*, vol. 29, no. 3, pp. 529–545, March 2001.
- W. Fan, B. Hu, J. Miller, and M. Li, "Comparative study between a new nonlinear model and common linear model for analysing laboratory simulated-forest hyperspectral data," *Remote Sensing of Environment*, vol. 30, no. 11, pp. 2951–2962, June 2009.
- A. Halimi, Y. Altmann, N. Dobigeon, and J.-Y. Tourneret, "Nonlinear unmixing of hyperspectral images using a generalized bilinear model," *IEEE Trans. Geosci. and Remote Sensing*, 2011.
- Keshava N.,"A Survey of Spectral Unmixing Algorithms", Vol.14, LINCOLN LABORATORY JOURNAL, 2003.
- N. Keshava and J. F. Mustard, "Spectral unmixing," *IEEE Signal Processing Magazine*, pp. 44–57, Jan. 2002.
- J. M. Bioucas-Dias and A. Plaza. Hyperspectral unmixing: Geometrical, statistical, and sparse
- regression-based approaches. In *SPIE Conf. Im. Sign. Proc. Rem. Sens.*, 2010.

Result

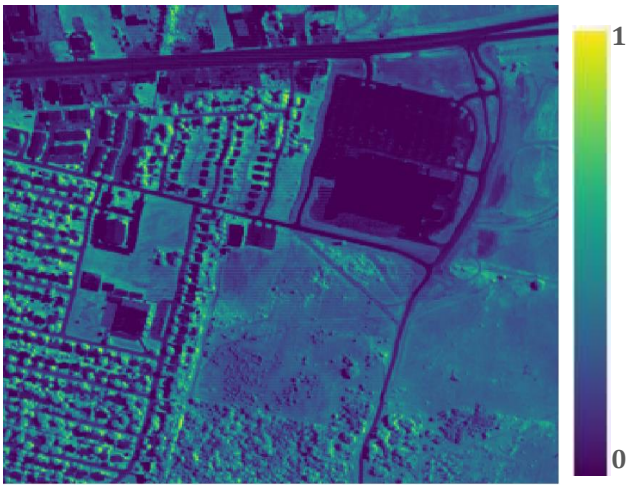


Figure: abundance maps

5.5 ITERATIVE SCATTER PLOY MIXING VISUALIZATION

QGIS 3.XX → Hyperspectral→ Unmixing → Iterative Scatterplot mixing Visualization

User Interface

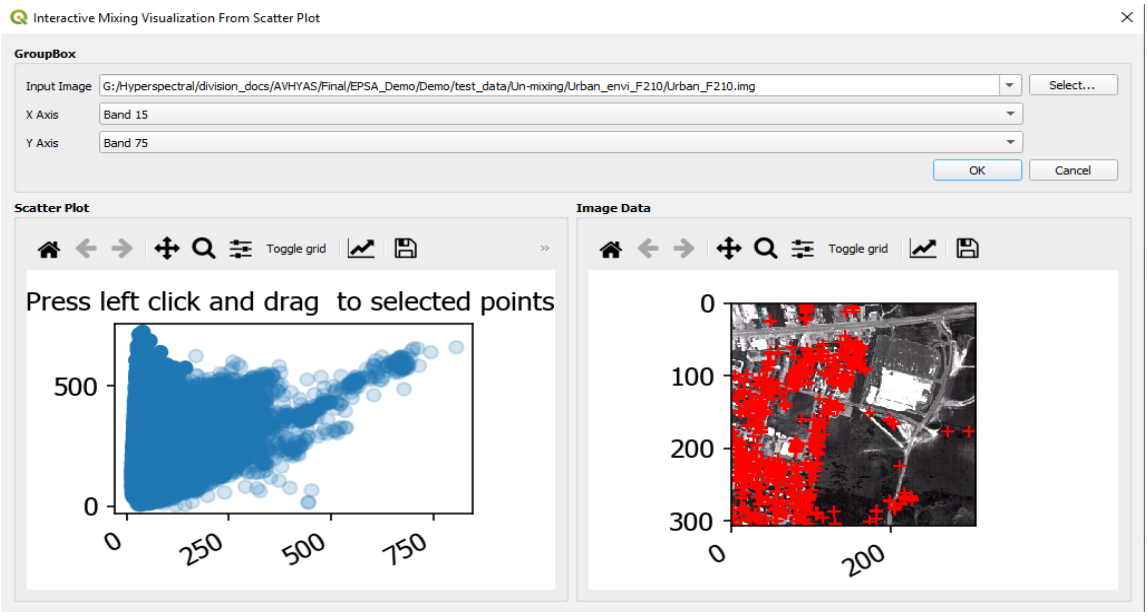


Figure User interface for Interactive Scatter plot visualization

5.6 UNMIXING RESULT VIEWER

QGIS 3.XX → Hyperspectral→ Unmixing → Unmixing Result Viewer

User Interface

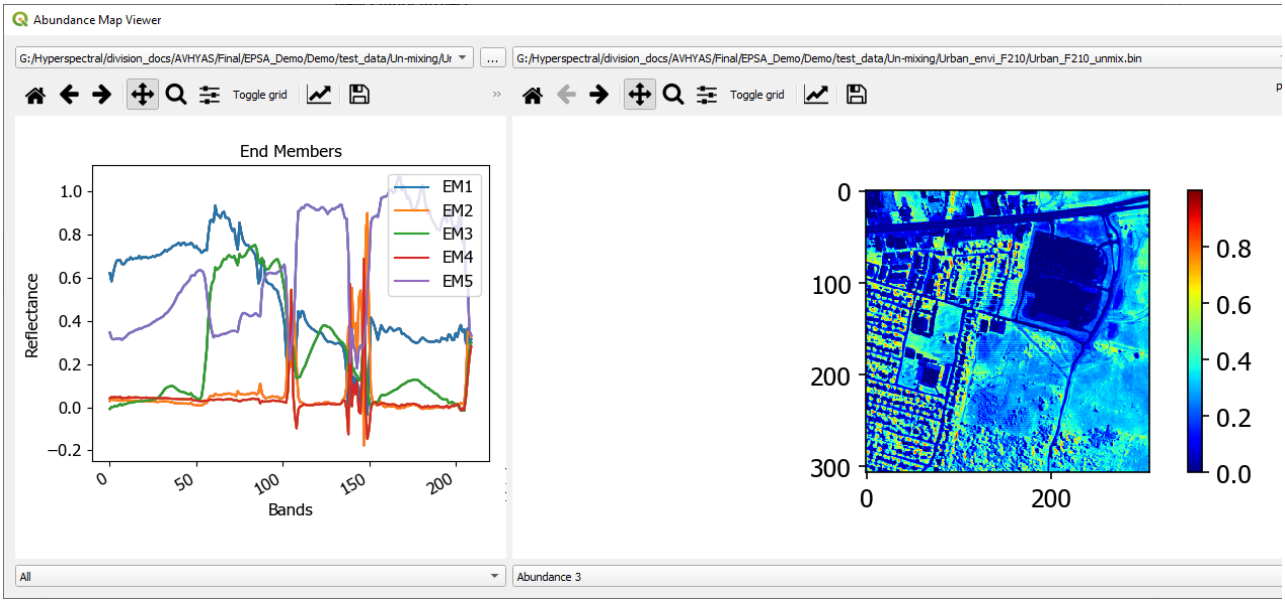


Figure User interface for Abundance Map Viewer

Overview

This module provides provide an interface for visualizing the Unmixing results. QGIS as such has no viewer for visualizing spectra and abundances so this utility has been implemented so as user can visualize the different endmembers along with their respective abundances. User requires to have an endmember csv file which were extracted by running end member extraction modules and the abundance map which can be obtained from the abundance estimation modules.

5.6 UNMIXING ERROR ANALYSIS-RMSE

QGIS 3.XX → Hyperspectral→ Unmixing → Unmixing Error Analysis-RMSE

User Interface

5 UNMIXING MODULE

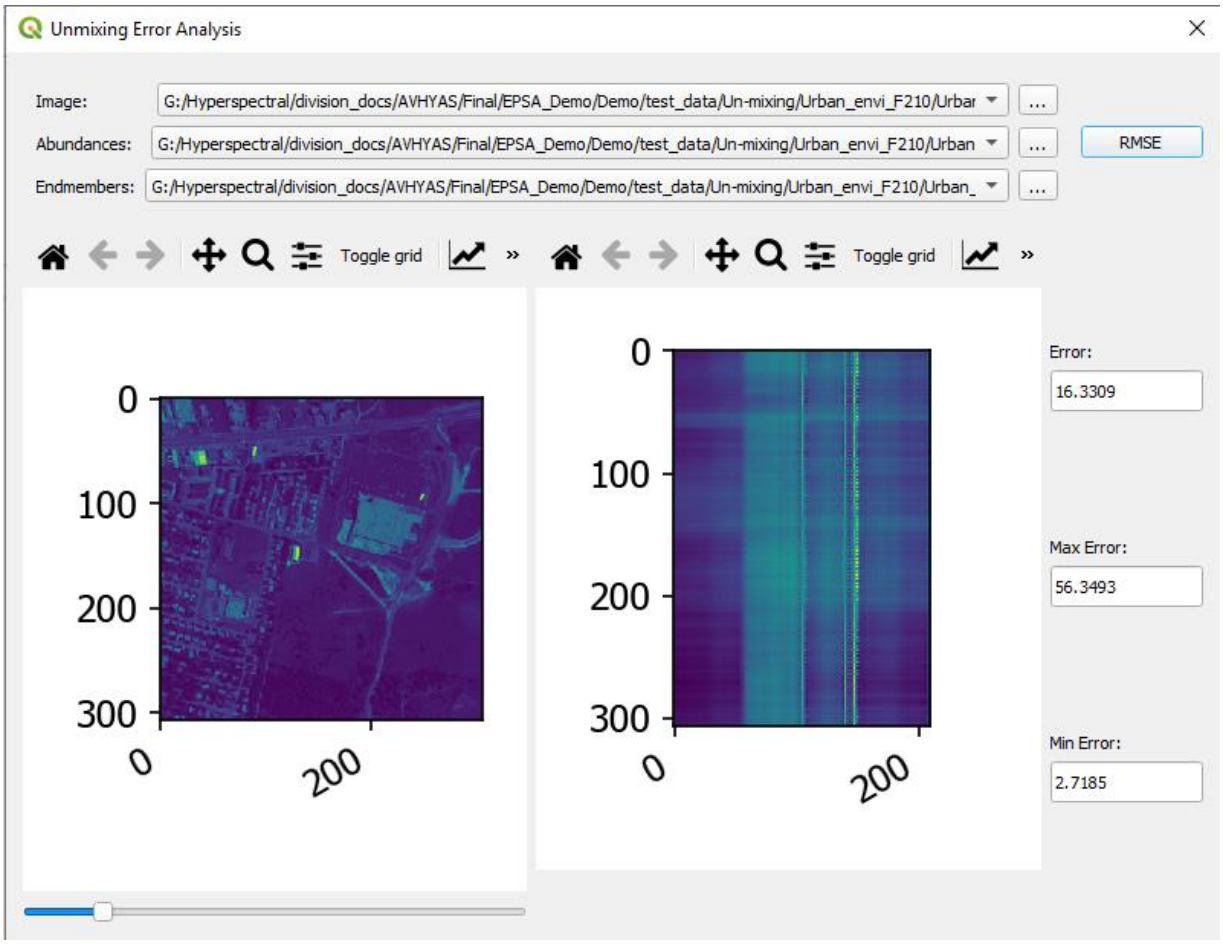


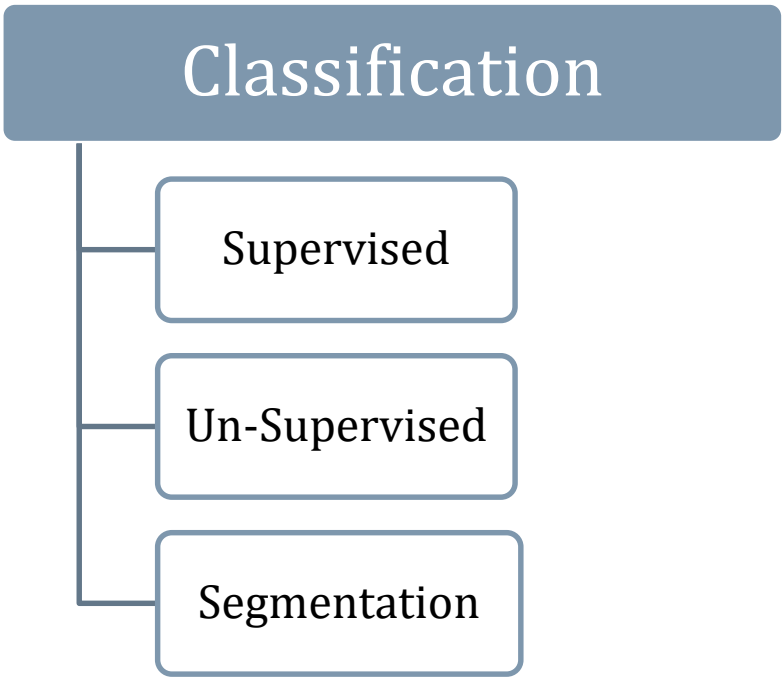
Figure Unmixing error analysis

Overview

This module is intended to estimate the reconstruction error of the abundances and end member during the process of un-mixing. User needs to have pre requisite file such as the original hyperspectral image, the abundances and the endmember file. On clicking on RMSE button the process for estimating the RMSE will be called and the error will be populated in the text file along with the reconstruction error.

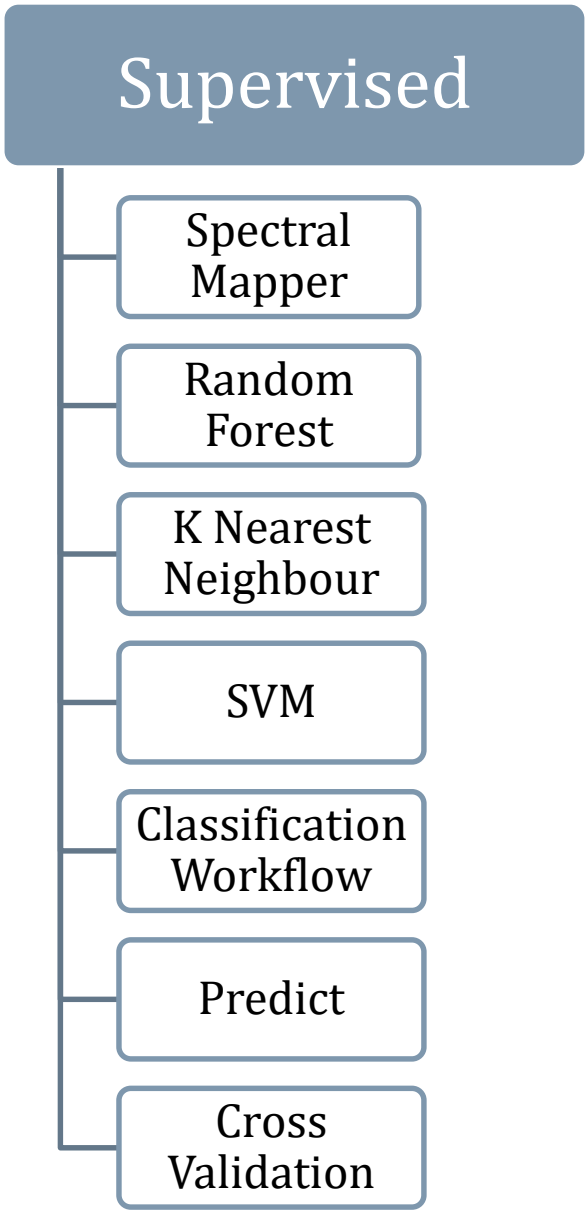
6 CLASSIFICATION MODULE

6 **CLASSIFICATION MODULE**



6 CLASSIFICATION MODULE

6.1 SUPERVISED CLASSIFICATION



6.1.1 SPECTRAL MAPPER

QGIS 3.XX → Hyperspectral→ Classification → Spectral Mapper

User Interface

6 CLASSIFICATION MODULE

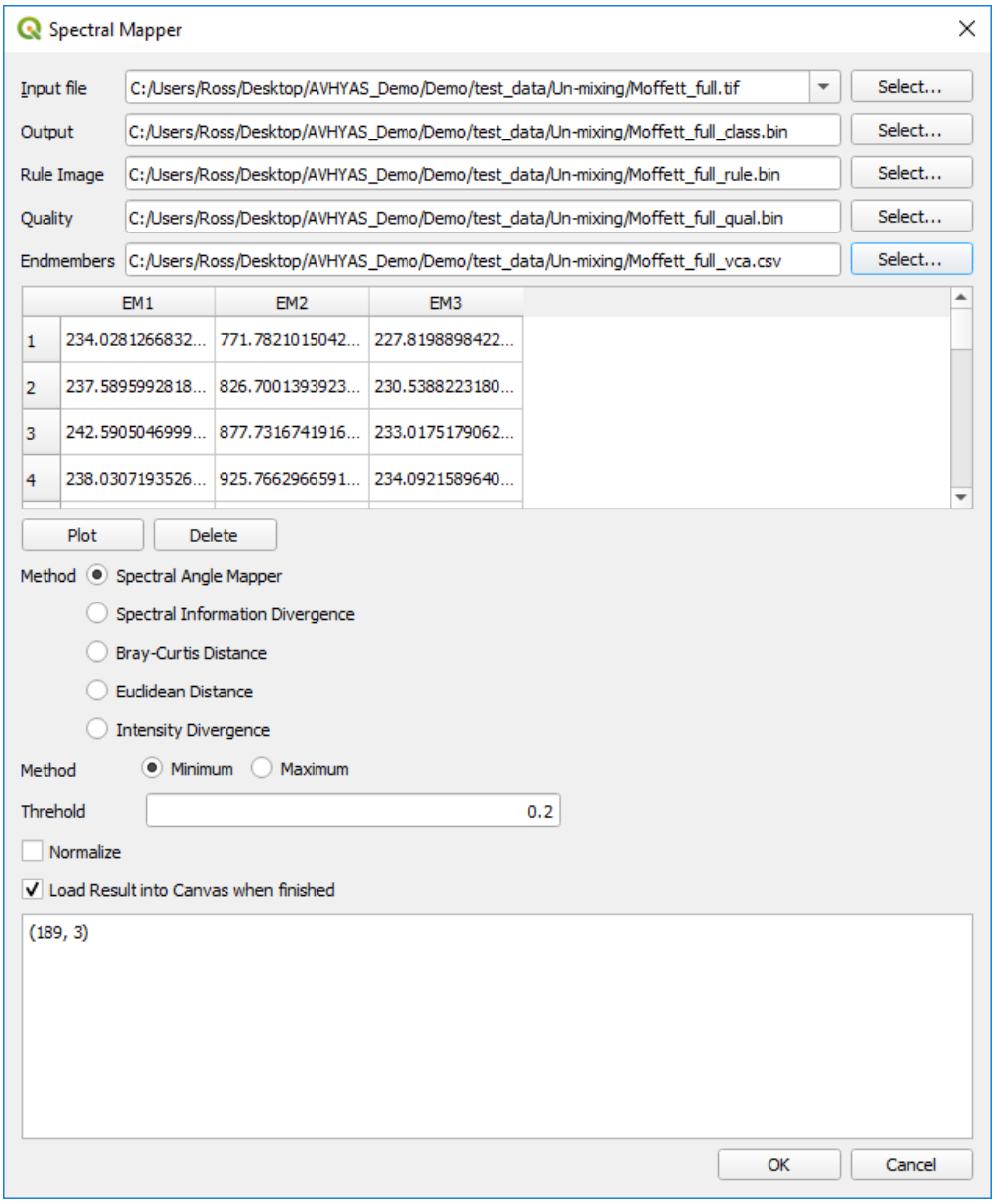


Figure User interface for Spectral Mapper

I/O PARAMETERS

Type	Data format	DATA type/Type	value range	Remark
Input				

6 CLASSIFICATION MODULE

Input File	GeoTiff / ENVI	Raster		Raster data to be used for classification Radiance/Reflectance
End Member File	CSV	File		The end member spectra's
Threshold		Float	Default 0.2	The threshold value for spectral mapper
Method				Distance measurement method to be used
Output				
Output	ENVI	Raster		Output Classified map
Rule Image	ENVI	Raster		Rule Image Map
Quality	ENVI	Raster		The classification Quality Raster Map

Input Arguments

Input File: Specify the input raster file to use for classification. After selecting the input file other output text box will be automatically populated

End Member File: Specify the End Member file for calculating the spectral mapping distance with

Output File: Specify the output path of the classification map to save to disk.

Rule Image: The raster image containing the spectral mapping distance of each pixel with the end members provided

Quality: The path of the image containing the quality of the classified image

Overview

The Spectral Mapper is analogous to what ENVI provides which is a module called Spectral Angle Mapper but offers more choice for the distance measure used. In addition to the traditional spectral angle also the Euclidean distance, intensity difference, Bray-Curtis distance and Spectral Information Divergence can be used.

6 CLASSIFICATION MODULE

For this module to run end-member are required to be collected. The endmember can be extracted from image by using the end member extraction tool given in the Unmixing Menu or by simple manually collecting the end members through the spectral library module in basic tools. However, the format supported so far for the beta version is only CSV format.

User can view the different end members by simply clicking on the plot button. Currently no resampling strategy has been implemented. This means that it is necessary for the input endmember to be in the same spectral resolution as that of the input.

For every selected end-member a rule image will be created as a separate band in the output. The band names of the output reflect the names of the end-members used. The Display program can be used for collecting end-member. The rule image automatically has '_map' suffix in its name which user has the option to change.

The rule images can then be further processed and give classified map. The rule based classifier classifies the rule image by per pixel selecting the class with minimum distance. The output is an ENVI classification image. Random Colors are assigned to the classes. Class names reflect the name of the original end members used in the spectral mapping operation in the rule image generation.

Finally, the quality image shows for every pixel the rule value of the chosen class. Using spectral angle, a high value in the quality image means a low confidence in the class membership.

Result:

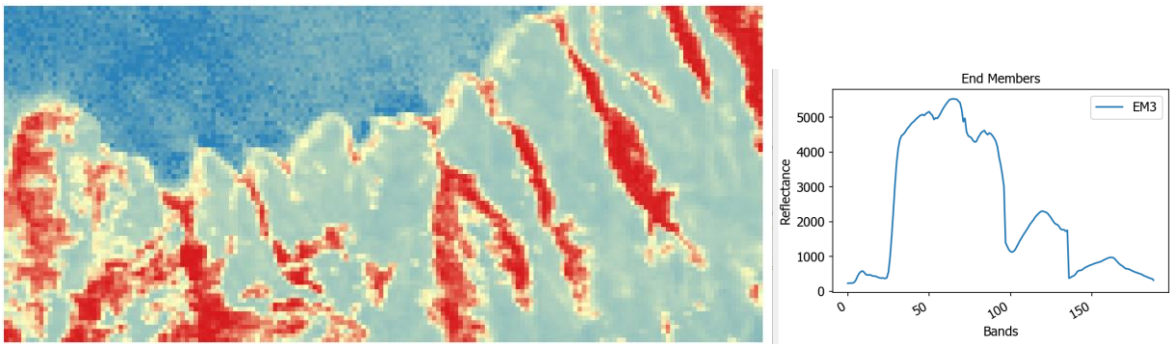


Fig Rule Image Output for end member 1

6 CLASSIFICATION MODULE

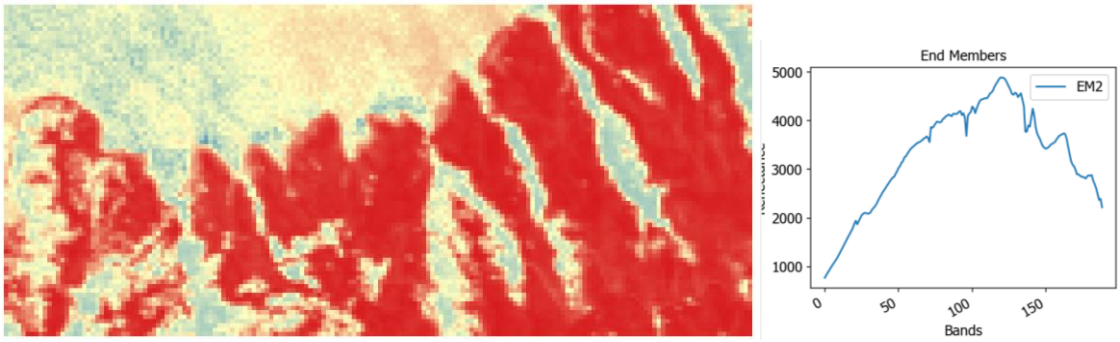


Fig Rule Image Output for end member 2

Fig Rule Image Output for end member 3

Fig Moffett Standard data with three end members

6 CLASSIFICATION MODULE

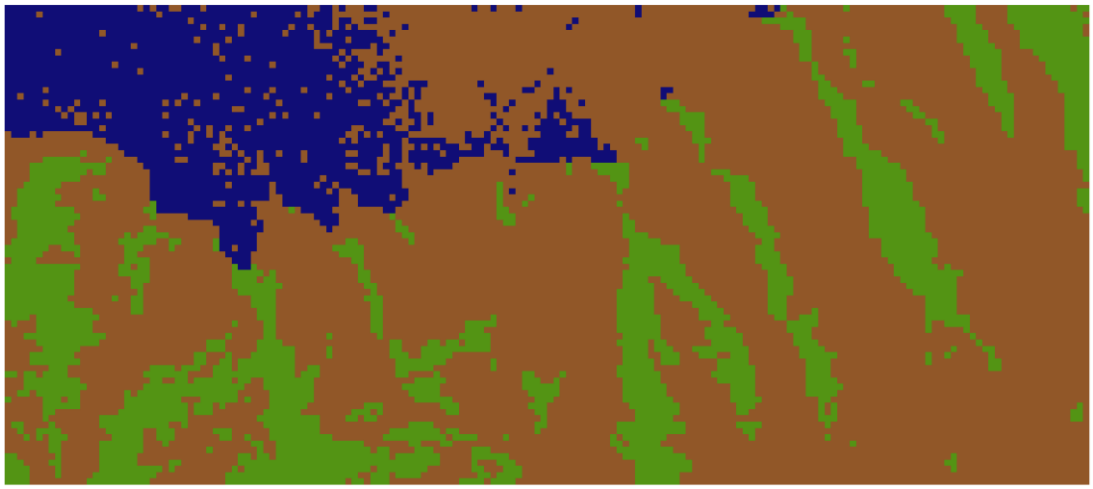


Fig Classified output using SAM on Moffett Standard data with three end members

6.1.2 RANDOM FOREST CLASSIFICATION TOOL

QGIS 3.XX → Hyperspectral→ Classification → Random Forest

User Interface

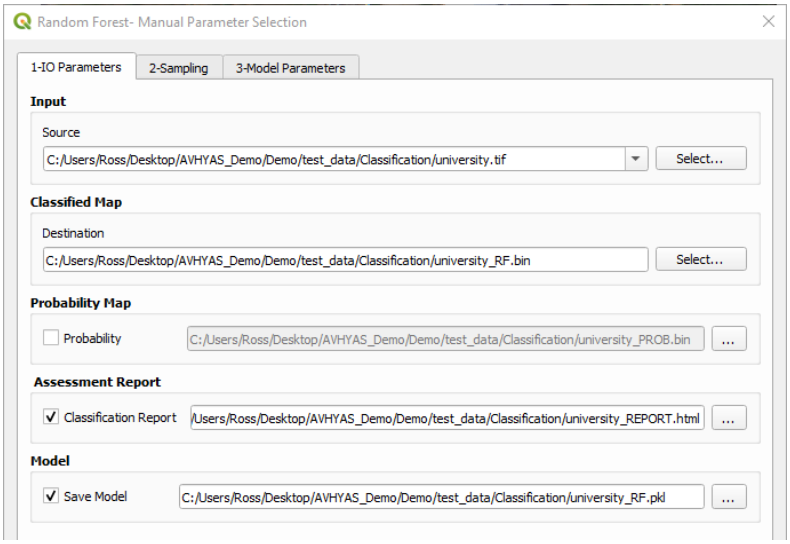


Figure User interface for Random Forest IO parameter tab

6 CLASSIFICATION MODULE

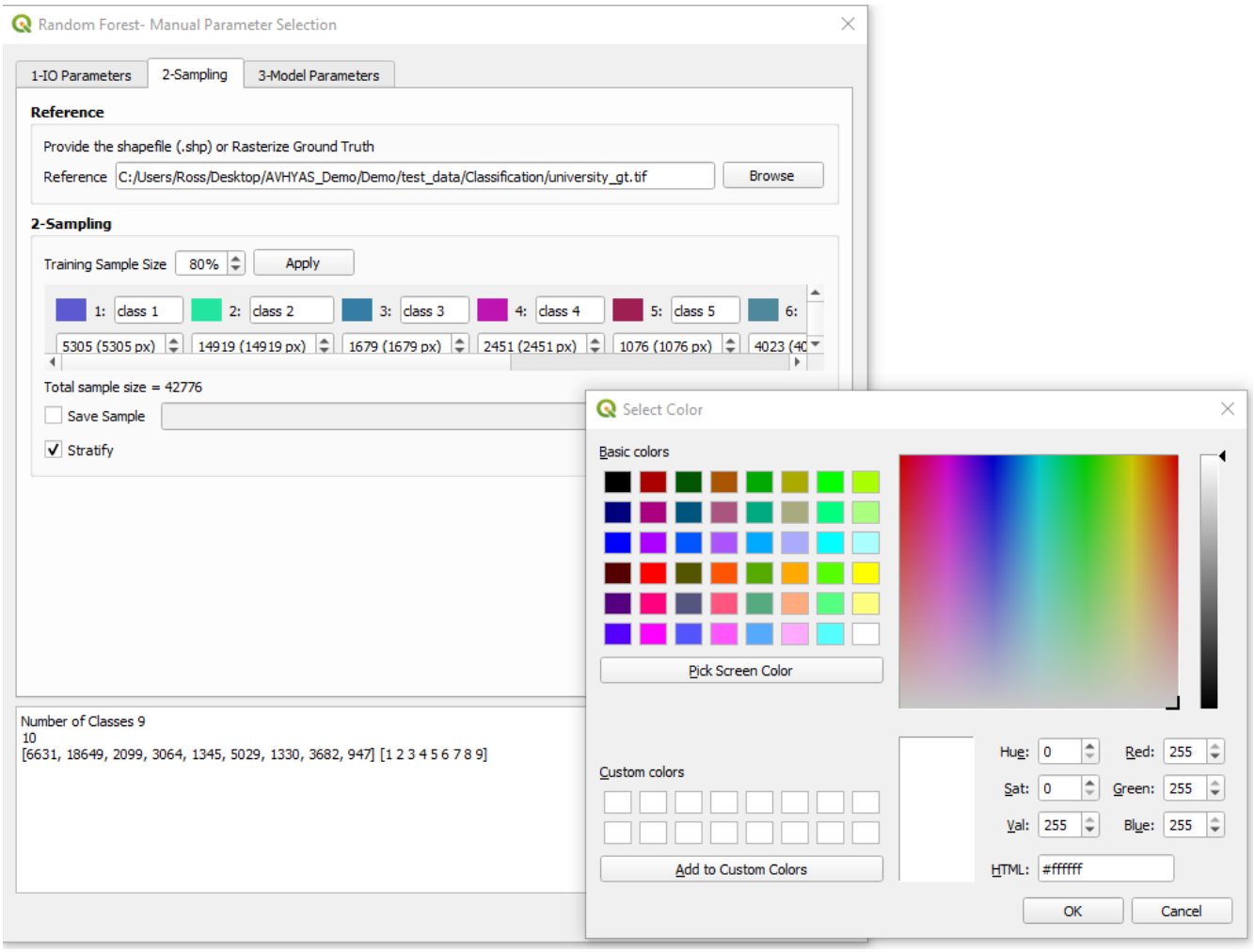


Figure User interface for Random Forest Sampling tab

6 CLASSIFICATION MODULE

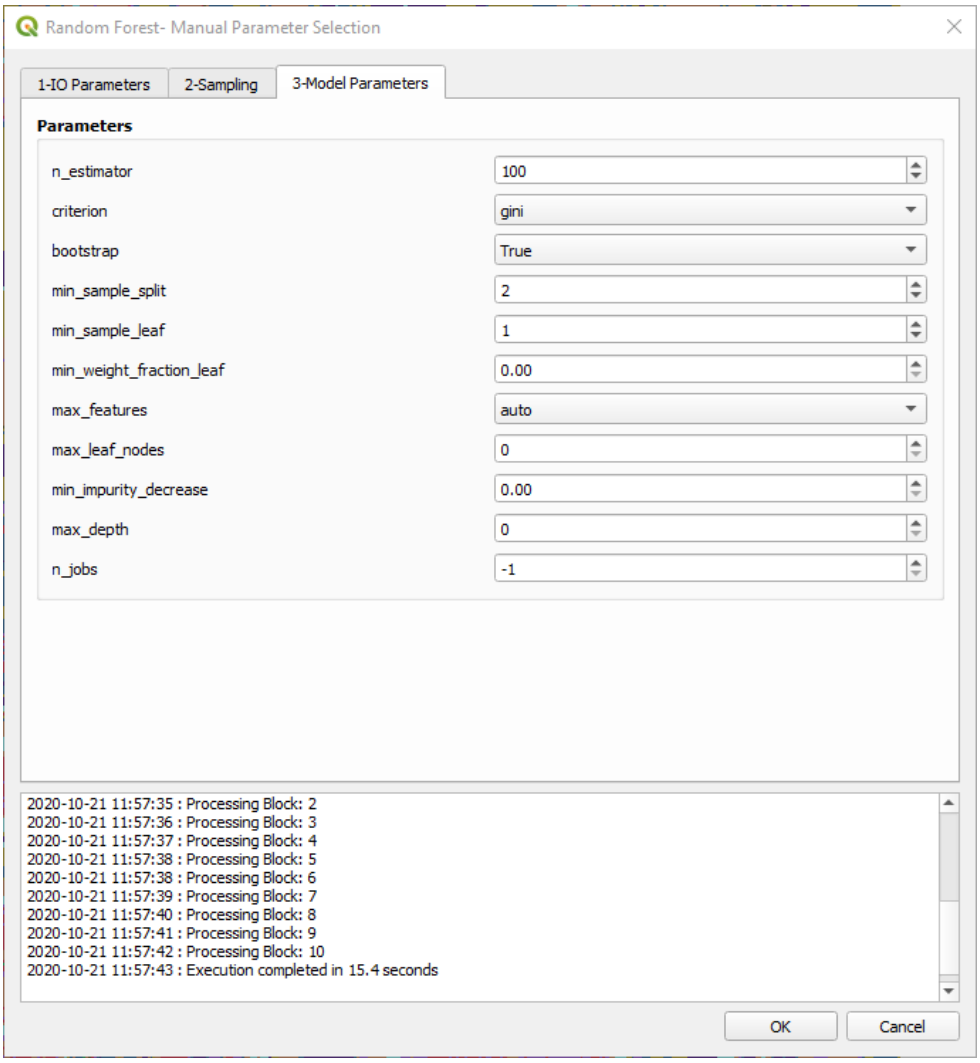


Figure User interface for Random Forest Model parameter tab

I/O PARAMETERS

Type	Data format	DATA type	value range	Remark
Input				
Input File	GeoTiff / ENVI			Raster data to be used for classification Radiance/Reflectance
References	Shapefile/Raster			Ground Truth for classification

training sample size				Percentage of sample to be used for classification with optional for stratification
n_estimators		Integer, default=100		Number of trees in the forest
critertion		String {gini, entropy}, default=gini		The function to measure the quality of a split
max_depth		Integer default=None		The maximum depth of the tree
min_samples_split		Integer default=2		The minimum number of samples required for split an internal node
min_samples_leaf		Integer/Float default=1		The minimum number of samples required to be at a leaf
min_weight_fraction_leaf		Float default=0.0		The minimum weighted fraction of the sum total weights required to be at the leaf node
max_features		String {auto,sqrt,log2) default=auto		The number of features to consider when looking for the best split
max_leaf_nodes		Integer default=None		Grow Trees with max_leaf_nodes in best first fashion
min_impurity_decrease		Float default=0.0		A node will be split if this split induces a decrease of the impurity greater than or equal to this value
min_impurity_split		Float default=None		Threshold for early stopping in tree growth
bootstrap		bool default=True		whether bootstrap samples are used when building trees

n_jobs		Integer, default=None		The number of jobs to run in parallel
Output				
Classified map	ENVI	Raster		Raster file with integer values for each class
probability map	ENVI	Raster		Raster file with probability of each class pixel wise for each band.
assessment report	HTML	Raster		The classification report and accuracy assessment
save model	pickle			Save the model for future inferences

Input Arguments

IO parameters

Input file: Specify the input raster for classification. Other field will be automatically populated with postfix added on the input name. User can change the default value if they want to otherwise they can proceed with the default values.

Classification map: Specify the destination file for the classification map to save to disk.

Probability map: ☑ Activate this setting in order to save the probability map ouput. Specify the destination file for the probability map.

Cross Validation and Assessment Report: ☑ Activate this setting to save the accuracy assessment report. User has the option to change the number of fold use in the cross validation process the default value is **3**. Specify the path to save the cross validation assessment report in HTML format.

Sampling

Reference: Specify the reference ground truth data either in the form of shapefile or rasterize file. If vector file is specified then user has to select the **Attribute** which contain the numeric class information. After selection of the attributes if vector else user are suppose to click on the **Apply** button so as the class names and class colors become visible.

Training Sample Size : User can define and alter the sampling size relative to percentage amount of samples. Optional argument for stratification is also provided which user can select or ignore if they wish to.

Model parameters

n_estimators : Define the number of trees in the forest
criterion: Define function to measure the quality of a split
max_depth: Define the maximum depth of the tree
min_samples_split: Define the minimum number of samples required to be at a leaf
min_weight_fraction_leaf: Define the minimum weighted fraction of the sum total weights required to be at the leaf node
max_features: Define the number of features to consider when looking for the best split
max_leaf_nodes: Define the number of grow Trees with max_leaf_nodes in best first fashion
min_impurity_decrease: Define the number of trees in the forest
min_impurity_split: Define the Threshold for early stopping in tree growth
bootstrap: Define whether bootstrap samples are used when building trees
njobs: Define the number of trees in the forest
See RandomForestClassifier for more information on different parameters and details of the algorithm

Overview

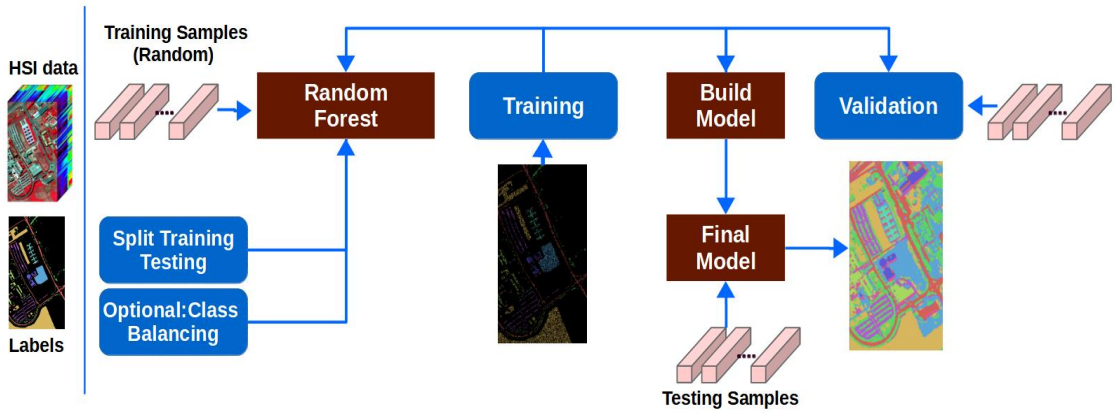


Figure: Data Flow Diagram

6 CLASSIFICATION MODULE

A big part of machine learning is classification — where we want to know what class (a.k.a. group) an observation belongs to. The ability to precisely classify observations is extremely valuable for various applications. Data science provides a plethora of classification algorithms such as logistic regression, support vector machine, naive Bayes classifier, and decision trees. But near the top of the classifier hierarchy is the random forest classifier

Random forest, like its name implies, consists of a large number of individual decision trees that operate as an ensemble. Each individual tree in the random forest spits out a class prediction and the class with the most votes becomes our model's prediction
The fundamental concept behind random forest is a simple but powerful one — the wisdom of crowds. In data science language, the reason that the random forest model works so well is because:

A large number of relatively uncorrelated models (trees) operating as a committee will outperform any of the individual constituent models. The low correlation between models is the key. Uncorrelated models can produce ensemble predictions that are more accurate than any of the individual predictions. The reason for this wonderful effect is that the trees protect each other from their individual errors (as long as they don't constantly all err in the same direction). While some trees may be wrong, many other trees will be right, so as a group the trees are able to move in the correct direction. So the prerequisites for random forest to perform well are:

1. There needs to be some actual signal in our features so that models built using those features do better than random guessing.

2. The predictions (and therefore the errors) made by the individual trees need to have low correlations with each other.

So how does random forest ensure that the behavior of each individual tree is not too correlated with the behavior of any of the other trees in the model? It uses the following two methods: Bagging (Bootstrap Aggregation) — Decisions trees are very sensitive to the data they are trained on — small changes to the training set can result in significantly different tree structures. Random forest takes advantage of this by allowing each individual tree to randomly sample from the dataset with replacement, resulting in different trees. This process is known as bagging. Notice that with bagging we are not sub setting the training data into smaller chunks and training each tree on a different chunk. Rather, if we have a sample of size N, we are still feeding each tree a training set of size N (unless specified otherwise). But instead of the original training data, we take a random sample of size N with replacement.

To sum up this module provides an interface to scikit-learn Random Forest Classifier where most the parameters are set as default as defined by the scikit-learn module. Important parameter to be considered is the **number of estimators** where the users can provide their choice depending on the problem at hand. The rest of the parameters can be used as the default

as it will work well in most of the problem. If however the user wants to experiment around then users are suggested to play around with the following parameters, **max_features, max_depth, min_samples_split , min_samples_leaf , bootstrap**.

Result

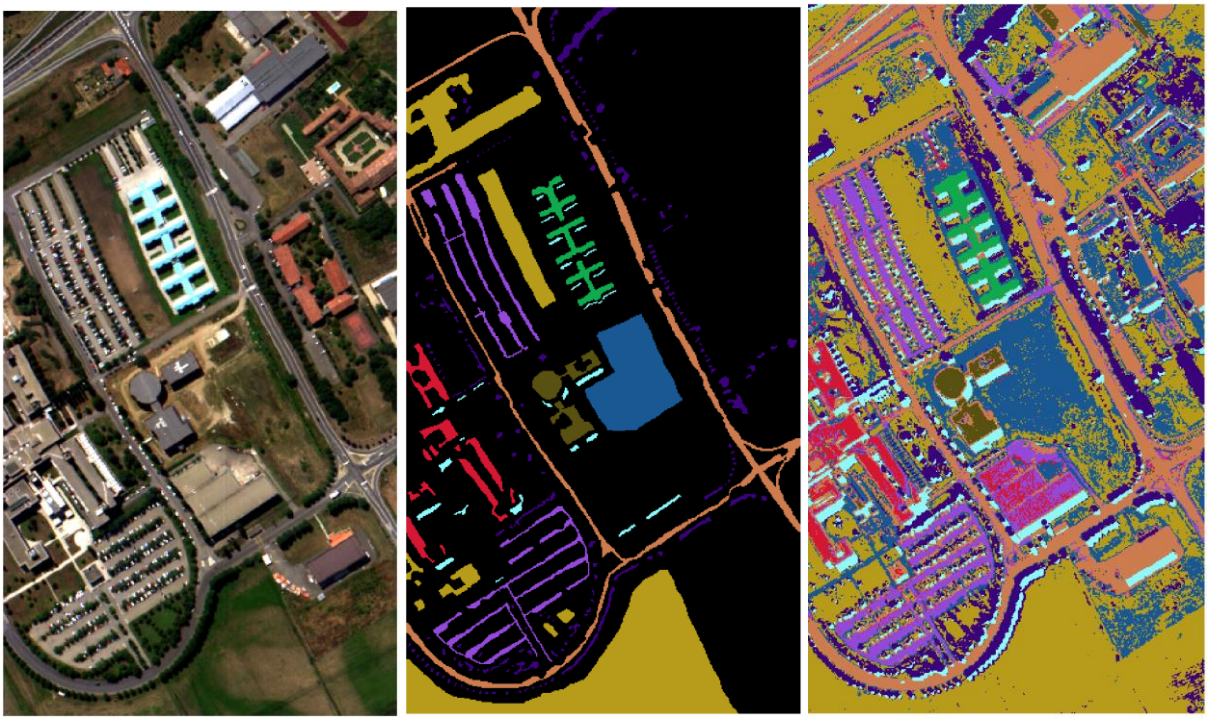
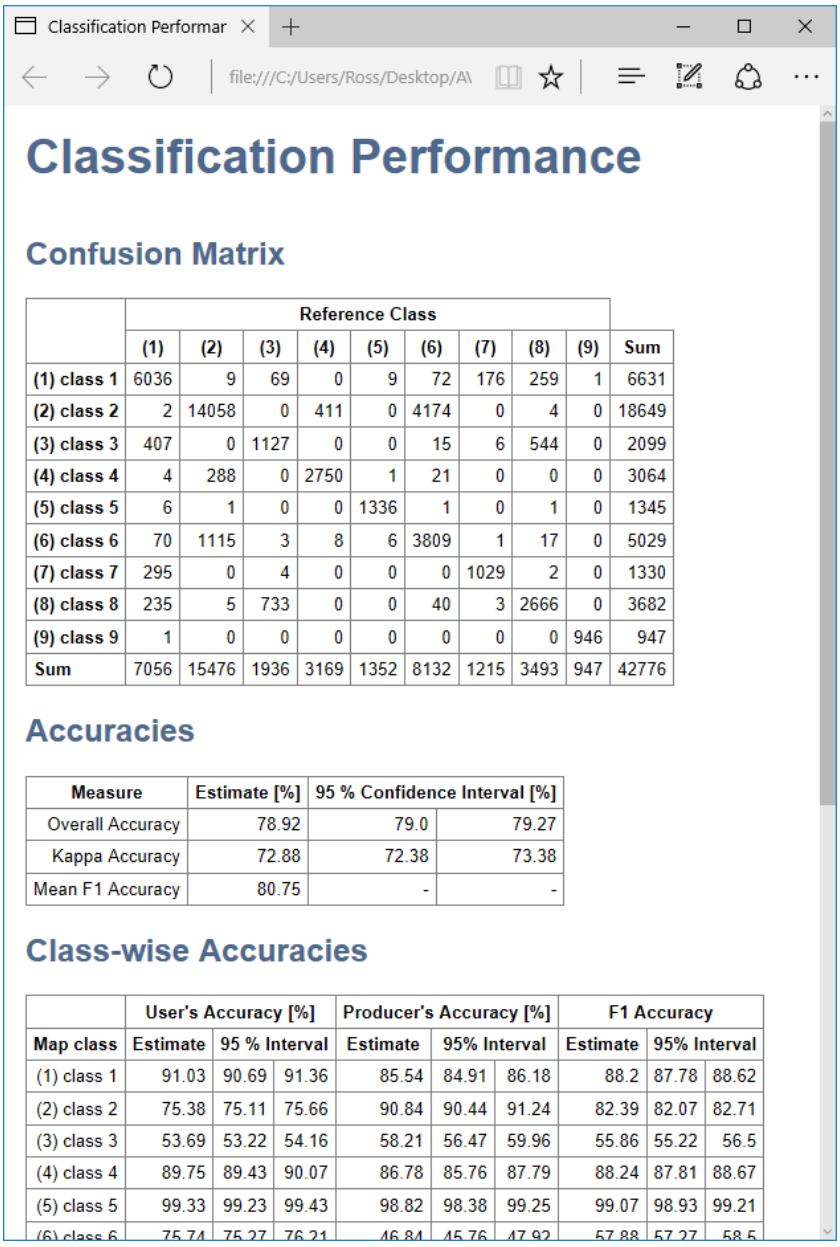


Figure (left) Pavia University Standard dataset (middle) Ground Truth and (right) Predicted result

6 CLASSIFICATION MODULE



Classification Performance

Confusion Matrix

	Reference Class									
	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	Sum
(1) class 1	6036	9	69	0	9	72	176	259	1	6631
(2) class 2	2	14058	0	411	0	4174	0	4	0	18649
(3) class 3	407	0	1127	0	0	15	6	544	0	2099
(4) class 4	4	288	0	2750	1	21	0	0	0	3064
(5) class 5	6	1	0	0	1336	1	0	1	0	1345
(6) class 6	70	1115	3	8	6	3809	1	17	0	5029
(7) class 7	295	0	4	0	0	0	1029	2	0	1330
(8) class 8	235	5	733	0	0	40	3	2666	0	3682
(9) class 9	1	0	0	0	0	0	0	0	946	947
Sum	7056	15476	1936	3169	1352	8132	1215	3493	947	42776

Accuracies

Measure	Estimate [%]	95 % Confidence Interval [%]	
Overall Accuracy	78.92	79.0	79.27
Kappa Accuracy	72.88	72.38	73.38
Mean F1 Accuracy	80.75	-	-

Class-wise Accuracies

	User's Accuracy [%]			Producer's Accuracy [%]			F1 Accuracy		
Map class	Estimate	95 % Interval		Estimate	95% Interval		Estimate	95% Interval	
(1) class 1	91.03	90.69	91.36	85.54	84.91	86.18	88.2	87.78	88.62
(2) class 2	75.38	75.11	75.66	90.84	90.44	91.24	82.39	82.07	82.71
(3) class 3	53.69	53.22	54.16	58.21	56.47	59.96	55.86	55.22	56.5
(4) class 4	89.75	89.43	90.07	86.78	85.76	87.79	88.24	87.81	88.67
(5) class 5	99.33	99.23	99.43	98.82	98.38	99.25	99.07	98.93	99.21
(6) class 6	75.74	75.27	76.21	46.84	45.76	47.92	57.88	57.27	58.5

Fig Classification Report in HTML format

Reference

Breiman, “Random Forests”, Machine Learning, 45(1), 5-32, 2001

6.1.3 K NEAREST NEIGHBOUR CLASSIFICATION TOOL

QGIS 3.XX → Hyperspectral → Classification → KNN

User Interface

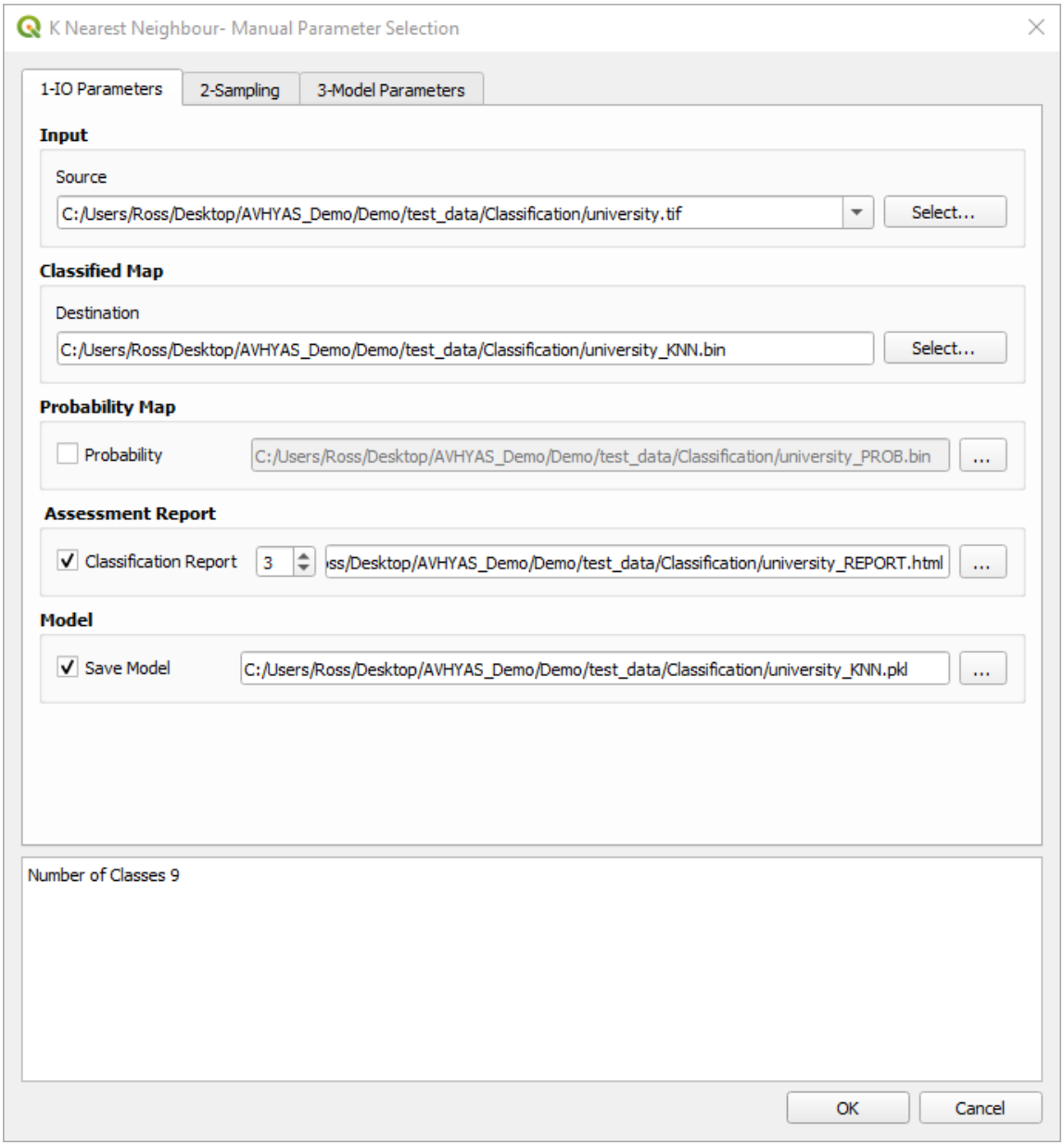


Figure User interface for KNN IO parameter tab

6 CLASSIFICATION MODULE

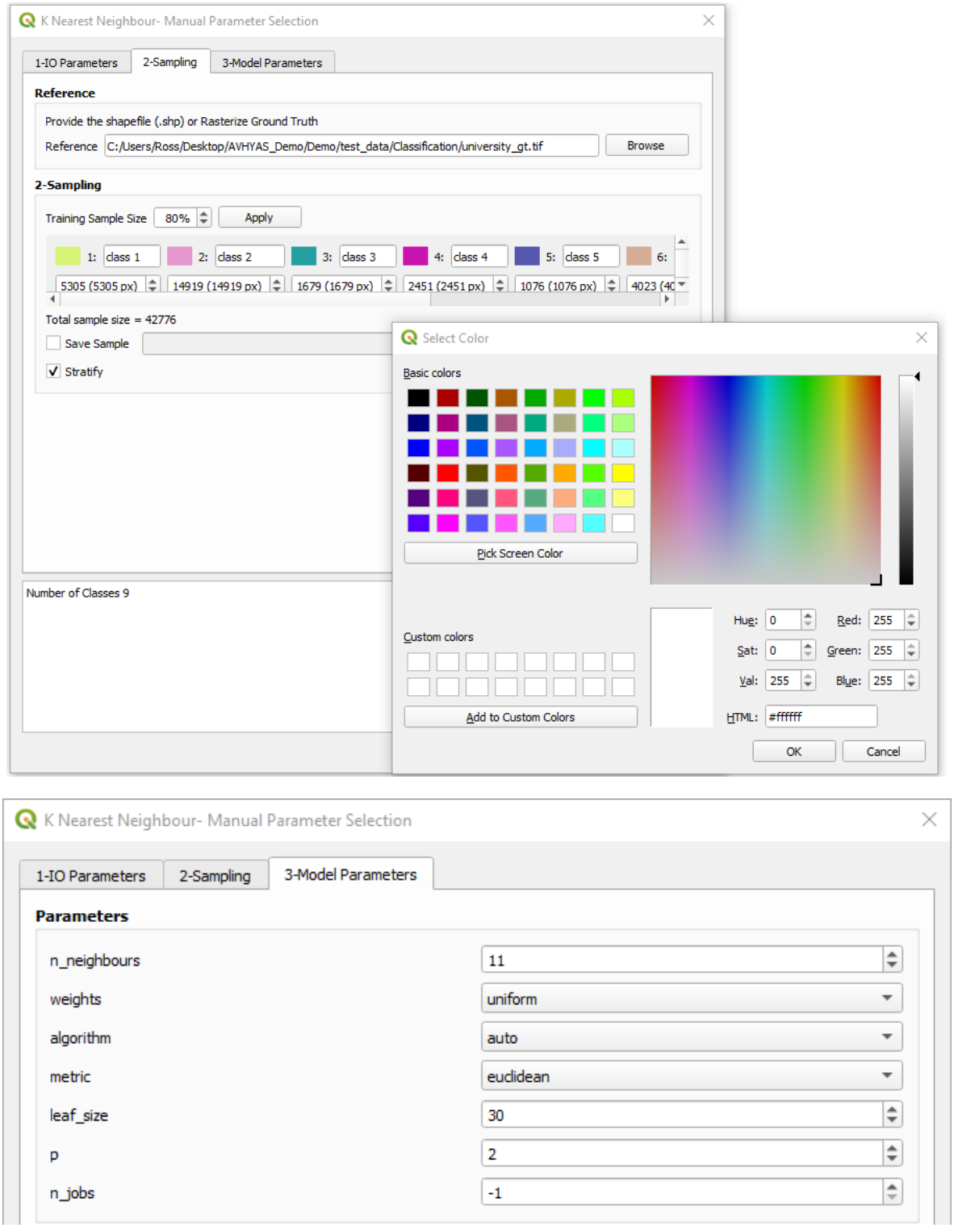


Figure User interface for Random Forest sampling and model paramter tab

6 CLASSIFICATION MODULE

I/O PARAMETERS

Type	Data format	DATA type	value range	Remark
Input				
Input File	GeoTiff / ENVI			Raster data to be used for classification Radiance/Reflectance
References	Shapefile/Raster			Ground Truth for classification
training sample size				Percentage of sample to be used for classification with optional for stratification
n_neighbours		int, default=5		Number of neighbors to use
weights		{'uniform', 'distance'} or callable, default='uniform'		weight function used in prediction. Possible values: ● 'uniform' : uniform weights. All points in each neighborhood are weighted equally. ● 'distance' : weight points by the inverse of their distance. in this case, closer neighbors of a query point will have a greater influence than neighbors which are further away. ● [callable] : a user-defined function

				which accepts an array of distances, and returns an array of the same shape containing the weights
algorithm		{'auto', 'ball_tree', 'kd_tree', 'brute'}, default='auto'		Algorithm used to compute the nearest neighbors: ● 'ball_tree' will use BallTree ● 'kd_tree' will use KDTree ● 'brute' will use a brute-force search. ● 'auto' will attempt to decide the most appropriate algorithm based on the values passed to fit method.
metric		int, default=30		The minimum number of samples required for split an internal node
leaf_size		Iint, default=2		leaf size passed to BallTree or KDTree. This can affect the speed of the construction and query, as well as the memory required to store the tree. The optimal value depends on the nature of the problem
p		str or callable, default='minkowski'		Power parameter for the Minkowski metric. When p = 1, this is equivalent to using manhattan_distance (l1), and euclidean_distance

				(l2) for p = 2. For arbitrary p, minkowski_distance (l_p) is used
n_jobs		int, default=None		The number of jobs ro run in parallel
Output				
Classified map	ENVI			Raster file with integer values for each class
probability map	ENVI			Raster file with probability of each class pixel wise for each band.
assessment report	HTML			The classification report and accuracy assessment
save model	pickle			Save the model for future inferences

Input Arguments

IO parameters

Input file: Specify the input raster for classification. Other field will be automatically populated with postfix added on the input name. User can change the default value if they want to otherwise they can proceed with the default values.

Classification map: Specify the destination file for the classification map to save to disk.

Probability map: ☑ Activate this setting in order to save the probability map ouput. Specify the destination file for the probability map.

Cross Validation and Assessment Report: ☑ Activate this setting to save the accuracy assessment report. User has the option to change the number of fold use in the cross validation process the default value is **3**. Specify the path to save the cross validation assessment report in HTML format.

Sampling

Reference: Specify the reference ground truth data either in the form of shapefile or rasterize file. If vector file is specified then user has to select the **Attribute** which contain

the numeric class information. After selection of the attributes if vector else user are suppose to click on the **Apply** button so as the class names and class colors become visible.

Training Sample Size : User can define and alter the sampling size relative to percentage amount of samples. Optional argument for stratification is also provided which user can select or ignore if they wish to.

Model parameters

n_neighbors: int, default=5

Number of neighbors to use by default for kneighbors queries.

weights{'uniform', 'distance'} or callable, default='uniform'

weight function used in prediction. Possible values:

- 'uniform' : uniform weights. All points in each neighborhood are weighted equally.
- 'distance' : weight points by the inverse of their distance. in this case, closer neighbors of a query point will have a greater influence than neighbors which are further away.
- [callable] : a user-defined function which accepts an array of distances, and returns an array of the same shape containing the weights.

algorithm :{'auto', 'ball_tree', 'kd_tree', 'brute'}, default='auto'

Algorithm used to compute the nearest neighbors:

- 'ball_tree' will use BallTree
- 'kd_tree' will use KDTree
- 'brute' will use a brute-force search.
- 'auto' will attempt to decide the most appropriate algorithm based on the values passed to fit method.

leaf_size: int, default=30

Leaf size passed to BallTree or KDTree. This can affect the speed of the construction and query, as well as the memory required to store the tree. The optimal value depends on the nature of the problem.

p: int, default=2

Power parameter for the Minkowski metric. When p = 1, this is equivalent to using manhattan_distance (l1), and euclidean_distance (l2) for p = 2. For arbitrary p, minkowski_distance (l_p) is used.

metric: str or callable, default='minkowski'

the distance metric to use for the tree. The default metric is minkowski, and with p=2 is equivalent to the standard Euclidean metric. See the documentation of DistanceMetric for a list of available metrics. If the metric is "precomputed", X is assumed to be a distance matrix and must be square during fit. X may be a sparse graph, in which case only "nonzero" elements may be considered neighbors.

metric_params: dict, default=None

Additional keyword arguments for the metric function.

n_jobs: int, default=None

The number of parallel jobs to run for neighbors search. None means 1 unless in a joblib.parallel_backend context. -1 means using all processors.

Overview

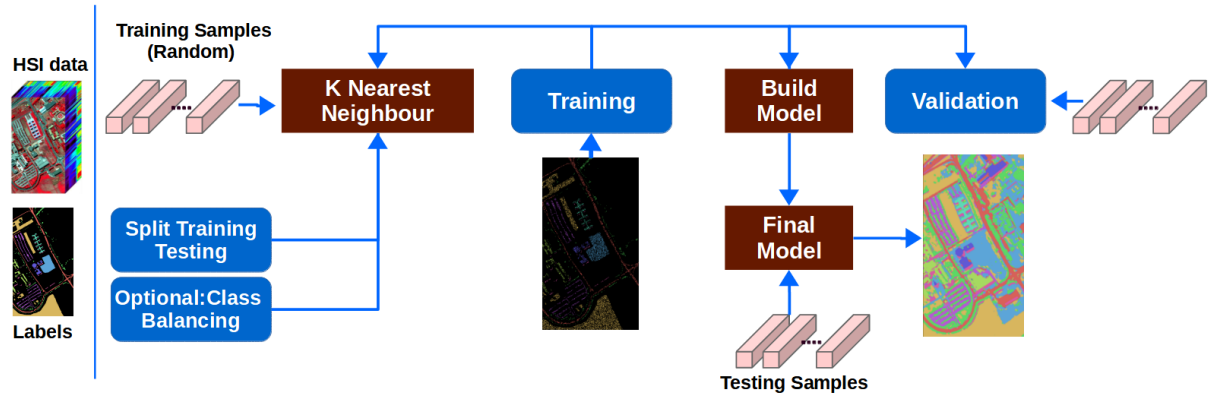


Figure: Data Flow Diagram

The k-Nearest Neighbors algorithm or KNN in short is a very simple but a powerful

technique. It works on the principle where the entire training dataset is stored and when a prediction is required, the k-most similar records to a new record from the training dataset are then located and have them vote on the label. Similarity between records can be measured in many different ways. A problem or data-specific method can be used. Generally, with numerical data, a good starting point is the Euclidean distance. Once the neighbors are discovered, the summary prediction can be made by returning the most common outcome or taking the average. As such, KNN can be used for classification or regression problems. There is no model to speak of other than holding the entire training dataset. Because no work is done until a prediction is required, KNN is often referred to as a lazy learning method.

This module similar to other classification algorithm provided here in AVHYAS relies on scikit-learn package in the back-end. The parameters are set as default based on scikit-learn documentation. User can change accordingly based on their data and their problem. Important parameters is the parameter *K* where user can set the number of neighbours to be considered for the voting process.

For more information kindly refer to scikit learn documentation on KNN.

Result

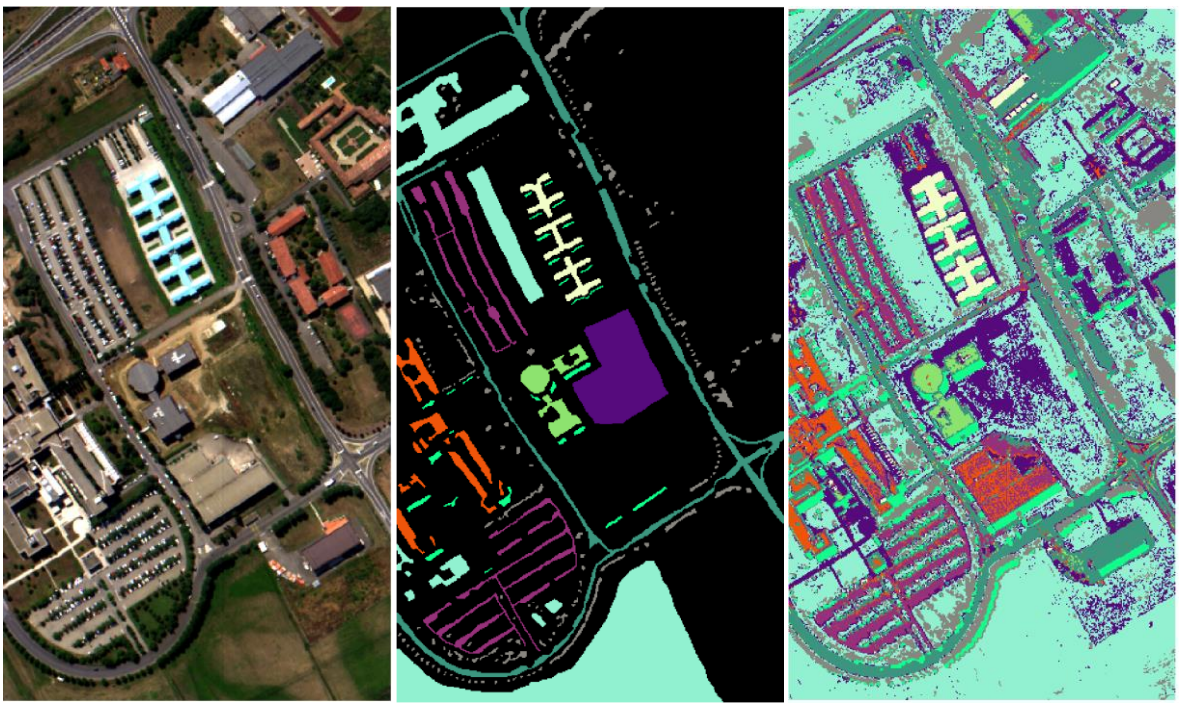
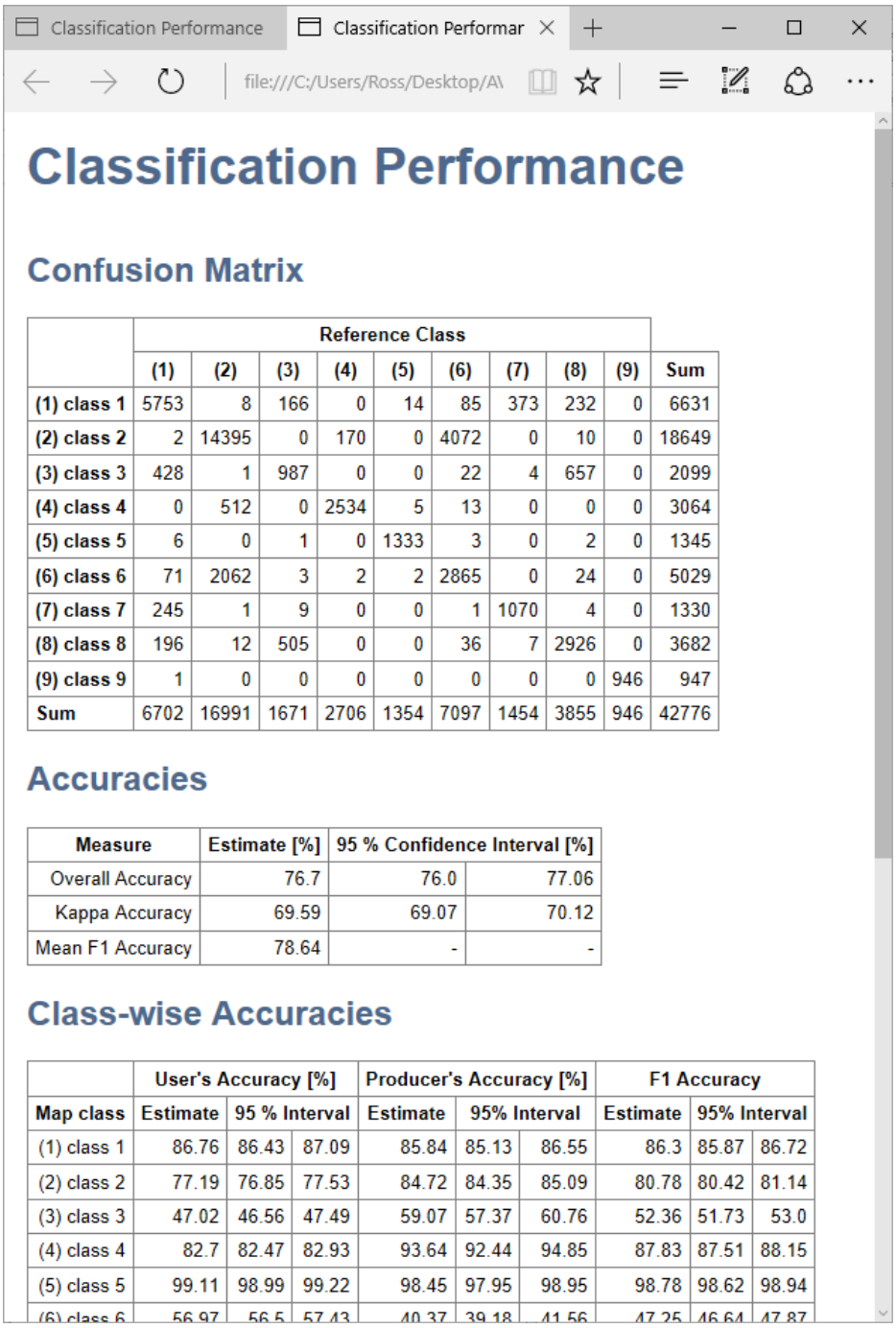


Figure: (Left) Pavia University standard data (Middle) Ground Truth (Right) Classified

6 CLASSIFICATION MODULE



Classification Performance

Confusion Matrix

	Reference Class									
	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	Sum
(1) class 1	5753	8	166	0	14	85	373	232	0	6631
(2) class 2	2	14395	0	170	0	4072	0	10	0	18649
(3) class 3	428	1	987	0	0	22	4	657	0	2099
(4) class 4	0	512	0	2534	5	13	0	0	0	3064
(5) class 5	6	0	1	0	1333	3	0	2	0	1345
(6) class 6	71	2062	3	2	2	2865	0	24	0	5029
(7) class 7	245	1	9	0	0	1	1070	4	0	1330
(8) class 8	196	12	505	0	0	36	7	2926	0	3682
(9) class 9	1	0	0	0	0	0	0	0	946	947
Sum	6702	16991	1671	2706	1354	7097	1454	3855	946	42776

Accuracies

Measure	Estimate [%]	95 % Confidence Interval [%]	
Overall Accuracy	76.7	76.0	77.06
Kappa Accuracy	69.59	69.07	70.12
Mean F1 Accuracy	78.64	-	-

Class-wise Accuracies

	User's Accuracy [%]			Producer's Accuracy [%]			F1 Accuracy		
Map class	Estimate	95 % Interval		Estimate	95% Interval		Estimate	95% Interval	
(1) class 1	86.76	86.43	87.09	85.84	85.13	86.55	86.3	85.87	86.72
(2) class 2	77.19	76.85	77.53	84.72	84.35	85.09	80.78	80.42	81.14
(3) class 3	47.02	46.56	47.49	59.07	57.37	60.76	52.36	51.73	53.0
(4) class 4	82.7	82.47	82.93	93.64	92.44	94.85	87.83	87.51	88.15
(5) class 5	99.11	98.99	99.22	98.45	97.95	98.95	98.78	98.62	98.94
(6) class 6	56.97	56.5	57.43	40.37	39.18	41.56	47.25	46.64	47.87

Figure: Cross Validation Assessment Report

6 CLASSIFICATION MODULE

6.1.4 SUPPORT VECTOR MACHINE CLASSIFICATION TOOL

QGIS 3.XX → Hyperspectral→ Classification → Support Vector Machine

User Interface

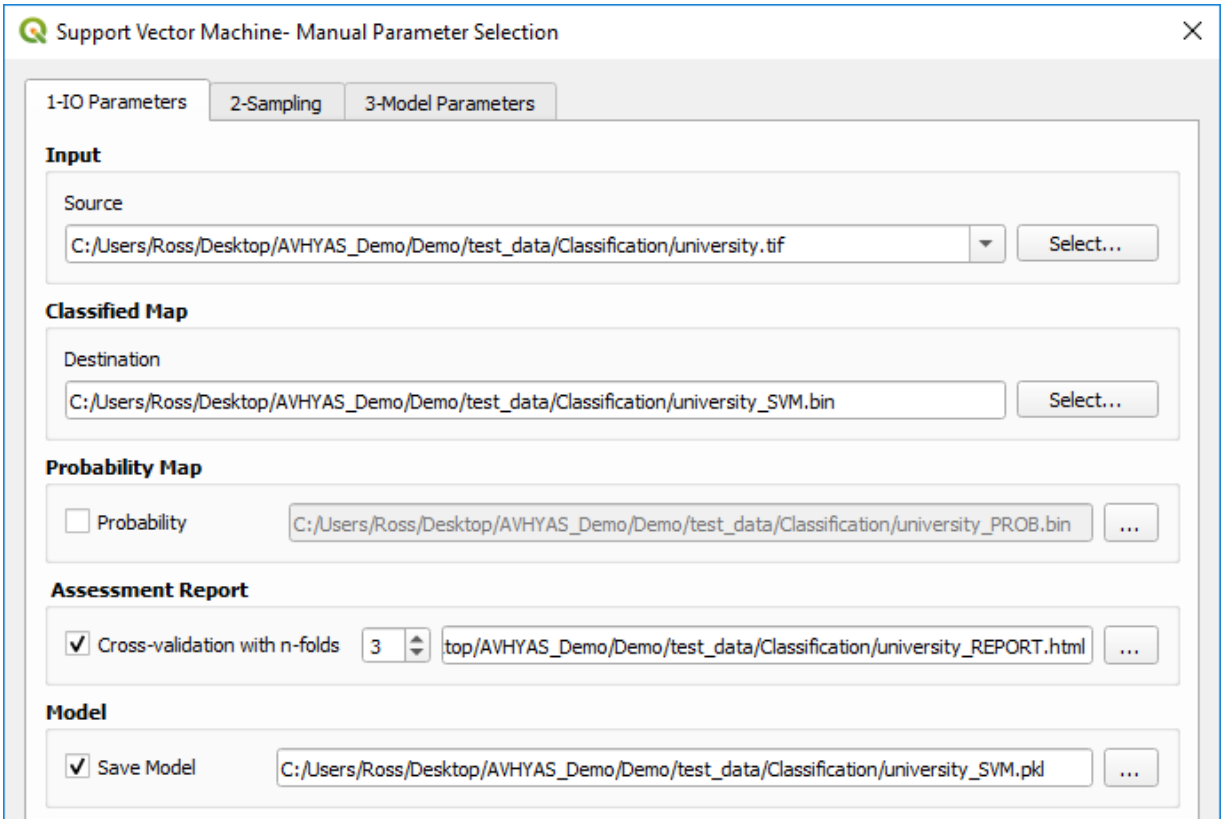


Figure: IO Parameters of SVM

6 CLASSIFICATION MODULE

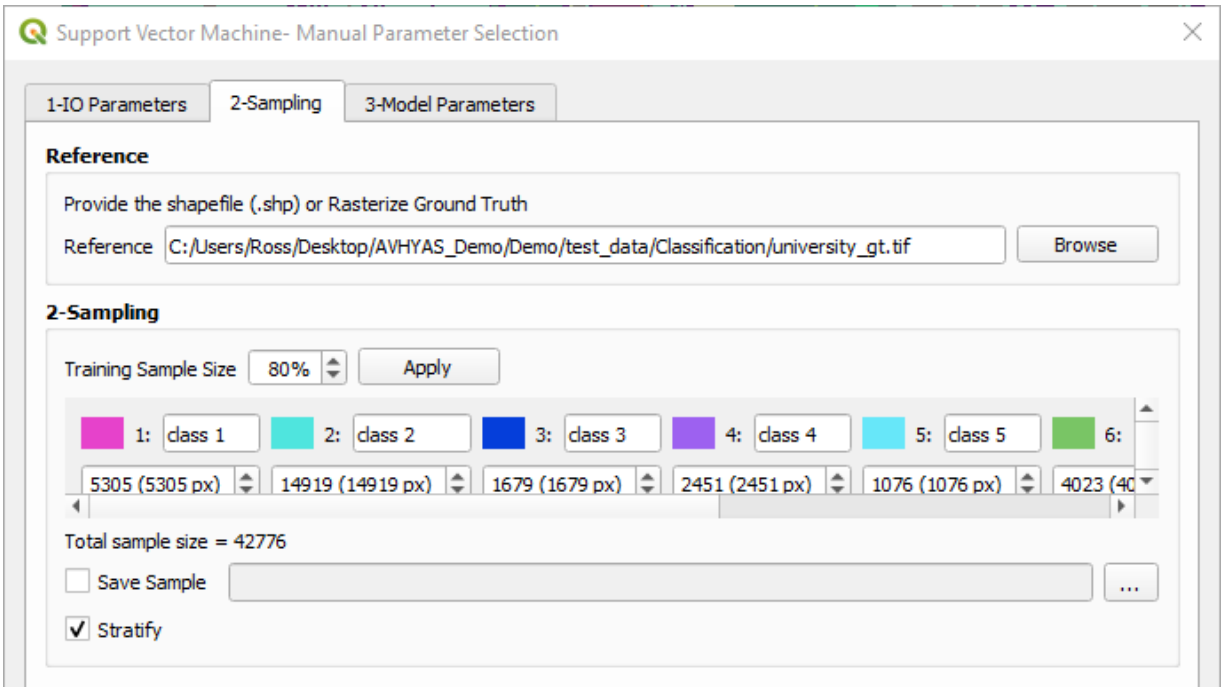


Figure: Sampling option of SVM

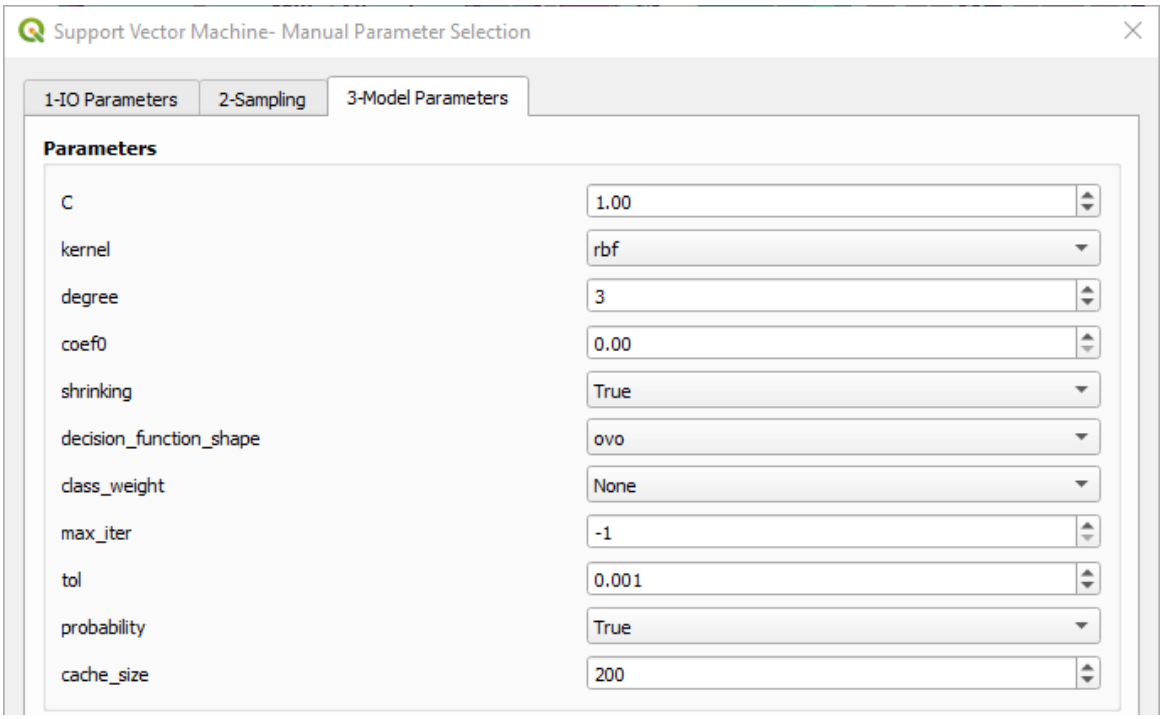


Figure: SVM Manual parameters

6 CLASSIFICATION MODULE

Type	Data format	DATA type	value range	Remark
Input				
Input File	GeoTiff / ENVI			Raster data to be used for classification Radiance/Reflectance
References	Shapefile/Raster			Ground Truth for classification
training sample size				Percentage of sample to be used for classification with optional for stratification
C		Float, default=1.0		Regularization parameter. The strength of the regularization is inversely proportional to C. Must be strictly positive. The penalty is a squared l2 penalty
kernel		String {'linear', 'poly', 'rbf', 'sigmoid'}, default='rbf'		Specifies the kernel type to be used in the algorithm. It must be one of 'linear', 'poly', 'rbf', 'sigmoid', 'precomputed' or a callable. If none is given, 'rbf' will be used. .
degree		Integer default=3		Degree of the polynomial kernel function ('poly'). Ignored by all other kernels.
gamma		String/Float {'scale', 'auto'} or float, default='scale'		Kernel coefficient for 'rbf', 'poly' and 'sigmoid'.
coef0		float, default=0.0		Independent term in kernel function. It is only

				significant in ‘poly’ and ‘sigmoid’
shrinking		bool, default=True		Whether to use the shrinking heuristic
probability		bool, default=False		Whether to enable probability estimates
tol		float, default=1e-3		Tolerance for stopping criterion.
cache_size		float, default=200		Specify the size of the kernel cache (in MB)
class_weight		dict or ‘balanced’, default=None		Set the parameter C of class i to class_weight[i]*C for SVC.
max_iter		int, default=-1		Hard limit on iterations within solver, or -1 for no limit.
random_state		int or RandomState instance, default=None		Controls the pseudo random number generation for shuffling the data for probability estimates
Output				
Classified map	ENVI			Raster file with integer values for each class
probability map	ENVI			Raster file with probability of each class pixel wise for each band.
assessment report	HTML			The classification report and accuracy assessment
save model	pickle			Save the model for future inferences

Input Arguments

IO parameters

Input file: Select the raster file to use for classification

6 CLASSIFICATION MODULE

Classification map: Select the destination file for the classification map

Probability map: Select the destination file for the probability map

Assessment Report: Select the destination file for saving the assessment report of the classification

Sampling

Reference: Select the reference ground truth data either shapefile or rasterize file.

Training Sample Size : Select the percentage of training samples to use for training and testing. Optional argument for stratification is also provided which user can select or ignore if they wish to.

Model parameters

n_estimators : Define the number of trees in the forest

criterion: Define function to measure the quality of a split

max_depth: Define the maximum depth of the tree

min_samples_split: Define the minimum number of samples required to be at a leaf

min_weight_fraction_leaf: Define the minimum weighted fraction of the sum total weights required to be at the leaf node

max_features: Define the number of features to consider when looking for the best split

max_leaf_nodes: Define the number of grow Trees with max_leaf_nodes in best first fashion

min_impurity_decrease: Define the number of trees in the forest

min_impurity_split: Define the Threshold for early stopping in tree growth

bootstrap: Define whether bootstrap samples are used when building trees

njobs: Define the number of trees in the forest

n_estimators : Define the number of jobs to run in parallel

C : float, default=1.0

Regularization parameter. The strength of the regularization is inversely proportional to C. Must be strictly positive. The penalty is a squared l2 penalty.

kernel: {'linear', 'poly', 'rbf', 'sigmoid', 'precomputed'}, default='rbf'

Specifies the kernel type to be used in the algorithm. It must be one of 'linear', 'poly', 'rbf', 'sigmoid', 'precomputed' or a callable. If none is given, 'rbf' will be used. If a callable is given it is used to pre-compute the kernel matrix from data matrices; that matrix should be an array of shape (n_samples, n_samples).

degree : int, default=3

Degree of the polynomial kernel function ('poly'). Ignored by all other kernels.

gamma : {'scale', 'auto'} or float, default='scale'

Kernel coefficient for 'rbf', 'poly' and 'sigmoid'.

- if gamma='scale' (default) is passed then it uses 1 / (n_features * X.var()) as value of gamma,
- if 'auto', uses 1 / n_features.

coef0: float, default=0.0

Independent term in kernel function. It is only significant in 'poly' and 'sigmoid'.

shrinking: bool, default=True

Whether to use the shrinking heuristic. **probability: bool, default=False**

Whether to enable probability estimates. This must be enabled prior to calling fit, will slow down that method as it internally uses 5-fold cross-validation, and predict_proba may be inconsistent with predict. tol: float, default=1e-3

Tolerance for stopping criterion.

cache_size: float, default=200

Specify the size of the kernel cache (in MB).

class_weight: dict or 'balanced', default=None

Set the parameter C of class i to class_weight[i]*C for SVC. If not given, all classes are supposed to have weight one. The "balanced" mode uses the values of y to automatically

adjust weights inversely proportional to class frequencies in the input data as n_samples / (n_classes * np.bincount(y))

verbose: bool, default=False

Enable verbose output. Note that this setting takes advantage of a per-process runtime setting in libsvm that, if enabled, may not work properly in a multithreaded context.

max_iter: int, default=-1

Hard limit on iterations within solver, or -1 for no limit.

random_state: int or RandomState instance, default=None

Controls the pseudo random number generation for shuffling the data for probability estimates. Ignored when probability is False. Pass an int for reproducible output across multiple function calls.

Overview

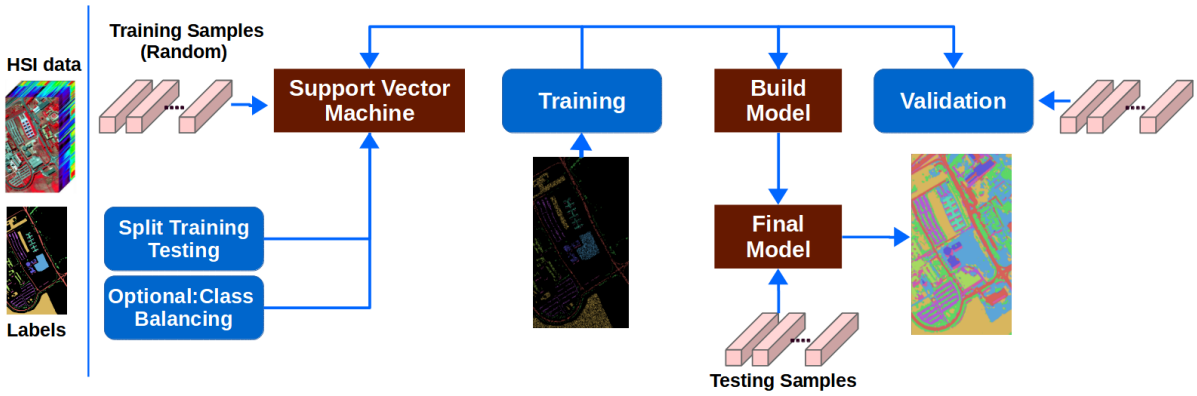


Figure: Data Flow Diagram

Support Vector Machines are perhaps one of the most popular and talked about machine learning algorithms developed by Vapnik. They were extremely popular around the time they were developed in the 1990s and continue to be the go-to method for a high-performing algorithm with little tuning. SVM is an exciting algorithm and the concepts are relatively simple.

6 CLASSIFICATION MODULE

Maximal-Margin Classifier

The Maximal-Margin Classifier is a hypothetical classifier that best explains how SVM works in practice. The numeric input variables (x) in your data (the columns) form an n-dimensional space. For example, if you had two input variables, this would form a two-dimensional space. A hyperplane is a line that splits the input variable space. In SVM, a hyperplane is selected to best separate the points in the input variable space by their class, either class 0 or class 1. In two-dimensions you can visualize this as a line and let's assume that all of our input points can be completely separated by this line. For example:

$$B_0 + (B_1 * X_1) + (B_2 * X_2) = 0$$

Where the coefficients (B1 and B2) that determine the slope of the line and the intercept (B0) are found by the learning algorithm, and X1 and X2 are the two input variables. We can make classifications using this line. By plugging in input values into the line equation, we can calculate whether a new point is above or below the line.

Above the line, the equation returns a value greater than 0 and the point belongs to the first class (class 0). Below the line, the equation returns a value less than 0 and the point belongs to the second class (class 1). A value close to the line returns a value close to zero and the point may be difficult to classify. If the magnitude of the value is large, the model may have more confidence in the prediction. The distance between the line and the closest data points is referred to as the margin. The best or optimal line that can separate the two classes is the line that as the largest margin. This is called the Maximal-Margin hyperplane. The margin is calculated as the perpendicular distance from the line to only the closest points. Only these points are relevant in defining the line and in the construction of the classifier. These points are called the support vectors. They support or define the hyperplane. The hyperplane is learned from training data using an optimization procedure that maximizes the margin.

Soft Margin Classifier

In practice, real data is messy and cannot be separated perfectly with a hyperplane. The constraint of maximizing the margin of the line that separates the classes must be relaxed. This is often called the soft margin classifier. This change allows some points in the training data to violate the separating line. An additional set of coefficients are introduced that give the margin wiggle room in each dimension. These coefficients are sometimes called slack variables. This increases the complexity of the model as there are more parameters for the model to fit to the data to provide this complexity. A tuning parameter is introduced called simply C that defines the magnitude of the wiggle allowed

across all dimensions. The C parameters defines the amount of violation of the margin allowed. A C=0 is no violation and we are back to the inflexible Maximal-Margin Classifier described above. The larger the value of C the more violations of the hyperplane are permitted. During the learning of the hyperplane from data, all training instances that lie within the distance of the margin will affect the placement of the hyperplane and are referred to as support vectors. And as C affects the number of instances that are allowed to fall within the margin, C influences the number of support vectors used by the model. The smaller the value of C, the more sensitive the algorithm is to the training data (higher variance and lower bias). The larger the value of C, the less sensitive the algorithm is to the training data (lower variance and higher bias).

Support Vector Machines (Kernels)

The SVM algorithm is implemented in practice using a kernel. The learning of the hyperplane in linear SVM is done by transforming the problem using some linear algebra, which is out of the scope of this introduction to SVM. A powerful insight is that the linear SVM can be rephrased using the inner product of any two given observations, rather than the observations themselves. The inner product between two vectors is the sum of the multiplication of each pair of input values. For example, the inner product of the vectors [2, 3] and [5, 6] is 2*5 + 3*6 or 28. The equation for making a prediction for a new input using the dot product between the input (x) and each support vector (xi) is calculated as follows:

$$f(x) = B_0 + sum(a_i * (x,x_i))$$

This is an equation that involves calculating the inner products of a new input vector (x) with all support vectors in training data. The coefficients B0 and ai (for each input) must be estimated from the training data by the learning algorithm.

Linear Kernel SVM

The dot-product is called the kernel and can be re-written as:

$$K(x, x_i) = sum(x * x_i)$$

The kernel defines the similarity or a distance measure between new data and the support vectors. The dot product is the similarity measure used for linear SVM or a linear kernel because the distance is a linear combination of the inputs. Other kernels can be used that transform the input space into higher dimensions such as a Polynomial Kernel and a Radial Kernel. This is called the Kernel Trick. It is desirable to use more complex kernels as it allows lines to separate the classes that are curved or even more complex.

This in turn can lead to more accurate classifiers.

Polynomial Kernel SVM

Instead of the dot-product, we can use a polynomial kernel, for example:

$$K(x,x_i) = 1 + sum(x * x_i)^d$$

Where the degree of the polynomial must be specified by hand to the learning algorithm. When d=1 this is the same as the linear kernel. The polynomial kernel allows for curved lines in the input space.

Radial Kernel SVM

Finally, we can also have a more complex radial kernel. For example:

$$K(x,xi) = exp(-gamma * sum((x - x_i^2))$$

Where gamma is a parameter that must be specified to the learning algorithm. A good default value for gamma is 0.1, where gamma is often 0 < gamma < 1. The radial kernel is very local and can create complex regions within the feature space, like closed polygons in two-dimensional space.

The SVM model needs to be solved using an optimization procedure. You can use a numerical optimization procedure to search for the coefficients of the hyperplane. This is inefficient and is not the approach used in widely used SVM implementations like LIBSVM. If implementing the algorithm as an exercise, you could use stochastic gradient descent. There are specialized optimization procedures that re-formulate the optimization problem to be a Quadratic Programming problem. The most popular method for fitting SVM is the Sequential Minimal Optimization (SMO) method that is very efficient. It breaks the problem down into sub-problems that can be solved analytically (by calculating) rather than numerically (by searching or optimizing).

Data Preparation for SVM

This section lists some suggestions for how to best prepare your training data when learning an SVM model. Numerical Inputs: SVM assumes that your inputs are numeric. If you have categorical inputs you may need to covert them to binary dummy variables (one variable for each category). Binary Classification: Basic SVM as described in this post is intended for binary (two-class) classification problems. Although, extensions have been developed for regression and multi-class classification.

6 CLASSIFICATION MODULE

This module provides an interface with all the parameters defined by scikit-learn for SVC classification. Kindly refer to scikit learn documentation on SVC for more information on the different parameters to be used.

Result

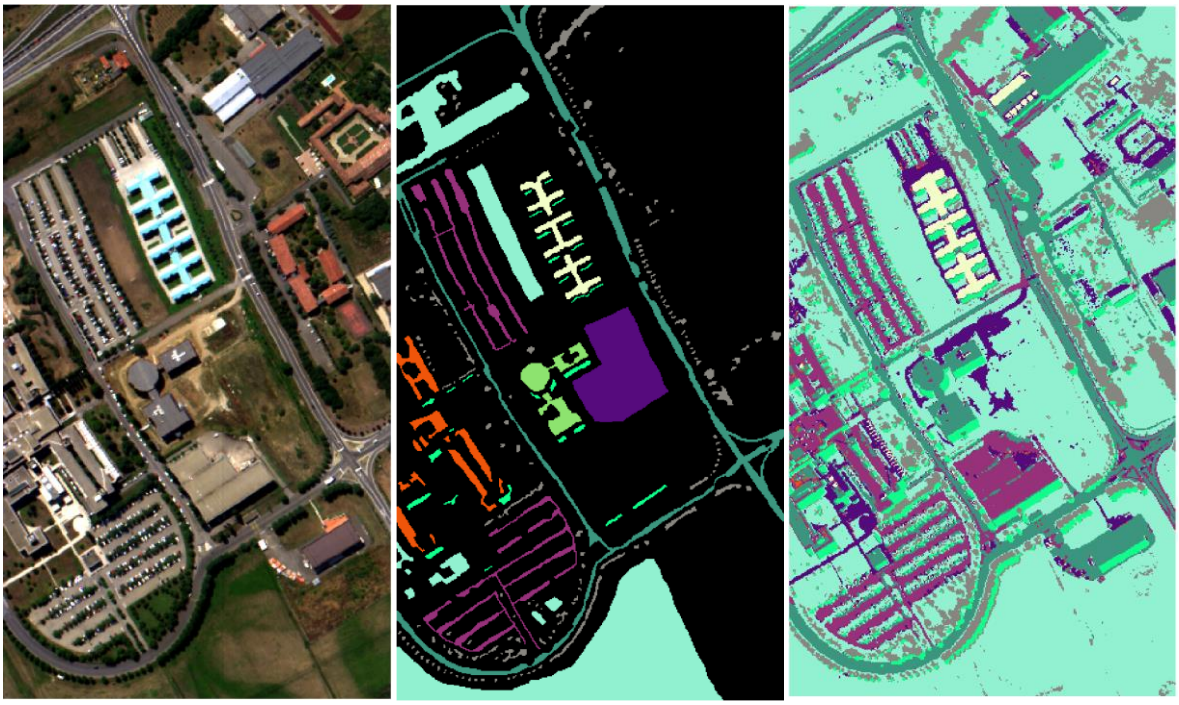
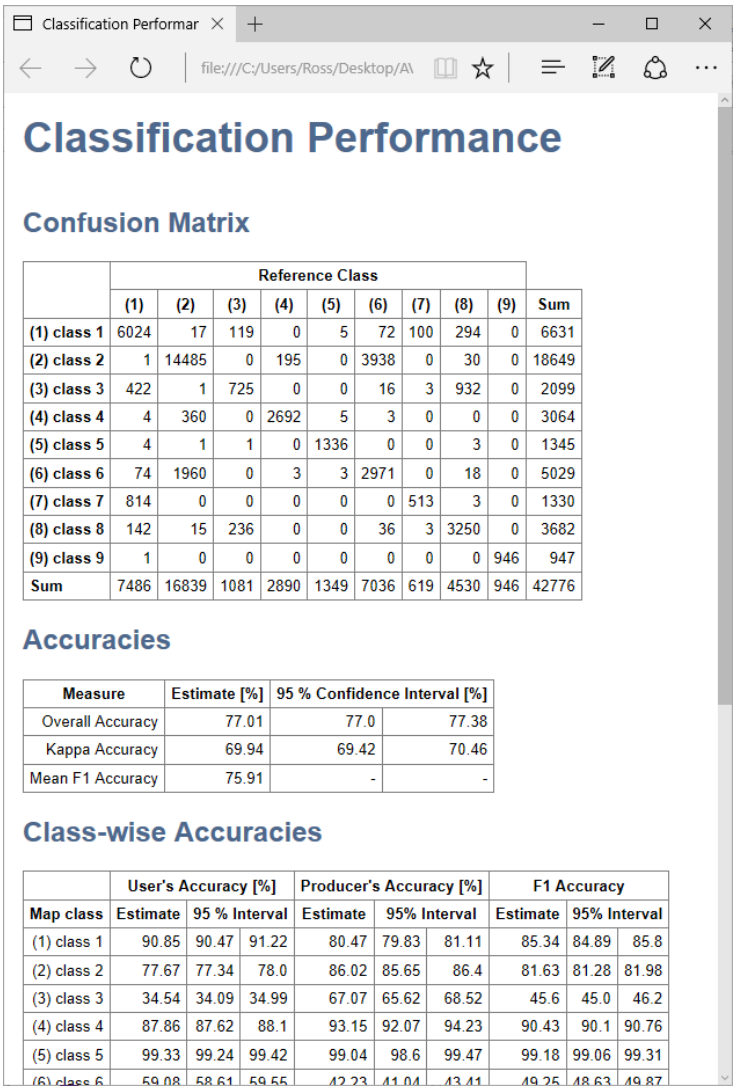


Figure: (Left) Pavia University standard data (Middle) Ground Truth (Right) Classified



Classification Performance

Confusion Matrix

	Reference Class									
	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	Sum
(1) class 1	6024	17	119	0	5	72	100	294	0	6631
(2) class 2	1	14485	0	195	0	3938	0	30	0	18649
(3) class 3	422	1	725	0	0	16	3	932	0	2099
(4) class 4	4	360	0	2692	5	3	0	0	0	3064
(5) class 5	4	1	1	0	1336	0	0	3	0	1345
(6) class 6	74	1960	0	3	3	2971	0	18	0	5029
(7) class 7	814	0	0	0	0	0	513	3	0	1330
(8) class 8	142	15	236	0	0	36	3	3250	0	3682
(9) class 9	1	0	0	0	0	0	0	0	946	947
Sum	7486	16839	1081	2890	1349	7036	619	4530	946	42776

Accuracies

Measure	Estimate [%]	95 % Confidence Interval [%]	
Overall Accuracy	77.01	77.0	77.38
Kappa Accuracy	69.94	69.42	70.46
Mean F1 Accuracy	75.91	-	-

Class-wise Accuracies

	User's Accuracy [%]			Producer's Accuracy [%]			F1 Accuracy		
Map class	Estimate	95 % Interval		Estimate	95% Interval		Estimate	95% Interval	
(1) class 1	90.85	90.47	91.22	80.47	79.83	81.11	85.34	84.89	85.8
(2) class 2	77.67	77.34	78.0	86.02	85.65	86.4	81.63	81.28	81.98
(3) class 3	34.54	34.09	34.99	67.07	65.62	68.52	45.6	45.0	46.2
(4) class 4	87.86	87.62	88.1	93.15	92.07	94.23	90.43	90.1	90.76
(5) class 5	99.33	99.24	99.42	99.04	98.6	99.47	99.18	99.06	99.31
(6) class 6	59.08	58.61	59.55	42.23	41.04	43.41	49.25	48.63	49.87

Reference

LIBSVM: A Library for Support Vector Machines

Platt, John (1999). “Probabilistic outputs for support vector machines and comparison to regularized likelihood methods.”

6 CLASSIFICATION MODULE

6.1.5 CLASSIFICATION WOKFLOW

QGIS 3.XX → Hyperspectral → Classification → classification workflow

I/O PARAMETERS

Field Name	Data format	DATA type/Type	value range	Remark
Input				
Input File	GeoTiff / ENVI	Raster		Specify input raster based on which samples will be drawn for training a classifier
References	Shapefile/ GeoTiff / ENVI	Vector/Raster		Specify the reference ground truth data either in shapefile or rasterize format
training sample size		Integer	0-100	Percentage of sample to be used for classification with optional for stratification
Model parameters		Valid Python Code		Scikit Learn Python code for different algorithms.
Output				
Classification	ENVI	Raster		Raster file with integer values for each class
probability	ENVI	Raster		Raster file with probability of each class pixel wise for each band.
Cross Validation accuracy assessment	HTML	Integer		The cross validation classification report and accuracy assessment
save model	pickle			Output path of the model for future inferences

6 CLASSIFICATION MODULE

User interface

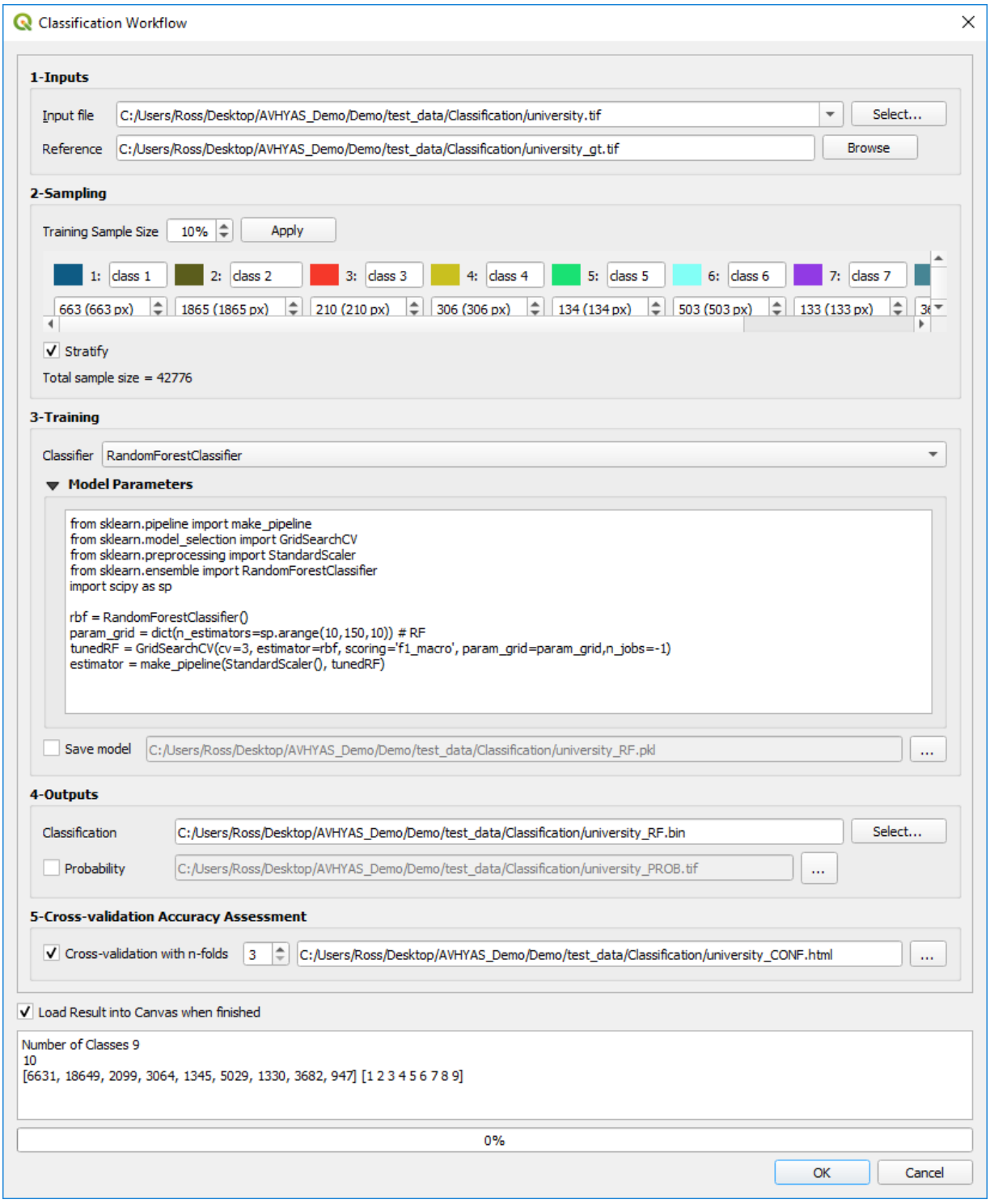


Figure User interface for Classification workflow

6 CLASSIFICATION MODULE

Input Arguments

Input:

Input file: Specify the input raster based on which samples will be drawn for training a classifier. The raster format supported so far is GeoTiff and ENVI.

Reference: Specify the reference ground truth data. It can either be vector dataset or a rasterize dataset. For vector data format it has to have a column in the attribute table with a unique class identifier (numeric).

Attribute: (if vector dataset is provided) attribute field in the reference vector layer which contains the unique class identifier

Output:

Classification: Output path where to write the classification image to.

Probability: Output path of the class probability image

Cross validation accuracy assessment: Clicking on the ☐ Cross validation with n-fold  activates this setting to assess the accuracy of the classification by performing cross validation. Specify the desired number of folds (default: 3). HTML report will be generated at the specified output path.

Sampling:

Training Sample Size:  Specify the sample size per class. Current supported sample size is based on percentage. The sample size will be shown below.

Training:

In the classifier  dropdown menu user can choose different classifiers (eg. Random Forest, Support Vector Machine)

Model Parameters: Specify the parameters of the selected classifier. A bare code template has been provided for the user to change accordingly for different classification algorithm. Scikit-learn python syntax is used here which user can specify model parameters accordingly. User can opt not to change the code and run straight away but however if they wish to change they can change and run directly. Have a look at the scikit-learn documentation on the individual parameters eg for the RandomForestClassifer

☐ **Save Model:** Activate this option to save the model file (.pkl) to disk

Overview

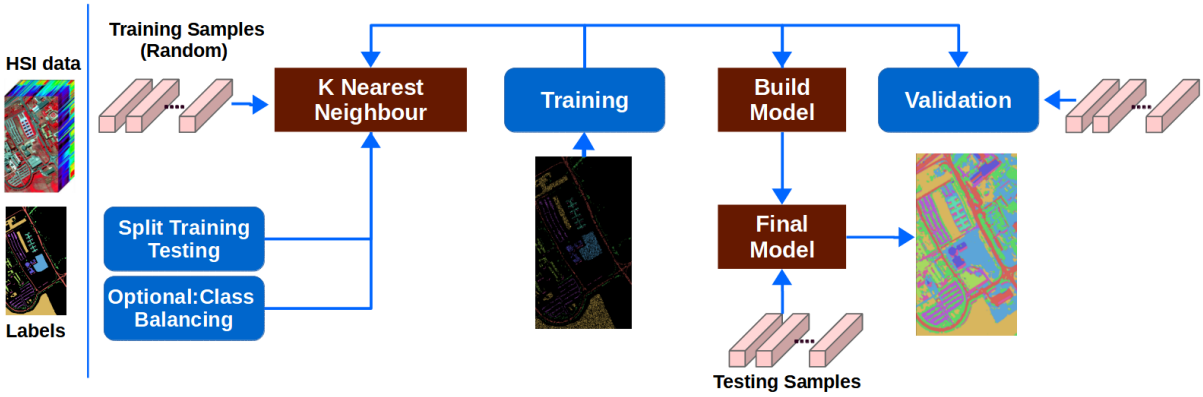


Figure: Data Flow Diagram

This module provides a flexible way of running different classification algorithms. The workflow designed and idea has been taken from Enmapbox Classification workflow with the backend module written and prepared in house. The whole purpose of a workflow is to provide an end to end framework for machine learning classification where user can feed in the data and the code and everything can be changed as per user's comfort and requirement. The bare template given in the model parameters however is sufficient for many classification problem but may not be optimum so user are requested to refer to scikit-learn python documentation on KNN, Random Forest, SVM for more detail options.

Steps for Classification workflow:

1. Provide the raster input file by clicking the select button
2. Provide the ground truth reference either in the form of a shapefile or a rasterize ground truth. When shapefile is provided make sure to have a categorical column. AVHYAS will auto populate categorical values in a combobox which user has to select.
3. Select the sampling ratio or percentage from the ground truth provided in step 2 and click on the button apply to run the sampling selection.
4. On clicking apply button selected sample size along with their colors will be loaded in the Ui, user has the option to change the final color of each classes based on their choice of colors or else leave them as default.

5. User are given option to select whether they want to save the probability map, assessment report and the trained model.
6. Output path will be prepopulated when in the input file is selected by adding a postfix in the input name such RF, SVM, KNN. User may change that if they want to.
7. Click ok to run and optionally user may click the load into canvas option if they want to load the classified output in the QGIS canvas.

6.1.6 PREDICT CLASSIFICATION

QGIS 3.XX → Hyperspectral→Classification → predict classification

User Interface

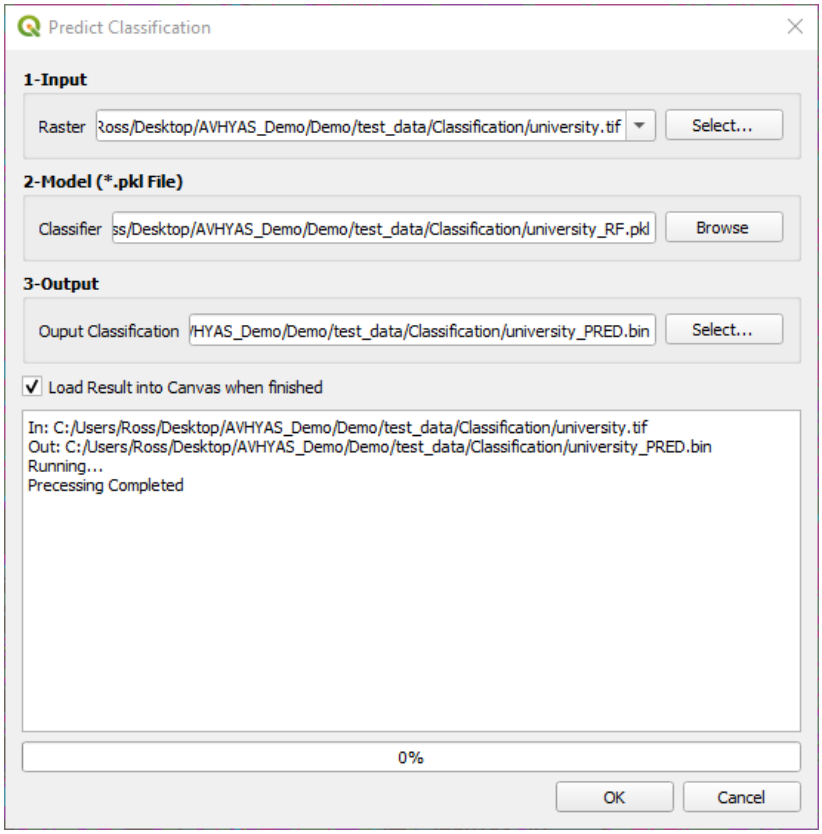


Figure User interface for predict classification

I/O PARAMETERS

Type	Data format	DATA type/Type	value range	Remark
Input				

6 CLASSIFICATION MODULE

Input File	GeoTiff / ENVI	Raster		Raster data to be used for classification Radiance/Reflectance
classifier	Pickle			Save trained model
Output				
Classified map	ENVI	Raster		Raster file with integer values for each class

Input Arguments

Input file : Specify the input raster where user wants to run a prediction based on the trained model. The raster format supported so far is GeoTiff and ENVI.

Classifier : Specify the path to the trained classifier model (.pkl)

Classification: Output path where to write the classification image to.

Overview

This module provides an interface for performing prediction based on a trained model on different datasets. It is important before running this module that user run the classification module and click on the save model option to save the train model in the file. AVHYAS uses pickle file format to save model in the file.

Steps:

1. Select the input raster where user wants to run a prediction based on the trained model
2. Provide a valid path to the trained model which is a pickle file save as with a pkl format extension
3. Like all the other module the output will be pre populated however user can change it if they want.
4. Click run and the process will start with the status of the processing shown in the log window.
5. If the load into canvas is selected, then the predicted result will be load in the QGIS canvas.

Result

6 CLASSIFICATION MODULE

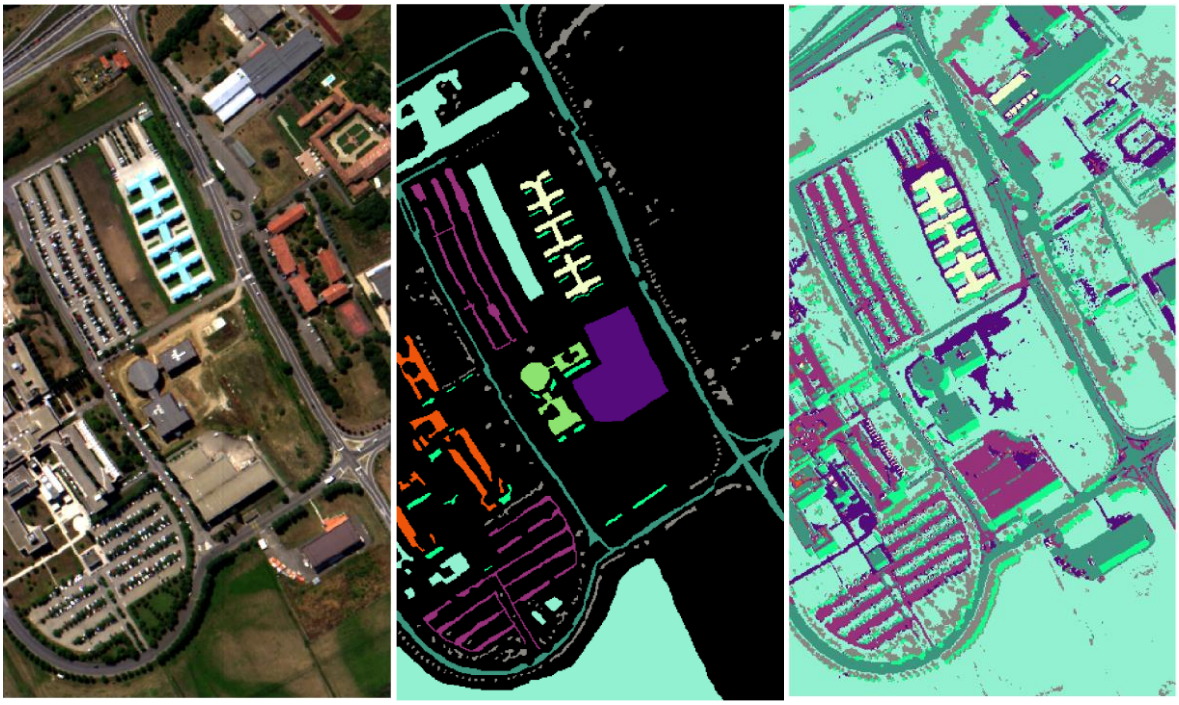


Figure: (Left) Pavia University standard data (Middle) Ground Truth (Right) Classified

6.1.7 CROSS VALIDATION CLASSIFICATION

QGIS 3.XX → Hyperspectral→ Classification → Cross Validation Classification

User Interface

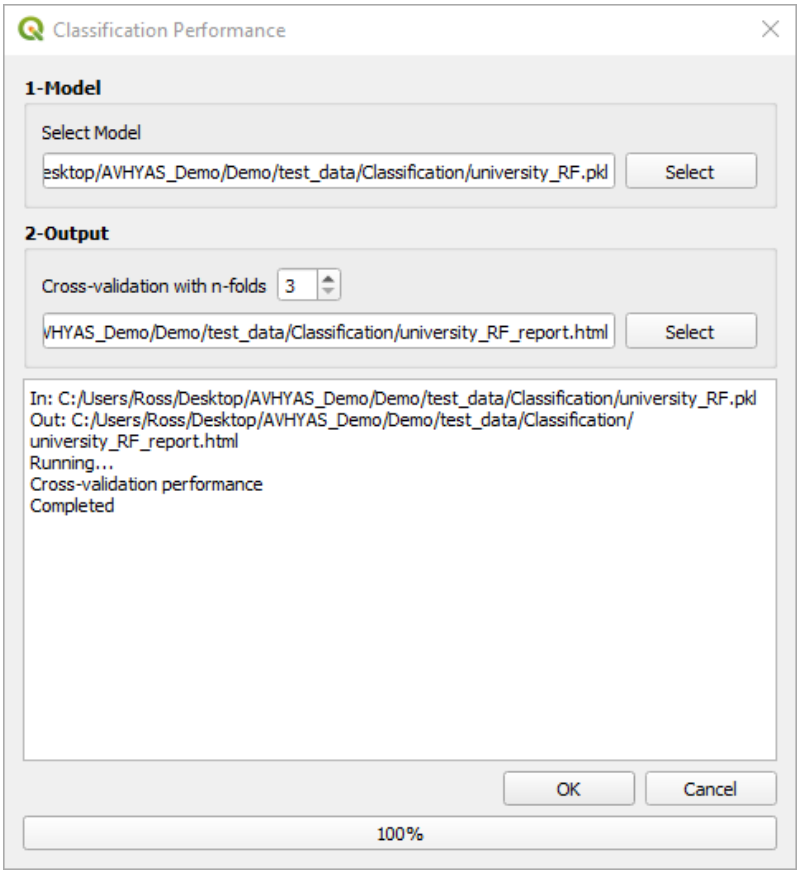


Figure User interface for classification performance

I/O PARAMETERS

Type	Data format	DATA type	value range	Remark
Input				
Model	Pickle			Save trained model
Output				

Assessment Report	HTML			HTML file containing different assessment report

Input Arguments

Model: Specify the input raster where user wants to run a prediction based on the trained model. The raster format supported so far is GeoTiff and ENVI.

Cross validation accuracy assessment: Clicking on the ☐ Cross validation with n-fold activates this setting to assess the accuracy of the classification by performing cross validation. Specify the desired number of folds (default: 3). HTML report will be generated at the specified output path.

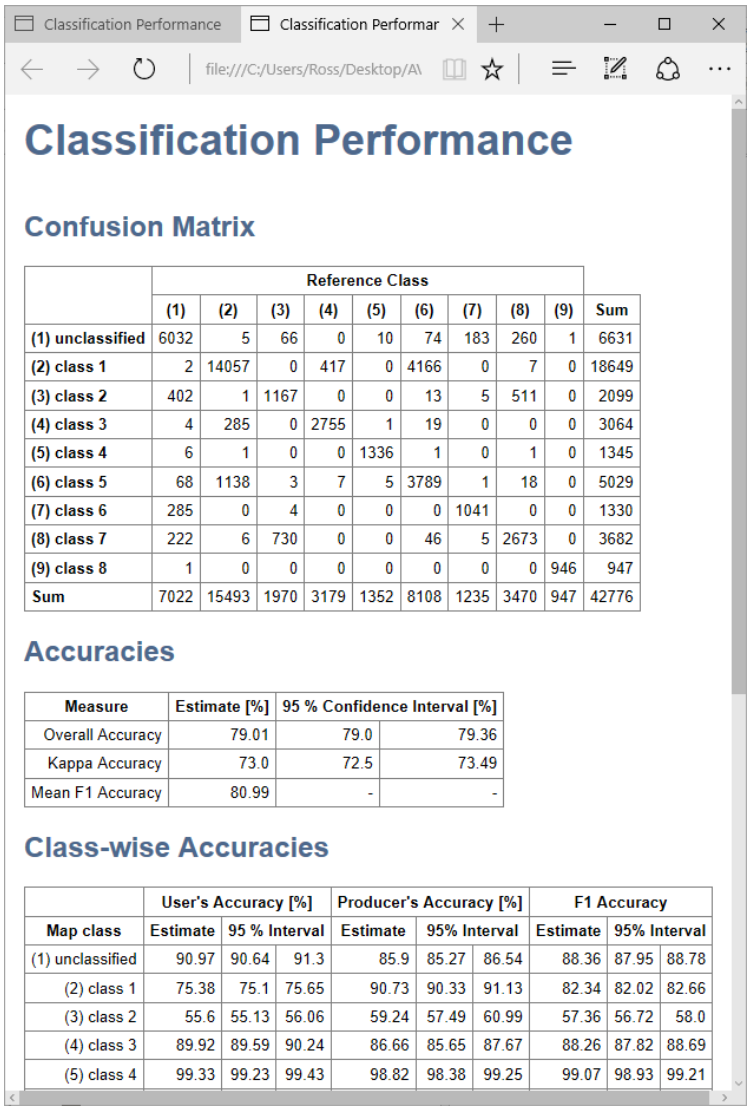
Overview

This module provides an interface for performing cross validation performance. The module requires the saved pickled trained model where along the model all the other necessary parameters are also saved for performing the cross validation performance.

Steps:

1. Provide a valid trained model which is a pickle file save with a pkl format extension
2. Provide the number of cross validation performance split to perform on the data
3. Click run and the process will start with the status of the processing shown in the log window.
4. On finished an assessment report will be pop up in your default browser.

Result



Classification Performance

Confusion Matrix

	Reference Class									
	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	Sum
(1) unclassified	6032	5	66	0	10	74	183	260	1	6631
(2) class 1	2	14057	0	417	0	4166	0	7	0	18649
(3) class 2	402	1	1167	0	0	13	5	511	0	2099
(4) class 3	4	285	0	2755	1	19	0	0	0	3064
(5) class 4	6	1	0	0	1336	1	0	1	0	1345
(6) class 5	68	1138	3	7	5	3789	1	18	0	5029
(7) class 6	285	0	4	0	0	0	1041	0	0	1330
(8) class 7	222	6	730	0	0	46	5	2673	0	3682
(9) class 8	1	0	0	0	0	0	0	0	946	947
Sum	7022	15493	1970	3179	1352	8108	1235	3470	947	42776

Accuracies

Measure	Estimate [%]	95 % Confidence Interval [%]	
Overall Accuracy	79.01	79.0	79.36
Kappa Accuracy	73.0	72.5	73.49
Mean F1 Accuracy	80.99	-	-

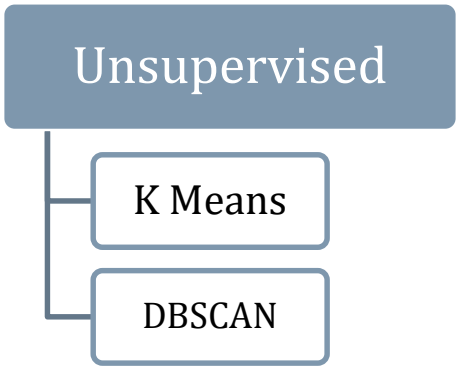
Class-wise Accuracies

	User's Accuracy [%]			Producer's Accuracy [%]			F1 Accuracy		
Map class	**Estimate**	**95 % Interval**		**Estimate**	**95% Interval**		**Estimate**	**95% Interval**	
(1) unclassified	90.97	90.64	91.3	85.9	85.27	86.54	88.36	87.95	88.78
(2) class 1	75.38	75.1	75.65	90.73	90.33	91.13	82.34	82.02	82.66
(3) class 2	55.6	55.13	56.06	59.24	57.49	60.99	57.36	56.72	58.0
(4) class 3	89.92	89.59	90.24	86.66	85.65	87.67	88.26	87.82	88.69
(5) class 4	99.33	99.23	99.43	98.82	98.38	99.25	99.07	98.93	99.21

Figure Classification performance HTML report

6 CLASSIFICATION MODULE

6.2 UNSUPERVISED



Overview

Cluster analysis, or clustering, is an unsupervised machine learning task. It involves automatically discovering natural grouping in data. Unlike supervised learning (like predictive modeling), clustering algorithms only interpret the input data and find natural groups or clusters in feature space. “Clustering techniques apply when there is no class to be predicted but rather when the instances are to be divided into natural groups”. A cluster is often an area of density in the feature space where examples from the domain (observations or rows of data) are closer to the cluster than other clusters. The cluster may have a center (the centroid) that is a sample or a point feature space and may have a boundary or extent.

Clustering can be helpful as a data analysis activity in order to learn more about the problem domain, so-called pattern discovery or knowledge discovery.

For example:

- The phylogenetic tree could be considered the result of a manual clustering analysis.
- Separating normal data from outliers or anomalies may be considered a clustering problem.
- Separating clusters based on their natural behavior is a clustering problem, referred to as market segmentation.

6 CLASSIFICATION MODULE

Clustering can also be useful as a type of feature engineering, where existing and new examples can be mapped and labeled as belonging to one of the identified clusters in the data.

Evaluation of identified clusters is subjective and may require a domain expert, although many clustering-specific quantitative measures do exist. Typically, clustering algorithms are compared academically on synthetic datasets with pre-defined clusters, which an algorithm is expected to discover.

6.2.1 K-MEANS CLUSTERING TOOL

QGIS 3.XX → Hyperspectral→ Classification → Clustering → K-Means

User Interface

6 CLASSIFICATION MODULE

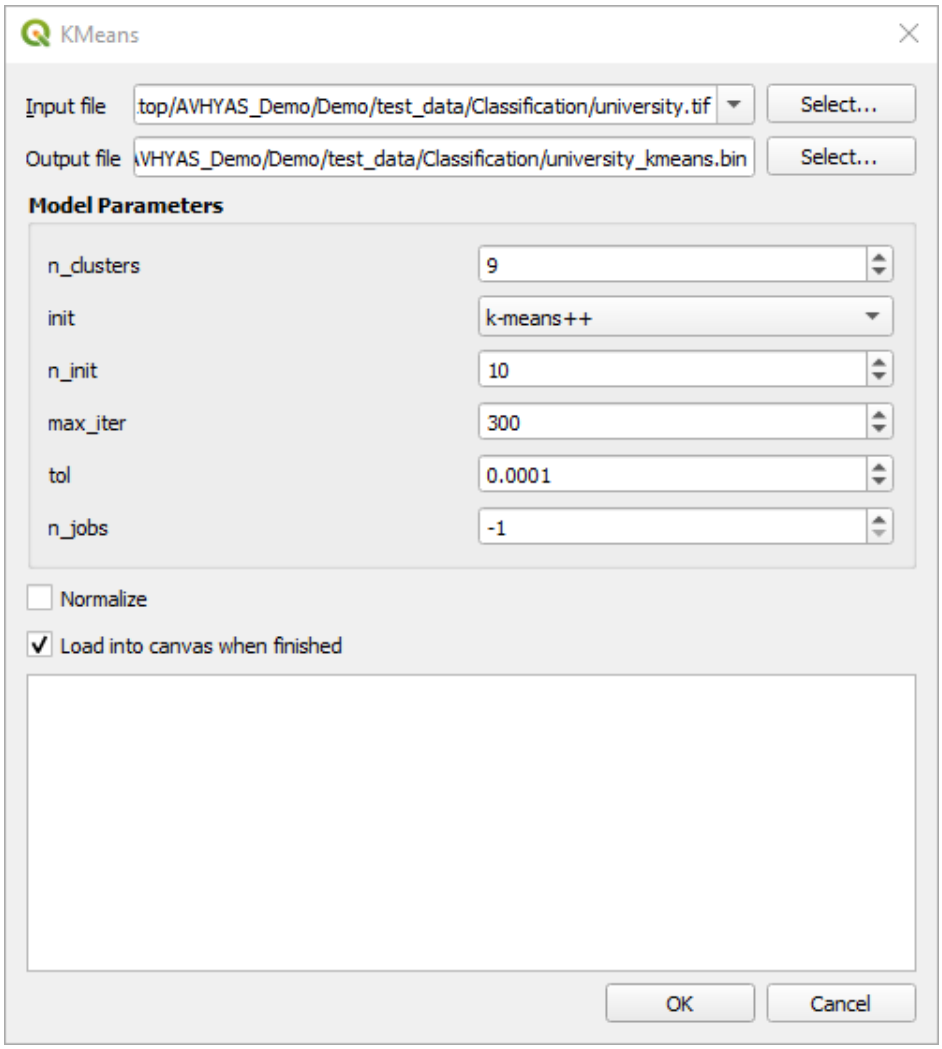


Figure User interface for K Means

I/O PARAMETERS

Type	Data format	DATA type	value range	Remark
Input				
Input file	ENVI/Geotiff Raster			Input Raster File Reflectance/Radiance
n_clusters				Number of output clusters
init			{'k-means++', 'random', ndarray,	Initialization algorithm

			callable}, default='k-means++'	
n_init			int, default=10	Number of times the k-means algorithm will be run with different centroid seeds.
max_iter			int, default=300	Maximum number of iterations of the k-means algorithm for a single run.
tol			Float, default=1e-4	Relative tolerance with regards to Frobenius norm of the difference in the cluster centers of two consecutive iterations to declare convergence
n_jobs			int, default=None	The number of OpenMP threads to use for the computation. Parallelism is sample-wise on the main cython loop which assigns each sample to its closest center.
Output				
Output file	ENVI			Output raster image

6 CLASSIFICATION MODULE

Input Arguments

Input File: Define input file for clustering.

Output File: Define output file

n_clusters: int, default=8 The number of clusters to form as well as the number of centroids to generate.

init: {'k-means++', 'random'}, default='k-means++,Method for initialization: 'k-means++' : selects initial cluster centers for k-mean clustering in a smart way to speed up convergence. See section Notes in k_init for more details. 'random': choose n_clusters observations (rows) at random from data for the initial centroids.

n_init: int, default=10 Number of time the k-means algorithm will be run with different centroid seeds. The final results will be the best output of n_init consecutive runs in terms of inertia.

max_iter: int, default=300 Maximum number of iterations of the k-means algorithm for a single run.

tol : float, default=1e-4 Relative tolerance with regards to Frobenius norm of the difference in the cluster centers of two consecutive iterations to declare convergence.

n_jobs: int, default=None The number of OpenMP threads to use for the computation. Parallelism is sample-wise on the main cython loop which assigns each sample to its closest center.

None or -1 means using all processors.

Overview

K-Means Clustering is the most widely known clustering algorithm and involves assigning examples to clusters in an effort to minimize the variance within each cluster.

1. To begin, we first select a number of classes/groups to use and randomly initialize their respective center points. To figure out the number of classes to use, it's good to take a quick look at the data and try to identify any distinct groupings.
2. Each data point is classified by computing the distance between that point and each group center, and then classifying the point to be in the group whose center is closest to it. The equation is given below:

$$d(p,q) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + ... + (q_n - p_n)^2}$$

$$d(p,q) = \sqrt{\sum_{i}^{n}((q_i - p_i)^2))}$$

3. Based on these classified points, we re-compute the group center by taking the mean of all the vectors in the group.
4. Repeat these steps for a set number of iterations or until the group centers don't change much between iterations. You can also opt to randomly initialize the group centers a few times, and then select the run that looks like it provided the best results.

K-Means has the advantage that it's pretty fast, as all we're really doing is computing the distances between points and group centers; very few computations! It thus has a linear complexity O(n).

On the other hand, K-Means has a couple of disadvantages. Firstly, you have to select how many groups/classes there are. This isn't always trivial and ideally with a clustering algorithm we'd want it to figure those out for us because the point of it is to gain some insight from the data. K-means also starts with a random choice of cluster centers and therefore it may yield different clustering results on different runs of the algorithm. Thus, the results may not be repeatable and lack consistency. Other cluster methods are more consistent.

Steps for performing K-Means Clustering

1. Select the input data where you want to run K-Means clustering
2. There are a number of parameters given in the user interface. Among the parameters the most important one is the number of clusters. The rest can be kept as default and it should work for almost all problems. However, if user wants to change and experiment around they can change each of the filters and play around. For more information regarding each parameters kind refer to scikit learn documentation on K-Means Clustering
3. Click on the ok button to run the module and additional there is an option to load the result on the canvas when finished.

Result

6 CLASSIFICATION MODULE

Figure Clustering result

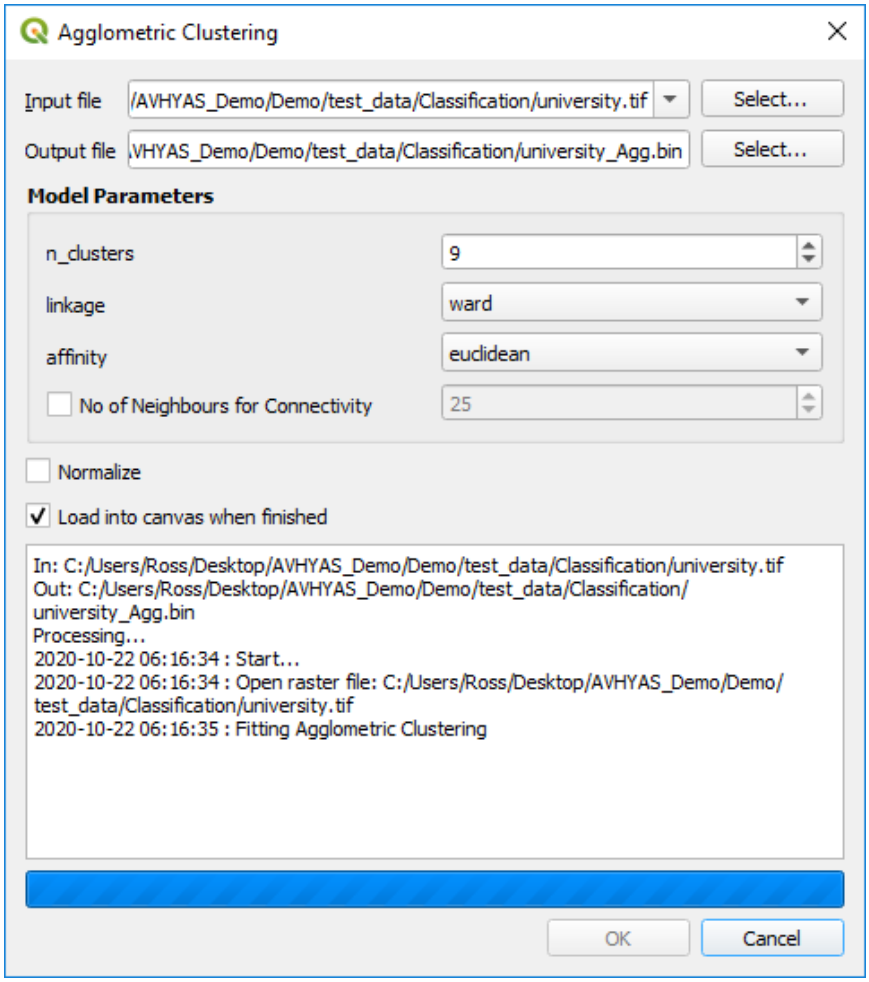
References

For detailed information kindly refer to scikit learn documentation on K-Means

6.2.2 AGGLOEMTETRIC CLUSTERING TOOL

QGIS 3.XX → Hyperspectral→Classification→Clustering→Agglometric Clustering

User Interface



I/O PARAMETERS

Type	Data format	DATA type	value range	Remark
Input				
Input file	ENVI/Geotiff Raster			Input Raster File Reflectance/Radiance
n_clusters			int,	Number of output clusters

			default=2	
linkage			{“ward”, “complete”, “average”, “single”}, default=”ward”	Linkage criterion to use
affinity			str, default=’euclidean’’	Metric use to calculate linkages
Output				
Output file	ENVI			Output raster image

Input Arguments

n_clusters: int or None, default=2 The number of clusters to find. It must be None if distance threshold is not None.

affinity: str or callable, default=’euclidean’ Metric used to compute the linkage. Can be “euclidean”, “l1”, “l2”, “manhattan”, “cosine”, or “precomputed”. If linkage is “ward”, only “euclidean” is accepted. If “precomputed”, a distance matrix (instead of a similarity matrix) is needed as input for the fit method.

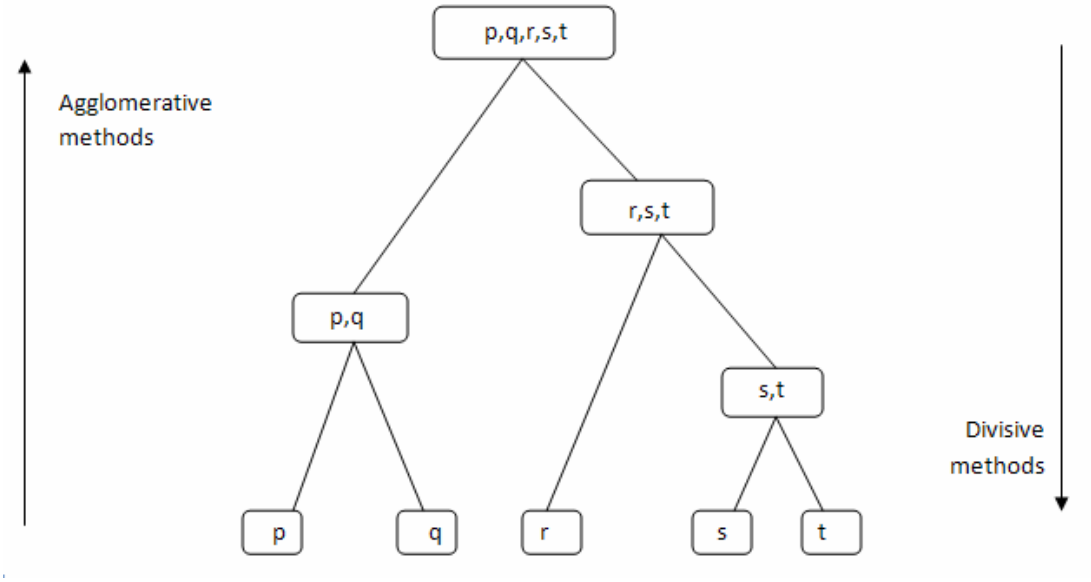
linkage : {“ward”, “complete”, “average”, “single”}, default=”ward” Which linkage criterion to use. The linkage criterion determines which distance to use between sets of observation. The algorithm will merge the pairs of cluster that minimize this criterion.

I. ward minimizes the variance of the clusters being merged.
II. average uses the average of the distances of each observation of the two sets.
III. complete or maximum linkage uses the maximum distances between all observations of the two sets.
IV. single uses the minimum of the distances between all observations of the two sets.

Overview

Hierarchical clustering algorithms group similar objects into groups called clusters. There are two types of hierarchical clustering algorithms Agglomerative which is a bottom up approach. It starts with many small clusters and merge them together to create bigger clusters. Divisive is a top down approach. Starts with a single cluster than

break it up into smaller clusters.



Some pros and cons of Hierarchical Clustering

Pros

- No assumption of a particular number of clusters (i.e. k-means)
- May correspond to meaningful taxonomies

Cons

- Once a decision is made to combine two clusters, it can't be undone
- Too slow for large data sets, O(n2 log(n))

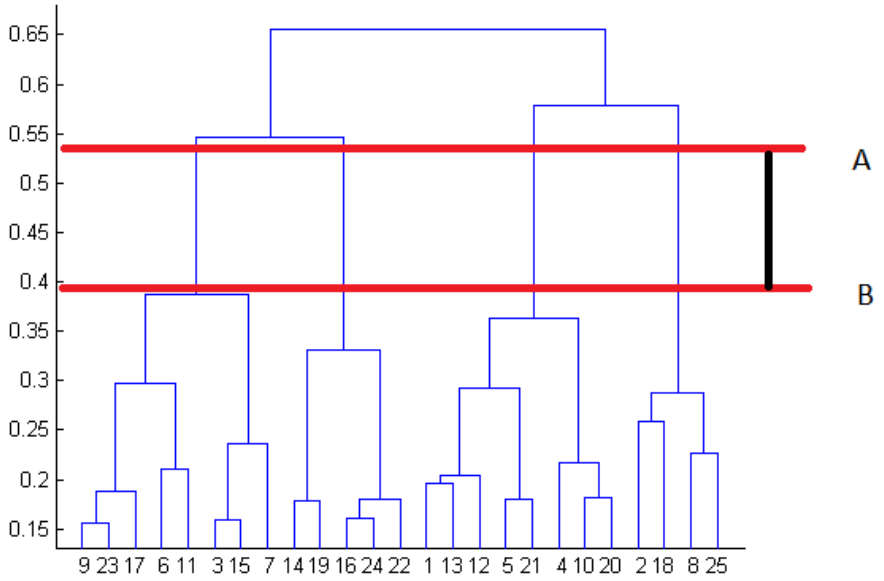
Steps in Agglomerative clustering

1. Make each data point a cluster
2. Take the two closest clusters and make them one cluster
3. Repeat step 2 until there is only one cluster

Dendrograms

Dendograms can be used to visualize the history of groupings and figure out the optimal number of clusters.

Determine the largest vertical distance that doesn't intersect any of the other clusters



Draw a horizontal line at both extremities

The optimal number of clusters is equal to the number of vertical lines going through the horizontal line

For eg., in the above case, best choice for no. of clusters will be 4.

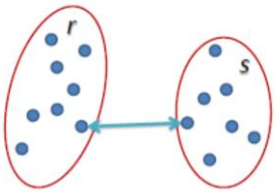
Linkage Criteria

Similar to gradient descent, you can tweak certain parameters to get drastically different results.

The linkage criteria refer to how the distance between clusters is calculated.

Single Linkage

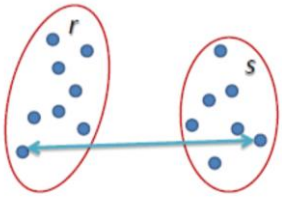
The distance between two clusters is the shortest distance between two points in each cluster



$$L(r,s) = \min(D(x_{ri}, x_{sj}))$$

Complete Linkage

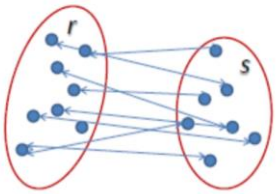
The distance between two clusters is the longest distance between two points in each cluster



$$L(r,s) = \max(D(x_{ri}, x_{sj}))$$

Average Linkage

The distance between clusters is the average distance between each point in one cluster to every point in other cluster

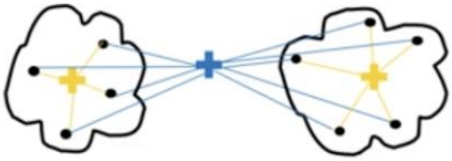


$$L(r,s) = \frac{1}{n_r n_s} \sum_{i=1}^{n_r} \sum_{j=1}^{n_s} D(x_{ri}, x_{sj})$$

Ward Linkage

The distance between clusters is the sum of squared differences within all clusters

6 CLASSIFICATION MODULE

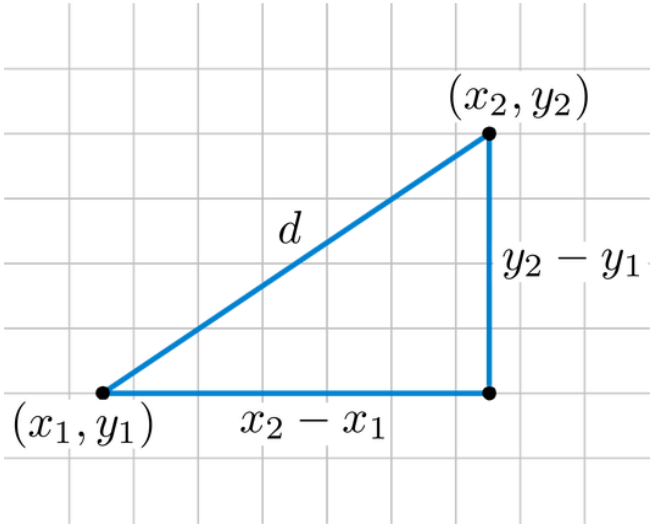


Distance Metric

The method you use to calculate the distance between data points will affect the end result.

Euclidean Distance

The shortest distance between two points. For example, if x=(a,b) and y=(c,d), the Euclidean distance between x and y is $\sqrt{(a-c)^2+(b-d)^2}$



Manhattan Distance

Imagine you were in the downtown center of a big city and you wanted to get from point A to point B. You wouldn't be able to cut across buildings, rather you'd have to make your way by walking along the various streets. For example, if x=(a,b) and y=(c,d), the Manhattan distance between x and y is |a−c|+|b−d|

Steps for performing Agglometric Clustering

1. Select the input data where you want to run clustering algorithm on

2. The user interface provided has minimal parameters provided which are sufficient for clustering of hyperspectral data as such. Other parameters are the default parameters defined by scikit learn module. User has to select the linkages and an affinity measurement metric as discussed in detail above. For more information regarding each parameters kind refer to scikit learn documentation on Agglometric Clustering .
3. User also has to provide the number of cluster which is the most important parameters of this module.
4. Click on the ok button to run the module and additional there is an option to load the result on the canvas when finished.

References

For detailed information kindly refer to scikit learn documentation on Agglometric Clustering

6.2.3 DBSCAN CLUSTERING TOOL

QGIS 3.XX → Hyperspectral→Classification→ Clustering→ DBSCAN Clustering

User Interface

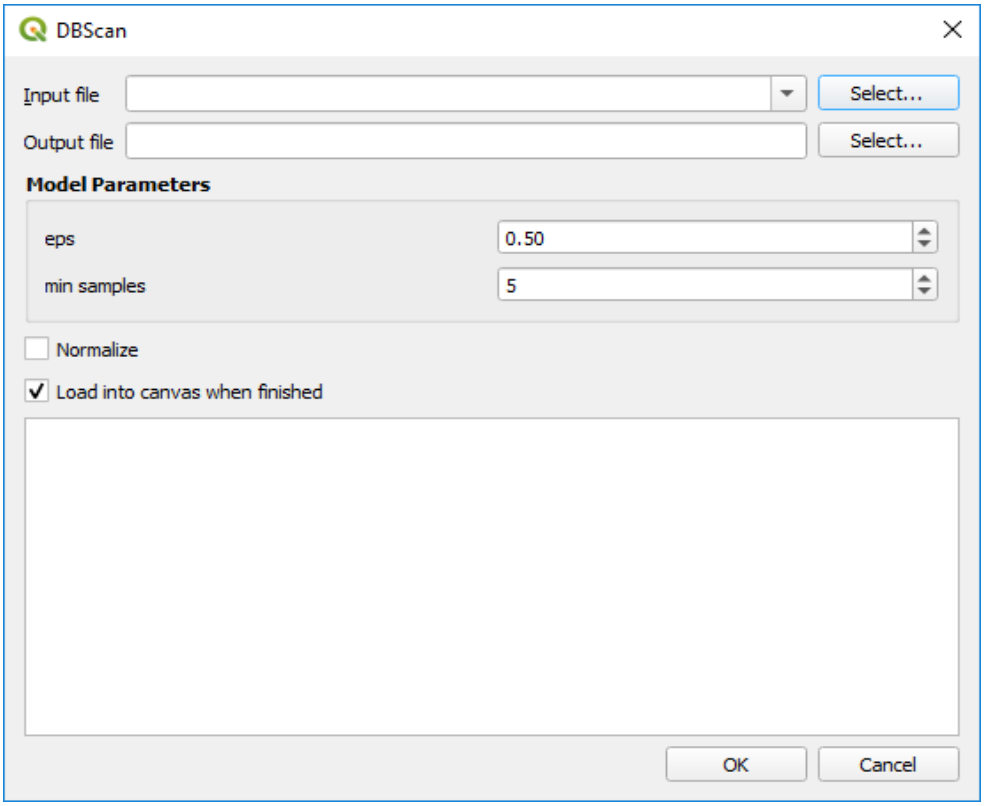


Figure User interface for DBSCAN Clustering

I/O PARAMETERS

Type	Data format	DATA type	value range	Remark
Input				
Input file	ENVI/Geotiff Raster			Input Raster File Reflectance/Radiance
eps			float, default=0.5	The maximum distance between two samples for one to be considered as in the neighborhood of the other

min samples			int, default=5	The number of samples (or total weight) in a neighborhood for a point to be considered as a core point.
Output				
Output file	ENVI			Output raster image

Input Arguments

eps : float, default=0.5 The maximum distance between two samples for one to be considered as in the neighborhood of the other. This is not a maximum bound on the distances of points within a cluster. This is the most important DBSCAN parameter to choose appropriately for your data set and distance function.

min_samples: int, default=5 The number of samples (or total weight) in a neighborhood for a point to be considered as a core point. This includes the point itself.

Overview

DBSCAN Clustering (where DBSCAN is short for Density-Based Spatial Clustering of Applications with Noise) involves finding high-density areas in the domain and expanding those areas of the feature space around them as clusters. It is similar to mean-shift, but with a couple of notable advantages.

1. DBSCAN begins with an arbitrary starting data point that has not been visited. The neighborhood of this point is extracted using a distance epsilon ε (All points which are within the ε distance are neighborhood points).
2. If there are a sufficient number of points (according to minPoints) within this neighborhood then the clustering process starts and the current data point becomes the first point in the new cluster. Otherwise, the point will be labeled as noise (later this noisy point might become the part of the cluster). In both cases that point is marked as “visited”.
3. For this first point in the new cluster, the points within its ε distance neighborhood also become part of the same cluster. This procedure of making all points in the ε neighborhood belong to the same cluster is then repeated for all of the new points that have been just added to the cluster group.

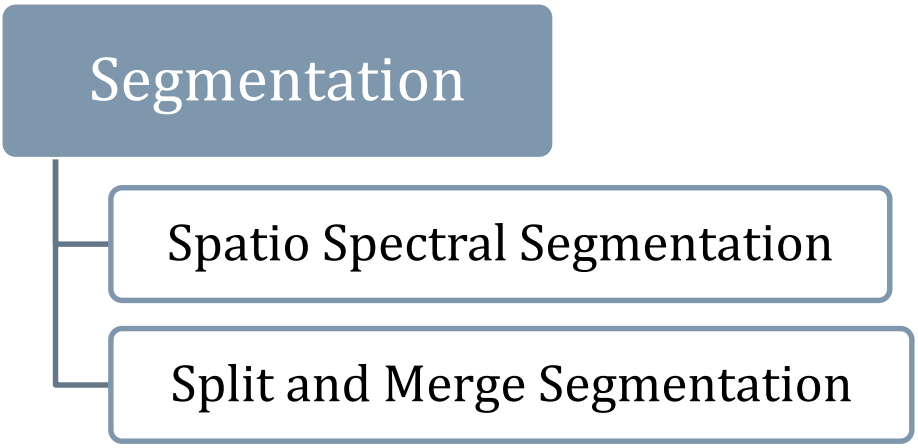
4. This process of steps 2 and 3 is repeated until all points in the cluster are determined i.e all points within the ε neighborhood of the cluster have been visited and labeled.
5. Once we're done with the current cluster, a new unvisited point is retrieved and processed, leading to the discovery of a further cluster or noise. This process repeats until all points are marked as visited. Since at the end of this all points have been visited, each point will have been marked as either belonging to a cluster or being noise.

DBSCAN poses some great advantages over other clustering algorithms. Firstly, it does not require a pre-set number of clusters at all. It also identifies outliers as noises, unlike mean-shift which simply throws them into a cluster even if the data point is very different. Additionally, it can find arbitrarily sized and arbitrarily shaped clusters quite well. The main drawback of DBSCAN is that it doesn't perform as well as others when the clusters are of varying density. This is because the setting of the distance threshold ε and minPoints for identifying the neighborhood points will vary from cluster to cluster when the density varies. This drawback also occurs with very high-dimensional data since again the distance threshold ε becomes challenging to estimate.

References

For detailed information kindly refer to scikit learn documentation on DBSCAN Clustering

6.3 SEGMENTATION



6.3.1 SPATIO-SPECTRAL SEGMENT TOOL

QGIS 3.XX → Hyperspectral→ Classification → Segmentation → Spatio-Spectral-Segment

User Interface

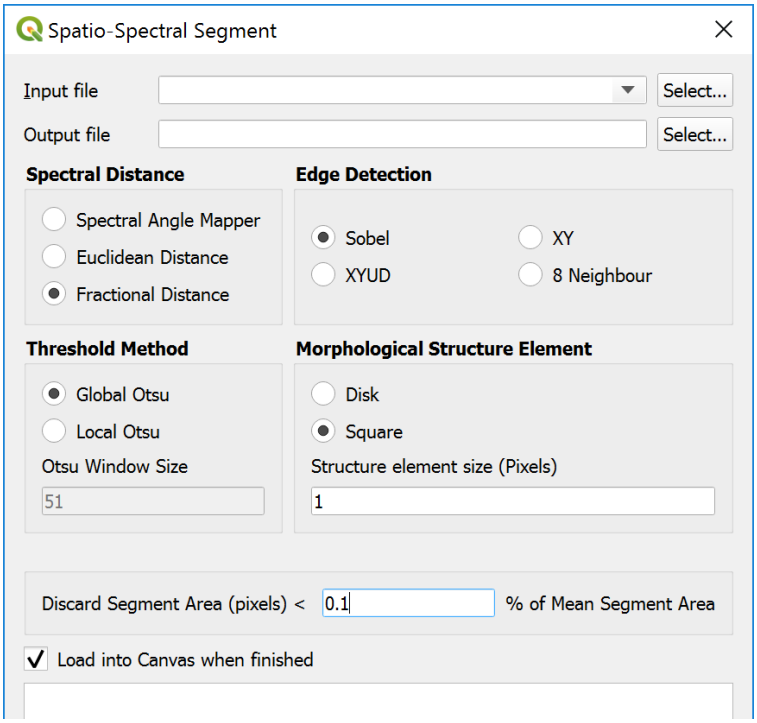


Figure. User-Interface for Spatio-Spectral Segment

6 CLASSIFICATION MODULE

I/O PARAMETERS

TYPE	DATA FORMAT	DATA TYPE	VALUE RANGE	REMARK
Input				
Input File	GeoTiff / ENVI			Bad band Removed Data, Radiance/Reflectance
Local Otsu Window Size		Integer	Odd-Number	Maximum value = Min (no. of rows, no. of columns)
Structure element Size		Integer		Default=1
Discard Segments		Integer	0-100%	
Output				
Output File	ENVI			Raster file with integer values for each segment

Input Arguments

Spectral Distance: Select the spectral distance which defines the similarity between two spectrum

Edge Detection: Select from the four different types of Edge detection methods which uses the spectral similarity within the 3x3 neighborhood

Threshold: Global otsu/Local otsu will be used for identifying the threshold for separating edge pixels from homogeneous area pixels.

Otsu Window Size: Defines the Block size of the dynamic thresholding algorithm

Structure element: Decides the morphological structuring element for removing the "salt and pepper" type noise in the outcome of the edge detector.

Discard Segment area: Segments with the area less than the "input" percentage of the mean segment area will be discarded from the final result.

6 CLASSIFICATION MODULE

Overview

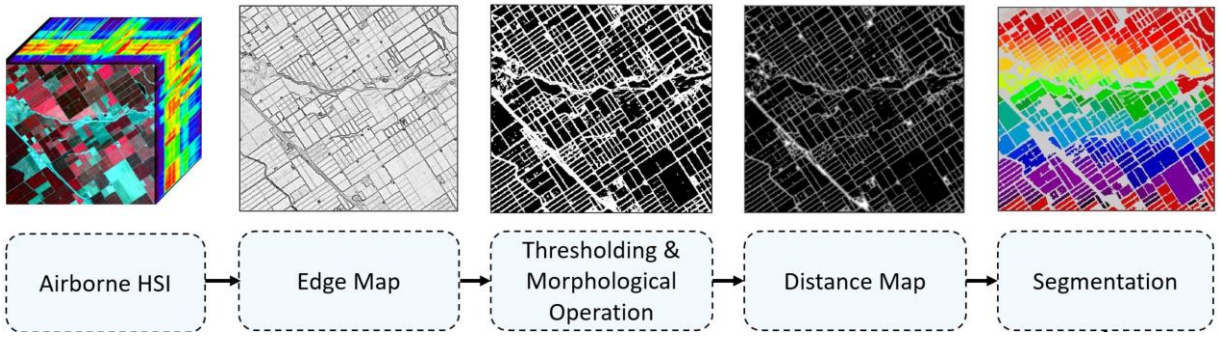
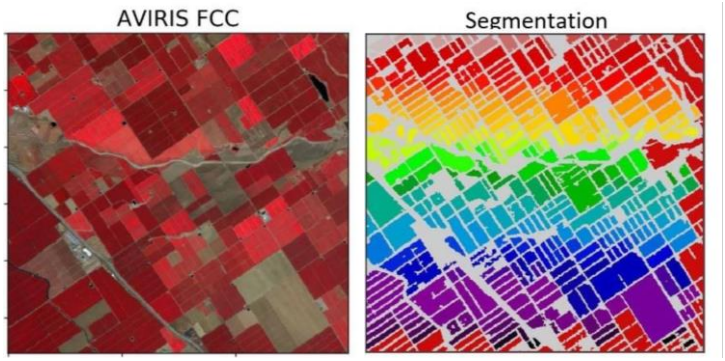


Figure: Data Flow Diagram

Spatio-Spectral segment is an n unsupervised method for extracting the spatial-spectral homogeneous areas in HSIs. Methodology is illustrated in the Figure above this paragraph. Edge map has been derived using Spectral Distance Method as mentioned in Bakker and Schmidt (Bakker and K. Schmidt, 2002). Otsu thresholding has been implemented to extract boundaries. Morphological opening and closing has been implemented sequentially for removing "salt and pepper" like noise in the edge map. Two pixels are connected when they are neighbors and have the same value. Based on this, connected regions were labeled based on their neighbors in 8-neighborhood pixels. These connected regions have been employed to find centroids for the segmentation algorithm. We consider watershed segmentation (WS) (Tarabalka et. al., 2010) method to segment the image. WS algorithm allocates pixels into marked basins. Here, marker is determined automatically using the local-maxima of the distance transform to the background for separating overlapping objects. Distance transform uses Euclidean distance to assign a value (distance) to the pixel which is the distance between the non-background pixels and the nearest background pixel. Segments which are small in size has been filter out using the minimum area threshold. Small segments (area less than specified threshold) can be removed from the final segmentation result using the area property of the segments.

6 CLASSIFICATION MODULE

Result



Reference

W. Bakker and K. Schmidt, “Hyperspectral edge filtering for measuring homogeneity of surface cover types,” ISPRS Journal of Photogrammetry and Remote Sensing, vol. 56, no. 4, pp. 246–256, 2002.

Y. Tarabalka, J. Chanussot, and J. A. Benediktsson, “Segmentation and classification of hyperspectral images using watershed transformation,” Pattern Recognition, vol. 43, no. 7, pp. 2367–2379, 2010.

6.3.2 SPLIT AND MERGE SEGMENTATION TOOL

QGIS 3.XX → Hyperspectral→ Classification → Segmentation → Split and Merge-Segmentation

User Interface

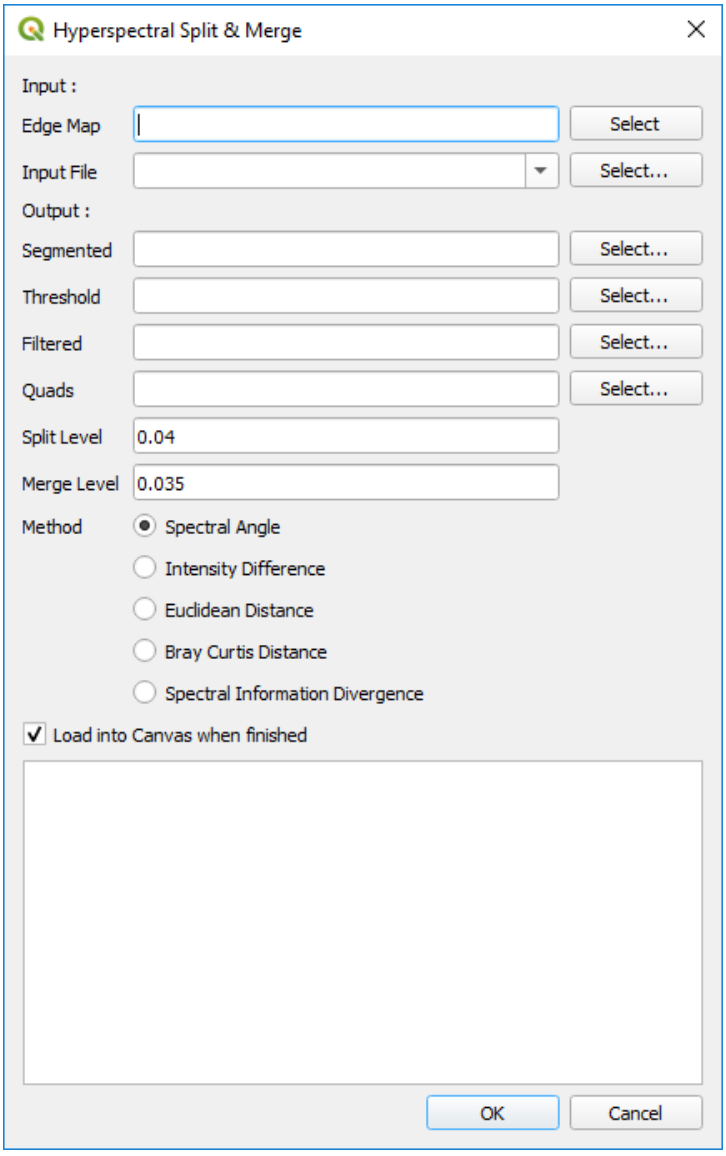


Figure. User-Interface for Split and Merge Segmentation

6 CLASSIFICATION MODULE

I/O PARAMETERS

Type	Data format	DATA type	value range	Remark
Input				
Input file	ENVI/Geotiff Raster			Input Raster File Reflectance/Radiance
Edge map	Raster			The spectral bandwidth
Split level			float, default=0.04	The splitting level threshold
Merge level threshold			float, default=0.035	The merge level threshold
Output				
Segmented file	ENVI			Output segmented image
Threshold file	ENVI			Output Threshold file
Filtered file	ENVI			Output filtered file
Quads file	ENVI			Output quad tree file

Input Arguments

Input file: Define input raster data in Geotiff or ENVI format. It can be either Reflectance or radiance data.

Edge Map: Define the edge map as input from the edge detection algorithm module under preprocessing.

Split Level Threshold Define the splitting level threshold for the splitting criterion

Merge Level Threshold Define the merging criteria.

Segmented file This file is the final result of the segmentation and contains the merged clusters.

Quad file This file contains the quads as they are generated by the split phase of the segmentation.

Threshold file The split level is used the threshold the edge map. This file contains the result. Black is considered 'homogeneous', white is considered 'heterogeneous'

Filtered file The script does a pepper & salt filtering on the edge image. This image contains the result of that filtering.

Overview

This module is based on the segmentation module given by W. Bakker. Before we can run the split & merge segmentation using quadtrees we have to create the (hyperspectral) edge image (edge map) first. The edge map can be generated using the Spectral Edge Filter or the Spectral Gradient Filter. The edge map must have one band only. An important choice for the generation of the edge map is the choice of the distance function. Usually this will be 'spectral angle', which is the default. Inspect the edge map with the Display program. Generate an x-profile. What is the level at which you call the original image homogeneous? This value will be your split level.

Inspect the edge map with the Display program. Generate an x-profile. What is the level at which you call the original image homogeneous? This value will be your split level.

The input file plus the filtered image are the input files of the next step, the image segmentation.

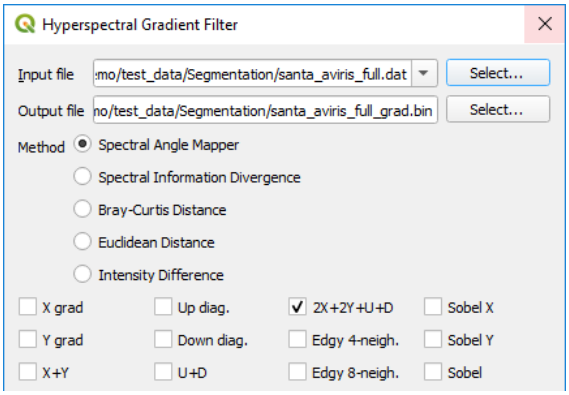
Steps:

- Open the Segmentation program.
- Select the edge map and input file.
- Select the output files. Some of the names may be automatically generated. The names can be changed.
- Set the split level and the merge level. Some values may be suggested by the program. These values are based on heuristics and may or may not give a good result. If you select a different distance function for the merge phase than was used for the generation of the edge map then the best merge level may actually be quite different from the split level.
- select the distance function that will be used for the merge level. Usually this will be the same as the one used for the generation of the edge map. The default is 'spectral angle'
- Press 'Ok.

6 CLASSIFICATION MODULE

The program will take a few minutes to run. A number of files are created by the process. These files can be viewed in the Display to check the quality of the segmentation.

Result



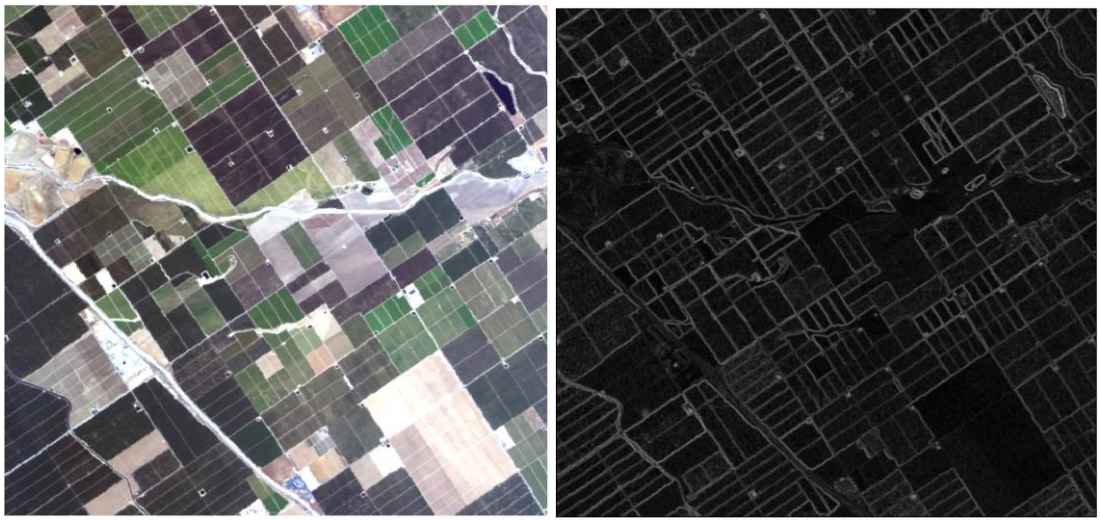


Fig (top) Edge Map Generation using Gradient 2x+2y+U+D filter (buttom 1st) Image (button 2nd) Edge Map

6 CLASSIFICATION MODULE

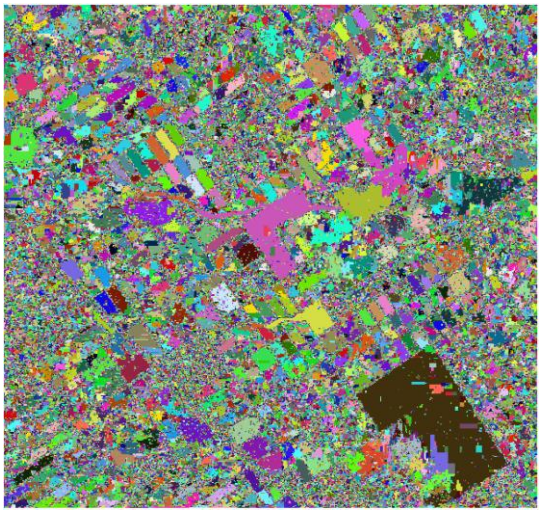


Fig Output segmented Image

References

S.L. Horowitz and T. Pavlidis. Picture segmentation by a directed split and-merge procedure. In Proceedings of the 2nd International Conference on Pattern Recognition, pages 424–433, 1974

S.L. Horowitz and T. Pavlidis. Picture segmentation by a tree traversal algorithm. Journal of the Association for Computing Machinery, 23(2):368– 388, April 1976.

B.G.H. Gorte. Multi-spectral quadtree based image segmentation. International Archives of Photogrammetry and Remote Sensing, XXXI:251–256, 1996.

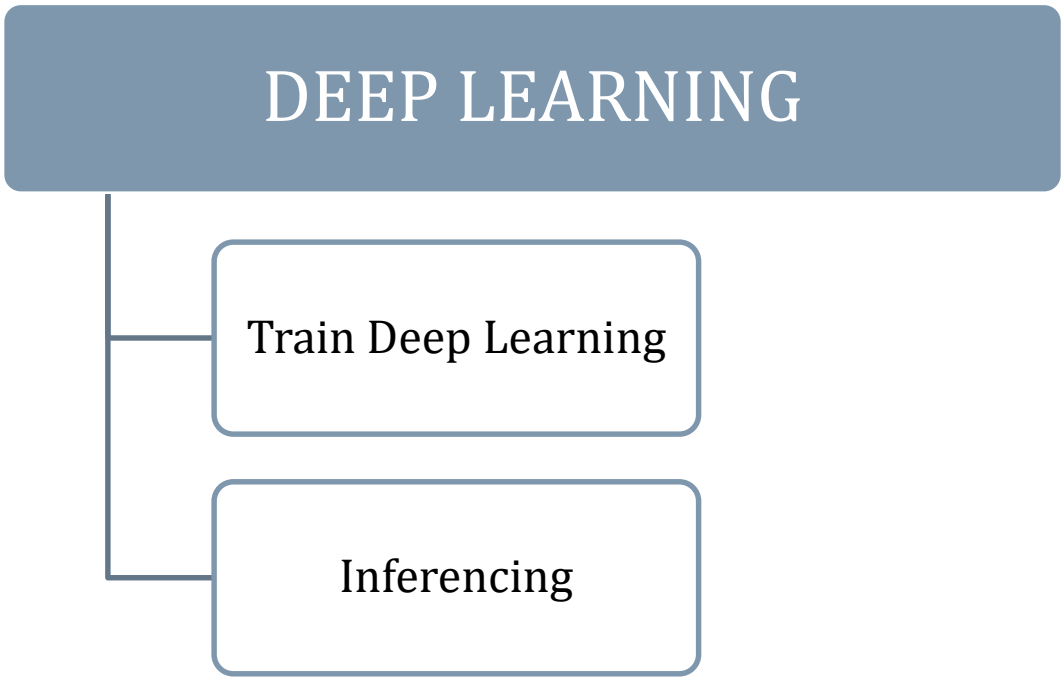
R.A. Finkel and J.L. Bentley. Quad trees: A data structure for retrieval on composite keys. Acta Informatica, 4(1):1–9, 1974.

Gueye Abdou Lat. Multi-spectral Quadtree based image segmentation. ITC, 1996. MSc thesis

W. Bakker and K. Schmidt, “Hyperspectral edge filtering for measuring homogeneity of surface cover types,” ISPRS Journal of Photogrammetry and Remote Sensing, vol. 56, no. 4, pp. 246–256, 2002.

7 DEEP LEARNING MODULE

7 DEEP LEARNING MODULE

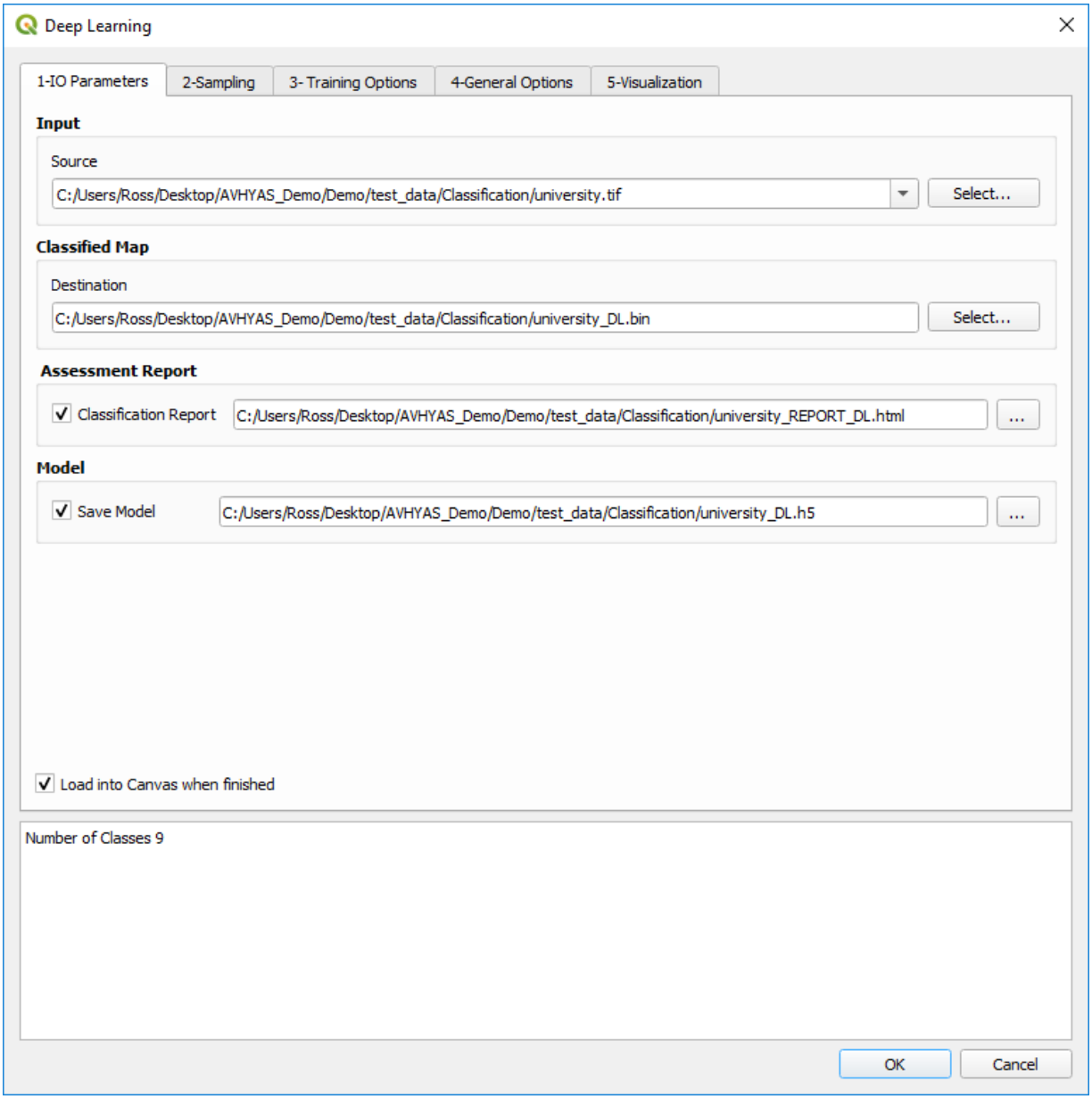


7 DEEP LEARNING MODULE

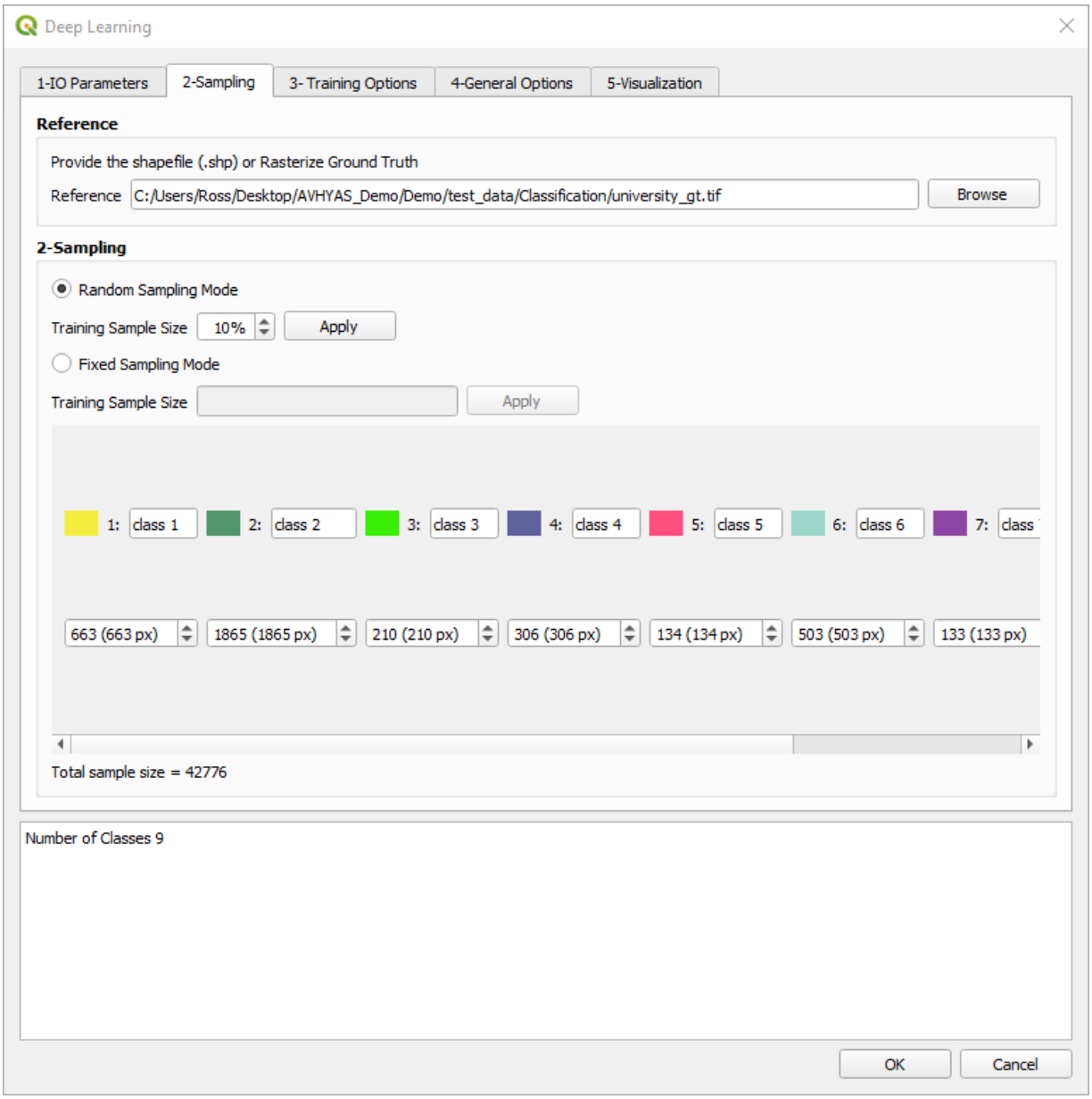
7.1 DEEP LEARNING

QGIS 3.XX → Hyperspectral→ Deep Learning → Deep Learning Workflow

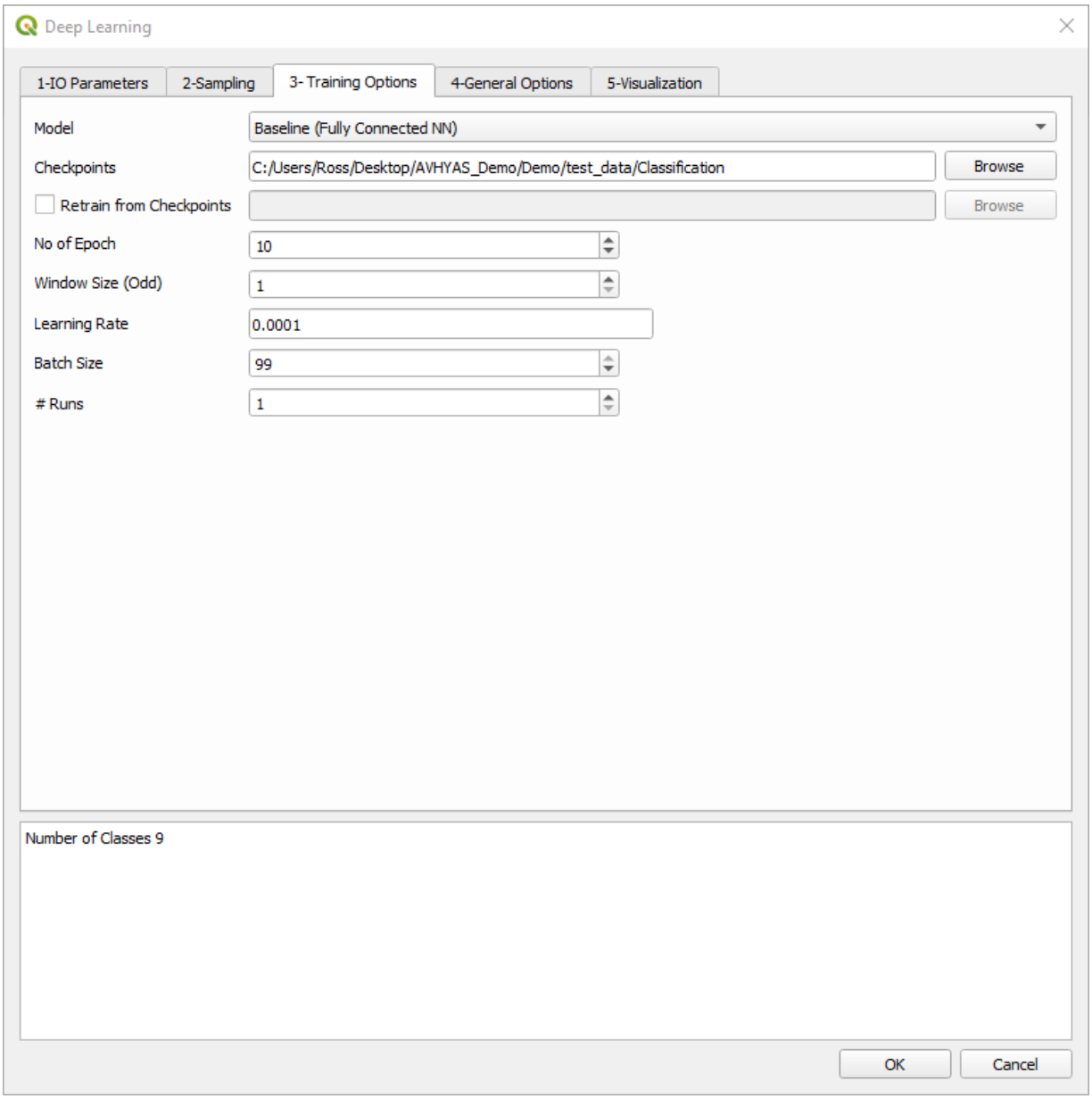
User Interface



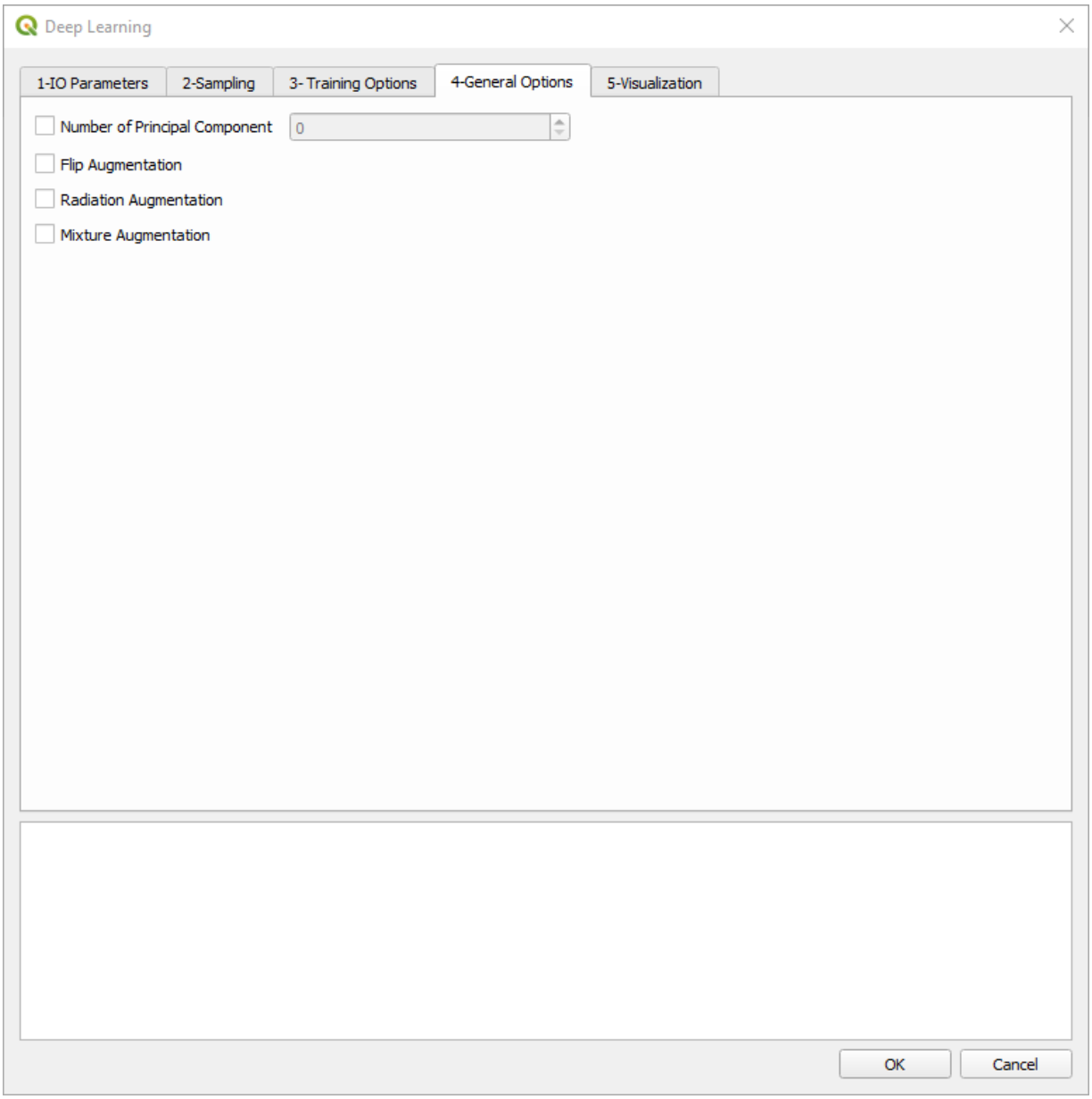
7 DEEP LEARNING MODULE



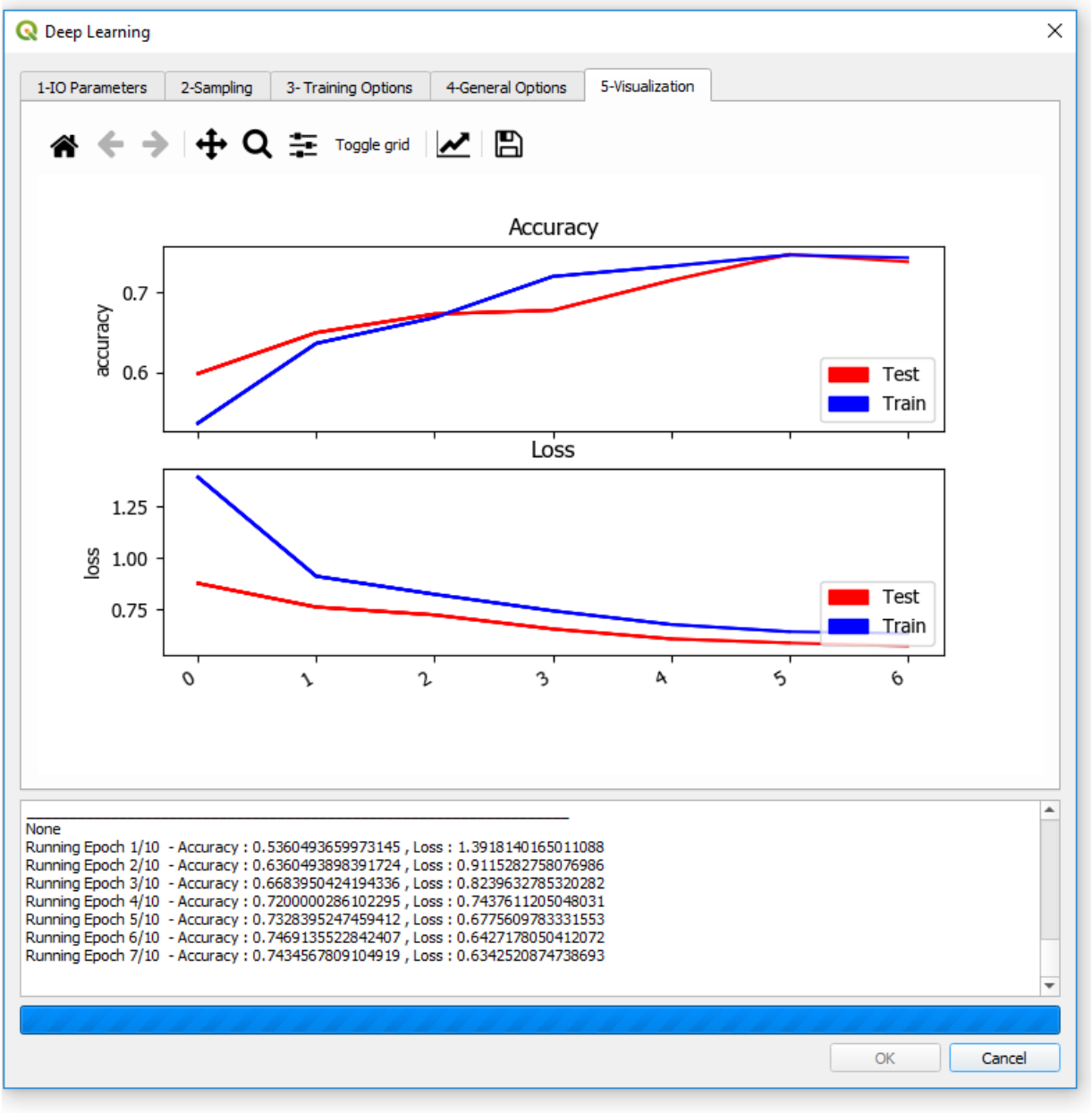
7 DEEP LEARNING MODULE



7 DEEP LEARNING MODULE



7 DEEP LEARNING MODULE



I/O PARAMETERS

Type	Data format	DATA type	value range	Remark
Input				
Input File	GeoTiff / ENVI			Raster data to be used for classification

				Radiance/Reflectance
Ground Truth	Shapefile/Raster			Ground Truth for classification
training sample size				Percentage of sample to be used for classification with optional for stratification
Model				Select any model from combobox.
Checkpoints				Provide the folder location in which check points will be saved.
Epoch				Number of times the entire sample will be seen by the model.
Window Size				The data window size
Learning Rate				The hyper parameter of the optimization method for which the next step of the gradient will be calculated
No of Runs				Number of times the module will run before the final output
Output				
Output Classification	ENVI			Raster file with predicted values
Probability map	ENVI			Raster file with probability of each pixel wise for each band.
Assessment report	HTML			The prediction report and accuracy assessment
Save model	pickle			Save the model for future inferences

7 DEEP LEARNING MODULE

Input Arguments

IO Parameters:

Input file: Specify the raster file to use for classification

Prediction: Specify the destination file for the output classification

Probability map: Specify the destination file for the probability map

Assessment Report: Specify the destination file for saving the assessment report of the regression

Sampling:

Reference: Specify the reference ground truth data either shapefile or rasterize file.

Model Parameters:

Model: Some of the implemented models are added which are meant specifically for hyperspectral data classification based on literature survey.

Checkpoints: Specify a folder location for which the model will save its weight and biases if it Is found to improve as compared to its previous accuracies.

Epoch: Define the number of times the model will go through the entire dataset

Window Size: Define the spatial window size for which the model will calculate its parameters

Learning rate: Define the hyper parameter for gradient descent algorithms. It basically gives the step size in the direction of the gradient for which the optimization will proceed in the next step.

No Runs: This parameter is meant for running the model multiple number of times so as to have a stable accuracy value.

Overview

Hyperspectral image (HSI) classification techniques have evolved quite extensively over the years. With deep convolutional neural network in recent times making huge progressed, emerging now as the state-of-the-art technique for HSI classification. The effectiveness of deep CNN model lies in its unique capabilities to extract hierarchical features automatically without human intervention. HSI data which are generally represented by a 3D data cube, contains rich

spectral information. The contiguous spectral channels add better discriminating power to HIS complementing its high spatial correlation making it useful in a wide range of precise mapping applications. As a result of recent advancement in hyperspectral technology now, finer spatial and narrower spectral resolution are made possible. With this aforementioned advances, the performance of the traditional vector based classifiers like SVM, random Forest, gaussian Mixture Model etc. are found to be no longer efficient due to the so called “curse of dimensionality”. With the number of spectral dimension increasing, the gap between dimensionality and the availability of ground truth samples became larger and larger. The other aspect which traditional classifiers often fails to capture is the spatial relatability of HSI data across all the bands.The issue of dimensionality and spectral redundancy often are resolved by using some pre-processing methods like principal component analysis (PCA), independent component analysis (ICA), and linear discriminant analysis (LDA). However, majority of these methods rely on the linear transformation of data, generally not applicable in all the cases because, light interaction with different targets on the ground are often complex and inherently non-linear by nature.

Since 2000, manifold learning and kernel based algorithm have become a major topic of research. The performance of the traditional classification algorithms can be improved considerably by handcrafting manually the spatial and spectral features either at the pre-processing or the post processing stage. The effectiveness of such methods relies majorly on the domain expertise and particularly very time consuming as well. Lately, a lot of research have been dedicated to investigate the full potential of deep CNN approaches for HSI classification. Chen et. al proposed the usage of stacked autoencoder (SAE) and deep belief network (DBN) for performing unsupervised classification of HSI. The deep learning models they proposed although outperforms other methods in terms of classification accuracy they, however, demands a lot of training parameters due to the fully connected architecture configuration. This in turn proves undesirable as there is limitation in the number of training samples. Furthermore, both the models suffer from spatial information loss, primarily caused by flattening of input data. Subsequently Zhao, Yue, Makantasis, and Liang et.al, came up with a 2D convolutional neural networks (CNN) for HSI classification, where spatial features were extracted by 2D-CNN using the first few Principle component(PC) of the original HSI data. The aforementioned approaches relies on using principal component analysis (PCA) for feature reduction prior to training, thus may fail to fully capture the joint spatial/spectral correlations information, which can be important for discrimination purposes. Recently, Chen et.al and Li et al proposed a 3D CNN based model to jointly learn the local changes in both the spatial and spectral dimension by using 3D kernels. Hamida et al proposed a simplified conventional 3D CNN based deep learning model specifically for HSI classification. The major setback often that 3D CNN suffered is the

lack of enough training samples as a result of which it prevents 3D CNN deep learning models to scale deeper. To tackle this issue, Lee et. al utilizes the concept of residual learning in their proposed network so as to scale the network much deeper even with limited training samples. The concept of using residual mapping brought in new concepts that focusses on learning of block of modules that are ultimate subgroup of layers. The optimization is applied on the difference between the desired output and the module input, through an identity mapping. With this configuration considerable increase in depth of the network can be achieved thereby improving the overall performance of the network. Zhong et.al very recently proposed an architecture that outperforms other networks in terms of performance through an architecture called spectral spatial residual network which utilizes separate spectral and spatial residual block to learn the discriminative features of HSI more distinguishably. Inspired by Zhong et.al model a modified version called faster dense spectral spatial residual network was proposed by Wang et.al where they have concentrates on concatenating and making the feature map as dense as possible with each passing layer. Other conventional computer vision inspired architecture like the dense network architecture by Gao et al. Google Inception and Residual Net (ResNet) also became prominent in between. The advantage of having a network with more depth and dense is that it effectively enhanced the feature extraction capability of the network by three fold. However DenseNet also has its own share of drawbacks and disadvantages, it takes longer to train the network, as well and the requirement of a larger training samples. In 2016, Iandola et.al , proposed a network known as squeezenet in which they stacked hierarchically the feature extracting module called ”fire modules”. The salient feature of the squeezenet is its ability to maintain high level of prediction accuracy even after strategically reducing the number of trainable parameters by replacing the larger filter size with the relatively cheaper1x1x1 filters as well as delaying the down sample step as late as possible.

Steps for Deep Learning workflow:

1. Provide the raster input file by clicking the select button
2. Provide the ground truth reference either in the form of a shapefile or a rasterize ground truth. When shapefile is provided make sure to have a categorical column. AVHYAS will auto populate categorical values in a combobox which user has to select.
3. If the input files are valid user has the option of selecting two sampling strategy, either by fixed sampling size or by random sampling. For fixed sampling kindly enter the number sample not more than the least sample number from all the classes. If random sampling is selected as the option user has to provide a percentage split of training and testing. User has to click on the Apply button after providing valid sampling strategy.

4. On clicking Apply, different classes along with the number of sample of each class will be automatically loaded along with their colors. User can change the color if they want to otherwise they can leave as the default color.
5. User has to then select any of the implemented Model provided in AVHYAS.
6. There is an optional parameter called checkpoints for which user are given the option to save intermediate result of each epoch of the deep learning models while training. The purpose of this is to reload the weights and biases so as not to start from scratch (random initialization).
7. On selection of any model provided, other options like epoch, learning rate, window size no of runs will be automatically loaded. User can change the values of these parameters in the text box. The value of the parameters provided are based on different heuristic experimental analysis done on different datasets.
8. Output path will be prepopulated when in the input file is selected by adding a postfix in the input name. User may change that if they want to.
9. Click ok to run and optionally user may click the load into canvas option if they want to load the classified output in the QGIS canvas.

Result

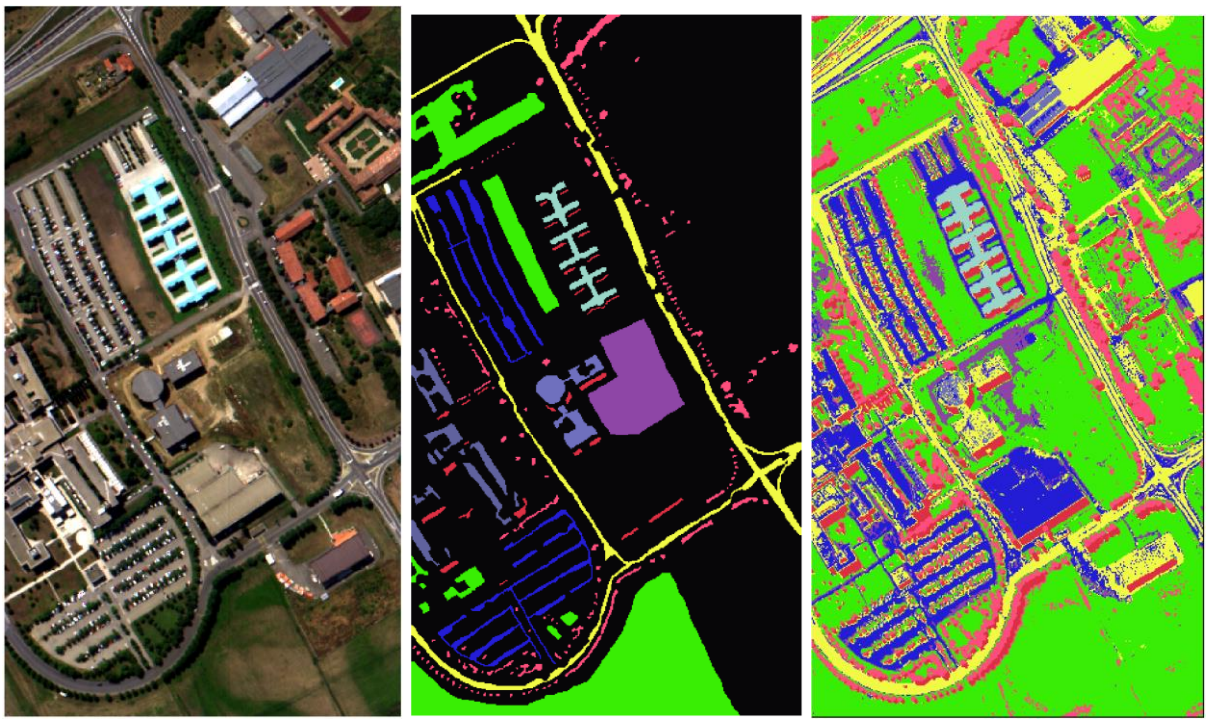


Figure (left) Pavia University Standard data, (middle) grown truth data and (right) Classified map

Classification Performance

Confusion Matrix

	Reference Class									
	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	Sum
(1) class 1	4866	43	0	0	17	49	253	725	2	5955
(2) class 2	9	15747	0	403	0	363	0	2	0	16524
(3) class 3	234	6	0	0	0	21	12	1615	1	1889
(4) class 4	0	529	0	2215	1	8	0	0	5	2758
(5) class 5	3	0	0	0	1184	0	18	5	1	1211
(6) class 6	23	3400	0	2	0	1005	2	94	0	4526
(7) class 7	714	4	0	0	0	1	407	71	0	1197
(8) class 8	51	24	0	0	0	116	9	3114	0	3314
(9) class 9	2	0	0	0	0	0	0	0	850	852
Sum	5902	19753	0	2620	1202	1563	701	5626	859	38499

Accuracies

Measure	Estimate [%]	95 % Confidence Interval [%]	
Overall Accuracy	76.33	nan	nan
Kappa Accuracy	67.66	67.11	68.21
Mean F1 Accuracy	66.04	-	-

Class-wise Accuracies

	User's Accuracy [%]			Producer's Accuracy [%]			F1 Accuracy		
Map class	Estimate	95 % Interval		Estimate	95% Interval		Estimate	95% Interval	
(1) class 1	81.71	81.33	82.09	82.45	nan	nan	82.08	82.08	82.08
(2) class 2	95.3	94.9	95.7	79.72	nan	nan	86.82	86.82	86.82
(3) class 3	0.0	nan	nan	0.0	nan	nan	0.0	0.0	0.0
(4) class 4	80.31	79.95	80.67	84.54	nan	nan	82.37	82.37	82.37
(5) class 5	97.77	97.65	97.89	98.5	nan	nan	98.14	98.14	98.14
(6) class 6	22.21	21.73	22.68	64.3	nan	nan	33.01	33.01	33.01
(7) class 7	34.0	33.51	34.49	58.06	nan	nan	42.89	42.89	42.89
(8) class 8	93.96	93.47	94.46	55.35	nan	nan	69.66	69.66	69.66
(9) class 9	99.77	99.66	99.87	98.95	nan	nan	99.36	99.36	99.36

Proportion Matrix

	Reference Class									
	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	Sum
(1) class 1	0.1264	0.0011	0.0	0.0	0.0004	0.0013	0.0066	0.0188	0.0001	0.1547
(2) class 2	0.0002	0.409	0.0	0.0105	0.0	0.0094	0.0	0.0001	0.0	0.4292
(3) class 3	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan
(4) class 4	0.0	0.0137	0.0	0.0575	0.0	0.0002	0.0	0.0	0.0001	0.0716
(5) class 5	0.0001	0.0	0.0	0.0	0.0308	0.0	0.0005	0.0001	0.0	0.0315
(6) class 6	0.0006	0.0883	0.0	0.0001	0.0	0.0261	0.0001	0.0024	0.0	0.1176
(7) class 7	0.0185	0.0001	0.0	0.0	0.0	0.0	0.0106	0.0018	0.0	0.0311
(8) class 8	0.0013	0.0006	0.0	0.0	0.0	0.003	0.0002	0.0809	0.0	0.0861
(9) class 9	0.0001	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0221	0.0221
Sum	nan	nan	nan	nan	nan	nan	nan	nan	nan	1.0

Figure Accuracy Report for 10 epoch

Reference

J. A. Benediktsson and P. Ghamisi, SpectralSpatial Classification of Hyperspectral Remote Sensing Images. Boston, MA, USA: Artech House, 2015.

G. Hughes, On the mean accuracy of statistical pattern recognizers, IEEE Trans. Inf. Theory, vol. IT-14, no. 1, pp. 5563, Jan. 1968.

J. B. Dias et al., Hyperspectral remote sensing data analysis and future challenges, IEEE Geosci. Remote Sens. Mag., vol. 1, no. 2, pp. 636, Feb. 2013.

G. Licciardi, P. R. Marpu, J. Chanussot, and J. A. Benediktsson, Linear versus nonlinear PCA for the classification of hyperspectral data based on the extended morphological profiles, IEEE Geosci. Remote Sens. Lett., vol. 9, no. 3, pp. 447451, May 2011.

A. Villa, J. A. Benediktsson, J. Chanussot, and C. Jutten, Hyperspectral image classification with independent component discriminant analysis, IEEE Trans. Geosci. Remote Sens., vol. 49, no. 12, pp. 48654876, Dec. 2011.

D. Lunga, S. Prasad, M. M. Crawford, and O. Ersoy, Manifold-learningbased feature extraction for classification of hyperspectral data: A review of advances in manifold learning, IEEE Signal Process. Mag., vol. 31, no. 1, pp. 5566, Jan. 2014.

T. Han, and D. Goodenough, Investigation of nonlinearity in hyperspectral imagery using surrogate data methods, IEEE Trans. Geosci. Remote Sens., vol. 46, no. 10, pp. 28402847, Oct. 2008.

Y. Tarabalka, M. Fauvel, J. Chanussot, and J. A. Benediktsson, SVM- and MRF-based method for accurate classification of hyperspectral images, IEEE Geosci. Remote Sens. Lett., vol. 7, no. 4, pp. 736740, Oct. 2010.

M. Fauvel, J. A. Benediktsson, J. Chanussot, and J. Sveinsson, Spectral and spatial classification of hyperspectral data using SVMs and morphological profiles, IEEE Trans. Geosci. Remote Sens., vol. 46, no. 11, pp. 38043814, Nov. 2008.

Mercier,G.andM.Lennon.Onthecharacterizationofhyperspectraltexture. In: IEEE International Geoscience and Remote Sensing Symposium. Vol. 5. 2002, 25842586 vol.5. doi: 10.1109/IGARSS.2002.1026708.

Moser, G., S. B. Serpico, and J. A. Benediktsson. Land-Cover Mapping by Markov Modeling of Spatial-Contextual Information in Very-HighResolution Remote Sensing Images. In: Proceedings of the

IEEE 101.3 (Mar. 2013), pp. 631651. issn: 0018-9219. doi: 10.1109/JPROC.2012.2211551. Tarabalka, Yuliya et al. SVM-and MRF based method for accurate classification of hyperspectral images.In:IEEE Geoscience and Remote Sensing Letters 7.4 (2010), pp. 736740

Y. Chen, Z. Lin, X. Zhao, and G. Wang, Deep learning-based classification of hyperspectral data, IEEE J. Sel. Topics Appl. Earth Observ.Remote Sens., vol. 7, no. 6, pp. 20942107, Jun. 2014.

Y. Chen, X. Zhao, and X. Jia, Spectralspatial classification of hyperspectral data based on deep belief network, IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens., vol. 8, no. 6, pp. 112, Jun. 2015.

Zhao, W.; Du, S. Spectral-Spatial Feature Extraction for Hyperspectral Image Classification: A Dimension Reduction and Deep Learning Approach. IEEE Trans. Geosci. Remote Sens. 2016, 54, 45444554.

Yue, J.; Zhao, W.; Mao, S.; Liu, H. Spectral-spatial classification of hyperspectral images using deep convolutional neural networks. Remote Sens. Lett. 2015, 6, 468477.

Makantasis, K.; Karantzalos, K.; Doulamis, A.; Doulamis, N. Deep supervised learning for hyperspectral data classification through convolutional neural networks. In Proceedings of the IEEE International Geoscience and Remote Sensing Symposium, Milan, Italy, 2631 July 2015; pp. 49594962

Liang, H.; Li, Q. Hyperspectral imagery classification using sparse representations of convolutional neural network features. Remote Sens. 2016, 8.

Chen, Y.; Jiang, H.; Li, C.; Jia, X.; Ghamisi, P. Deep Feature Extraction and Classification of Hyperspectral Images Based on Convolutional Neural Networks.IEEE Trans. Geosci.Remote Sens. 2016, 54, 62326251.

Ying Li, Haokui Zhang and Qiang Shen, "SpectralSpatial Classification of Hyperspectral Imagery with 3D Convolutional Neural Network", Remote Sens. 2017, 9, 67; doi:10.3390/rs9010067

H. Lee and H. Kwon, "Going Deeper With Contextual CNN for HyperspectralImageClassification," inIEEE Transactionson ImageProcessing, vol. 26, no. 10, pp. 4843-4855, Oct. 2017. doi: 10.1109/TIP.2017.2725580

Zhong, Z.; Li, J.; Luo, Z.; Chapman, M. Spectral-Spatial Residual Network for Hyperspectral Image Classification: A 3-D Deep Learning Framework. IEEE Trans. Geosci. Remote Sens. 2017, 99, 112.

Wang, Wenju; Dou, Shuguang; Jiang, Zhongmin; Sun, Liujie. 2018. "A Fast Dense SpectralSpatial Convolution Network Framework for Hyperspectral Images Classification." Remote Sens. 10, no. 7: 1068.

Huang, G.; Liu, Z.; Weinberger, K.Q. Densely Connected Convolutional Networks. In Proceedings of the 2017 IEEE Conference on Pattern Recognition and Computer Vision (CVPR), College Park, MD, USA, 2526 July 2017; pp. 19.

Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going Deeper with Convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, 712 June 2015; pp. 19.

He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2730 June 2016; pp. 770778.

SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and¡ 0.5MB model size FN Iandola, S Han, MW Moskewicz, K Ashraf, WJ Dally, K Keutzer arXiv preprint arXiv:1602.07360 [28] A. Krizhevsky, I. Sutskever, and G. E. Hinton, ImageNet classification with deep convolutional neural networks, in Proc. Adv. Neural Inf. Process. Syst., 2012, pp. 11061114.

Y. LeCun, Y. Bengio, and G. E. Hinton, Deep learning, Nature, vol. 521, pp. 436444, May 2015.

V. Mnih et al., Human-level control through deep reinforcement learning, Nature, vol. 518, pp. 529533, Feb. 2015.

S.Ioffe and C.Szegedy, Batchnormalization:Acceleratingdeepnetwork training by reducing internal covariate shift, in Proc. 32nd Int. Conf. Mach. Learn., 2015, pp. 448456 [32] He, K.; Zhang, X.; Ren, S.; Sun, J. Delving Deep into Rectifiers: Surpassing Human-Level Performance on Imagenet Classification. In Proceedings of the 15th IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile,1118 December 2015; pp.10261034.

Tijmen, T.; Hinton, G. Lecture 6.5-Rmsprop: Divide the Gradient by a Running Average of Its Recent Magnitude. COURSERA Neural Netw. Mach. Learn. 2012, 4, 2631.

Kingma, D.P.; Ba, J.L. Adam: A Method for Stochastic Optimization. In Proceedings of the International Conference on Learning Representations 2015, San Diego, CA, USA, 79 May 2015; pp. 115.

Sutskever, I.; Martens, J.; Dahl, G.; Hinton, G. On the Importance of Initialization and Momentum in Deep Learning. In Proceedings of the 30thInternationalConferenceonMachineLearning,ICML2013,Atlanta, GA, USA, 1621 June 2013; pp. 11391147.

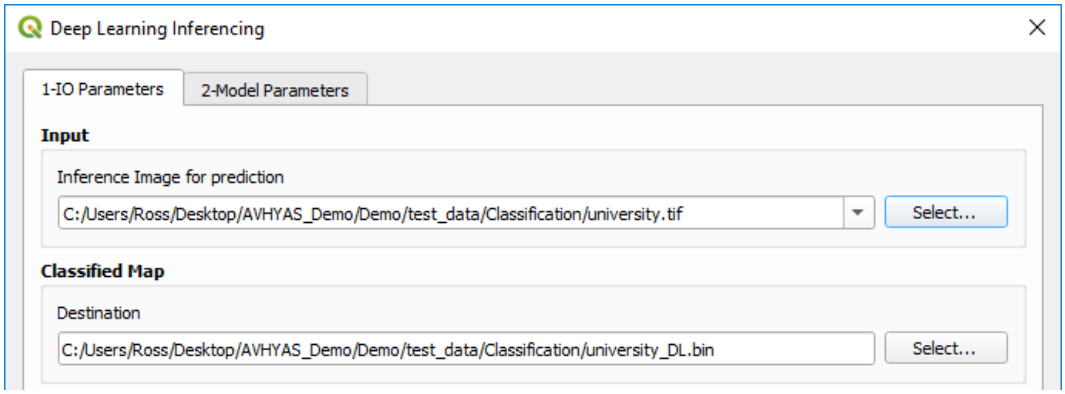
He, K.; Zhang, X.; Ren, S.; Sun, J. Delving Deep into Rectifiers: Surpassing Human-Level Performance on Imagenet Classification. In Proceedings of the 15th IEEE International Conference on Computer Vision,ICCV2015,Santiago,Chile,1118December2015;pp.10261034.

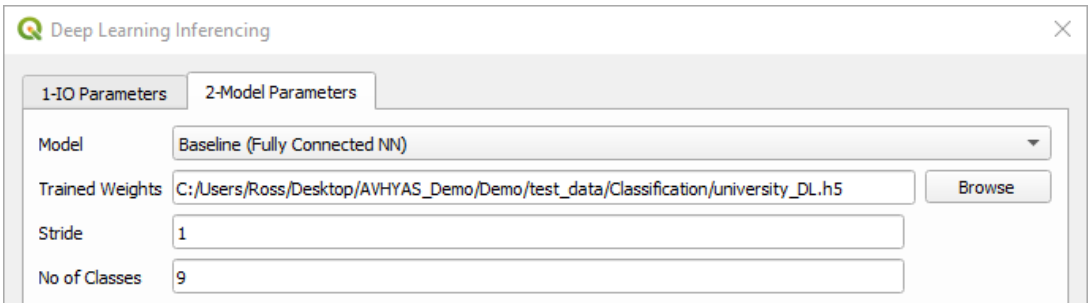
7 DEEP LEARNING MODULE

7.2 INFERENCE DEEP LEARNING

QGIS 3.XX → Hyperspectral→Deep Learning → Inference Deep Learning

User Interface





I/O PARAMETERS

Type	Data format	DATA type	value range	Remark
Input				
Input File	GeoTiff / ENVI			Raster data to be used for classification Radiance/Reflectance
Model				Select the model
Trained weights				Save weight and biases of the trained model
Stride				Number of step or moving window step size

7 DEEP LEARNING MODULE

No of Classes				Number of Classes in the output
Output				
Classified map	ENVI			Raster file with integer values for each class

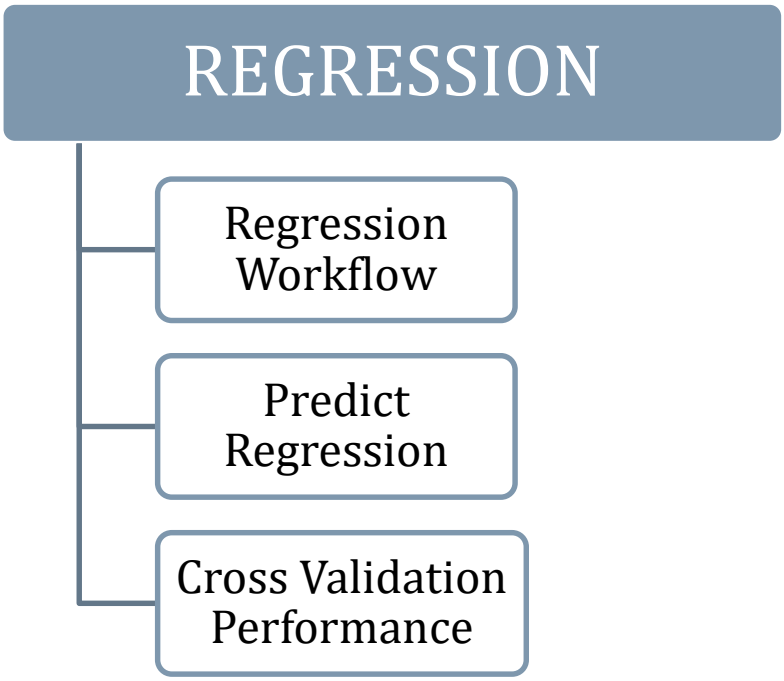
Overview

This module provides an interface for performing prediction based on a trained model of deep learning models. User has to provide a valid H5 file which contains the weight and biases of the model being trained. It is important before running this module that user run the deep learning module and click on the save model option to save the train model as a file. AVHYAS uses H5 file format to save the model.

Steps:

1. Select the input raster where user wants to run a prediction based on the trained model
2. Like all the other module the output will be pre populated however user can change it if they want.
3. Select the model which was used for training
4. Provide the trained weights, stride and number of classes
5. Click run and the process will start with the status of the processing shown in the log window.
6. If the load into canvas is selected, then the predicted result will be load in the QGIS canvas.

8 REGRESSION MODULE

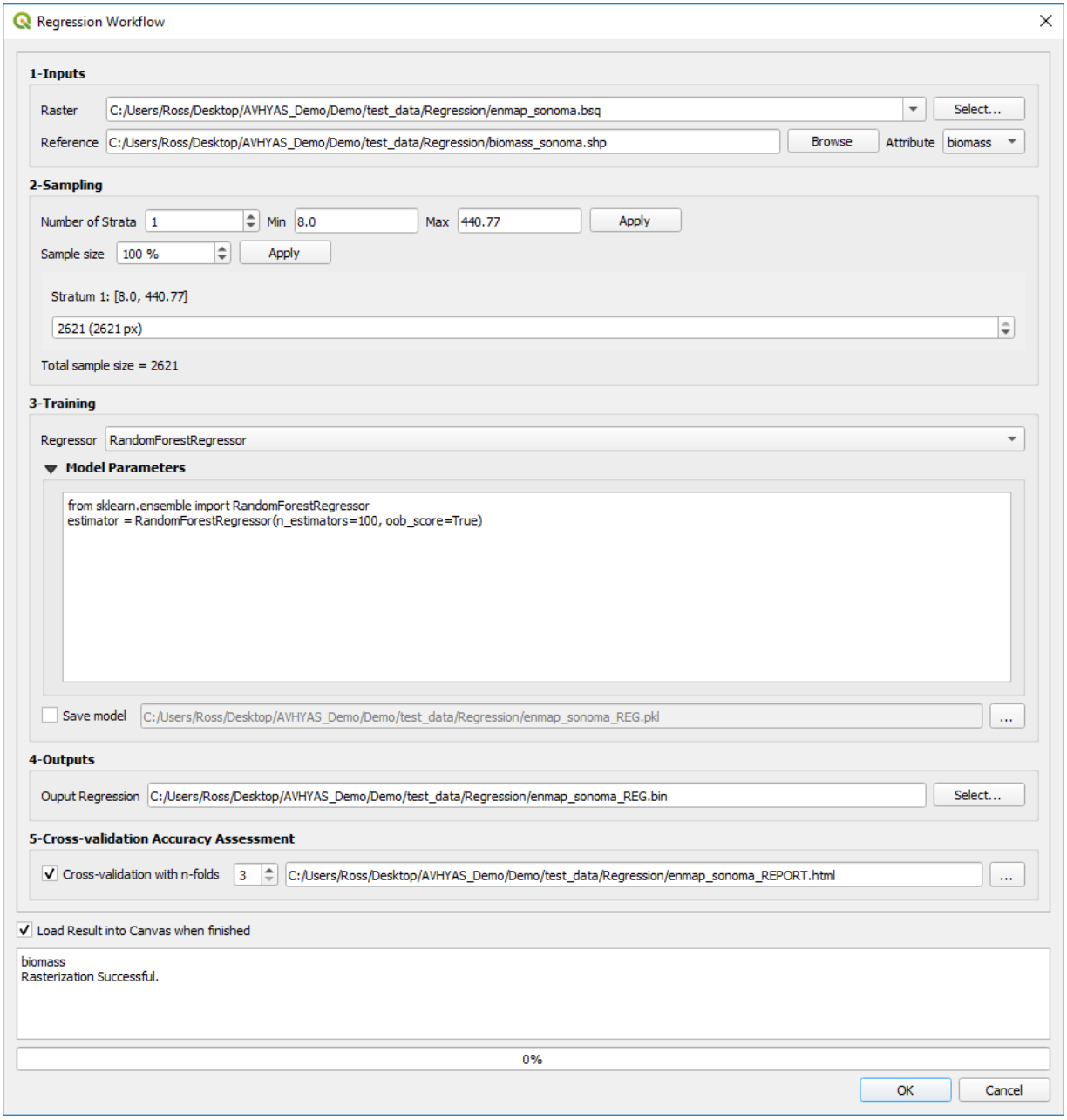


8 REGRESSION MODULE

8.1 REGRESSION WORKFLOW

QGIS 3.XX → Hyperspectral→ Regression → Regression Workflow

User Interface



8 REGRESSION MODULE

I/O PARAMETERS

Type	Data format	DATA type/Type	value range	Remark
Input				
Input File	GeoTiff / ENVI	Raster		Raster data to be used for classification Radiance/Reflectance
References	Shapefile/ GeoTiff/ENVI	Raster		Ground Truth for classification
Number of strata		Integer	Default=1	Number of strata to be used during the sampling process
training sample size		Integer	0-100	Percentage of sample to be used for classification with optional for stratification
Min & Max		Integer	Data dependent	Value range in which sample should be drawn
Scikit-learn python code		Python script		Scikit Learn Python code for different algorithms.
Output				
Output regression	ENVI	Raster		Raster file with predicted values
probability map	ENVI	Raster		Raster file with probability of each pixel wise for each band.
assessment report	HTML	Raster		The prediction report and accuracy assessment
save model	pickle	Raster		Save the model for future inferences

Input Arguments

IO Parameters:

Input file: Specify input raster band on which samples will be drawn for training a regressor.

Reference: Specify vector/rasterize dataset with reference information. Has to have a numeric column in the attribute table with a target variable of interest if vector dataset is provided.

Attribute: If vector dataset is specified then specify the attribute field in the reference vector layer which contains the regression target variable.

Prediction: Specify the output path of the raster you like to save to disk

Probability map: Select the destination file for the probability map

☐ **Cross Validation Accuracy Assessment:** Activate this setting to assess the accuracy of the regression by performing cross validation . Specify the desired number of folds (default :3). HTML report will be generated at the specified output path.

Sampling:

Number of strata: Specify the desired number of strata sampling. If stratified sampling is not required, then just enter the value 1

Min & Max: Define the value range in which samples should be drawn. The value will be auto populated based on histogram as per defined strata value

Sample Size: Specify the sample size per stratum, relative in percent

Training:

In the **Regressor** dropdown menu user can choose different regressors (eg. Random Forest, Support Vector Regression, Kernel Ridge Regression)

Model Parameters: Specify the parameters of the selected regressor. A bare code template has been provided for the user to change accordingly for different regression algorithm. Scikit learn python syntax has been used here. User can opt not to change the code and run straight away but however if they wish to change they can change and run accordingly.

8 REGRESSION MODULE

User can have a look at the scikit-learn documentation of the individual paramters eg for the RandomForestRegressor

☐ **Save Model:** Activate this option to save the model file (.pkl) to disk

Overview

This module provides a flexible way of running different regression algorithms. The idea of workflow has been borrowed and inspired from Enmapbox plugin with the backend module written and prepared in house. The whole purpose of a workflow is to provide an end to end framework for machine learning regression where user can feed in the data and the code and everything can be changed on the fly as per user's comfort and requirement. The bare template given in the model parameters however is sufficient for many regression problem but may not be optimum so user are requested to refer to scikit-learn python documentation of Scikit Learn on different regression algorithms available for more detail options.

Steps for Classification workflow:

1. Provide the raster input file by clicking the select button
2. Provide the ground truth reference either in the form of a shapefile or a rasterize ground truth. When shapefile is provided make sure to have a categorical column. AVHYAS will auto populate categorical values in a combobox which user has to select.
3. If the input files are valid user has the option to provide a stratified sampling based on the number of strata option and proceed forward by clicking apply. The input samples will then be drawn from samples from each of the strata.
4. Select the sampling ratio or percentage from the ground truth provided in step 2 and click on the button apply to run the sampling selection.
5. User are given option to select whether they want to save the probability map, assessment report and the trained model.
6. Output path will be prepopulated when in the input file is selected by adding a postfix in the input name. User may change that if they want to.
7. Click ok to run and optionally user may click the load into canvas option if they want to load the classified output in the QGIS canvas.

Result

8 REGRESSION MODULE

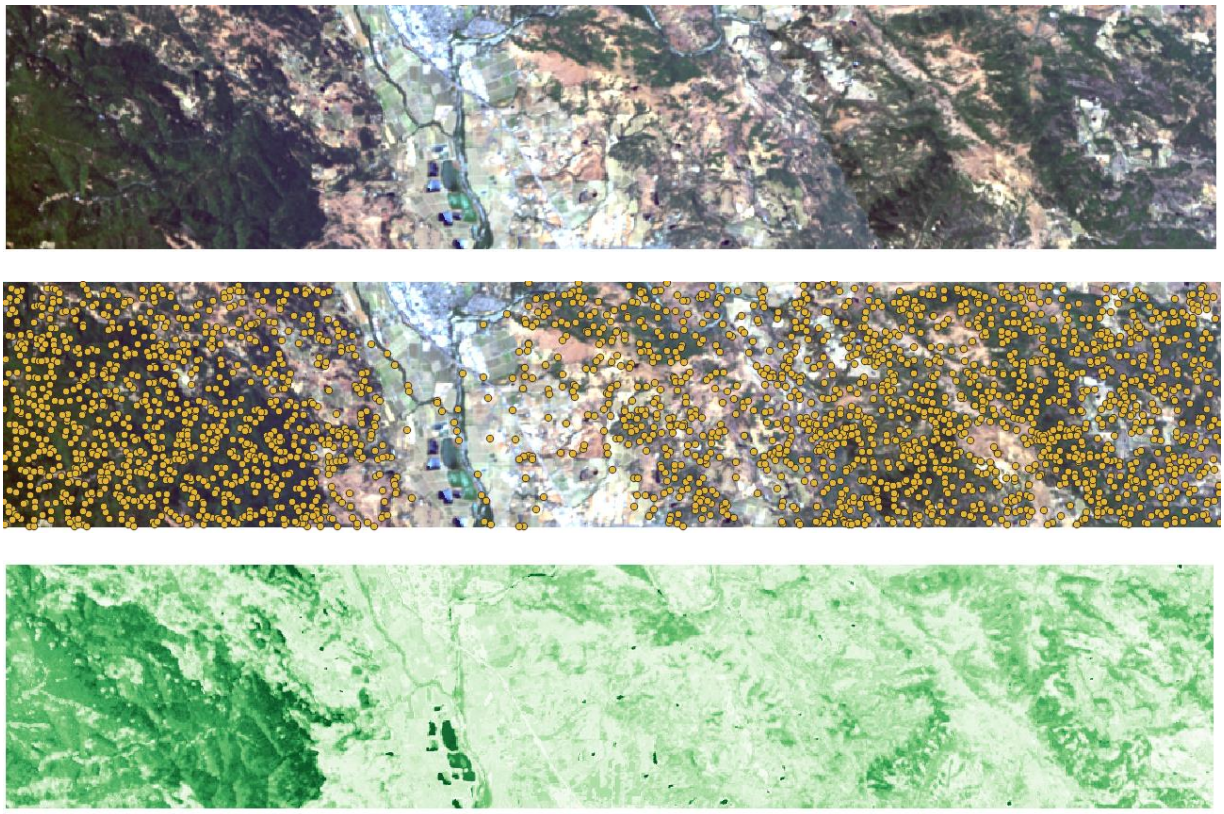


Figure (top row) Enmap ENSOMA test data, (middle row) overlayed biomass grown truth data and (buttom row) predicted biomass

Regression Performance

Outputs Overview

	Outputs
Reference	biomass
Prediction	biomass

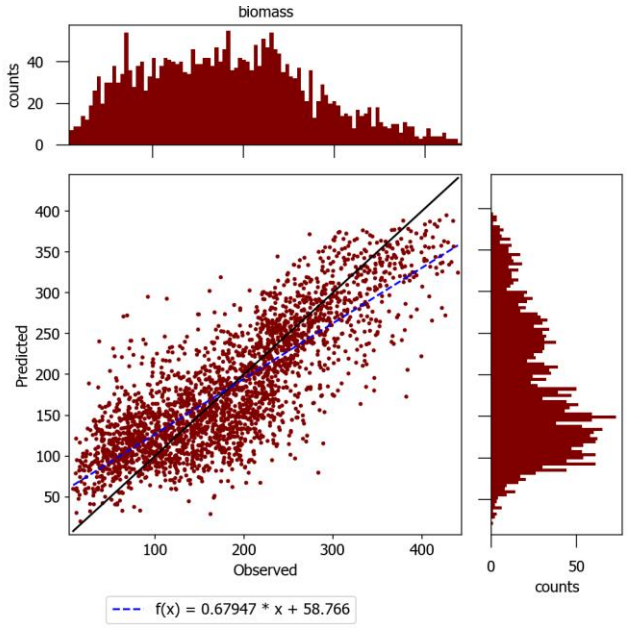
Metrics

Number of samples: 2621

	Outputs
	biomass
Mean absolute error (MAE)	42.7968
Root MSE (RMSE)	54.3418
Ratio of performance to deviation (RPD)	1.7114
Mean error (ME)	0.1228
Mean squared error (MSE)	2953.0292
Median absolute error (MedAE)	35.9763
Squared pearson correlation (r^2)	0.6592
Explained variance score	0.6586
Coefficient of determination (R^2)	0.6586

See Scikit-Learn documentation for details.

See Scipy documentation for details on pearson correlation.



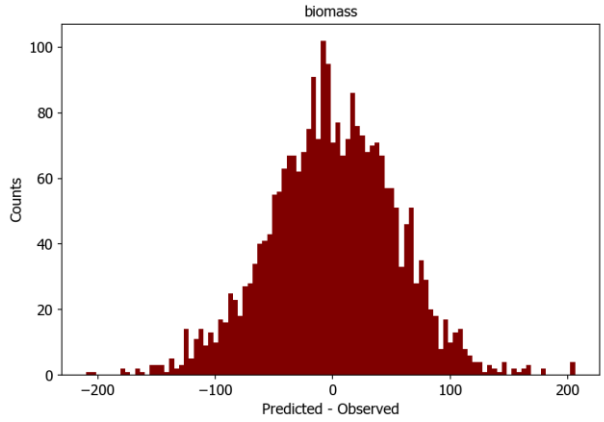


Figure Performance measure of regression

8 REGRESSION MODULE

8.2 PREDICT REGRESSION

QGIS 3.XX → Hyperspectral→Classification → predict classification

User Interface

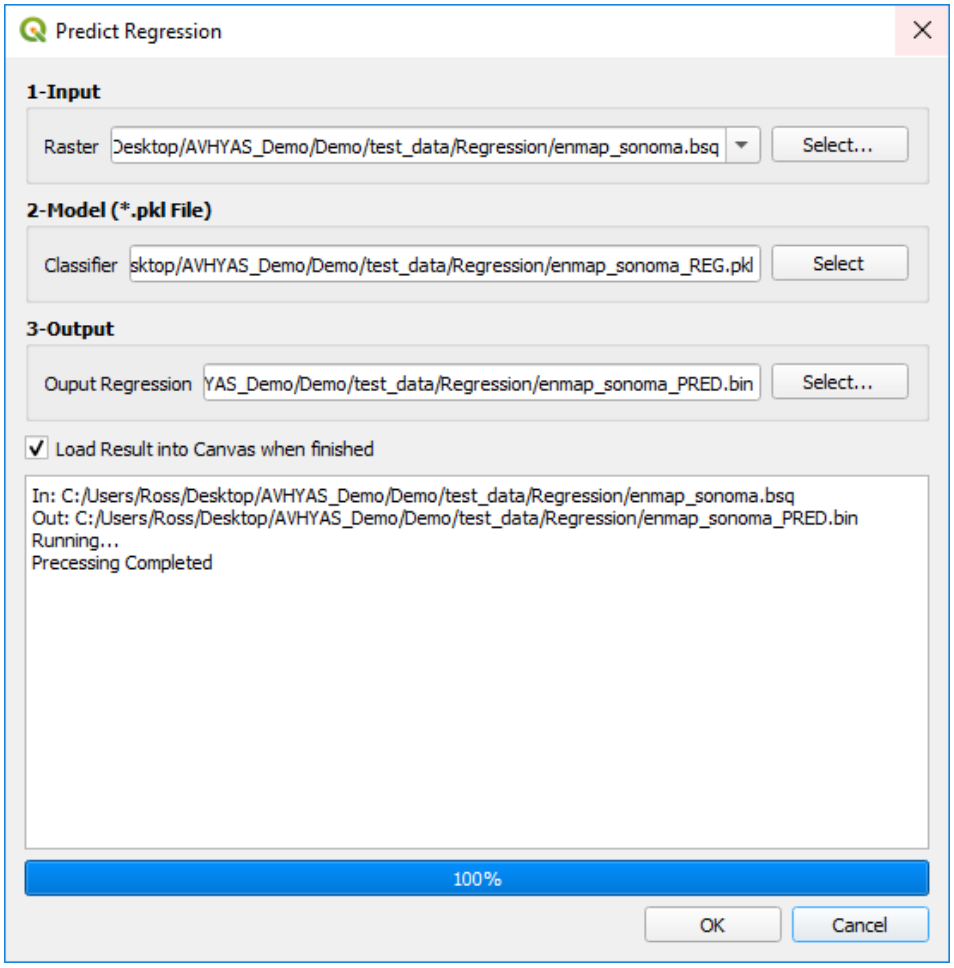


Figure Predict regression GUI with parameters

8 REGRESSION MODULE

I/O PARAMETERS

Type	Data format	DATA type/Type	value range	Remark
Input				
Input File	GeoTiff / ENVI	Raster		Raster data to be used for classification Radiance/Reflectance
Regression	Pickle			Save trained model
Output				
Classified map	ENVI	Raster		Raster file with integer values for each class

Input Arguments

Input file : Specify the input raster where user wants to run a prediction based on the trained model. The raster format supported so far is GeoTiff and ENVI.

Regressor : Specify the path to the trained classifier model (.pkl)

Classification: Output path where to write the regression image to.

Overview

This module provides an interface for performing prediction based on a trained model on different datasets. Like in the classification module here also it is important before running this module that user run the regression module and click on the save model option to save the train model in the file. AVHYAS uses pickle file format to save model in the file.

Steps:

7. Select the input raster where user wants to run a prediction based on the trained model
8. Provide a valid path to the trained model which is a pickle file save as with a pkl format extension
9. Like all the other module the output will be pre populated however user can change it if they want.
10. Click run and the process will start with the status of the processing shown in the log window.
11. If the load into canvas is selected, then the predicted result will be load in the QGIS canvas.

8.3 CROSS VALIDATION CLASSIFICATION

QGIS 3.XX → Hyperspectral→ Classification → Cross Validation Classification

User Interface

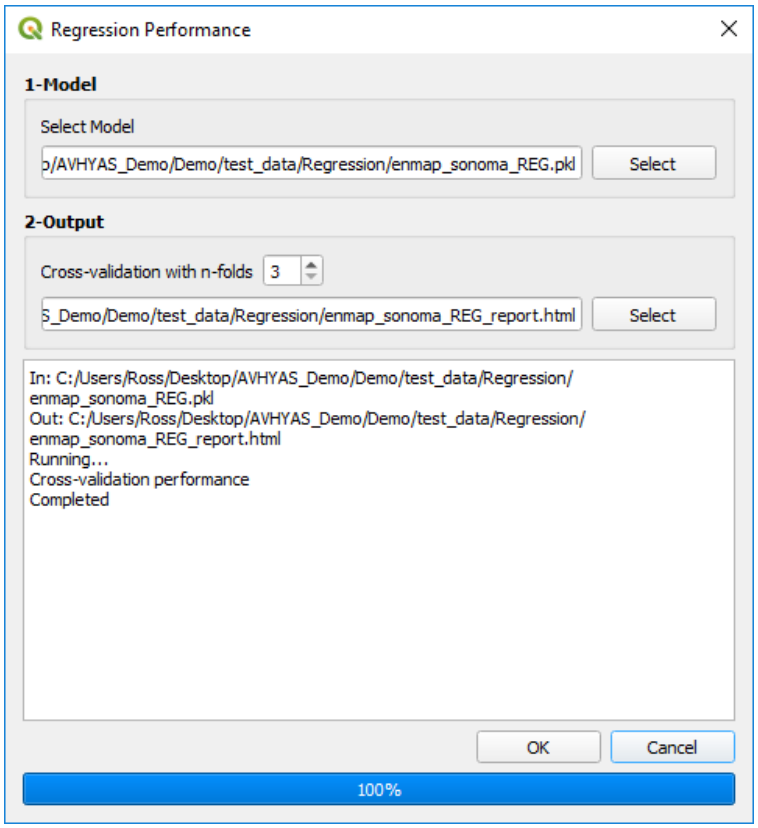


Figure Cross validation regression GUI with parameters

I/O PARAMETERS

Type	Data format	DATA type	value range	Remark
Input				
Model	Pickle			Save trained model
Output				
Assessment Report	HTML			HTML file containing different assessment report

8 REGRESSION MODULE

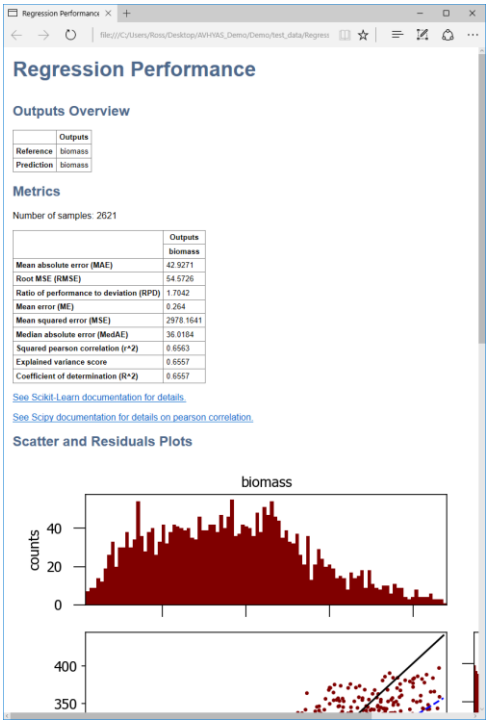
Overview

This module provides an interface for performing cross validation performance. The module requires the saved pickled trained model where along the model all the other necessary parameters are also saved for performing the cross validation performance.

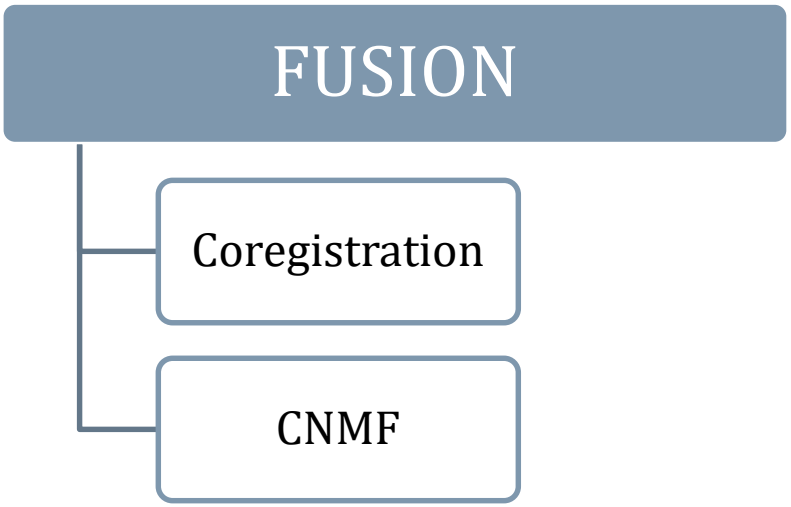
Steps:

1. Provide a valid trained model which is a pickle file save with a pkl format extension
2. Provide the number of cross validation performance split to perform on the data
3. Click run and the process will start with the status of the processing shown in the log window.
4. On finished an assessment report will be pop up in your default browser.

Results



9 **FUSION MODULE**



9 FUSION MODULE

9.1 COREGISTRATION

QGIS 3.XX → Hyperspectral→ Fusion → Coregistration

User Interface

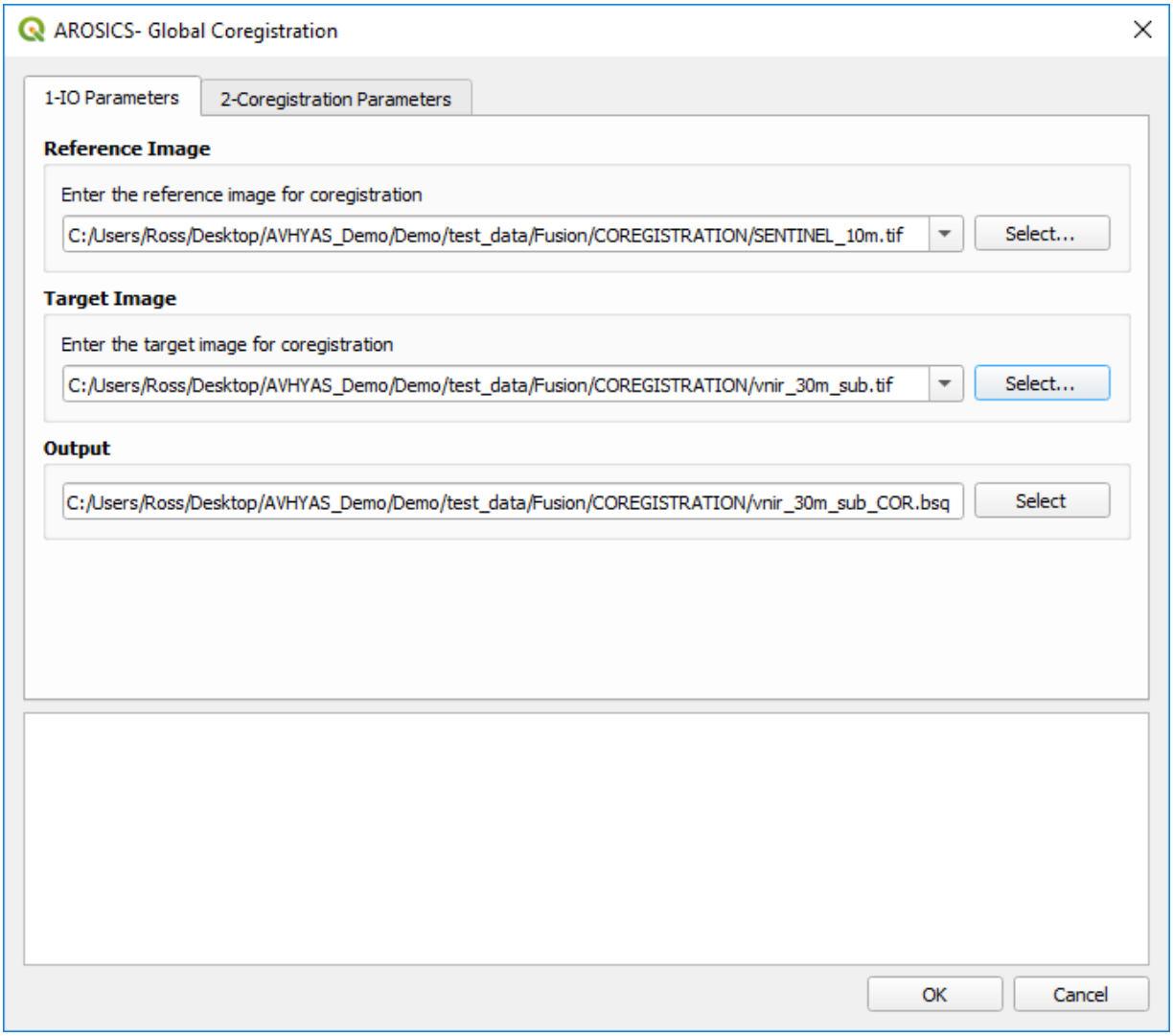


Figure Coregistration using AROSICS

9 FUSION MODULE

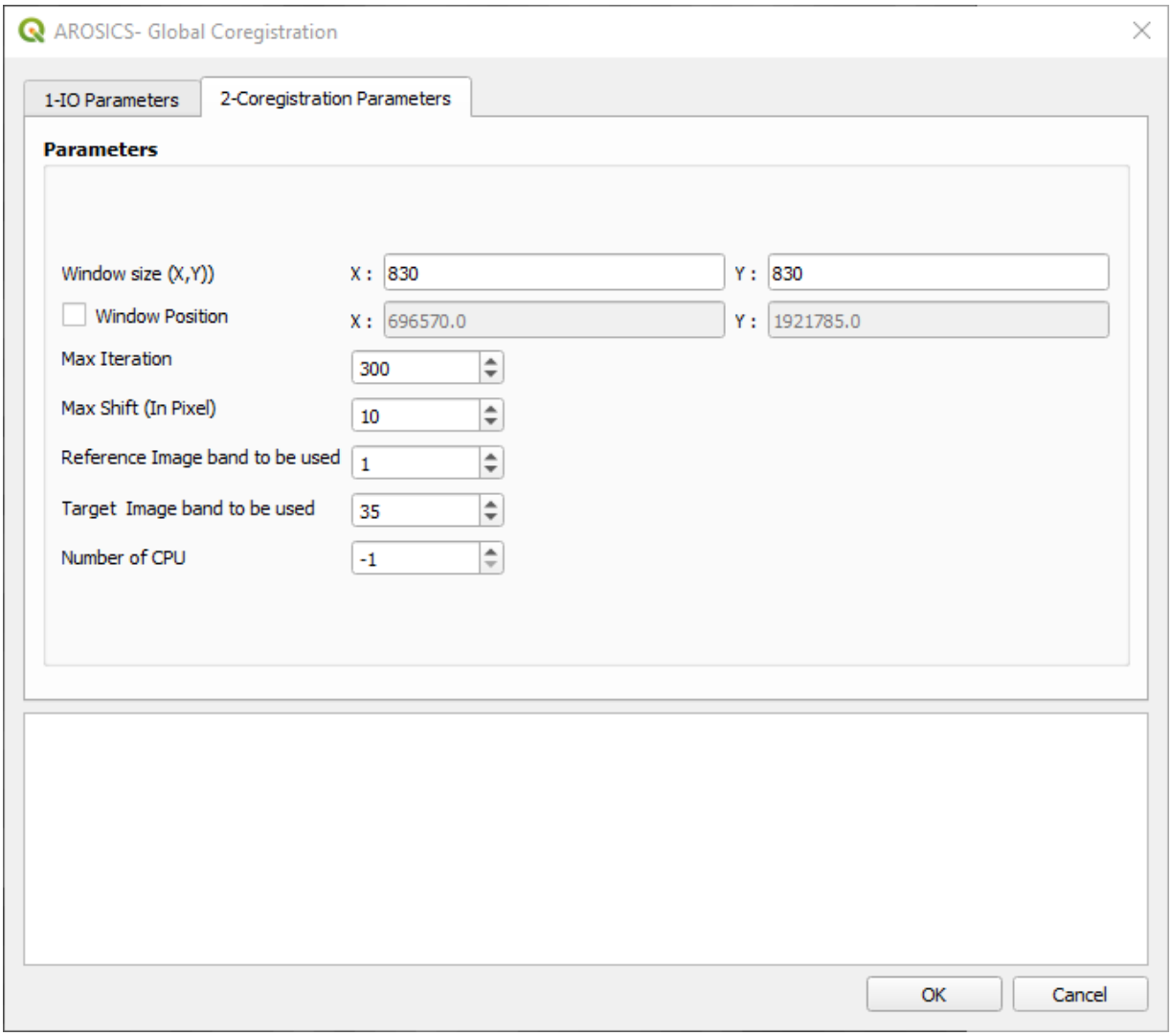


Figure Coregistration using AROSICS

I/O PARAMETERS

Type	Data format	DATA type	value range	Remark
Input				
Input File	GeoTiff / ENVI			Source raster data for coregistration Radiance/Reflectance
Reference Image	GeoTiff / ENVI			Reference data to be use as a reference when coregister

Window Size (X,Y)				Window size for calculating
Window Position		Integer, default=None		Initial Window position
Max Iteration		Integer, default=300		The maximum iteration
Max shift		Integer default=10		The maximum acceptable shift
Reference band to be used		Integer default=1		The reference band number for calculating the shift
Source band to be used		Integer default=1		The source band number for calculating the shift
CPU		Integer, default=-1		The number of jobs to run in parallel
Output				
Classified map	ENVI			Raster file with integer values for each class
probability map	ENVI			Raster file with probability of each class pixel wise for each band.
assessment report	HTML			The classification report and accuracy assessment
save model	pickle			Save the model for future inferences

Input Arguments

IO parameters

Source file: Select the input source raster file for coregistration

Reference file: Select the reference raster data for coregistation

Output file: Select the destination file for the coregistered result.

Window size: The window size to be considered for finding features to coregister two images.

Window position: The optional window position in the scene. By default, it is set to be at the canter

Max iteration: Maximum iteration needs to be provided.

Max Shift: Maximum allowable shift between two images in terms of pixel number beyond which the tool will throw an error.

Reference band to be used: The band number is a mandatory parameter. The band number provided here will be used in the coregistration process

Source band to be used: Likewise, the source band number also has to be provided.

CPU: Define the number of jobs to run in parallel

Overview

This tool is used to coregister two different datasets. The dataset can either be from the same sensor or from different sensors as well. However, it is important that the two datasets provided should have some common area and at least be geocoded. The backend module for coregistration relies on one of the most recent robust software for coregistation known as **AROSICS (An Automatic And Robust Open Source Image Coregistration Software for Multi Sensor Data)**. At the moment only global coregistration has been implemented and the purpose of coregistation is solely for the purpose of data fusion module. Most of the parameters defined by AROSICS are used as default.

In order to use this module the user has to provide two datasets (Most commonly the same area). One has to be provided as the source and the other as reference. On selection of source and reference most of the values will be suggested and automatically populated by the program. The output file name is automatically generated by appending a postfix COREG to the input file name. This output name can be overridden by changing the name in the text box or by picking a new output file. Other values like the Window size and window position are generated based on the dimension of the input data and scene center. The window size is kept to be as large as possible so as to allow more flexibility in terms of features available to search and match between the two images.

It is important to note that when hyperspectral data is used, most of the time the first band tends to have a very low SNR. So kindly make sure to select an appropriate source and

reference band number for the coregistration process to perform accurately. In most cases if the program returns “No Match found”, kindly change either the window size or the band number

Reference

Scheffler, D.; Hollstein, A.; Diedrich, H.; Segl, K.; Hostert, P. AROSICS: An Automated and Robust Open-Source Image Co-Registration Software for Multi-Sensor Satellite Data. Remote Sens. 2017, 9, 676

9.2 COUPLED NON NEGATIVE MATRIX FACTORIZATION

QGIS 3.XX → Hyperspectral→ Fusion → CNMF

User Interface

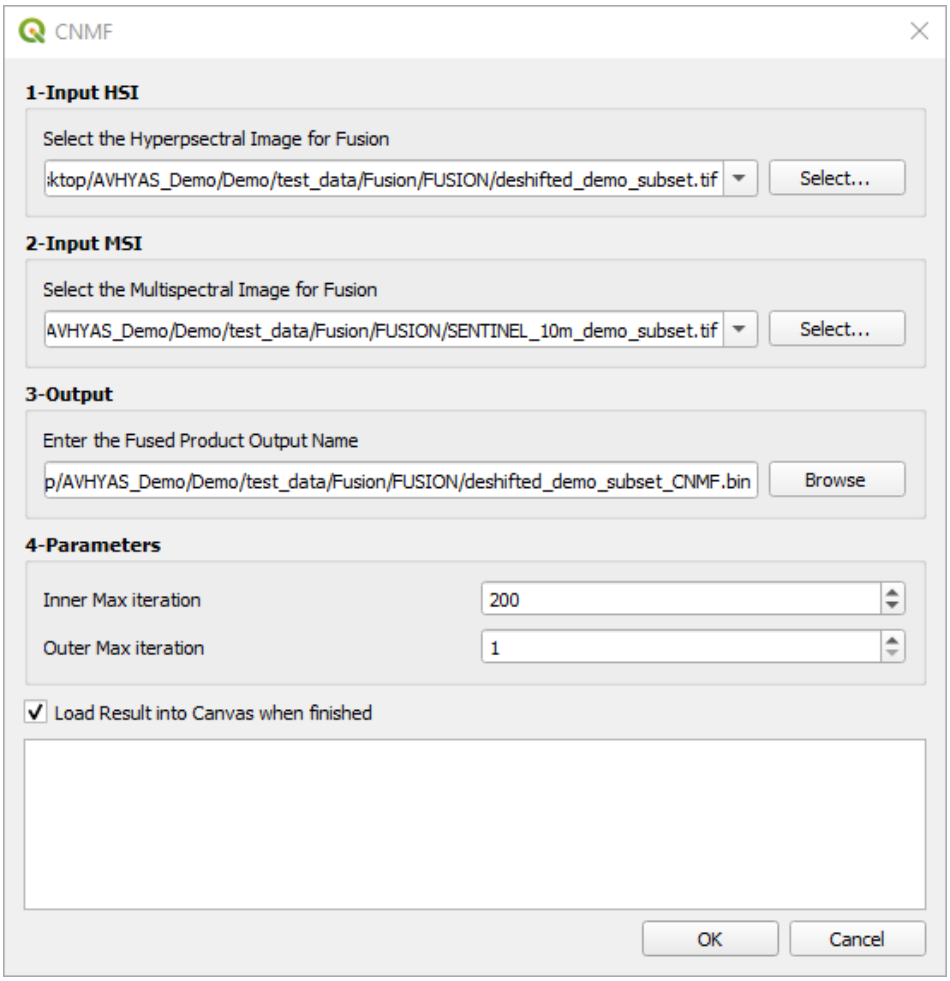


Figure User interface of CNMF

9 FUSION MODULE

I/O PARAMETERS

Type	Data format	DATA type	value range	Remark
Input				
Input HSI Image	GeoTiff / ENVI			Input Hyperspectral raster data Radiance/Reflectance
Input MSI Image	GeoTiff / ENVI			Input Multispectral raster data Radiance/Reflectance
Inner Max Iteration		Integer, default=200		Inner loop max iteration
Outer Max Iteration		Integer, default=1		Outer loop max iteration
Output				
Output file	ENVI			Fused raster file with hyperspectral data spectral resolution and with multispectral data spatial resolution

Input Arguments

IO parameters

Input MSI Image: The input multispectral data for fusion

Input HSI Image: Input hyperspectral data

Output file: Destination file of fused raster file with hyperspectral data spectral resolution and with multispectral data spatial resolution

Inner Max Iteration: The maximum iteration of inner loop during the optimization process.

Outer Max Iteration: The maximum outer iteration loop during the optimization process.

Overview

9 FUSION MODULE

Hyperspectral imaging is characterized by continuous narrow spectral channels ranging from 400nm to 2500 nm of the electromagnetic spectrum. However, this rich spectral information's comes with an inevitable trade-off, between spatial resolution and signal-to-noise ratio (SNR). Because of this reason, majority of the space borne imaging sensors are usually designed to provide data at moderate spatial resolution of 30m. In recent years due to limitation in terms of technology instead different fusion based approaches have been proposed to enhance the spatial resolution of space borne HSI datasets.

This tool implement one such technique where hyperspectral data spatial resolution is enhanced using an auxiliary multispectral data with temporal timestamp as close to that of the HSI dataset. There are different categories of data fusion namely 1) Component Substitution (CS) 2) Multiresolution Analysis (MRA) 3) Bayesian Method and 4) Unmixing. This module used the Unmixing approach proposed by Yokoya et al. He came out with an idea of coupled non negative matrix factorization(CNMF) for HS-MS fusion by using an alternating optimization scheme based on NMF incorporating both the SRF and point spread function (PSF).

In order to run the module properly kindly consider the following step.

Pre-Requisite:

- Make sure both Hyperspectral data and multispectral are coregistered. If not kindly run the coregistration module describe in the previous section before running this module otherwise the results will be not satisfactory.
- The date difference between the two datasets should not be too large otherwise there will be artefacts generated in the output which may lead to misleading results during analysis

In order to run the module once the pre-requisite is satisfied all the module requires is two datasets to be provided. One the multispectral data and the other hyperspectral data. The output file name will be automatically generated with a postfix CNMF added to the input file (Users can change the name if they want to). The number of iteration may be kept as default, however it may be noted that the default parameters may not give the best result as these parameters are set based on heuristic approach. If users want to change the number of iteration it is optional, the larger the value the longer it will take to complete the process.

9 FUSION MODULE

Result

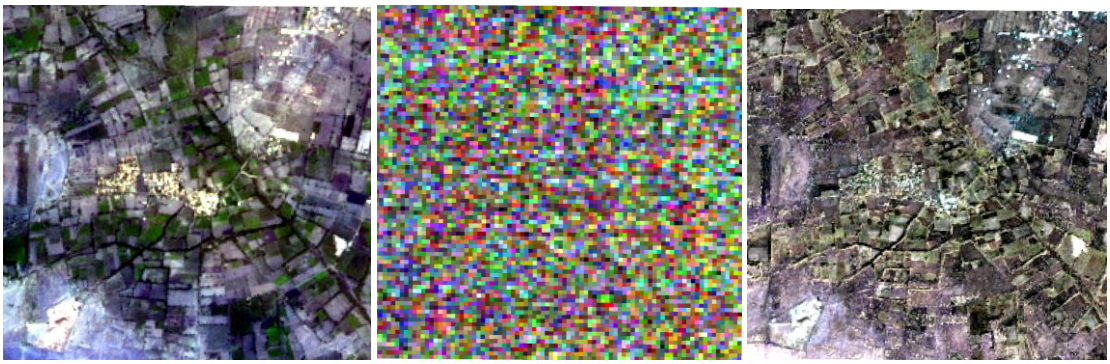


Figure left(Sentinel 10m data), Middle (HySIS 30m data), Right (Fused 10m Hyperspectral data)

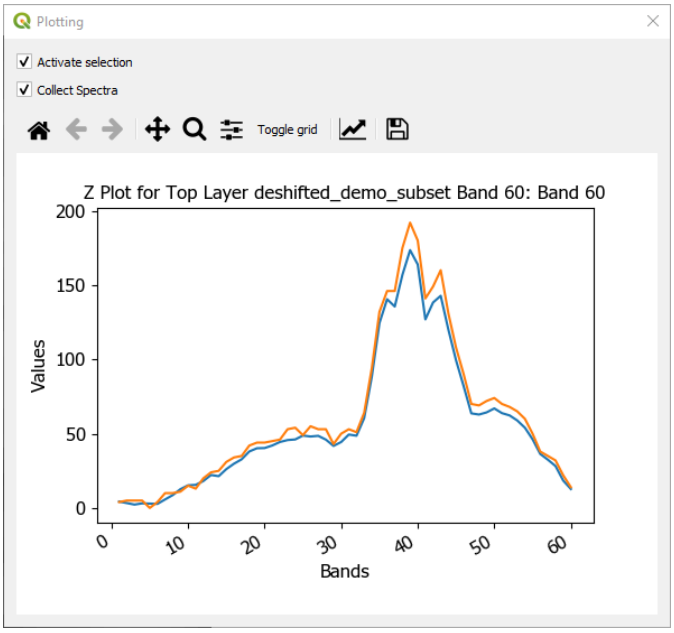
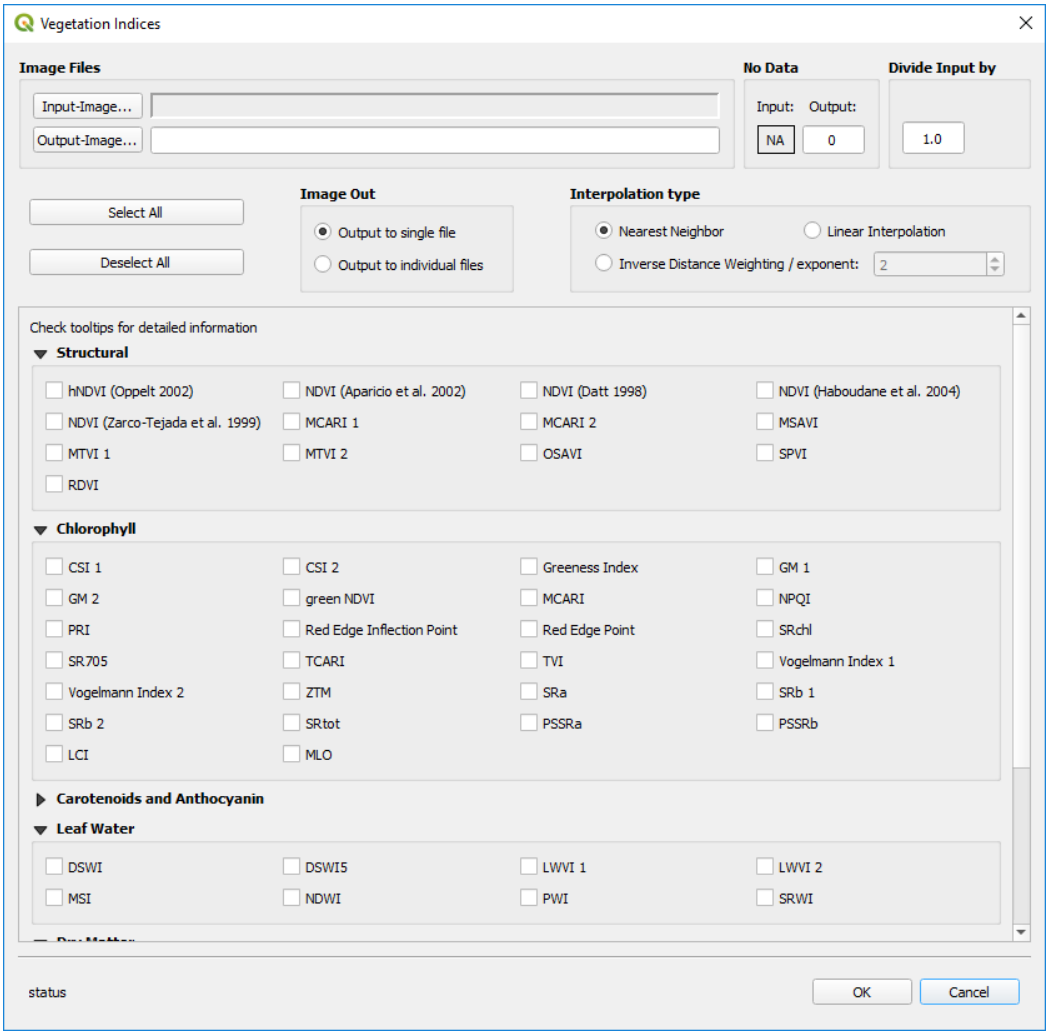


Figure Spectral plot of fused and original data

Reference

Scheffler, D.; Hollstein, A.; Diedrich, H.; Segl, K.; Hostert, P. AROSICS: An Automated and Robust Open-Source Image Co-Registration Software for Multi-Sensor Satellite Data. Remote Sens. 2017, 9, 676

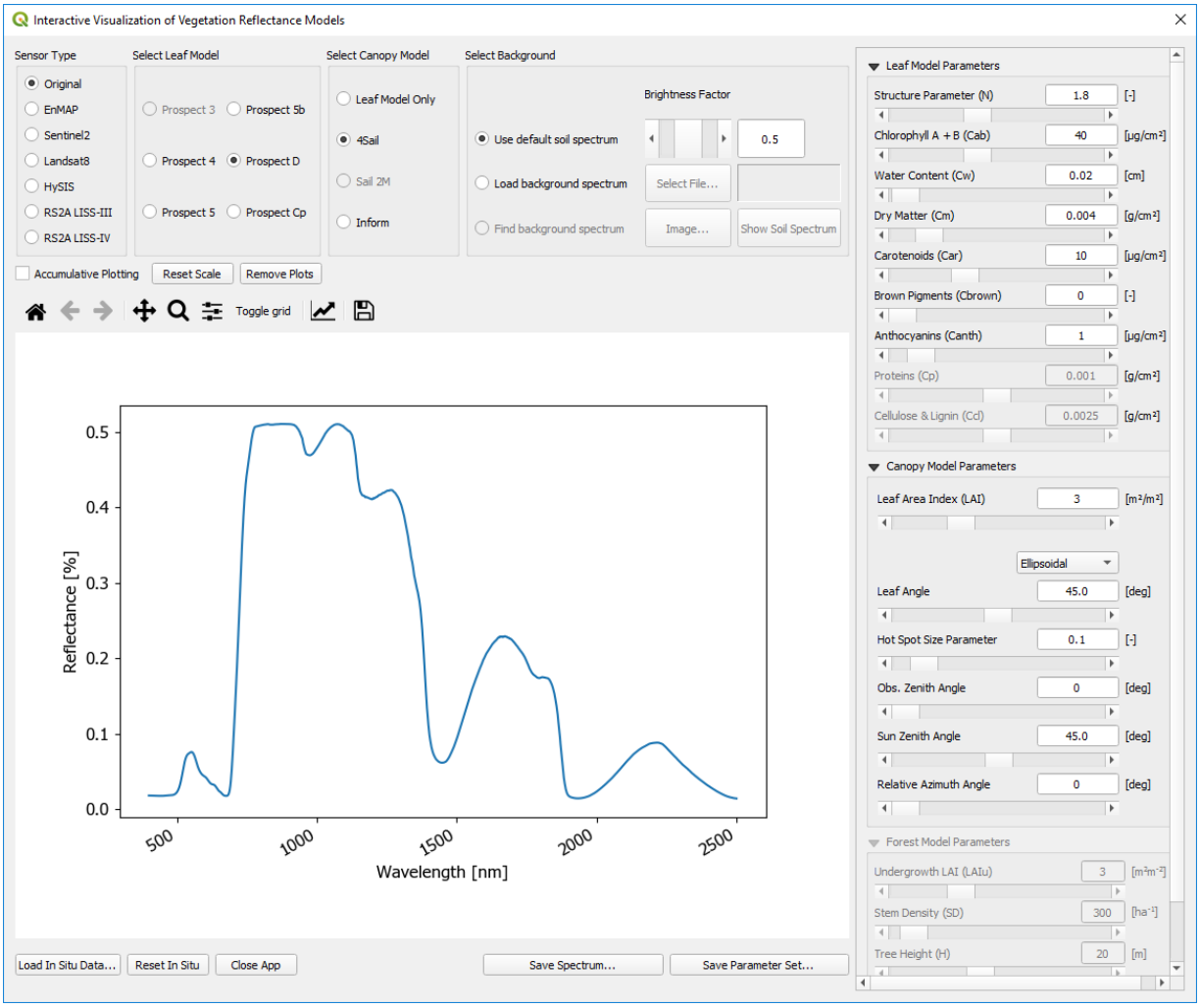
10 INDICES



Overview

This module provides a ready to use interface for deriving more than 50 vegetation indices. Pre requisite of the module is the availability of wavelength information in the metadata. The module assumes that the data has the wavelength information if not it will fail to process. There are different mode of saving output to disk provided either as a single file or multiple individual files. The module has been taken from Enmapbox plugin with few additional indices of our own as well.

11 INTERACTIVE VISUALIZATION OF VEGETATION



Overview

This module is an initial workflow for developing modules for biophysical parameters estimation using radiative transfer modelling of vegetation. AVHYAS has extended the sensors capability of Enmapbox plugin to indian sensors as well. User can interactively select any PROSAIL model (Prospect+ SAIL) and different leaf and canopy parameters and see the impact of each parameters on the spectra. Subsequent version will incorporate lookup table generation and inverse modelling for retrieval of parameters. The beta version so far has only visualization capabilities.