Geography Department

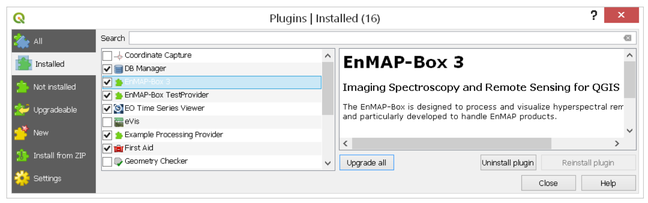
HUMBOLDT-UNIVERSITÄT ZU BERLIN

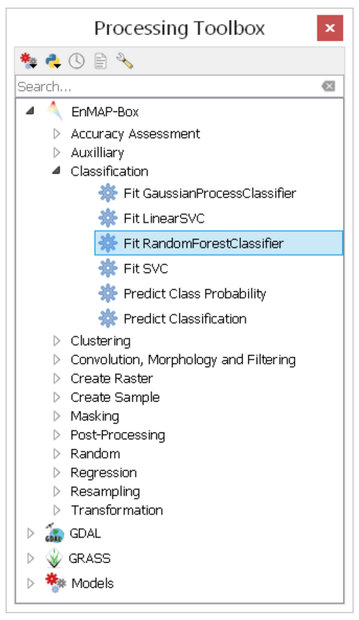
EnMAP-Box 3
A QGIS Plugin to explore and process imaging spectroscopy data

Andreas Rabe, Benjamin Jakimow, Fabian Thiel, Sebastian van der Linden and Patrick Hostert
Contact: andreas.rabe@geo.hu-berlin.de | +49.30.2093.9331 | www.hu-geomatics.de

Motivation

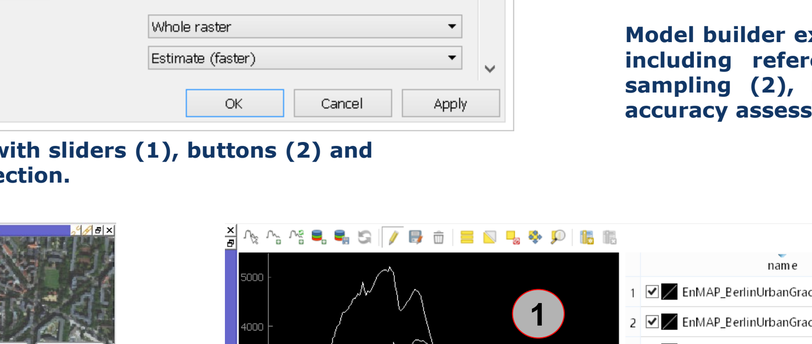
- free and open source plug-in for QGIS
- process imaging spectroscopy data
- state-of-the-art applications
- intuitive graphical user interface



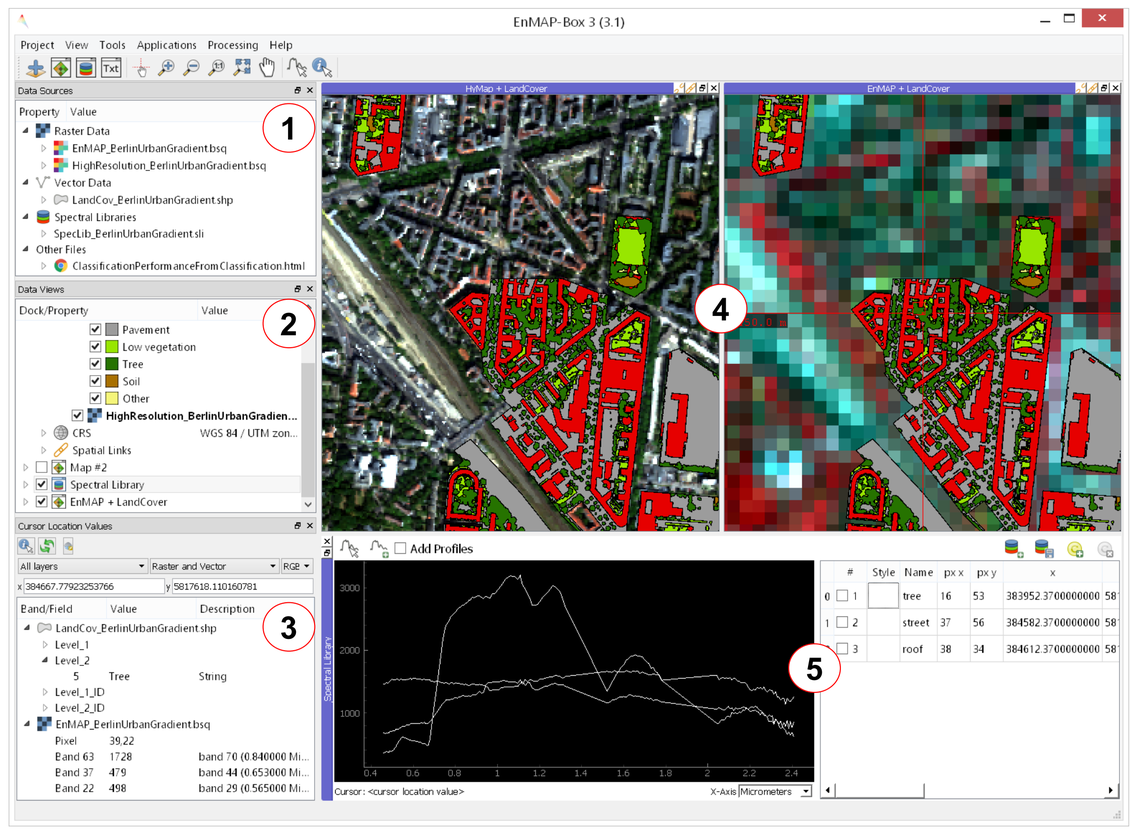


Processing algorithms

- algorithms are integrated into QGIS processing framework (toolbox and model builder)
- API with high level data types and operators
- tools for data exploration and preparation

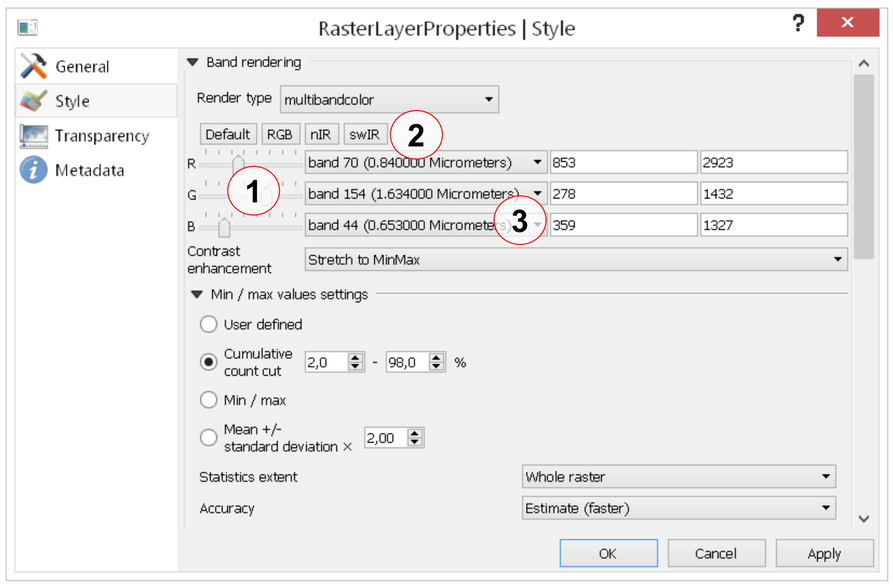


Package and dependency overview

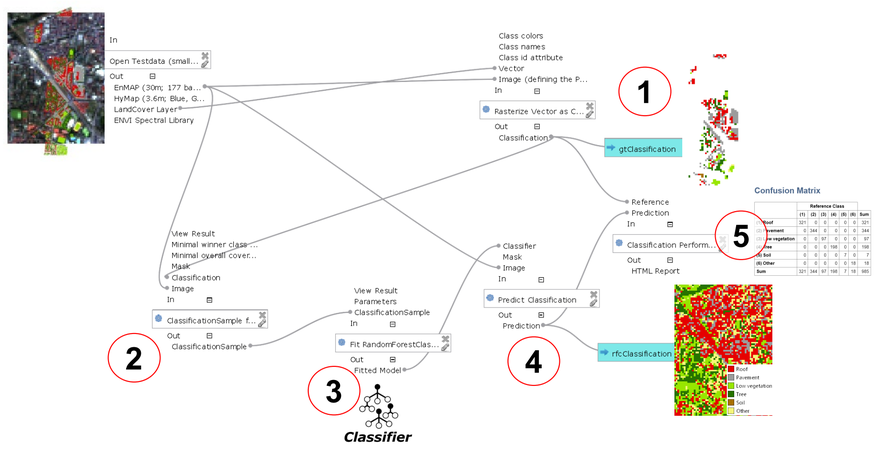


EnMAP-Box main window with panels for (1) data sources, (2) data views, (3) cursor location values and windows for (4) raster/vector maps, and (5) spectral libraries.

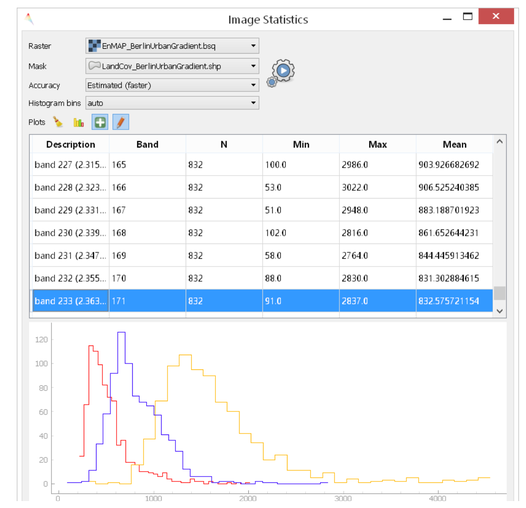
GRAPHICAL USER INTERFACE



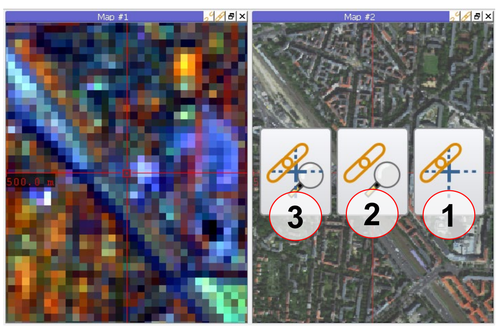
Enhanced layer styling dialog with sliders (1), buttons (2) and droplists (3) for waveband selection.



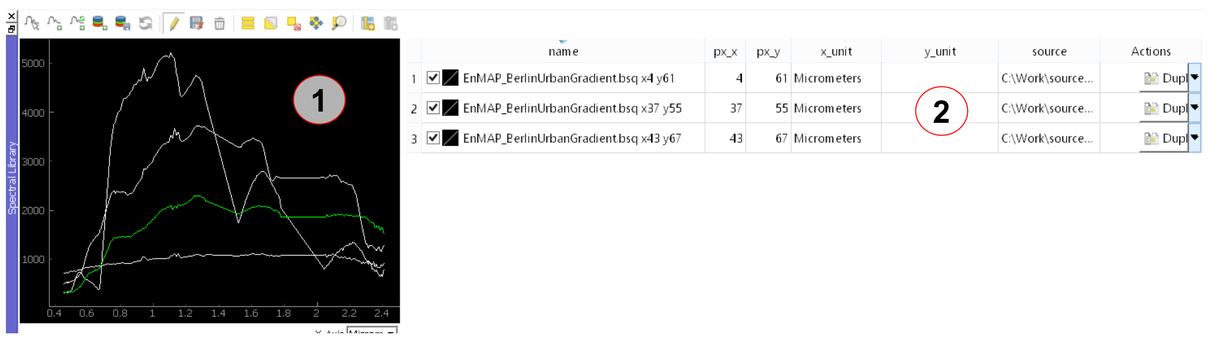
Model builder example showing an image classification workflow including reference polygon rasterization (1), training data sampling (2), model fitting (3), model application (4), and accuracy assessment (5).



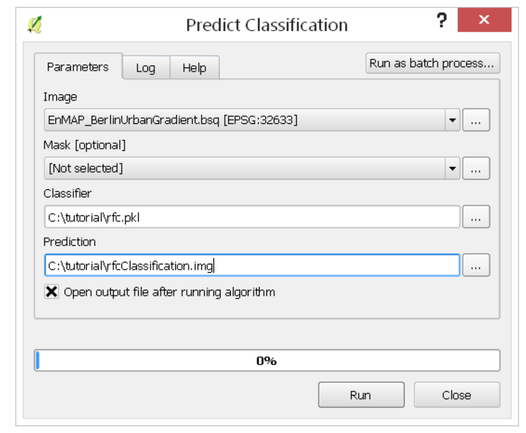
Interactive application example for image statistics and histogram plotting.



Map Views can be linked by position (1), scale (2) or both (3).



Spectral view for visualizing (1) and annotating (2) collected spectra. Pixel profiles selected inside a map view can be overlaid (green line in plot window).



Processing algorithm example showing the auto-generated dialog for applying a fitted Classifier to an image.

APPLICATION PROGRAMMING INTERFACE

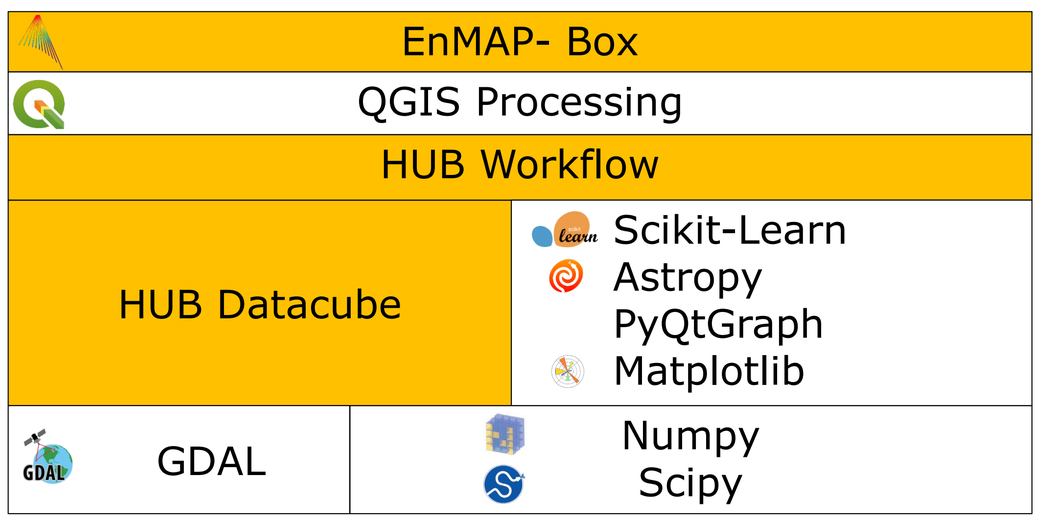
HUB Datacube API

- API for conveniently reading and writing data with on-the-fly re-projection, resampling and vector rasterization
- user defined processing functions for block-wise i/o and (parallel) processing

HUB Workflow API

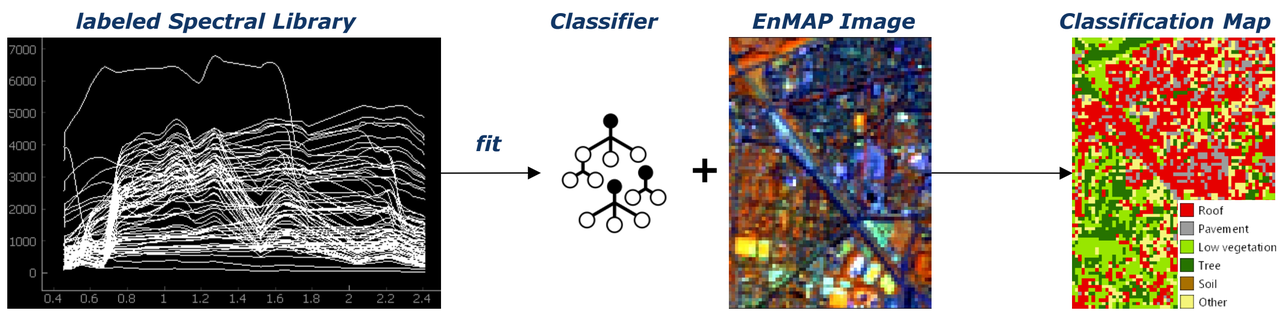
- API for higher-level workflows
- data types ensure data integrity
- better code readability due to domain-specific type and method naming

API Stack




```python
from hubflow.core import *
from sklearn.ensemble import RandomForestClassifier
import enmapboxtestdata

# resample library spectra to EnMAP wavelength
library = ENVISpectralLibrary(filename=enmapboxtestdata.speclib)
enmapSensor = SensorDefinition.EnMAP()
enmapSpectra = enmapSensor.resampleRaster(filename='/vsimem/enmapSpectra.bsq',
                                          raster=library.raster())
# fit Random Forest Classifier and apply to an image
# - get class labels from library metadata
labels = Classification.fromRasterMetadata(filename='/vsimem/labels.bsq',
                                           raster=library.raster(),
                                           classificationSchemeName='level 2')
# - fit classifier on labeled spectra
classifier = Classifier(sklEstimator=RandomForestClassifier())
classifier.fit(sample=ClassificationSample(raster=enmapSpectra,
                                           classification=labels))
# - apply classifier to another image
classification = classifier.predict(filename='/vsimem/classification.bsq',
                                    raster=Raster(filename=enmapboxtestdata.enmap))
```



HUB Workflow processing chain example