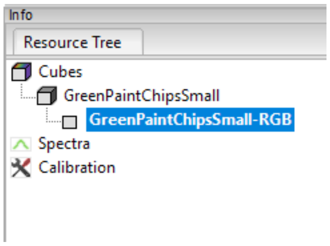
Introduction to Data Processing with Spectronon

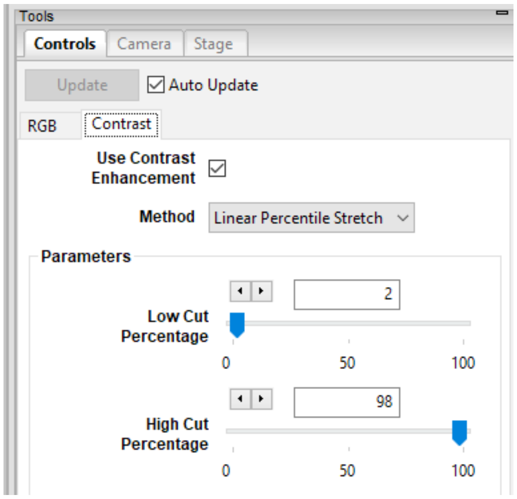
This tutorial will step you through a few example operations in order to get more familiar with the use of Spectronon for processing data, as well as introduce algorithms and techniques useful for working with hyperspectral data.

Opening and Viewing

1. Open Spectronon, then select *File->Open Datacube*. Navigate to the datacube you would like to open. The *GreenPaintChip* datacube provided with this tutorial is an interesting (if artificial) example to start with, but any datacube could work for the purposes of this tutorial.
2. When the datacube opens, you will see a cube icon appear in the *Resource Tree* on the left-hand side of the application window, and a square icon underneath the cube. The cube represents the datacube, while the square represents a 2D representation of the datacube. Typically, this representation, called an *Image* (or occasionally *Render*) in Spectronon, is a RGB image using select wavelength bands for the red, green, and blue components.



3. By default, Spectronon applies a 2% linear stretch to the Image, which usually makes the Image look more natural but depending on the scene can have side effects. Go to the *Contrast* tab on the right-hand side of the application, then unclick the *Use Contrast Enhancement* box. Press *Update* if the *Auto Update* box is not checked. Enable Contrast again and then adjust with different *Cut Percentages*.



☑ Auto Update

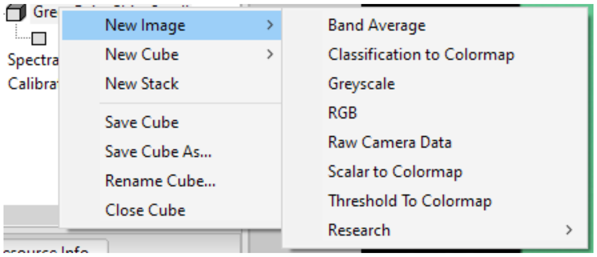
Use Contrast Enhancement ☑

Method: Linear Percentile Stretch

Low Cut Percentage: 2

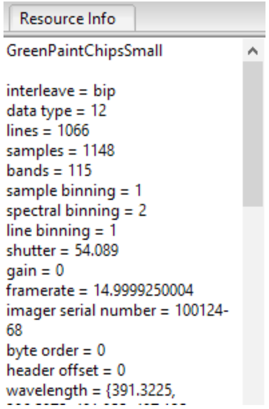
High Cut Percentage: 98

4. Back in the *Resource Tree*, right click on the Datacube object and select *New Image->Greyscale*. A new Image should appear, along with a new tab in the *Controls* box to the right. Make sure *Auto Update* is selected, then use the slider to look at different Images at selected wavelengths.



Data Pre-processing

1. A potential way of reducing data size and increasing the Signal-To-Noise Ratio (SNR) is to average adjacent channels together. Since most hyperspectral data is over-resolved spectrally for most applications, spectral averaging can be useful. Try this by right-clicking on the datacube, then selected *New Cube->Utilities->Average Channels*. Enter a 4 into the *Spectral Ave* window and press OK.
2. A new datacube will be created and added to the *Resource Tree*, below and indented from the original cube. These are the parent and daughter cubes. Select the Inspector ⊕ (looks like crosshairs) tool from the menu bar. Now click on the datacube to see the spectrum underneath the cursor show up in the Spectra Plot window. Drag the cursor to different parts of the datacube to see the spectrum at different parts of the cube.
3. Click on the new datacube in the *Resource Tree*, then view the *Resource Info* window, which contains all of the metadata for the datacube. Verify that the number of bands is a factor of 4 smaller than the parent cube.



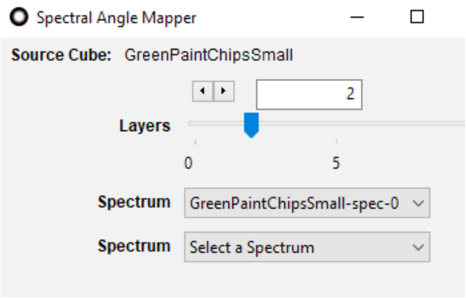
```
GreenPaintChipsSmall

interleave = bip
data type = 12
lines = 1066
samples = 1148
bands = 115
sample binning = 1
spectral binning = 2
line binning = 1
shutter = 54.089
gain = 0
framerate = 14.9999250004
imager serial number = 100124-
68
byte order = 0
header offset = 0
wavelength = {391.3225,
```

4. Another useful pre-processing step is to smooth the data using a Saviztky-Golay[1] filter, also located in the Utilities plugin folder. Perform this action. Note: The Saviztky-Golay plugin can also return a smoothed derivative of the data by using the *Differential Order* field. Derivatives can be a useful analysis tool.

Spectral Matching Classification

One useful classification technique is to define a reference spectrum of interest and try to match this spectrum to all others in a datacube. The canonical algorithm is the Spectral Angle Map[2] (SAM), which treats spectra as vectors and computes the angle between them. As such, it is fairly insensitive to brightness differences, which may be an advantage in some datasets.

1. Select the Lasso ROI tool from the menu. Use it to encircle an area of interest in your datacube.
2. Right-click in the ROI, and select *Mean Spectrum*, which will show up in the Spectra Plot window and also as a new object in the *Resource Tree.*
3. Create 2 more ROIs and Mean Spectra from different areas of the datacube. Now right-click on these spectra, one at a time, and use the *Rename Spectrum* option to give the spectra meaningful names.
4. Right-click on the parent cube and select *New Cube->Classify->Spectral Angle Mapper*. Increase the layer count to 3, then select one the Spectrum objects created above in each of the *Spectrum* drop down boxes. Press OK.



Spectral Angle Mapper

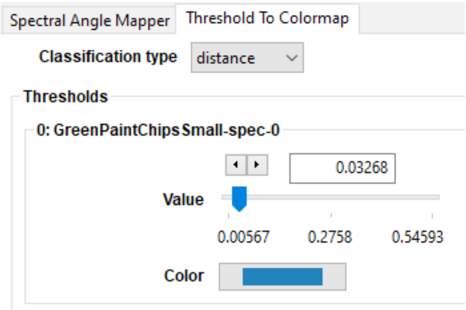
Source Cube: GreenPaintChipsSmall

Layers: 2

Spectrum: GreenPaintChipsSmall-spec-0

Spectrum: Select a Spectrum

5. A SAM classification map with a *Threshold to Colormap* image will appear. Click on the *Threshold to Colormap* tab in the *Tools* window to the lower right. Slide the *Threshold* sliders around to include or exclude pixels. You can also change the colors of each layer in this window.



Spectral Angle Mapper | Threshold To Colormap

Classification type: distance

Thresholds

0: GreenPaintChipsSmall-spec-0

Value: 0.03268

Color: 🟦

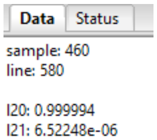
6. Use the Inspector tool in the SAM image and view the data values in the Data window to the lower left side of Spectronon. Here you can see the computed angles for all three spectra.

Statistical Classification

Spectronon contains many algorithms that can take advantage of the statistical distribution of a collection of spectra, which can be a powerful approach in datasets with large amounts of variance.

Since these algorithms use a collection of spectra for training, we must provide them with a ROI datacube instead of a *Mean Spectrum* of a ROI. We will start with the Logistic Regression[3] algorithm to demonstrate statistical classification.

1. Select the square Marquee selector tool from the menu. Create a ROI and right-click in it, selecting *Create Cube From Selection*. Give it a meaningful name and press OK.
2. Try the same thing with the Lasso selector tool. Notice that in the creation of a square datacube from a non-square ROI, the spatial relationship between pixels is scrambled.
3. Create one or two more datacubes from ROIs. Make sure to give them clear names.
4. Right-click on the parent cube, then select *New Cube -> Classify -> Logistic Regression*. Increase the *Classes* count to match the number of ROI datacubes you have. Uncheck the *Save Coefficients* to File checkbox, then select a ROI datacube for each Class dropdown box and press OK.
5. A classified image with the *Classification to Colormap* image will appear. The Logistic Regression algorithm (with the Probability checkbox checked) returns a probability that a pixel belongs to each entered class (ROI datacube). Use the Inspector tool to view the probability of each class by looking in the Data window to the lower left.



sample: 460
line: 580

I20: 0.999994
I21: 6.52248e-06

6. Now, uncheck the Probability? check box in the Logistic Regression tab on the lower right side of Spectronon. The plugin will now return a classification (by *Group Number* listed under the *Class* drop down) instead of a probability. The *Classification to Colormap* image is not appropriate for the returned cube any longer, so we need to switch it before the results make sense.
7. Right click on the returned Logistic Regression cube in the *Resource Tree*, and select *New Image -> Scalar to Colormap*. You can now see the predicted *Group Number* by color. The color scheme can be changed in the *Scalar To Colormap* tab.

Other Interesting Tools to Explore

Below are a few other interesting tools to try in order to understand the power and versatility of spectral image data. These examples also work well with the GreenPaintChips datacube, but any data could be used. The CIE Colorspace Conversion assumes units of Reflectance.

Principle Component Analysis

Principle Component Analysis[4] (PCA) transforms data onto a new coordinate system which maximizes the variance of the variables. In hyperspectral data where adjacent bands are highly correlated, this technique can be used to compress data, reduce noise, or explore data in preparation for other forms of classification or analysis.

1. Right-click on the datacube and select *New Cube->Analysis->PCA*. Uncheck the *Save Transformation Matrix* box and hit OK.
2. The 8 bands in the new coordinate space with the highest variance are returned. In the *RGB* tab on the lower right side, use the sliders to select the bands 2, 1, and 0 for red, green, and blue. These are the three most informative bands. Now slide one color out to band 7. This band is likely very noisy, showing that a 100+ band datacube can be transformed to a very small number of informative bands with a large amount of noise eliminated.

Colorspace Conversion

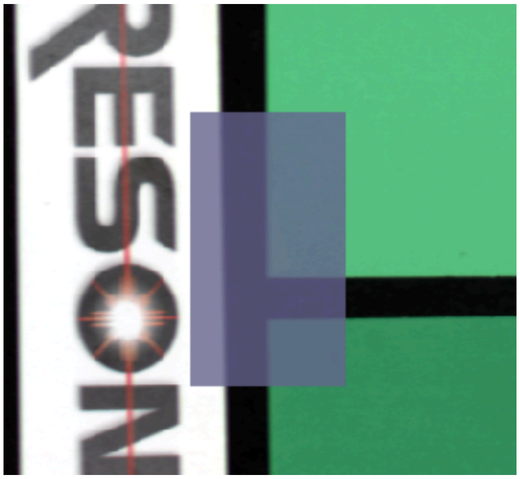
Print and manufactured goods are often specified in CIE[5] colorspaces. These spaces standardize colors into a small set of numbers and provide a measure of distances between colors that compliments human perception. Datacubes must be in units of reflectance for this tool to work.

1. Right-click on the datacube and select *New Cube->Color->CIE Colorspace Conversion*. Press OK.
2. Create a ROI with one of the selection tools, then right click and select *Show Mean of Selection* to see the CIE values of that region.

Anomaly Detection

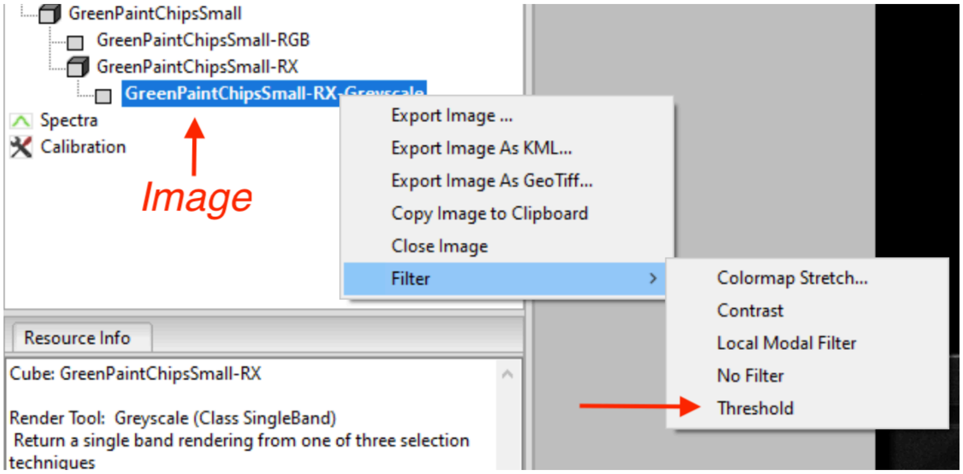
Anomaly Detection refers to a class of algorithms that attempt to classify pixels that are significantly different than the majority of the rest of the image. Some anomaly detection algorithms operate completely unsupervised, while others require a datacube of *clutter*, which is a collection of expected or *normal* spectra. This example uses the Reed-Xiaoli[6] (RX) algorithm.

1. Create a clutter datacube of the expected pixels. In the case of the GreenPaintChips datacube, create an ROI of white, black, and two of the green chips and create a cube from this selection, as shown below.



2. Right click on the datacube in the *Resource Tree*, and select *New Cube->Anomaly->RX Anomaly Detector*.
3. In the *Method* drop down, select *ed*. This will bring up a *User Select Kernel* dropdown. Select the *clutter* datacube that you created in step 1.
4. Select a kernel size of 256, and press OK.

5. The resulting datacube is a single layer, with higher values more likely to be anomalies. Right click on the *Image* of the datacube, located under and indented from the datacube object shown in the *Resource Tree.* This is shown below.



Cube: GreenPaintChipsSmall-RX

Render Tool: Greyscale (Class SingleBand)
Return a single band rendering from one of three selection techniques

6. Select *Filter->Threshold*. The tab for the threshold Filter will replace the *Contrast* tab in the *Tools* section, located on the lower right side of Spectronon.
7. Select the *Lock Min-Max* box, and slide either the *Min* or *Max Threshold* slider until the desired results are achieved.

Clustering

Clustering is an unsupervised method (meaning no ground truth or labeled data is used) of classifying pixels of a datacube into groups, or clusters. In general, supervised methods (in which ground-truth or labeled data is used) are more accurate than unsupervised, but sometimes ground truth data is not available. Clustering can be slow, so typically it is best to reduce the number of features (channel averaging, PCA, statistical feature selection, etc) first. This example uses the K-Means[7] algorithm.

1. Right click on the datacube in the *Resource Tree*. Select *New Cube->Clustering->K-Means Clustering*. Select 7 *Clusters* (4 greens, black, white, and red) and 20 *Iterations*. More iterations will refine the result but will also increase the processing time. Press OK.
2. The results are interesting, and not necessarily intuitive. Play with the number of clusters and iterations, or try some feature reduction before the clustering.

References:

1. Savitzky-Golay: https://en.wikipedia.org/wiki/Savitzky–Golay_filter
2. Spectral Angle Map: http://www.ijiset.com/v1s4/IJISET_V1_I4_27.pdf
3. Logistic Regression: https://en.wikipedia.org/wiki/Logistic_regression
4. Principle Component Analysis: https://en.wikipedia.org/wiki/Principal_component_analysis
5. CIE Colorspaces: https://en.wikipedia.org/wiki/CIELAB_color_space
6. Reed-Xiaoli: https://www.hindawi.com/journals/jece/2012/162106/
7. K-Means: https://en.wikipedia.org/wiki/K-means_clustering