Hyperspectral Imaging, or imaging spectroscopy, refers to the creation of a digital image containing very high spectral (color) resolution. Each spatial point (pixel) in a hyperspectral image represents a continuous curve of incoming light intensity versus wavelength. For example, red, green, and blue arrows below show the spectra for three pixels of an image of a leaf in the image below. The data can also be interpreted as a stack of images, with each layer in the stack representing the scene at a different wavelength – this “stack” of two-dimensional images is referred to as a “datacube.”

The benefit of the additional information provided by hyperspectral imaging is that it generally allows you to tell the difference between materials better than you can with traditional color images. This capability can be applied to a wide range of current and emerging applications in remote sensing, sorting, quality control, research and development, and more.
Resonon imaging spectrometers are **line-scan** imagers, which means they collect data one line at a time. To assemble a complete two-dimensional image, multiple lines are imaged as the object (or imager) is translated. The multiple line-images are then assembled line-by-line to form a complete image, as indicated below. A scanning system is often needed to use a Resonon imaging spectrometer.

To obtain hyperspectral data, the signal from each pixel is dispersed (or diffracted) into its spectral components, much like passing the light from each pixel through a prism. This process occurs for every pixel in the line (red squares in the image below). The dispersed signals from all the pixels are imaged onto a focal plane array. One of the benefits of this approach is that all the spectral (color) information is collected at the same time for each pixel. The result is a detailed spectral curve for every pixel in the image.
Resonon’s benchtop hyperspectral imaging system is comprised of a Pika hyperspectral imaging camera, linear translation stage, mounting tower, lighting assembly, and software control system. The positions of the imager and lighting assembly are adjustable along the length of the tower. See Figure 1 below.

Resonon’s hyperspectral imagers are line-scan imagers (also referred to as push-broom imagers). Two-dimensional images are constructed by translating the sample relative to the camera. This is typically accomplished by placing the sample on a linear translation stage.

Resonon’s Pika imaging spectrometers are compact, high fidelity, digital instruments for industrial and scientific applications. Spectronon is a powerful hyperspectral data visualization and analysis software package we provide as a free download. Spectronon is easy to learn, offers efficient workflow, and is highly extensible by the user for custom applications. Additionally, a number of datacubes can be downloaded from the Resonon website (www.resonon.com) so you can begin exploring hyperspectral data within a few minutes.
SpectrononPro has all the features of Spectronon, but also includes data collection tools that are highly integrated with our Pika imaging spectrometers to streamline the collection of spectral images.
3.1 Assembly of Stage, Tower, and Lighting

Begin system assembly by removing all items from the shipping package. Confirm that the items shown in the images below. If you are missing components, please contact Resonon.

Install lighting onto the tower. Begin by sliding the T-nuts into the slots on the tower. Slide in one T-nut into each slot on the same side as the adjustable post holder.
Position the T-nuts side-by-side at the location you wish to attach your lights. (Note: You may easily adjust the lighting position later.)

Place the lighting assembly on the post such that the mounting holes line up with the T-nuts.
Attach the lighting assembly to the T-nuts using the supplied bolts.

Next, attach the post to the baseplate. Begin by removing the four bolts from the bottom of the post.
Align the post to the baseplate, then tip the baseplate up and attach using the four bolts just removed from the bottom of the post.

After tightening all four bolts, tip the tower upright. Connect the lighting assembly to the regulated power supply.
Take the red and black banana plugs, and insert them into the red and black connectors on the regulated power supply.

Push the wire into a slot on the side of the tower.

Plug the regulated power supply into a 120 V socket and turn on to test the lighting. Turn the lights off after testing to continue with assembly.
If the lights do not turn on, turn off the red power switch and check all connections. If the lights continue to fail, contact Resonon.

**Note:** When illuminating your sample with Resonon’s lighting system, allow 20 minutes for the lighting to fully stabilize.

Using the supplied USB cable, connect the stage to your computer. Plug the cable into a black USB 2.0 port, not a blue USB 3.0 port.

Plug the mini-USB connector end into the stage motor.
Next, connect the power to the stage using the provided DC power supply.

Screw the posts onto the bottom of your Pika imaging spectrometer. The studs in the posts are ¼”-20. Install both posts in neighboring mounting holes on the bottom of the imaging spectrometer.
Insert the posts into the adjustable post holder on the tower.

Tighten setscrew on the adjustable post holder to secure your Pika imaging spectrometer.

To adjust the height of your Pika imaging spectrometer, loosen the handle on the side of the adjustable post holder, move to your desired position, and then re-tighten the handle.

Assembly of your linear scanning stage system with stabilized lighting assembly is now complete.
3.2 Setup of Hyperspectral Imager

3.2.1 Connecting Pika imaging spectrometer

All Pika imaging spectrometers must be interfaced to a computer for data acquisition. Depending on which Pika imaging spectrometer you have, this interface may utilize a USB or CameraLink interface. Additionally, some Pika spectrometers require external power. The image below shows interfacing with USB 3.

Using the provided interface cable, connect your Pika imaging spectrometer to the appropriate port on your computer.

3.2.2 Installing Calibration Values

Older Pika imaging spectrometers required the user to manually insert calibration values into SpectrononPro software. This is now done during factory set-up of the imager and requires no additional attention from the user. However, your Pika’s calibration numbers are provided on a sheet that comes with your imaging spectrometer and should be checked prior to use. If you cannot locate your calibration numbers, contact Resonon at: ProductSupport@resonon.com.

Connect your Pika imaging spectrometer to your computer. Launch SpectrononPro by double-clicking on the SpectrononPro icon, or navigate to it from your Start menu.

From the Main menu, select File → Preferences…. This will reveal the Preferences menu. Select the tab for your Pika imaging spectrometer.
Check the $A$, $B$, and $C$, Spectral Calibration values from the sheet provided against those that are installed in SpectrononPro. If they are different please contact Resonon at: ProductSupport@resonon.com.

Initial setup is now complete. Instructions for how to collect hyperspectral images begin with Section 4: Basic Data Acquisition.
3.2.3 Objective Lens Installation

Resonon imaging spectrometers are designed to operate with Schneider c-mount lenses. To install the Schneider lens, first remove the lens and imager from their packaging. Unlock the set screw on the chrome lens collar of the Schneider lens with a 2 mm hex wrench. NOTE: Do NOT remove the screw all the way, just loosen it a few turns.

Holding onto the objective lens by the chrome collar, unscrew the black lens barrel out until there is approximately a 10mm gap between the barrel and collar.

Do NOT install the lens onto the imager if the barrel is screwed in too far as shown below.
While holding and squeezing the chrome collar, thread the objective lens onto the imager.

The objective lens is now installed.

### 3.2.4 Setting Objective Lens Aperture

Next, set the f-stop (f/#) of the objective lens. There is a range of acceptable options and the end user will need to choose the appropriate value based on his or her application. Large f/#s will provide large depths of field, but limit the amount of light collected, whereas small f/#s will have small depths of field but will exhibit better light gathering.
To set the f/# on Schneider lenses, for example, loosen the setscrew on the objective lens itself. This will allow you to rotate the f/# collar on the objective lens, which can be read by noting the location of a small white alignment line. Align the white line with your desired f/#, then retighten the setscrew.

**Acceptable f/#s**

<table>
<thead>
<tr>
<th>Lens</th>
<th>f/# Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pika L</td>
<td>2.4 and larger</td>
</tr>
<tr>
<td>Pika XC2</td>
<td>2.4 and larger</td>
</tr>
<tr>
<td>Pika NIR</td>
<td>1.8 and larger</td>
</tr>
<tr>
<td>Pika NUV</td>
<td>2.4 and larger</td>
</tr>
</tbody>
</table>
4.1 Data Modes

Hyperspectral data from Resonon imaging systems can be utilized in three forms, as summarized below.

4.1.1 Raw Data

This data is spectrally calibrated but contains the instrument sensor response, sample reflectivity and illumination functions. This is the least useful form, as the spectral curves do not have real units or real physical meaning. The units are in Digital Number (DN).

4.1.2 Radiance

The raw data can be post-processed to radiance. This requires the imager to be specially calibrated (radiometric calibration) by Resonon at the desired aperture. This data form does not include the instrument sensor response function. This function is corrected for by using the Imager Calibration Pack (ICP file) with the Radiance From Raw Data plugin. The resulting data is the product of illumination and sample reflectivity. It has the advantage of possessing real units (in Microflicks) and physical meaning [microflick = 1 microwatt per steradian per square centimeter of surface per micrometer of span in wavelength].

4.1.3 Reflectivity/Reflectance

In reflectance mode, both the instrument sensor response and illumination functions are removed. This leaves the data in absolute reflectance. Data can be converted to reflectance in one of four ways described below. These explanations apply mainly to our airborne and outdoor system users as most benchtop users will be in a lighting controlled environment.

1. **White reference**: Data can be processed to reflectance with a quick calibration against a reflection standard. The highest quality reflection standard is Spectralon, but Teflon is acceptable for many applications. (Note: Teflon needs to be sanded with 100 grit sandpaper on an orbital sander to eliminate any specular properties). This calibration is done with the Record Correction Cube feature, as described in Section 4.6: Imager Calibration. It is important to note that reflection values are only accurate if the solar illumination (cloud, sun angle, etc) does not change between the collection of the correction cube and the collection of datacubes. Data can also be converted to reflectance using the Reflectance from Raw Data and Spectrally Flat Reference Cube plugin.

2. **Known spectral reference in scene**: Once the data is in radiance, the spectrum of a reference object in the scene can be used to correct the rest of the cube. The reference spectrum must be known and in a tab or space delimited file. Use the Reflectance from Radiance Data and Spectrally Flat Reference Spectrum plugin.
3. **Downwelling irradiance sensor:** An alternative method for converting data to reflectance is to use a downwelling irradiance sensor. This sensor records the solar spectrum during data acquisition. This data is used, along with the Imager Calibration Pack (ICP files) supplied by Resonon for both the spectral imager and downwelling sensor. This method uses the *Reflectance from Radiance Data and Downwelling Irradiance Spectrum* plugin.

4. **Atmospheric Correction:** Data can be converted to reflectance data with the use of atmospheric correction algorithms such as FLAASH (Fast Line of Sight Atmospheric Analysis of Spectral Hypercubes). Please contact Resonon for more information.

### 4.2 Start The System

If you have a lighting system, turn on the lights and let them warm up. It may require 15 to 20 minutes for the illumination to fully stabilize.

With the camera and scanning system connected to your computer, launch **Spectronon** or **SpectrononPro** by double-clicking on the Spectronon icon or starting **Spectronon [Pro]** from your Start menu. The Spectronon user interface is shown below with the various windows labeled.
Once the software has started, make sure that the imager and stage controls (if used) are enabled. The imager and stage tools will be greyed out if not enabled, as shown below.

You can get the latest version of SpectrononPro by clicking on Help → Check For Updates. This won’t download the latest version, but will give an alert if there is a newer version.

4.2. Start The System
4.3 Camera Controls

Exposure parameters can be controlled by clicking on the Camera tab in the Tools Panel.

Frame Rate is equal to the number of images acquired each second, and limits the maximum exposure time (Max Exposure Time = 1.0 ÷ Frame Rate).

Integration Time (also known as exposure time) is the duration of data acquisition for each individual line image.

Gain is a factor which increases the signal, but at the expense of signal-to-noise ratio. Try to keep gain as low as possible (preferably zero), unless absolutely necessary. (Note: The Pika NIR camera does not have a gain control tool.)

4.4 Stage Controls

You can move the stage manually by clicking on the Jog Stage buttons, located on the toolbar. The buttons will move the stage incrementally in either direction. Use the buttons to center the stage underneath the Pika imaging spectrometer.

The stage can be further controlled by clicking on the Stage tab in the Tools Panel.
**Speed Units** is a setting used for different types of stages. **Linear** is used for the standard linear translation stage that is installed on most benchtop systems. **Rotation** is used for a tripod-mounted rotational scanning stage, typically used in outdoor applications. **Motor** displays speed in “motor pulses per second.”

**Stepping Mode** controls the way the stage moves in relation to the imager. When the Stepping Mode box is not checked, the stage and imager both run continuously during the scan time.

When the **Stepping Mode** box is checked, the stage moves incrementally, an image is acquired while the stage is stationary, the stage moves incrementally again, another image is acquired while the stage is stationary, and so on. This behavior is preferable when the scanning speed is very slow, when the integration time is very long, or to guarantee there is no motion blur in your scan.

**Scanning Speed** is the linear speed of the stage during a scan.

If the “Go Home After” box is checked the stage will return to its starting position after a scan. The speed at which the stage returns to its original position is the **Homing Speed**, and the **Jog Speed** is used for the **Jog Stage** buttons, described above.
4.5 Focus Objective Lens

You are now ready to focus the objective lens of your Pika imaging spectrometer. At first, this process is somewhat challenging, but with a little practice it becomes quite easy. Begin by clicking on the Focus button located on the SpectrononPro tool bar. This will reveal a live image from the camera within your Pika imaging spectrometer. (Wave your hand in the field of view of your Pika imaging spectrometer to confirm that the image is a live view.) One axis of this image represents the spatial (position) axis of your object, and the other is the spectral (wavelength) axis. (To understand this view better, move colored objects within the field of view of your imager after you have focused the objective lens.)

Place an object with multiple light and dark regions within your Pika imaging spectrometer’s field of view. A sheet of paper with dark lines, as provided in Section 11: Focusing & Calibration Sheets, works well. If you are in the field and are observing objects at a distance, direct your Pika towards an object with multiple features, such as a tree with many branches. Unless your lens is already focused, you will see a series of blurry or barely discernible lines in the Image Panel of SpectrononPro.

To adjust the focus, first unlock the focus adjustment. With Schneider lenses, this is done by loosening the locking metal collar on your objective lens using an Allen wrench, size 5/64 inch.

Then rotate the objective lens until you see dark lines from your object come into focus, as shown. Maximize the sharpness of the lines.
Hint: Clicking the Inspector Tool in the Image Panel, and then selecting X tab in the Plots window will reveal a cross-section plot of your image. Viewing this plot allows you to graphically see the sharpness of your focusing. You can zoom in on the X-axis by clicking the Zoom tool and then clicking on the X-axis.

Focusing the outdoor system can be a little more challenging than the benchtop system. If you are focusing on objects that are further than 40 feet start the process with the objective lens screwed in close to the collar. A method that has proven useful is to start the focusing process by increasing the number of lines scanned to 1000. This will give you a large scan area to begin the process. You will need to be out of live focus mode when performing this focusing procedure (“F” button should not be red). When the scan begins make a quarter turn with the lens. Continue to make quarter turns until you are confident your scene is in focus. When making the quarter turn intentionally place your hand in front of the lens. This will create a thin black line in the scan separating one focus length from its neighbor, allowing for easier comparison. This process can take some time, so be patient and remember that it will get easier with practice.

Once you have completed focusing, re-tighten the lock to the focus adjustment. Then click on the Focus tool again to toggle the camera live view off.

Hint: See our You Tube video on focusing the benchtop system at http://www.youtube.com/resonon.

4.5. Focus Objective Lens
4.6 Imager Calibration

The following discussion describes how to set up your system to scan for reflectance scaled to a reference object. If you wish to collect raw data and convert it to radiance do not perform the following correction process.

4.6.1 Remove Dark Current

SpectrononPro makes it easy to remove the average dark current noise from your scans. Begin by clicking on the Dark Current button on the SpectrononPro toolbar. You will be instructed to block all light entering your Pika imaging spectrometer by blocking the objective lens.

Once you have the objective lens blocked, click OK as instructed. SpectrononPro will then collect multiple dark frames and use these measurements to subtract the dark current noise from your measurements. The Dark Current button on the toolbar will appear with a red check through it as soon as the dark frames have been collected. Once you see the red check, unblock the objective lens.

4.6.2 Set Reflectance Reference

Measuring absolute reflectance of an object requires correction to account for illumination effects. To do this, click on the Response Correction Cube button on the SpectrononPro toolbar. A message will appear telling you to place a reference material within your Pika imaging spectrometer’s field of view. The reference material should be uniform across the imager’s field of view. Examples of reference materials include Spectralon® or sheets of white Teflon.

Once the reference material is in place, click on OK. This will trigger a short scan of the reference material. Once complete, the Response Correction Cube button will appear with a red check mark, indicating that the data you collect will be scaled in reflectance to your reference material, including flat-fielding to compensate for spatial variations in your lighting.
Note: The Dark Current button and the Response Correction Cube button are disabled in live camera view mode. Click on the Focus tool to toggle the camera live view off.

Once the imager is calibrated for both dark current and reflectance reference, the imager will remain calibrated until the references are removed by the user, or the machine is turned off. If the integration time is changed by the user after calibration, the reference signals will be adjusted accordingly. To manually remove the references, click Spectrometer → Remove Dark Current Cube, and Spectrometer→ Remove Response Correction Cube.

For additional help with this process see our Calibration video at http://www.youtube.com/resonon.

4.6.3 Adjust Aspect Ratio

Scanning objects continuously (i.e., with Stepping Mode off) is faster and usually the preferred method. However, when scanning continuously the initial image will usually be distorted along the scan dimension. To correct this distortion, you must calibrate the Scanning Speed of the stage relative to the or Framerate, which is the line acquisition rate of the camera.

Recall that Pika imaging spectrometers are line-scan instruments. By adjusting the stage speed of the scanning system relative to the frame rate, you are adjusting the spacing of the lines used to assemble your image. Thus, some adjustments are necessary to scan images with a unity aspect ratio that gives the proper proportion between the scan direction and the line direction.

To adjust your scan’s aspect ratio, it is useful to first image an object whose distortion is easy to observe, such as a circle. Print out the Pixel Aspect Ratio Calibration Sheet provided in Section 11: Focusing & Calibration Sheets for this purpose. After placing an object with circles within the field of view of your Pika imaging spectrometer, record a scan with enough lines that you can see the complete circle. (200 lines are usually sufficient.) Record the scan by clicking on the Scan Button, and a waterfall image should appear in the Image Panel of SpectrononPro. You may need to record several trial images to determine how many lines to scan and where to best position the object.

Note: You can stop a scan by re-clicking on the Scan Button, which changes to the Stop Button during the scan.

If your image is distorted along the scan direction, you will need to change the Scanning Speed on the Stage tab on the Tools panel. If your image is elongated along the scan direction, then the stage was moving too slowly in relation to the frame rate (over-sampling), and you need to increase the scanning speed. Conversely, if your image is compacted along the scan direction, then the stage was moving too quickly in relation to the frame rate (under-sampling), and you should decrease the scanning speed.

Note: Alternatively, the Framerate can be adjusted in relation to a fixed Scanning Speed. However, a faster Framerate corresponds to a lower maximum Integration Time and thus limits the available signal.

Repeat the above process until your image is no longer distorted. For additional help setting the aspect ratio see our Setting Aspect Ratio video at http://www.youtube.com/resonon.

4.6. Imager Calibration
A distortion-free image with unit aspect ratio is achieved when the stage/object advances the distance of the projected image of one spatial pixel per each line acquired at the imager’s given frame rate. Thus, changing the Framerate setting requires updating the stage speed to maintain unity aspect ratio, while changing only the Integration Time does not. Similarly, changing the working distance of the imager to the stage changes the object magnification, and thus changes the aspect ratio for any particular Framerate and Scanning Speed combination.

4.7 Scanning and Saving Datacubes

To scan an image of an object after calibrating the system, place the object on the stage within the imager’s field of view and type in the number of lines you would like to scan in the window just to the left of the Scan Button. Record a hyperspectral datacube (the image) by pressing the Scan Button. (Click to terminate the scan early, if needed.) Adjust the stage position and increase/decrease the number of lines as needed to cover the object’s feature(s) of interest.

A waterfall image of your datacube will appear in the Image Panel of Spectronon, and a new entry labeled Current Scan will appear in the Resource Tree. To save the scanned datacube, use your mouse to select Current Scan and then either right-click or select Datacube → Save Cube. This will open a dialog window that allows you to name the datacube and save it in a folder of your choosing. If you do not save your datacube, the current scan will be overwritten when you record another datacube, and a warning should appear.

For additional help scanning and saving datacubes see our Scanning and Saving video at http://www.youtube.com/resonon. Once an image is scanned, you can use all the visualization and analysis tools of Spectronon on your image.
5.1 Camera Controls

Control of the imager can be controlled from two windows: the Camera tab in the Tools Panel, and the Preferences window (accessed by File → Preferences).

![Camera tab in tools window]

Fig. 1: Camera tab in tools window
When the *Preferences* window opens, select the tab for your specific Pika imager (Pika L, Pika XC2, Pika NUV, or Pika NIR). The options for different Pika models will vary.
5.1.1 Basic Camera Controls

Integration Time (also known as exposure time) is the duration of data acquisition for each individual line image.

If you use the Auto Expose button in the toolbar, the integration time will be adjusted automatically for your lighting conditions such that there are no saturated pixels in the image. Make certain the reflectance calibration tile is being imaged when using this option.

Frame Rate is equal to the number of images acquired each second, and limits the maximum integration time (Max Integration Time (seconds) = 1.0 ÷ Frame Rate). If you do not have enough light, decreasing the frame rate will enable
a longer integration time.

*Gain* is a factor which increases the signal, but at the expense of Signal-To-Noise ratio, in units of dB. We recommend that you operate at low Gain values (preferably zero) to minimize noise. However, for situations where you need to operate at high frame rates and need more signal, you may increase the Gain manually. The *Auto Expose* button in the toolbar will always set the gain to zero. (Note: The Pika NIR camera does not have a Gain control tool.)

**Hint:** When adjusting the camera settings, you may observe the effects of your adjustments in live view. To do this, click on the *Focus* button.

**Note:** If you observe broken or torn images when recording data at high speeds, you may be able to eliminate this problem by collecting data at slower speeds. These errors may occur due to a number of shortcomings within the data acquisition system including limitations in disk-write speed, chipset, motherboard bus speed, and CPU speed.

### 5.1.2 Spectral Calibration Values

As discussed in Section 3: Benchtop System Assembly, the spectral calibration values should be set by the factory when you receive your hyperspectral imager. The values are provided on a paper calibration sheet with your imager. Contact Resonon support if you have lost your spectral calibration values.

### 5.1.3 Auto Exposure Parameters

Spectronon features a user-configurable auto exposure algorithm that can be controlled in the camera settings tab. Two parameters are available: *Target Brightness* and *Target Pixel*. During the auto exposure routine, Spectronon calculates the histogram of pixel brightness of each frame. It then iteratively adjusts the integration time such that the histogram percentile represented by the *Target Pixel* parameter is at least as bright as the *Target Brightness*, but is not saturated. For example, if the *Target Pixel* is equal to 100% and the *Target Brightness* is equal to 3890 (the default settings for 12 bit cameras), the algorithm will attempt to find an exposure time such that the brightest pixel has a brightness of at least 3890 but less than 4095 (saturation). Similarly, if the *Target Pixel* = 95%, the pixel that represents the 95th percentile of total brightness will be targeted in each frame, instead of the brightest pixel. Selecting a *Target Pixel* less than 100% may cause the brightest pixels to be saturated, but may be useful for achieving desired exposure when features of interest are not the brightest in the field of view). For typical conditions, *Target Pixel* is set to 100% and *Target Brightness* is set to 95% of the camera’s bit depth.

### 5.1.4 Binning of Hyperspectral Data

**Binning** involves combining data from multiple adjacent pixels into a single data point. Binning reduces overall data volumes and increases SNR, at the expense of decreased resolution. Binning is possible in the spectral dimensions only.

There are two types of binning. One type is software binning, which occurs in Spectronon, and is available on all systems. The other type is hardware binning, which is available on the Pika L, Pika XC2, and Pika NIR. Hardware binning has the advantage of decreasing data volumes and increasing SNR. See the below table for more information on the affects of hardware binning with the specific imagers.
It is preferable to bin with hardware than with software, which will utilize fewer CPU resources.

Software binning is available on the Scanner tab in the Preferences window. See Section 5.1.4 below. Hardware binning controls are accessed through the appropriate imager tab in the Preferences window. The On Camera Binning shows the available spectral binning values. These values may vary between imager models.
Fig. 3: Pika XC2 imager tab in the Preferences window
GigE Connection Setup

The Pika II imager with GigE connection requires some additional steps for complete setup. If you purchased your computer from Resonon, these steps may have already been completed for you.

First, enable jumbo packets for your network adapter in Windows. In Windows, click the start menu, right click on My Computer, and click the Manage option. Click on Device Manager in the left column, click on Network Adapters in the middle column, right click your wired network adapter and choose Properties. In the Advanced tab click Jumbo Packet and set the Value to ~9014 Bytes. Click OK and close the Computer Management window.

![Device Manager - Network Adaptor - Jumbo Packets]

Fig. 4: Device Manager - Network Adaptor - Jumbo Packets

In Spectronon, two additional settings will appear under the subgroup “GigE Properties” in the Pika II tab under the File → Preferences . . . menu: Packet size and Packet Delay. These settings allow for management of bandwidth between the camera and the controlling computer. The settings selected here will, in part, determine the achievable framerate. Larger packet sizes require less CPU usage to transfer a given amount of data from the camera to the computer than small packet sizes. The maximum packet size is limited by your network adapter, any other network hardware, and the camera. Typically, the largest available packet size is recommended.

The packet delay sets the gap between data packets transmitted to the computer from the camera, and allows for time for the CPU to process the data it has received before new data is transmitted. If your scans result in dropped frames, try increasing the packet delay (which will impact the maximum framerate achievable).
Typically problems with these settings will manifest as image inconsistency errors, which are reported in the bottom status bar as a scan progresses. See the figure below for an example warning.

Inconsistency warning. Try increasing packet delay.

5.2 Stage Controls

You can move the stage manually by clicking on the Jog Stage buttons, located on the tool bar of SpectrononPro. The buttons will move the stage incrementally in either direction. Use the buttons to center the stage underneath the Pika imaging spectrometer.
Further control of the stage can be controlled from two windows: the Stage tab in the Tools Panel, and the Preferences window (accessed by File → Preferences).

**Fig. 7: Stage tab in Tools Panel**

*Speed Units* is a setting used for different types of stages. *Linear* is used for the standard linear translation stage that is installed on most benchtop systems. *Rotation* is used for a tripod-mounted rotational scanning stage, typically used in outdoor applications. *Motor* displays speed in “motor pulses per second.”

*Stepping Mode* controls the way the stage moves in relation to the imager, and is ideal for very slow scanning, very long integration times, or to guarantee there is no motion blur in your scan.

*Scanning Speed* is the linear speed of the stage during a scan.

If the *Go Home After* box is checked the stage will return to its starting position after a scan. The speed at which the stage returns to its original position is the *Homing Speed*, and the *Jog Speed* is used for the *Jog Stage* buttons, described above.

### 5.3 Additional Controls

#### 5.3.1 Scanner Preference Tab

The *Scanner* tab allows you to set the binning options for recording your datacubes. These options are useful for situations where you wish to reduce the size of collected datacubes or where you wish to improve your signal-to-noise ratio. This tab also allows you to select the correct camera for Spectronon to search for. For example the Pika L and Pika XC2 use a Basler camera.

The *Spectral Bin* option indicates how many spectral channels to bin. For many situations, this is your best choice to reduce the size of your datacubes and to improve your signal-to-noise ratio, as the spectral resolution is often higher than needed. With the Pika II, a spectral bin of 3 will result in 80 spectral bands (240 / 3).
The *Spatial Bin* option indicates how many spatial channels to bin. This option is typically useful when imaging an object known to be largely homogeneous.

![Scanner tab in the Preferences window](image)

**Fig. 8: Scanner tab in the Preferences window**

**Note:** The Bin options listed above performs an average of the binned pixels. This allows you to directly compare binned and un-binned data.
5.3.2 Workbench Preference Tab

The Selection options relate to visualizing selected regions of datacubes.

The Floodfill options relate to the flood fill (wand) tool located on the menu bar, which is used to select a contiguous region of spectrally similar pixels. After selecting a region with either the marquee or lasso tool, right-click to reveal a pop-up menu with several options. Holding control while selecting ROIs allows you to append to the existing selection. Metric options are either Euclidean or Spectral Angle Mapper (SAM), and Tolerance is a unitless parameter that determines the sensitivity of Floodfill.

Default Image and Open Cubes pertain to opening existing datacubes. The options for Default Image are to open the image from your datacubes in a Red Green Blue (RGB) representation, or as a single grayscale image. The Open Cubes options are either to memory or disk. The memory option is faster, but may not work well for computers with limited memory.
When checked, the Warn on Unsaved option will provide you with a warning message that you have not saved your last scan when closing the software. Similarly, when clicked, the Last Scan option will provide you with a warning message that your previous scan will be overwritten by your next scan if you have not already saved it.

The Default Scan Image option allows you to set whether your scanned image is displayed as a Red Green Blue (RGB) image or as a grayscale image. This setting does NOT impact the hyperspectral data itself; it only changes how the image of the data is presented in the image panel.

The Recording option allows you to record your scans into computer memory or directly to disk. Recording to memory is faster and preferred unless your data acquisition computer does not have sufficient memory for your scans. Be sure to click on the Set button if you change this setting.

The Scale 100% Reflectivity to option allows you to record reflectance scaled to 1.0 (floats), 10,000 (integers), and Effective Bit Depth (integers). This process will scale the brightness of your reference sample to the option you choose. This saves your data as a standard reflectance value IF you use a reference sample with a reflectivity of 1. (Spectralon is a good material for this, and Teflon is a less expensive and reasonable substitute reference material, although less accurate.) If you use the 1.0 (floats) option, each channel of data will require 4 bytes instead of 2. Thus, the size of your datacube will double with Float as compared to not using this option.

The Scale Uncorrected Scan to option allows you to scale the raw data taken with your imager to Effective Bit Depth (integers) or 1.0 (floats). We recommend using the Effective Bit Depth unless you require the other option.

Note: The use of “uncorrected” above refers to the user not using the Dark Current (D) or Response Correction (R) tools prior to collecting a datacube. You would do this if you wanted to collect raw data and convert it to Radiance using radiometric calibration data provided by Resonon. Prior to converting, raw data is in digital number (DN).

5.3.3 Spectrometer Menu

The Spectrometer Main Menu button provides alternative approaches to useful actions, as well as some timesaving options.

Spectrometer → Detect Imager will prompt Spectronon software to detect an imager. This option allows you to detect an imager that perhaps was not plugged in when Spectronon was initially opened.

Spectrometer → Reload Imager option will restart the Pika imager without restarting SpectrononPro software. This option will save you time if you have, for example, accidentally disconnected your Pika spectrometer or crashed the software.

Spectrometer → Reload Stage will restart your scanning stage without restarting SpectrononPro software. This option saves you time if your stage needs to be restarted.

Spectrometer → Auto Exposure allows you to let the software set the exposure setting for your lighting conditions. To use this, place an object within the field of view of your Pika, and then click on Spectrometer → Auto Exposure.

Alternatively, you can use the Exposure button from the toolbar.
Spectrometer → Start Focus Tool option will provide you with a live view from your Pika imager in the image panel. This tool is useful for adjusting the focus of your objective lens or checking the illumination of your system. You can also start the focus tool by clicking on the Focus button on the SpectrononPro tools bar.

Spectrometer → Stop Focus Tool turns off the live view, which you need to do before you record a scan. The last live view will remain in a tab of the image panel. Another way to turn off the live view is to click the Focus button, which toggles live view off and on.

Spectrometer → Record Dark Current Cube is used to remove dark current noise from the imager. When you select this option you will be instructed to block the imager and then click OK. Another way to do this is select the Dark Current button on the SpectrononPro toolbar.

Spectrometer → Remove Dark Current Cube will remove your dark current calibration cube. This will also remove the red check from the Dark Current button.

Spectrometer → Set Dark Current Cube... allows you to import a saved dark current correction cube. This option is rarely utilized.

Spectrometer → Record Response Correction Cube allows you to use a reference material against which all measurements will be scaled. Typically this is done with a reference material whose reflectivity is approximately equal to 1. When you select this option, you will be instructed to place a reference material within the imager’s field of view, and then click on OK. A small reference cube will then be recorded and used for scaling the data to the bit-level of the camera. (E.g., for the Pika II, 12-bits, or 4096, and for the Pika NIR, 14-bits or 16,384.) If you have selected Normalize to Float, as described above, the data will be scaled to 1 rather than the bit level of the camera. Another way to perform this function is to use the Response Correction Cube button on the SpectrononPro toolbar.

Spectrometer → Remove Response Correction Cube will remove the existing response correction cube. This will also remove the red check from the ResponseCorrection Cube button.

The Set Response Correction Cube... allows you to import a saved reference cube. This option is rarely utilized, as one should generally record a correction cube regularly and with the current lighting to obtain accurate results.

Spectrometer → Start Recording and Spectrometer → Stop Recording will start and stop a scan with your Pika imaging spectrometer. This can also be done using the Scan button on the SpectrononPro toolbar. The Scan Button will toggle Start and Stop functions.
Spectronon provides visualization and manipulation capabilities for hyperspectral images. SpectrononPro software has all the features of Spectronon, but also enables data acquisition from Resonon’s family of imagers. Spectronon software can be downloaded for free on Resonon’s website http://www.resonon.com/. SpectrononPro comes bundled with any of Resonon’s imaging spectrometers. Additional analysis capabilities are available in software packages such as ENVI®.

This section begins with basic operation of Spectronon, such as opening a hyperspectral datacube and viewing the data. A complete description of visualization tools is provided in sections 7, 8, and 9 on Advanced Data Analysis. The Spectronon custom plugin manual discusses how to implement user-written algorithms into Spectronon, enabling custom data analyses.

### 6.1 Spectronon Tools

To open a datacube, select *File → Open Datacube.*

This will open a dialog that allows you to browse to find your datacube. Select your datacube and click on *Open* to load your datacube. This will result in the following:

- An image of your datacube will appear in the image panel
- A listing of the open datacube will appear in the Resource Tree
- Tabs will appear in the Parameters window that allow you to change the image (more on this later)
- Header information on your datacube will appear in the information panel
**Note:** Spectronon can open any datacube with an ENVI® formatted header. This includes .bip, bil, and .bsq formats.

This chapter employs an example datacube of M&M® and Reese’s® Pieces candies. (This datacube can be downloaded from Resonon’s website at [http://downloads.resonon.com/](http://downloads.resonon.com/).) By default, the datacube is opened with a true color image of the data, which approximates the appearance of the object under normal lighting conditions by combining red, green, and blue wavelengths from the datacube.

With a few minutes of practice using the available tools, you will be able to manipulate and visualize hyperspectral data quickly and efficiently.

### 6.2 Zoom, Pan, Flip, and Rotate Tool

To zoom to a specific area of the image, select the *magnify* tool in the toolbar and the cursor will change. Click the *magnify* tool in the image, and the view will zoom in. It is also possible to click and drag a selection within the image to zoom into the selected area.

To zoom out, select the *demagnify* tool and click anywhere in the image.
To zoom all the out and recover the original image, select the *original size* tool and click anywhere in the image.

The user may also zoom in and out using the mouse scroll wheel, if available.

To pan the image while zoomed in, select the pan tool. Click and drag inside of the Image to pan.

Click these tools to rotate left, rotate right, flip vertically, or flip horizontally the image.

### 6.3 The Inspector Tool – Spectral Plots

The *inspector* tool allows you to see the spectrum associated with a pixel. Choose the *inspector* from the toolbar, and then click a point inside the image. This will:

- Plot the spectrum for the pixel in the *spectrum plot panel*
- List the pixel location (sample and line number) in the *data panel*
- List the red (R), green (G), and blue (B) brightness values in the *data panel*
Click on other pixels to see the spectra from other pixels, click and hold while dragging the inspector tool to update the plot panel continuously.

The red, green, and blue vertical lines in the spectral plot indicate the hyperspectral wavelength bands used to generate the current image.

### 6.4 Region Of Interest (ROI) Tools

It is often useful to consider a group of pixels within the image. The ROI tools enable this capability and provide a number of options. As will be seen later, the ROI tool is often used during one of the first steps in classifying different objects within a hyperspectral image.

To select a Region of Interest (ROI), select either the marquee, lasso, or flood fill (wand) tool from the menu bar. Click and drag a rectangle of interest with the marquee tool, or click and drag any closed shape with the lasso. The floodfill tool can be used to select a contiguous region of spectrally similar pixels. After selecting an area, right-click to reveal a pop-up menu with several options. Holding control while selecting ROIs allows you to append to the existing selection.
Fig. 1: A small ROI on one of the red candies has been selected and a right-click has revealed the popup selection menu.

**Hint:** The *selection* menu is also available in the main menu.

The *floodfill* tool shows pixels that are spectrally similar to the chosen pixel. Spectral similarity is assessed with either Euclidean distance or Spectral Angle Mapper (SAM), along with a tolerance value. The user can set these options by accessing the *Workbench* tab from *File* → *Preferences* menu.
The use of the floodfill tool depends on an adjustable tolerance parameter. Floodfill operates on a representation of the datacube scaled from zero to one in each band. It calculates the Euclidean distance or SAM angle in spectral space between the clicked pixel and all contiguous pixels and expands the selection until the selected area contains all of the contiguous pixels for which the spectral distance to the clicked pixel is less than the selected tolerance. Increasing the tolerance will result in a larger selected region with greater spectral variability within that region (i.e. it allows pixels that are less similar to the clicked pixel to be included in the selection). Decreasing the tolerance will result in smaller selected regions with greater spectral similarity. As with the other selection tools, holding control while using the floodfill tool will allow a selection to be built up through multiple clicks of the tool.

An ROI consisting of multiple parts of the image has been selected by using the floodfill tool several times.

One of the most useful selection options is mean spectrum. (Descriptions for the other ROI options can be found in Basic Data Acquisition.) Selecting the mean spectrum option calculates the mean spectrum of all the pixels within the ROI area you selected and plots the result in the spectral plotter. Standard deviation can be shown as an envelope around the spectrum plot. Show or hide the standard deviation envelope with the Show/Hide Standard Deviation menu items.
Individual spectra can be selected by clicking on the graph of the spectra. You can crop an individual spectrum by selecting it, selecting a rectangular region in the plot window, right clicking to bring up the menu, and selecting crop spectrum. You can also set the range of the plot by selecting Plots → Spectral Plotter → Set Range.

**Hint:** To examine the plots in more detail, you may resize the plot panel boundary by dragging the edges. Alternatively, click on the magnify tool, then click or drag in the spectral plotter to zoom in. The pan tool will allow you to pan within the spectral plotter as well.

Selecting Mean Spectrum creates a new entry in the resource tree under a new heading, spectra. Right-clicking on the spectrum in the resource tree or selecting Spectrum from the menu bar will reveal a menu of options. Some of the most used options are listed below.
Save Spectrum

Change the name and save the spectrum as a file.

Set Label Color

Open a color picker dialog to change the label color of the spectral plot, and as shown later, classification areas based on this spectrum.

Show Region

Show the originally selected area for the ROI in the current image. This option is useful after you have selected several ROIs.

The color and transparency of the selection can be modified by selecting File → Preferences. The selection options are located in the workbench tab of the preferences window. A selection transparency of 1 represents a completely transparent selection (the selection will not be visible), while a selection transparency of 0 represents a completely opaque selection (the underlying render of the datacube will not be visible through the selection).
6.5 Image Visualization

Hyperspectral data can be visualized in far more ways than conventional color images. Image controls are provided in the tool control panel.

By default, the image is displayed in *True Color*, which means three representative bands are used to generate a red-green-blue (RGB) image, approximating how it appears to a human eye. The *Color Infrared* preset option provides a false-color RGB image with the red band set to an infrared wavelength. This option is useful for live vegetation datacubes.

Any time you wish to restore the image to true color, simply click on the *True Color* button under *Presets*. 
To generate false color images, use the sliders or arrows to change the wavelength bands used to create the RGB Image. This tool is often useful when trying to visualize specific spectral features associated with an object in your image. If Auto Update is not selected, click Update to generate the new image.

The Mode menu allows you to identify the band by wavelength (typically the most useful), or by band number.

As an example, of how false-color images can reveal interesting features, move the red slider to approximately 593 nm, and the green slider to approximately 516 nm, then click Update. This generates a new false-colored image, shown below, that reveals there are actually two kinds of red candy, and suggests there are two kinds of yellow candy – each candy type is positioned in the shape of an “I”. In Section 8 a classification technique shows this more clearly.
Note: The Red, Green, and Blue vertical lines in the Spectral Plot show the location of the bands chosen to create the false-color RGB image.
The *Contrast* tab in the tool control panel allows you to adjust the image contrast. If *Use Contrast Enhancement* is not checked, no image enhancement will be done and the tools in the Contrast tab will be not be active.

**Note:** Contrast enhancement does NOT change the hyperspectral data. It only changes the way the image appears.

Generally, contrast enhancement is beneficial. The 2% stretch is the default, and it sets the darkest 2% of the pixels in the image to a value of 0, and the brightest 2% of the pixels in the image to maximum brightness (255). This choice minimizes the impact of glare. You can customize the percentage of the dark pixels set to 0 and the percentage of the bright pixels set to 255 with the sliders. The Linear stretch option sets these percentages to zero.

The *Inverse* checkbox is useful if you wish to highlight dark pixels.

The *Individual Bands* checkbox controls whether the brightness levels of the three image layers are considered all together or as individual layers.
It is often useful to view a single band in a standard grayscale (black-and-white) image to visualize the impact of a single spectral feature. To do this, go to the main menu and select **Datacube → New Image → Greyscale**.

The controls are similar to the RGB controls. If **Auto Update** is not checked, be sure to click **Update** after moving the slider to see the grayscale image for a new band.

As with the RGB images, a vertical line in the Spectral Plot shows the band you have chosen. Note that even though the image is from a single band, the Inspector and ROI Tools will continue to plot and operate on all wavelengths.
**Hint:** With *Auto Update* selected in the tool control panel, you can quickly scroll through single band images.

**Warning:** For large datacubes or slow computers, the *Auto Update* refresh rate may be slow.

### 6.6 Plot Panel

The plot panel allows you to visualize hyperspectral data graphically. This has already been seen with the use of the Inspector and ROI tools, but here we explore the plot panel in more detail.

The plot panel has three tabs: *Spectra*, *X*, and *Y*. These three tabs provide you with plots along the three axes of a datacube using the Inspector Tool, as shown below.

Clicking on the *X* and *Y* tabs in the plot panel accesses the corresponding cross-sectional plots. The plot will show the intensity versus position value for the RGB bands used to create the image or the Grayscale band if used with a grayscale image.

**Note:** The direction of *X* and *Y* depends on the orientation of your cube. Moving the Inspector Tool should reveal which axis you are plotting.
6.7 Saving Spectra, Plots, and Images

Spectronon makes it easy to save the results of your work for further investigations or for making presentations.

To save a spectrum, click the spectrum you wish to save in the Resource Tree, and then you may either (1) use the Spectrum menu in the main menu, or (2) right-click on the spectrum in the Resource Tree to reveal the menu shown below. From this menu, select either Save Spectrum or Save Spectrum As... This will open a save dialog. Once saved, the new name will appear when the file is plotted in the spectral plotter, and the file can be re-opened for use in later sessions.

Select the menu option Copy Spectrum As Text to copy the data onto your clipboard, from which you can paste it into other applications such as Notepad and Excel.

To save a plot use the Plots menu as shown above. Select which plot you wish to save (Spectral Plotter, X Cross Section, or Y Cross Section), and then select Save as Image to save as an image or Save as Text to save the plotted data as tables in text file. Both options will pop up a save Dialog.
To Save an Image, select *Image* from the main menu, and then *Export Image*… This will pop up a save dialog.
Spectronon’s data visualization and analysis tools are presented in this section, following the structure of the main menu.

### 7.1 File Menu

*File → Open Datacube…*

Open an ENVI® compatible datacube file. The datacube is added to the resource tree and an image is generated in the image panel.

**Hint:** Dragging a .bil, .bip or .bsq datacube file into Spectronon will open it.

*File → Open Spectrum…*

Opens a saved spectrum file.

**Hint:** Dragging a .spec spectrum file into Spectronon will open it.

*File → Preferences…*

Opens a window that allows you to set a variety of preference options. Options associated with data visualization can be found under the Plotters and workbench tabs at the top of the window.
The *Plotters* tab opens a window that allows you to control the presentation of the plots presented in the spectral plotter. Additionally, this window also allows you to set the parameters for exporting plot data so it can be manipulated or plotted using other software tools.

The *workbench* tab opens a window that allows you to set the default image preference. One of the most useful settings is *RGB*, which generates a Red-Green-Blue image based on the values of three chosen hyperspectral bands, which you can choose in the tool control panel. Selecting the *True Color* button in the tool control panel produces an image that approximates the colors you would see looking at the object. Adjusting the sliders allows you to generate false-color images. Be sure to click *Update* after adjusting the sliders. *Greyscale* is another useful option, which presents a greyscale (black and white) image based on a single hyperspectral band. Again, you can adjust this band using sliders in the tool control panel. Be sure to click *Update* after moving a slider.

**Hint:** Check the *Auto Update* box in the tool control panel so you do not have to continually click *Update*.

*File* → *Built in Scripts*

Spectronon enables you to run scripts to best suit your application. A variety of scripts are built in to Spectronon.

*File* → *Run Script*...

You may also run Python scripts you write. *File* → *Run Script* opens a window that allows you to browse to your script.

*File* → *Batch Processor*...

For information on Batch Processing your data go to Section 10: Batch Processing.

*File* → *Mosaic Tool*...

The mosaic tool is designed for an airborne application and requires georegistered data to operate. This allows the user to create one image by merging several individual images from a raster dataset.

*File* → *Reload Plugins*

When making changes to user-written plugins, clicking this will make Spectronon aware of the changes.

*File* → *Exit*

Closes Spectronon.

### 7.2 Datacube Menu

#### 7.2.1 New Image

Allows you to create a new image in the Image panel. The items in this submenu are implemented as plugins. See Section 12.2: Render Plugins for full documentation.

The most generally useful options are RGB, Greyscale and Stack.
Datacube \rightarrow New Image \rightarrow RGB

Generates a color image based on three bands of your hyperspectral data. These bands can be chosen and adjusted using sliders in the tool control panel. See Section 12.2.7: RGB.

Datacube \rightarrow New Image \rightarrow Greyscale

This produces a single-band image from a single hyperspectral band that can be chosen using a slider in the tool control panel. See Section 12.2.4: Greyscale.

Datacube \rightarrow New Image \rightarrow Stack

This tool enables you to overlay multiple images. This tool is particularly useful for presenting classification results. For example, we may have a True Color RGB image of an object, such as shown below, as well as a classification map of one of the candy types. Using the Stack tool, we can combine these images to show the classification map on top of the RGB True Color image.
The stack tool opens a Stack tab in the tool control panel. Use the slider to select how many images you wish to combine (Stack height). Then click Update. Pull-down menus appear that allow you to select the images you wish to combine. A slider (Alpha) is available for each image that allows you to set its transparency in the combined image. Once set, click Update and your combined image will appear.

See Section 12.2.10: Stack.
7.2.2 New Cube

Datacube -> New Cube allows the user to create a new datacube from an existing datacube. The items in this submenu are implemented as plugins. See Built-in Plugins: Cube Plugins for full documentation.

Utilities

Allows you to generate a new, modified datacube from the currently open datacube. A few of the more commonly used utilities are explained below. See Section 12.1.1: Utilities for full documentation.

Datacube -> New Cube -> Utilities -> Bin Channels

Generates a new cube by binning spectral and/or spatial channels, which is often useful to either reduce the size of the datacube or improve the signal-to-noise ratio. This choice creates a new image in the Images Window and also generates a Bin Cube tab in the tool control panel with 3 sliders. The Sample Bin and Line Bin allow you to bin pixels along a spatial (x,y) axis. The Sample axis refers to the cross-track axis of the imaging spectrometer, and the Line axis refers to the along-track axis of the imaging spectrometer. The Spectral Bin slider allows you to bin spectral channels, and is likely the most useful of the binning options. After clicking Update, a new binned datacube is generated.

Note: This is a summation, not an average. The datatype of the returned cube may be promoted as required. If you choose Float Mode floating-point data will be returned.

See Section 12.1.1: Bin Channels.

Datacube -> New Cube -> Utilities -> Crop Wavelengths

Generates an image in the Image panel and a new tab with sliders in the tool control panel that allow you to crop wavelength bands by choosing a new minimum and maximum wavelength within the cube. Once chosen, click the Update button.
**Hint:** You may want to click on the RGB tab in the tool control panel to reset the bands used to create the image after cropping wavelengths.

See Section 12.1.1: Crop Wavelengths.

**Datacube → New Cube → Utilities → Subtract Spectrum**

Generates a new cube by subtracting a background spectrum from all pixels in the datacube. This option is useful, for example, if you are monitoring fluorescent dyes and wish to subtract the background fluorescence of the substrate. See Section 12.1.1: Subtract Spectrum.

**Classify**

Allows you to generate classification maps of different objects within your hyperspectral data. More information on hyperspectral classification can be found in Section 8: Advanced Data Analysis 2: Hyperspectral Classification. Also see Section 12.1.2: Classify for full classification plugin documentation.

Some of the classification algorithms (SAM, Euclidean Distance, Spectral Unmix) use Spectrum objects as inputs, while the statistics-based classifiers (Logistical Regression, Quadratic Discrimient, Support Vector) require datacubes of the classification target. These are made by selecting the ROI with the lasso or marquee tool, then selecting Create Cube from Selection.

When you select one of the classification options, (e.g. SAM), a new cube will be generated in the datatree panel. In many plugins, you will need to specify how many Layers you wish to classify (i.e., how many materials you wish to classify) – you will then need to select a Spectrum or Cube for each of these Layers in a pull-down menu. The Spectra and Cubes available in the pull-down menu can be generated either by using one of the ROI tools, marquee or lasso (or ) or by loading a previously saved spectrum or cube.

**Note:** a detailed discussion of hyperspectral data classification is beyond the scope of this document. Spectronon provides some of the more commonly used algorithms. For more advanced algorithm capability, please see other packages such as ENVI®. Additionally, custom algorithms can be utilized with Spectronon using the custom plugin capability described in the Spectronon custom plugin manual.

**Datacube → New Cube → Classify → Binary Encoding Classification**

7.2. Datacube Menu
The binary encoding classification technique encodes the data and endmember spectra into zeros and ones, based on whether a band falls below or above the spectrum mean, respectively. See Section 12.1.2: Binary Encoding.

**Datacube → New Cube → Classify → Euclidian Distance**

This is a commonly used classification algorithm for hyperspectral data that is more sensitive to pixel brightness than SAM. See Section 12.1.2: Euclidean Distance.

**Datacube → New Cube → Classify → Spectral Angle Mapper (SAM)**

The SAM classification routine is described in detail in Section 8.2: Spectral Angle Mapper (SAM) Classification. Also see Section 12.1.2: Spectral Angle Mapper.

**Datacube → New Cube → Classify → Spectral Unmix**

Spectral unmixing deconvolves the signal in each pixel into a linear combination of known spectra. See Section 12.1.2: Spectral Unmix.

**Analyze**

**Datacube → New Cube → Analyze** allows you to perform useful operations on hyperspectral data that will generate new datacubes based on applying analytical functions to your datacube. A few of these are documented here. See Section 12.1.3: Analyze for full documentation.

**Datacube → New Cube → Analyze → First Derivative**

This tool generates a new datacube with the first derivative of the spectral curve for each pixel. A new image of the first derivative datacube is presented in the Image panel. To see the spectral derivatives, utilize the inspector tool, or use one of the ROI tools marquee or lasso. The first derivative curves appear in the spectral plotter. See Section 12.1.3: First Derivative.

**Datacube → New Cube → Analyze → Principal Component Analysis (PCA)**

A detailed discussion of PCA is beyond the scope of this document (see, for example, Wikipedia for a discussion). This tool generates a new datacube and image with the principal component values for each pixel. Additionally, a PCA tab appears in the tool control panel with a slider that allows you to select the number of Bands, or PCA components. To see the PCA component magnitudes, utilize the inspector tool, or use one of the ROI tools marquee or lasso.
The PCA magnitude curves appear in the spectral plotter. As with standard hyperspectral data, classification algorithms can be performed on the new PCA datacube. See Section 12.1.3: PCA.

**Correct**

 Allows you to correct / calibrate a datacube from a measured reference. This tool also allows you to convert your data to either radiance or reflectance values, as described below. See Section 12.1.4: Correct for full documentation.

---

**Datacube → New Cube → Correct → Radiance From Raw Data**

Converts raw data to units of radiance (microflicks) based on the instrument’s radiometric calibration file (.icp). See Section 12.1.4: Radiance From Raw Data.

**Datacube → New Cube → Correct → Reflectance from Radiance Data and Downwelling Irradiance Spectrum**

Converts radiance data to reflectivity based on a downwelling irradiance spectrum collected with a single point spectrometer and cosine corrector. See Section 12.1.4: Reflectance from Radiance Data and Downwelling Irradiance Spectrum.

**Datacube → New Cube → Correct → Reflectance from Radiance Data and Measured Reference Spectrum**

This approach to converting data to reflectance assumes the input datacube has been corrected spatially, typically by converting to radiance but other approaches would be acceptable. A spectrum of measured reference material is used to correct the datacube spectrally. See Section 12.1.4: Reflectance from Radiance Data and Measured Reference Spectrum.

**Datacube → New Cube → Correct → Reflectance from Radiance Data and Spectrally Flat Reference Spectrum**

This approach assumes the input datacube has been corrected spatially, typically by converting to radiance but other approaches would be acceptable. A spectrum of spectrally flat reference material (typically Spectralon™ or Fluorilon™) is used to correct the datacube spectrally. See Section 12.1.4: Reflectance from Radiance Data and Spectrally Flat Reference Spectrum.

**Datacube → New Cube → Correct → Reflectance from Raw Data and Downwelling Irradiance Spectrum**

Converts raw data to reflectivity based on a downwelling irradiance spectrum collected with a single point spectrometer and cosine corrector. See Section 12.1.4: Reflectance from Raw Data and Downwelling Irradiance Spectrum.

**Datacube → New Cube → Correct → Reflectance from Raw Data and Measured Reference Cube**
Convert raw data to reflectance based on a input datacube of measured reference material. See Section 12.1.4: Reflectance from Raw Data and Measured Reference Cube.

Datacube → New Cube → Correct → Reflectance from Raw Data and Spectrally Flat Reference Cube

Converts raw data to reflectivity based on another datacube of a spectrally flat reference material. See Section 12.1.4: Reflectance from Raw Data and Spectrally Flat Reference Cube.

**Anomaly**

Anomaly detection tries to determine anomalous outliers in a datacube that do not conform to the expected spectra. This is usually based on a datacube of ‘clutter’, which contains the expected spectra.

See Section 12.1.5: Anomaly for full documentation.

**Color**

Allows you to transform your hyperspectral data into CIE colorspace, providing XYZ, xyY, and LAB values for each pixel. Additionally, you can determine the $\Delta E$ values for each pixel as compared to a standard set by the user.

See Section 12.1.6: Color for full documentation.

**Agricultural**

Create maps of Hyperspectral Vegetation Indices (HVI’s). HVI’s in Spectronon are discussed in greater depth in Section 9. Also see Section 12.1.7: Agricultural.

**Clustering**

Clustering tools are unsupervised classification routines that do not require any a priori information of a spectral image. They group the scene into clusters based on spectral similarity. Supervised classification routines can almost always produce better results, if scene knowledge available. See Built-in Plugins: Cube Plugins - Clustering for full documentation.

**7.2.3 Datacube Menu Misc.**

Datacube → Save Cube

Saves the selected datacube.

Datacube → Save Cube As

Saves the currently open datacube under a new name or location.

Datacube → Rename Cube

Renames the selected datacube.

Datacube → Close Cube

Removes the selected datacube from the Resource Tree.
7.3 Image Menu

Tools for saving or closing images generated from a datacube.

*Image → Export Image*

Exports an image as TIFF, PNG, BMP, JPG, or GIF.

*Export Image as KML*

Exports a georectified image in Keyhole Markup Language (KML) format.

*Export Image as GeoTiff*

Exports a georectified image as a TIFF file.

*Copy Image to Clipboard*

Copies an image to the Windows clipboard.

*Close Image*

Closes the selected image.

7.3.1 Filter

Changes the filter applied to a generated image. Filters alter the image presentation in some way. Spectronon usually chooses the most appropriate filter automatically. Most commonly this is the *contrast* filter.

All filter menu items are implemented as plugins. See Section 12.3: Filter Plugins for full documentation.

7.4 Spectrum Menu

The Spectrum menu provides tools for manipulating individual spectra.
**Hint:** If the Spectrum menu is greyed out, verify that a spectrum is selected in the resource tree.

*Spectrum → Save Spectrum*

Saves the spectrum selected in the Resource Tree.

*Spectrum → Save Spectrum As*

Saves a spectrum in a file name of your choice.

*Spectrum → Export Spectrum as Text File*

Saves a spectrum as a text file.

*Spectrum → Rename Spectrum*

Renames a spectrum.

*Spectrum → Integrate Spectrum*

Computes the area under the spectral plot for a given spectrum.

*Spectrum → Convert Spectrum To Irradiance*

Converts a spectrum to irradiance. A downwelling calibration pack is required.

*Spectrum → Close Spectrum*

Closes the spectrum selected in the Resource Tree and removes the associated plot from the spectral plotter.

*Spectrum → Show Plot*

Plots the spectral curve of the selected spectrum in the spectral plotter.

*Spectrum → Hide Plot*

Hides the spectral curve plot in the spectral plotter.

*Spectrum → Show Standard Deviation*

Shows the standard deviation envelope in the spectral plotter.

*Spectrum → Hide Standard Deviation*

Hides the standard deviation envelope in the spectral plotter.

*Spectrum → Set Label Color*

Changes the color of the selected spectrum’s spectral curve and generated classification maps.

*Spectrum → Show Region*

Highlights the datacube region used to generate the selected spectrum.

*Spectrum → Copy Spectrum as Text*

Copies a spectrum’s spectral data to the Windows clipboard.

*Spectrum → Crop Spectrum*

Select a spectrum, then select a region in the spectra window, and then click Spectrum → Crop Spectrum. The spectral plot will be cropped to the chosen range.
7.5 Selection Menu

The Selection menu items provide a number of options for use with the selection tools (\(\text{\textcircled{1}}, \text{\textcircled{2}}, \text{\textcircled{3}}\)). For all of the following options, you must first select a Region of Interest (ROI).

<table>
<thead>
<tr>
<th>Selection Menu Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Selection → Select All</td>
<td>Selects the entire dataset.</td>
</tr>
<tr>
<td>Selection → Select None</td>
<td>Clears the current selection.</td>
</tr>
<tr>
<td>Selection → Create Cube From Selection</td>
<td>Creates a new datacube from the current selection.</td>
</tr>
<tr>
<td>Selection → Mean Spectrum</td>
<td>This tool will calculate and plot the mean spectrum for the pixels selected in your ROI.</td>
</tr>
<tr>
<td>Selection → Show Mean of Selection</td>
<td>Shows a dialog box with the mean of each band in the selected region.</td>
</tr>
<tr>
<td>Selection → Mean Z profile</td>
<td>Create and plot the mean profile of the selected region in the Z (band) dimension.</td>
</tr>
<tr>
<td>Selection → Mean First Derivative</td>
<td>Plot the mean first derivative (discrete difference) of the selected area.</td>
</tr>
<tr>
<td>Selection → Spectral Correlation Matrix</td>
<td>Creates a heatmap showing band-to-band correlations using the Pearson coefficient.</td>
</tr>
</tbody>
</table>

Note: The Selection menu can be accessed from the main menu or by right-clicking in the image after creating an ROI.

Note: Some of these menu items are implemented as selection plugins. See Section 12.4: Select Plugins for references.
A correlation coefficient aligns reflectance data created with downwelling irradiance data to a ground truth target such as a tarp or reflectance standard. Once created, these coefficients are used in the Reflectance Conversion with Downwelling Irradiance plugins to improve the accuracy of the result. See Section 12.4.6: Create Correlation Coefficients.

Selection → Find Closest RAL Classic Color

Uses spectral information to determine the closest RAL Classic color to selected region. Wavelength and reflectance scale factor metadata must be present. Useful for color matching. See Section 12.4.7: Find Closest RAL Classic Color.

Selection → Median Spectrum

Plot the median spectrum within selected area. See Section 12.4.8: Median Spectrum.

Selection → Mean Nth Derivative

Compute and plot the mean Nth derivative (discrete difference) of the selected region. See Section 12.4.9: Mean Nth Derivative.

Selection → Save random spectra to text file

Randomly selects pixels within the region of interest and exports their spectra (or bands) to a text file. See Section 12.4.10: Save random spectra to text file.

7.5.1 Send to Clipboard

This tool allows you to transfer the data within your selected ROI. The first option is Copy Mean as Text, which allows you to paste the mean spectrum values into other applications such as Notepad or Excel. The second option, All Spectra as Text copies the spectral curves of all the pixels within your selected ROI. This tool is useful for those who want to perform specific statistical analysis on the data from small regions.

Selection → Send to Clipboard → Copy Mean as Text

Copies mean spectrum as text to clipboard for pasting into other applications. See Section 12.4.11: Copy Mean as Text.

Selection → Send to Clipboard → All Spectra as Text

Copies all spectra in region as text to clipboard for pasting into other applications. See Section 12.4.11: All Spectra as Text.

7.6 Plots Menu

The Plots menu items provide tools for the data shown in the plot panel.

Plots → [plot type] → Set Range

Set X-axis and the Y-axis ranges.

Plots → [plot type] → Save as Image
Save a plot to an image file.

Plots → [plot type] → Save as Text

Save plot data values as a .txt file.

Plots → Spectral Plotter → Clear

Clears the spectral plot.

Plots → [plot type] → Copy to Clipboard as Text

Copy plot data to the Windows clipboard as text.

7.7 Datacubes and Header Files

Datacubes collected with Resonon imaging spectrometers require two files, one that contains the hyperspectral data, and also a header file that contains important information about the data. Hyperspectral data can be accessed with many tools, for example, Matlab and ENVI, and is compatible with a variety of languages including: Python, C++, Fortran.

7.7.1 Hyperspectral data files

ENVI-compatible hyperspectral data can be stored in the following three formats:

**Band-Sequential** In Band-Sequential (BSQ) format each line of the data is followed by the next line in the same spectral band. Hyperspectral data stored in BSQ format will end with .bsq.

**Band-Interleaved-by-Line** Data saved in Band-Interleaved-by-Line (BIL) format have the first line of the first band followed by the first line of the second band, followed by the first line of the third band, and so forth. This pattern is repeated for subsequent lines. Hyperspectral data stored in BIL format will end with .bil.

**Band-Interleaved-by-Pixel** Band-Interleaved-by-Pixel (BIP) data have the first pixel for all bands in sequential order, followed by the second pixel for all bands, and so forth for all pixels. Hyperspectral data stored in BIP format will end with .bip.

Resonon imaging spectrometers will save hyperspectral data in BSQ, BIL, or BIP formats, and Spectronon software will open hyperspectral data in all of these formats.

7.7.2 Hyperspectral data header files

Header files for hyperspectral data have the same name as the BSQ, BIL, or BIP files followed by .hdr. The following discussion provides an overview of header files.

Header files can be opened and edited using WordPad. The following list describes some, but not all, of header file commands. Bold indicates the command, quantities in (brackets) are acceptable arguments for the commands, and statements in [square brackets] are additional explanation. Note that the order of these commands can be different from file to file.

**ENVI** [All ENVI-compatible header files begin with this]

interleave = (bil, bip, bsq) [Tells the software the hyperspectral data format]

data type = (4, 12) [4 is 4-byte floating point; 12 is 2-byte unsigned; other data types are also supported, but these are common]

lines = (###) [The number of lines in the hyperspectral data where ### is a number]
samples = (###) [The number of samples in the hyperspectral data where ### is a number. This is typically the number of cross-track spatial pixels of the hyperspectral imager.]

bands = (###) [The number of bands in the hyperspectral data where ### is a number]

bit depth = (12, 14) [The bit depth of the hyperspectral data]

shutter = (###) [The shutter time in units of msec where ### is a number. This is mostly for reference, but it may be used by certain plugins]

gain = (#.#) [The camera gain in dB where #.# a number. This is mostly for reference, but it may be used by certain plugins]

framerate = (####) [The framerate at which the data were recorded in frames/sec where ### is a number. This is mostly for reference, but it may be used by certain plugins]

reflectance scale factor = (4095, 1023, 1) [If you divide the data by this factor, the data will scale from 0 to 1, typically for reflectance data]

byte order = (0) [byte order is the ordering of byte significance (http://en.wikipedia.org/wiki/Endianness) Data for Spectronon are LSF or Little Endian which means the least significant bytes come first. Spectronon will not properly open BSF or Big Endian files]

header offset = (0) [This is the number of bytes at the beginning of the binary file that should be ignored].

wavelengths = (###, ###, ###, . . . ) [List of the spectral band centers in units of nm]

rotation = ((#,#), (#,#), (#,#), (#,#)) [This is a default orientation of the view of a datacube in Spectronon. This is a Spectronon only header option].

cube file = (name) [The label is used by Spectronon to give a human readable alternative to the file name of a file. If omitted, Spectronon will use the file name]

timestamp = (day & time of datacube) [If available, this is when the datacube was recorded. The timestamp is often found on airborne datacubes, and datacubes where a script entered this information explicitly into the header]

The following are for spec.hdr files only, headers for spectrum. Spectrum files are ENVI compatible because they are 1x1 datacubes, but Spectronon adds in these data for its own use.

original cube file = (name of original cube) [This is a Spectronon-only header item for recording the history of a generated spectrum. In the case of a derived datacube, you will see a “history” header value that shows the way this cube was generated]

pointlist = (long list of values) [This is the set of points from the original cube that were averaged together to make this spectrum if the spectrum is a mean]

label color = (#FF00FF) [This is the color used to plot the spectrum]
8.1 General Approach

The detailed spectral information in hyperspectral data enables one to distinguish between very similar objects. Additionally, machine vision algorithms enable rapid, accurate, and repeatable classification of objects. With these capabilities, hyperspectral imaging has a broad range of current and potential applications, including sorting (food, raw materials, recycled materials), quality control (pharmaceuticals, food, printed goods), remote sensing (defense, search & rescue, mineral exploration, agriculture), to name just a few.

Critical to all of these applications is implementation of algorithms that classify the pixels within an image based on their spectral profiles. This section provides an introduction to hyperspectral data classification. There are many approaches to classifying objects. However, the general approach to most of these algorithms can be understood by considering the following simplified hypothetical examples.

In general, the algorithms used to classify objects scale with the number of spectral channels. Understanding how the general approaches work with a simple 2-color system makes it relatively easy to understand hyperspectral classification with tens or hundreds of channels. Therefore, consider a 2-color camera that provides digital numbers for how much red and how much blue is in each pixel. (Thus, this example is even easier than a conventional digital color camera that provides digital values for the three colors: red, green, and blue.)

Much like humans need to learn to distinguish between objects, classification algorithms generally also must be trained. Typically this is done by imaging samples of interest, and then using results from this “training set” to learn how to distinguish between objects in general.

As an example, assume we wish to distinguish between the three objects shown below, a red ellipse, a blue triangle, and a purple (mixture of red and blue) parallelogram. (Ignore the grey rectangles for now.) An x and y axis is drawn to provide location coordinates for each pixel.

Note that each object has a distribution of light and dark pixels, although each object is approximately the same “color”. If we image these objects with our hypothetical 2-color imager that senses only “red” and “blue” channels, there will be two spectral channels per pixel. To train our imaging system, we first select a representative set of training
pixels from the image of each of the objects of interest. This might be done, for example, by selecting the pixels within each grey rectangle indicated in the image above.

A useful way to visualize the color information in these training pixels is to plot the red and blue brightness values for each selected pixel in “color space” along blue and red axes, as shown below. (Note: This color plot uses only the training pixels selected within the three grey rectangles shown in the image above.)

![Color space plot](image)

We can immediately recognize the training pixels that align along the vertical (red) axis are associated with the red ellipse, as those pixels clearly have large red brightness values, but small blue brightness. Similarly, the blue triangle pixels lie primarily along the horizontal (blue) axis. The purple pixels, however, lie in between the two axes because the color purple has significant red and blue brightness (i.e., it is a mixture or red and blue). Because some of the pixels are dark and others are light, the distribution of training pixels from each object is spread out from small values to large values in a near-linear manner.

One can see that the pixels from the spectrally distinct objects are separated in this “color space”. Although this example has a 2-dimensional color space, one could create a 3-dimensional color space for RGB cameras, or 100 dimensions for 100-band hyperspectral imagers. In general, additional dimensions provide additional “opportunities” for points from different objects to be distinct in the color space, and thus they are easier to classify in practice, but otherwise the number of dimensions need not concern us now. Admittedly it is difficult to visualize a 100-dimension space, but the mathematical transition is often straight-forward. Consequently, one can invent techniques that work in two dimensions, and they can generally be applied to 100 dimensions.

Consider the following example of the well-known hyperspectral classification algorithm known as Spectral Angle Mapper (SAM). Note that the groups of training pixels from the three different objects in our 2-dimensional color plot are located at different angles relative to the horizontal axis. The SAM technique utilizes this property to classify ALL pixels.

Consider a reference vector at the center of the group of purple training pixels. This vector is shown as a large orange vector below. Similarly, we can envision vectors from the origin to each pixel in the plot (only three are shown to reduce clutter).
One can see that the angle between the reference orange vector is small for the purple pixels, and relatively large for the red and blue pixels. Thus, if we calculate the angle between the reference orange vector and all pixels in the image, we recognize that those pixels with a small angle are from the purple parallelogram, and those pixels with a large angle from the orange reference vector are not purple pixels.

Fortunately there is an easy way to calculate the angle between two vectors by utilizing the vector dot product. To do this, write the orange reference vector in component form as

\[ \vec{R} = (R_r, R_b) \]

where the subscript \( r \) indicates the red brightness value component and \( b \) indicates the blue brightness value component. Similarly, for all pixels in the image, write

\[ \vec{P}(x, y) = (P_r(x, y), P_b(x, y)) \]

where \( x \) and \( y \) indicate the location of the pixel in the original image with an ellipse, triangle, and parallelogram. The vector dot product of \( \vec{R} \) and any pixel \( \vec{P}(x, y) \) is

\[ \vec{R} \cdot \vec{P}(x, y) = R_r P_r(x, y) + R_b P_b(x, y) = |\vec{R}| |\vec{P}(x, y)| \cos(\theta(x, y)) \]

where \( |\vec{R}| \) and \( |\vec{P}(x, y)| \) are the magnitudes of \( \vec{R} \) and \( \vec{P}(x, y) \). E.g., \( |\vec{R}| = [R_r R_r + R_b R_b]^{1/2} \) and \( \theta(x, y) \) is the angle between the reference vector \( \vec{R} \) and the pixel vector \( \vec{P}(x, y) \).

Solving for \( \theta(x, y) \) yields

\[ \theta(x, y) = \cos^{-1} \left( \frac{\vec{R} \cdot \vec{P}(x, y)}{|\vec{R}| |\vec{P}(x, y)|} \right) \]

By choosing only those pixels that have an angle \( \theta(x, y) \) less than some small threshold value, one can effectively identify all the purple pixels. Graphically, this is equivalent to choosing only those pixels within a narrow green cone, as shown below.
By finding the orange reference vector and establishing the threshold of acceptable angles from this vector, we have “trained” our imaging system. To identify objects with the same color as the purple parallelogram, one images the objects and then calculates the angle as described above for each pixel. Pixels with an angle smaller than the threshold are identified to be the same material (color) as the purple parallelogram.

One could also train the system to find pixels the same color as the red ellipse or blue triangle by finding the appropriate reference vector for these objects, thereby enabling one to identify multiple objects within one image by calculating the angle between multiple object reference vectors.

To extend this approach to hyperspectral data with multiple spectral channels, one utilizes the more general definition of a vector dot product

\[
\vec{R} \cdot \vec{P}(x, y) = \sum_i R_i P_i(x, y) = |\vec{R}| |\vec{P}(x, y)| \cos(\theta(x, y))
\]

where the sum is over all spectral bands (dimensions) indexed by \(i\). The angle \(\theta(x, y)\) is still the angle between the two vectors, \(\vec{R}\) and \(\vec{P}(x, y)\), although now in a complex multi-dimensional color space that is more difficult to visualize – the mathematics and approach are identical.

Of course SAM is not always the best algorithm to use, as one can see with the example below, where pixels from two hypothetical objects we wish to distinguish between are shown as red and blue points. In this case, if one choose representative vectors in the middle of each cluster to perform SAM classification, it is easy to see there would be substantial misclassification. (E.g., two red points lie nearly along the blue representative vector.)
For the case shown above, a different algorithm is needed. One approach that would clearly be superior to SAM, at least for these data, is to find the center points of each set of training points, and then all points within a certain radius would be classified as that object. This is also a well-known hyperspectral classification algorithm known as Euclidean distance. Again, this approach is readily extended to hyperspectral data because the concept of distance between points is well established for multi-dimensional space.

The Euclidean distance approach can readily be improved upon by recognizing that a circle is not the best “enclosure”, and an ellipse whose axes were scaled to the point distribution width would be far better. The hyperspectral classification algorithm that utilizes this information on the distribution of training points is called the Mahalanobis distance approach. As one would expect, the cost of additional algorithm sophistication is often the need for additional processing.

There are, of course, many other approaches to classifying objects using hyperspectral data, such as the one indicated...
below where one draws a line between the two sets of training points. The system is trained by noting that all pixels that map to the left of the line are associated with one object, whereas the pixels that map to the right of the line are associated with the other. Extrapolating this approach to higher dimensions is more difficult, as the line becomes a plane in three dimensions, and a hyper-plane in color spaces with dimensions larger than three.

8.2 Spectral Angle Mapper (SAM) Classification

In this example, different kinds of M&M® and Reese’s® Pieces candies are classified using the same hyperspectral datacube shown in Section 6. (This datacube can be downloaded from Resonon’s website.) To perform SAM, reference spectra must be collected for the objects of interest. In this case, based on prior knowledge, we know that the four candies indicated below (see arrows in image) are all different. Using the marquee tool or lasso, select small ROIs on each of the four candies (avoid the glare spots) indicated in the figure below. After selecting the ROI, right-click, and then select Mean Spectrum. This will generate four spectral curves in the spectral plotter, and also list the four spectral curves in the Resource Tree, as shown below.
The four spectral curves shown in the spectral plotter will be used as the Reference Spectra to perform a SAM classification. To perform the classification, click on Datacube → New Cube → Classify → Spectral Angle Mapper (SAM).

This will open a new window as shown. We wish to classify 4 objects, so use the slider or arrow keys to select a Spectrum for each box.
This will bring up 4 Spectrum pull-down menus. Click into each one of these and select one of the 4 spectra created with the ROI tool.

Then click OK, and Spectronon calculates the spectral angle (as described above) for each pixel for all 4 Reference spectra.

Typically, several seconds are required for the calculation, which generates a new classification map in the image panel using a default set of Threshold values.
Adjust the Threshold values to obtain a more accurate classification by clicking on the Threshold to Colormap tab in the tool control panel.

To adjust the Thresholds, move the sliders, select the arrow keys, or enter values by hand. Each Reference spectrum has its own threshold. After adjusting a threshold, click Update and a new classification map will be generated in the Image panel. With a few tries, a classification rendering similar to the one shown below can be generated. Only those pixels within the threshold are colored (the classification colors match those shown in the spectral plotter and Resource Tree for the Reference spectrum). If a pixel’s Spectral Angle is within the threshold for more than one Reference spectrum, the quantity (Spectral Angle)/(Threshold Value) is calculated for each Threshold and classified for the class that minimizes this quantity. This weighting scheme allows you to emphasize or deemphasize each class to fine-tune your classification map.
SAM is only one of many, many possible classification algorithms. Other classification algorithms are accessed in a similar manner, as described in Section 12.1.2. These algorithms can be found by clicking Datacube → New Cube → Classify. Additionally, user-defined scripts can be written and used with Spectronon for custom classifications algorithms.
9.1 Introduction

Hyperspectral Vegetation Indices (HVI’s) are a common tool for analyzing hyperspectral data of agricultural crops. They are typically simple arithmetic combinations of sums and differences of spectral reflectance values at particular wavelengths. For example, the Structure Insensitive Pigment Index (SIPI) is defined by the following expression:

\[
SIPI = \frac{\rho_{800} - \rho_{445}}{\rho_{800} + \rho_{680}}
\]

where the values \(\rho_{445}\), \(\rho_{680}\), and \(\rho_{800}\) are shown graphically in the figure below:

HVI’s have found many applications in research using airborne remote sensing data, and are currently subjects of intense research efforts.
9.1.1 Useful References

The developers at ENVI have written an excellent description of HVI’s, which is available at their website. We at Resonon consider ENVI (owned by Harris Corporation) to be excellent hyperspectral data analysis software. Here is a link to their HVI website:

http://www.harrisgeospatial.com/docs/VegetationAnalysis.html

Another very good reference discussing HVI’s is this book:


9.1.2 HVI’s in Spectronon

HVI’s calculated by Spectronon are listed in the following table. Also listed are the equations that Spectronon uses for each HVI.
<table>
<thead>
<tr>
<th>Hyperspectral Vegetation Index</th>
<th>Equation in Spectronon</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anthocyanin Reflectance Index 1</td>
<td>$ARI1 = \frac{1}{\rho_{990}} - \frac{1}{\rho_{700}}$</td>
</tr>
<tr>
<td>Anthocyanin Reflectance Index 2</td>
<td>$ARI2 = \rho_{990} \left( \frac{1}{\rho_{990}} - \frac{1}{\rho_{700}} \right)$</td>
</tr>
<tr>
<td>Atmospherically Resistant Vegetative Index</td>
<td>$ARVI = \frac{NIR - (Red - \gamma (Blue - Red))}{NIR + (Red - \gamma (Blue - Red))}$</td>
</tr>
<tr>
<td>Carotenoid Reflectance Index 1</td>
<td>$CRI1 = \frac{1}{\rho_{570}} - \frac{1}{\rho_{690}}$</td>
</tr>
<tr>
<td>Carotenoid Reflectance Index 2</td>
<td>$CRI2 = \frac{1}{\rho_{570}} - \frac{1}{\rho_{700}}$</td>
</tr>
<tr>
<td>Enhanced Vegative Index</td>
<td>$EVI = \frac{NIR - Red}{NIR + 6.0 \cdot Red - 7.5 \cdot Blue + 1}$</td>
</tr>
<tr>
<td>Modified Chlorophyll Absorption Reflectance Index</td>
<td>$MCARI = \rho_{700} - \rho_{670} - 0.2(\rho_{700} - \rho_{550}) \left( \frac{\rho_{700}}{\rho_{670}} \right)$</td>
</tr>
<tr>
<td>Modified Red Edge Normalized Vegetation Index</td>
<td>$MRENDVI = \frac{\rho_{735} - \rho_{705}}{\rho_{750} + \rho_{735} - 2 \cdot \rho_{445}}$</td>
</tr>
<tr>
<td>Modified Red Edge Simple Ratio Index</td>
<td>$MRESR = \frac{\rho_{700} - \rho_{695}}{\rho_{700} - \rho_{445}}$</td>
</tr>
<tr>
<td>Normalized Difference Vegetative Index</td>
<td>$NDVI = \frac{NIR - Red}{NIR + Red}$</td>
</tr>
<tr>
<td>Photochemical Reflectance Index</td>
<td>$PRI = \frac{\rho_{531} - \rho_{770}}{\rho_{531} - \rho_{570}}$</td>
</tr>
<tr>
<td>Plant Senescence Reflectance Index</td>
<td>$PSRI = \frac{\rho_{680} - \rho_{690}}{\rho_{750}}$</td>
</tr>
<tr>
<td>Red Edge Normalized Difference Vegetation Index</td>
<td>$RENDVI = \frac{\rho_{730} - \rho_{700}}{\rho_{750} + \rho_{700}}$</td>
</tr>
<tr>
<td>Simple Ratio Index</td>
<td>$SR = \frac{NIR}{Red}$</td>
</tr>
<tr>
<td>Structure Insensitive Pigment Index</td>
<td>$SIPI = \frac{\rho_{680} - \rho_{445}}{\rho_{800} + \rho_{680}}$</td>
</tr>
<tr>
<td>Transformed Chlorophyll Absorption Reflectance Index</td>
<td>$TCARI = 3 \left[ (\frac{\rho_{730} - \rho_{670} - 0.2(\rho_{700} - \rho_{550})}{\rho_{670}}) \right]$</td>
</tr>
<tr>
<td>Vogelmann Red Edge Index 1</td>
<td>$VREI1 = \frac{\rho_{740}}{\rho_{720}}$</td>
</tr>
<tr>
<td>Vogelmann Red Edge Index 2</td>
<td>$VREI2 = \frac{\rho_{740} - \rho_{742}}{\rho_{715} - \rho_{720}}$</td>
</tr>
<tr>
<td>Vogelmann Red Edge Index 3</td>
<td>$VREI3 = \frac{\rho_{740} - \rho_{742}}{\rho_{715} + \rho_{720}}$</td>
</tr>
<tr>
<td>Water Band Index</td>
<td>$WBI = \frac{\rho_{570}}{\rho_{400}}$</td>
</tr>
</tbody>
</table>
9.2 Generating HVI maps in Spectronon

The datacube in the following example was acquired using a Resonon airborne hyperspectral imaging system. This datacube is available for free download from http://downloads.resonon.com/.

HVI’s in Spectronon can be found at the path: Datacube → New Cube → Agriculture:

Choosing an HVI from this menu will generate a grey-scale image:
The value of the HVI at any pixel can be obtained with the Inspector Tool:

![Inspector Tool Image]

The value of the HVI, along with the coordinates of the inspector tool, can be found in the lower left corner of the screen:
Another tool for visualizing HVI maps is to transform the grey-scale image into a colormap. This can be accomplished with Datacube → New Image → Scalar to Colormap:
Below is a “spectral” colormap of the MCARI index for the example datacube:

After generating a colormap, the user can change the color scheme from the Colormap menu on the Controls tab. It may be helpful to toggle the Contrast Enhancement feature. Experiment with different colormaps, normalization, and contrast settings, all available on the Controls tab.

**Note:** If an HVI is selected that requires spectral bands that do not exist in your datacube, the warning shown below will appear. As described in the warning, clicking “OK” will generate an image using the closest bands available, but the resulting image will often be a poor approximation to a true index mapping.
Manually running the same processing sequence on a series of datacubes can be a tedious process. Spectronon’s batch processor allows the user to choose a number of datacubes at once and set up a sequence of processing steps (plugins) to execute on the chosen files. This may be of particular interest to users of our Airborne product, but applies to any customer who needs process multiple datacubes at once.

To start the batch processor, select File → Batch Processor. The batch processing window appears.

Expand the directory tree on the left side of the window to find the datacubes you wish to process. Use the buttons in the middle of the screen to select the desired datacubes. Datacubes that will be processed appear on the right side of the window. In the figure above 3 datacubes are selected for processing.

When you are happy with your selection, click Ok. A new window appears to define the plugins you wish to run.
Use the directory tree to select the cube plugins you wish to run. Cube plugins generate new datacubes and will run in the order selected, as shown in the **Cube plugin sequence** panel. Each plugin operates on the output of the plugin before it. Each plugin’s setup window appears as you add it to the processing queue. In the figure above the “Airborne Radiance Conversion” plugin is set to run, followed by the NDVI plugin.

Select any render plugins you wish to run. Render plugins generate images from the final result of the cube plugin sequence, and appear in the **Renders** panel. You can choose to save renders as .png, .tiff or .jpg with **Render Export Format**. In the figure above no render plugins are selected, so no images will be generated.

Use the controls at the lower right to select the location to save results. **Copy LCF and times files?** and **Export render KMLs?** are useful for our airborne customers. If desired, the resultant datacubes can be added to the workbench to show in Spectronon when processing is finished.

Press the **OK** button to begin processing. A progress dialog appears.
Press the *Close* button to dismiss the dialog after processing is complete. The results are saved to the selected location.
FOCUSING & CALIBRATION SHEETS

Use the focusing sheets to focus the objective lens, and use the aspect ratio calibration sheet to set the stage speed and imager framerate. See the Basic Data Acquisition for details.

11.1 Small Focusing Sheet

Focusing Sheet
11.2 Large Focusing Sheet

Focusing Sheet
11.3 Aspect Ratio Calibration Sheet

Aspect Ratio Calibration Sheet

scan direction

- increase stage speed or decrease frame rate
- calibrated 1:1 ratio
- decrease stage speed or increase frame rate
12.1 Cube Plugins

12.1.1 Utilities

Append Cube

Description:
Append one cube to another by either lines, bands, samples, or any spatial combination of lines and samples. To use bands, the datacubes must be the same size spatially (lines and samples). To use the lines or samples option, the datacubes must match the number of lines and samples respectively. The option of any spatial will scramble the line/sample representation in order to make a spatially rectangular datacube. For this option, the number of bands between cubes must match.

Usage:

New Name: New name of output cube.

Append Dir: Direction to append cube (lines, bands, samples, or any spatial).

Append Cube: Cube to append.

Outputs:

• Appended datacube.

Average Channels

Description:
Averages adjacent bands, samples, and/or lines.

Usage:

Spectral, Sample, and Line Averaging: Select the number of adjacent bands (spectral), sample and/or lines to average together.

Return floating points: Option to return floating point values. Otherwise, the data type of input cube is returned.

Outputs:

• Datacube with adjacent bands, samples and/or lines averaged.
Bad Band Removal

Description:
Remove selected bands from a datacube and optionally interpolate replacements.

Usage:
Start band: First band to remove.
End band: Last band to remove.
Interpolate: If checked, the removed bands will be linearly interpolated from the adjacent bands.

Outputs:
• Datacube with the selected bands removed or interpolated.

Bin Channels

Description:
Bins (sums) adjacent bands, samples, and/or lines.

Usage:
Spectral, Sample, and Line Binning: Select the number of adjacent bands (spectral), sample and/or lines to bin together. This is a summation, not an average.
Return floating points: Option to return floating point values. Otherwise, the data type of input cube is returned.

Outputs:
• Datacube with adjacent bands, samples and/or lines binned.

Convert to BIP, BIL or BSQ

Description:
Converts one datacube interleave to another.

Usage:
Input format: Shows current interleave.
Output format: Select desired output interleave.

Outputs:
• Datacube rotated to the desired interleave.
Count Pixel Values

**Description:**
Counts unique class labels, maximum pixel probabilities, or minimum spectral angles in a datacube and displays the counts.

**Usage:**
Apply to a classified cube class labels, classification probabilities, or spectral angles. The tool will compute the number of class label counts, maximum probability counts, or minimum spectral angle counts.

*Input Cube Type:* Select Probability, Class Label, or Spectral Angle depending on the input datacube type.

*Band Number:* For Class Label cubes, select the band the pixel counts are desired for.

**Outputs:**
- Launches a dialog box containing the counts.

Crop Spatially

**Description:**
Crop a datacube spatially by starting and ending samples and lines.

**Usage:**
- *Start Sample:* First sample to keep in resulting datacube.
- *End Sample:* Last sample to keep in resulting datacube.
- *Start Line:* First line to keep in resulting datacube.
- *End Line:* Last line to keep in resulting datacube.

**Outputs:**
- Cropped datacube.

Crop Wavelengths

**Description:**
Crops beginning and/or ending bands from datacube by wavelength.

**Usage:**
- *Min Wavelength:* Starting wavelength of new cube.

**Outputs:**
- Datacube with beginning and/or ending bands cropped.
Crop Wavelengths by Band Number

Description:
Crops beginning and/or ending bands from datacube, with bands specified by band number.

Usage:
Min Band: Starting band of new cube.
Max Band: Ending band of new cube.

Outputs:
• Datacube with beginning and/or ending bands cropped.

Normalize Datacube

Description:
Normalize each pixel (point spectrum) in a datacube.

Usage:
Method:
• Unit: Divide each brightness value within a pixel by the sum of all bands in that pixel, such that the sum of brightness across all bands in the pixel after normalization is 1.
• Root Mean Square: Divide each brightness value within a pixel by that pixel’s root mean square brightness.
• Maximum: Divide each brightness value within a pixel by the maximum that occurs in that pixel, such that the band that contained the maximum value now contains 1 and all other bands contain a number between 0 and 1.
• Min-Max: Scales each brightness value within a pixel by subtracting the minimum value that occurs in the pixel and dividing by the maximum, such that the new minimum and maximum values are 0 and 1.

Outputs:
• Normalized datacube.

Savitzky-Golay Smoothing

Description:
The Savitzky-Golay filter fits data with successive low-degree polynomials using linear least squares, resulting in smoothed data while preserving much of the original signal’s structure. This implementation will optionally return the n-th derivative of the smoothed signal.

Usage:
Number of points: Size of sliding window for polynomial fitting.
Polynomial Degree: Degree of polynomial used to fit.
Differential order: Order of derivative to return.

Outputs:
• Smoothed datacube or n-th derivative of smoothed datacube.
References:

Scale To One Utility

Description:
Rescales a datacube to 1 by using either the cube’s Reflectance Scale Factor, Ceiling, Bit Depth, or user entered factor.

Usage:
Reflectance Scale Factor: The scaling factor to divide data by. It suggests the value of the cube’s Reflectance Scale Factor, Ceiling, and Bit Depth metavalues, if present and in that order.

Outputs:
- Datacube scaled to 1.

Spectrally Interpolate Linear Cube

Description:
Spectrally interpolates a datacube to the bands that would have been produced by a linearly calibrated imager with a different slope and intercept. This may be useful for using data from different spectral imagers in a single model or other classification routine.

Usage:
Slope: Enter new slope.
Intercept: Enter new intercept.
Slope multiplier: Set this to 2 for Pika III imagers using cameras in format 7 mode 2 or other values to appropriately compensate for on-camera binning.
New band count: Number of output bands desired.
Extrapolate linearly outside original cube waves: Extrapolate linearly if desired cube falls outside of existing data. An error will occur if this condition is true but the option left unchecked.

Outputs:
- Datacube interpolated to new wavelengths.

Spectrally Resample Cube

Description:
Resamples (interpolates) a datacube to match the wavelengths of a selected text or header file.

Usage:
This tool will prompt the user to select a text file (txt or csv) or a header file (hdr) containing the desired wavelengths. If txt or csv is chosen, the wavelength values should be in the first column in units of nanometers.

Outputs:
- Resampled datacube.
Spectrally Resample Spectrum to Datacube

**Description:**
Resamples (interpolates) a Spectrum to match the wavelengths of source datacube.

**Usage:**
- **Spectrum:** Select the Spectrum to resample.

**Outputs:**
- Resampled Spectrum.

Subset Cube Bands

**Description:**
Build a cube consisting of a subset of the bands of the source cube.

**Usage:**
- **3 Band RGB:** Preset to return a 3 band cube using default Red, Green, and Blue bands.
- **3 Band Color IR:** Preset to return a 3 band cube using default Infrared, Red, and Green bands.
- **Bands:** If not using the above presets, select the number and wavelengths of each band to return.

**Outputs:**
- Datacube containing the selected subset of bands.

Subtract Spectrum

**Description:**
Subtracts a background spectrum from a datacube. This could be a noise or background fluorescence spectrum, for example.

**Usage:**
- **Subtract by:** Select the Spectrum to subtract.
- **Return floating points:** Option to return a floating point cube. Otherwise, the original cube’s data type is used.

**Outputs:**
- Datacube with spectrum subtracted.

Unit Conversion Utility

**Description:**
Convert a cube’s intensity values by multiplying each element by a constant scale factor.

**Usage:**
- **Scale Factor:** Number to multiply datacube by.
- **Return Floating Point:** If selected, the cube will be converted to a 32 bit float. Otherwise, the original datatype will be maintained.
12.1.2 Classify

Auto Classifier

Description:
This tool incorporates many proven processes of hyperspectral image classification into an easy-to-use pipeline, while minimizing decisions that must be made by the user. It includes: 1. Feature engineering by concatenation of 1st derivatives to Savitsky-Golay smoothed spectra. 2. Dimensionality reduction with Principal Component Analysis 3. Random Forest classifier

Usage:

Classes: Classes are the discrete training datacubes used to build the predictive model. Enter the number of Classes to be used for training, and that number will be reflected in the number of Class datacube selectors below.

Class Number: For each Class number, select the datacube associated with the class.

Outputs:

• A floating point datacube, with each band containing the probability (0-1) of a pixel belonging that Class, ordered by Class Number.

References:
2. https://en.wikipedia.org/wiki/Principal_component_analysis

Binary Encoding

Description:
Binary encoding is a simple classification method with a small computational load. Each band within an input spectrum is encoded into zeros and ones depending on whether the band is above or below the spectral mean. An additional encoding is created, using a zero or one to represent whether the first derivative of that band is negative or positive. This encoding is concatenated with the first. The datacube is encoded in the same way. An exclusive or is then used to compare the datacube’s encoding with the input spectrum’s encoding, and the result summed. This provides a single number that provides a measure of similarity between the input spectrum and individual pixels of a datacube.

Usage:

Layers: Select the number of input spectra to classify the datacube against.

Spectrum: Select the input spectrum used for classification.

Outputs:

• A datacube with each band containing the distance of a pixel from a Spectrum, ordered by input position.

References:
**Euclidean Distance**

**Description:**
Computes the ordinary straight-line distance in Euclidean space.

\[
||u - v||_2 = \left( \sum |u_i - v_i|^2 \right)^{1/2}
\]

where u and v are input vectors (spectra).

**Usage:**

*Layers:* Select the number of input spectra to classify the datacube against.

*Spectrum:* Select the input spectrum used for classification.

**Outputs:**
- A datacube with each band containing the Euclidean distance of a pixel from a Spectrum, ordered by input position.

**References:**

**Linear Discriminant**

**Description:**
A generalization of Fisher’s linear discriminant that finds a linear combination of features that separate two or more classes, as defined by input training datacubes. It assumes that the independent variables (bands) are normally distributed. It is closely related to logistic regression, which does not make this assumption.

**Usage:**

*Classes:* Classes are the discrete training datacubes used to build the predictive model. Enter the number of Classes to be used for training, and that number will be reflected in the number of Class groupings at the bottom of the dialog box. For each Class grouping, select the datacube in the Class field and the associated dependent value in the Group Number field.

*Solver:* One of either Singular Value Decomposition (svd), Least Squares (lsqr), or Eigenvalue decomposition (eigen). See references below for more information.

*N components:* Number of components to keep after dimensionality reduction. Must be less or equal to the smaller value between the number of classes minus one, and the number of spectral bands.

*Probability:* Check to return a probability of class membership, uncheck to return binary predictions.

*Tolerance:* Threshold for determining whether a singular value is significant. Only used in Singular Value Decomposition.

**Outputs:**
- Datacube with each band containing a probability of a pixel being a member of that class, or a single band with integer encoding that represents class membership by input position order.

**References:**
1. [https://en.wikipedia.org/wiki/Linear_discriminant_analysis](https://en.wikipedia.org/wiki/Linear_discriminant_analysis)
Logistic Regression

Description:
Logistic regression is a statistical model that uses a logistic function to model a binary dependent variable, in this case membership of a class. Despite the name, it is a linear classification algorithm, not a true regression. It is generally fast and powerful.

Usage:
Reg. Method: Regularization method or penalty. See references for more information.
Probability: Check to return a probability of class membership, uncheck to return integer values indicating the predicted class.
Auto Weight?: Check to automatically adjust training weights to be inversely proportional to class frequencies in the input data. If this option is not checked, the classifier may be more likely to assign pixels to classes that occur more frequently in the training data.
C: Inverse of regularization strength; smaller values specify stronger regularization.
Save coefficients to file: Check to save model coefficients to file for use with other software packages.
Classes: Classes are the discrete training datacubes used to build the predictive model. Enter the number of Classes to be used for training, and that number will be reflected in the number of Class groupings at the bottom of the dialog box. For each Class grouping, select the datacube in the Class field and the associated dependent value in the Group Number field.

Outputs:
• Datacube with each band containing a probability of a pixel being a member of that class, or a single band with integer encoding that represents class membership by input position order.

References:
2. https://scikit-learn.org/0.21/modules/linear_model.html#logistic-regression

Mahalanobis Distance

Description:
Mahalanobis distance quantifies the multivariate similarity between each pixel in a datacube and a distribution defined by all of the pixels in a training datacube (or cubes). The distance used is normalized by the inverse of the covariance matrix computed from the training cubes to adjust for correlations between bands and differences in the magnitude of variance across bands.

\[ d_M(p, q) = \sqrt{(q - p)^T C^{-1} (q - p)} \]

Where p is the spectrum at the point being evaluated, q is the mean spectrum computed from the input training cube, and C is the covariance matrix computed from the input training cube. For uncorrelated variables with distributions of equal variance, this reduces to the Euclidean distance.

By applying a threshold to the computed Mahalanobis distance, pixels can be classified as members or outliers of a class.

Usage:
Classes: Classes are the discrete training datacubes used to build the predictive model. Enter the number of Classes to be used for training, and that number will be reflected in the number of Class entries at the bottom of the dialog box. Select Class datacubes for each class.

Outputs:
- Datacube with each band containing the Mahalanobis distance between a pixel and the distribution defined by an input class, as ordered by Class datacube input position. Small numbers indicate greater similarity.

References:

Quadratic Discriminant

Description:
Quadratic discriminant analysis is closely related to linear discriminant analysis, but it does not assume the covariance of each of the classes is identical. As a result, it tends to fit data better but has more parameters to estimate.

Usage:
Classes: Classes are the discrete training datacubes used to build the predictive model. Enter the number of Classes to be used for training, and that number will be reflected in the number of Class groupings at the bottom of the dialog box. For each Class grouping, select the datacube in the Class field and the associated dependent value in the Group Number field.

Probability: Check to return a probability of class membership, uncheck to return binary predictions.

Regularization: Parameter to regularize the covariance estimate.

Outputs:
- Datacube with each band containing a probability of a pixel being a member of that class, or a single band with integer encoding that represents class membership by input position order.

References:
2. https://scikit-learn.org/0.21/modules/generated/sklearn.qda.QDA.html

Random Forest

Description:
Random Forest classification is an ensemble method constructed of a multitude of decision trees, each constructed from a random subset of training data and input features. The final output class is the mode of the classes of the individual trees, which helps correct for individual decision trees’ tendency to overfit the training set.

Usage:
Probability: Check to return a probability of class membership (one band for each class), uncheck to return integer class membership predictions as a single band.

Estimators: The number of trees in the forest. Higher numbers tend to produce better predictive models, but require more memory and computational time.
Classes: Classes are the discrete training datacubes used to build the predictive model. Enter the number of Classes to be used for training, and that number will be reflected in the number of Class groupings at the bottom of the dialog box. For each Class grouping, select the datacube in the Class field and the associated dependent value in the Group Number field.

Outputs:

- Datacube with each band containing a probability of a pixel being a member of that class, or a single band with integer encoding that represents class membership by input position order.

References:


Spectral Angle Mapper

Description:

Compares spectra in a datacube to reference (input) spectra by computing the spectral angle between them. By treating the spectra as vectors, the angle between them is simply the inverse cosine of the normalized dot product of the two.

\[
\theta(p, q) = \arccos \left( \frac{\sum_{i=1}^{n} p_i q_i}{\left( \sum_{i=1}^{n} p_i^2 \right)^{1/2} \left( \sum_{i=1}^{n} q_i^2 \right)^{1/2}} \right)
\]

Thus, two parallel vectors have a spectral angle of zero, while orthogonal vectors have a spectral angle of \(\pi/2\). These angles are typically thresholded to determine class membership.

Usage:

Layers: Select the number of input spectra to classify the datacube against.

Spectrum: Select the input spectrum used for classification.

Outputs:

- Datacube with each band containing the spectral angle between input pixels and reference spectra in radians, ordered by input position of spectrum. Smaller numbers indicate greater similarity.

References:

1. https://pdfs.semanticscholar.org/c192/da3f6560c0305926149e7f6324dab441201b.pdf?_ga=2.90400281.2089631239.1589487588-904881142.1587062196

Spectral Unmix

Description:

Assuming that a signal at a given pixel is a linear combination of pure endmember spectra, this tool will return the relative abundances of the input endmember, constrained to a defined range. Useful for both fluorescence and remote sensing data.

Usage:

Constrain To: Constrain the output to the defined range.

Endmembers: Number of input endmember spectra. Select individual endmember spectra in the drop-down boxes.

Outputs:
• A datacube with each band containing the relative abundance of the endmember, ordered by input position order.

References:

Support Vector Machine

Description:
A SVM model is a representation of the training examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on the side of the gap on which they fall. In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces.

Usage:
Method: One of Linear, Poly, RBF, or LinearSVC, plus the option of returning probability (“Proba”) instead of discrete class predictions. Linear, Poly, or RBF specify the kernel function to use in SVC, while LinearSVC is very similar to Linear but scales better to large number of samples.

Auto Gamma?: Kernel coefficient for RBF and Polynomial functions.

Auto Weight?: Check to automatically adjust weight inversely proportional to class frequencies in the input data.

C: Inverse of regularization strength; smaller values specify stronger regularization.

Poly Degree: Degree of polynomial kernel function. Not used in other methods.

Classes: Classes are the discrete training datacubes used to build the predictive model. Enter the number of Classes to be used for training, and that number will be reflected in the number of Class groupings at the bottom of the dialog box. For each Class grouping, select the datacube in the Class field and the associated dependent value in the Group Number field.

Outputs:
• Datacube with each band containing a probability of a pixel being a member of that class, or a single band with integer encoding that represents class membership by input position order.

References:
2. https://scikit-learn.org/0.21/modules/svm.html
3. https://scikit-learn.org/0.21/modules/svm.html#svm-kernels
**k-Nearest Neighbors**

**Description:**
Neighbours-based classification is computed from a simple majority vote of the nearest neighbors of each point based on a distance metric. To avoid prohibitively long computation times, it is recommended to reduce dimensionality before using k-NN.

**Usage:**

*N Neighbors*: Number of neighbors to use in vote.

*Weights*: Select *uniform* to have all points in neighborhood weighted equally. Select *distance* to weight by inverse distance.

*Distance Metric*: Distance metric used to compute similarity of neighbors.

- *euclidean* - ordinary straight line distance in n-dimensional space

\[
d_{u,v} = \sqrt{\sum_{i=1}^{\text{nbands}} (u_i - v_i)^2}
\]

- *manhattan* - the sum of the absolute differences in each dimension in n-dimensional space

\[
d_{u,v} = \sum_{i=1}^{\text{nbands}} |u_i - v_i|
\]

- *mahalanobis* - a statistically weighted distance that accounts for correlations and scale differences between bands.

\[
d_{u,v} = \sqrt{(u - v)^T C^{-1} (u - v)}
\]

where u and v are the point vectors being compared and C is the covariance matrix computed from the complete dataset.

- *minkowski* - a generalization of euclidean and manhattan distance defined as

\[
d_{u,v} = \left( \sum_{i=1}^{\text{nbands}} |u_i - v_i|^p \right)^{1/p}
\]

Setting P to 1 is equivalent to Manhattan distance, P = 2 is equivalent to Euclidean.

- *chebyshev* - the maximum distance in a single dimension between two point vectors

\[
d_{u,v} = \max_{i=1}^{\text{nbands}} (|u_i - v_i|)
\]

See references for more information.

*Power (P)*: Power to be used in the Minkowski distance calculation. Setting P to 1 is equivalent to Manhattan distance, P = 2 is equivalent to Euclidean.

*Probability*: Check to return a probability of class membership, uncheck to return discrete predictions.
Classes: Classes are the discrete training datacubes used to build the predictive model. Enter the number of Classes to be used for training, and that number will be reflected in the number of Class groupings below. For each Class grouping, select the datacube in the Class field and the associated dependent value in the Value field.

Outputs:
• Datacube with each band containing a probability of a pixel being a member of that class, or a single band with integer encoding that represents class membership by input position order.

References:
2. https://scikit-learn.org/0.21/modules/generated/sklearn.neighbors.KNeighborsClassifier.html
3. https://scikit-learn.org/0.21/modules/generated/sklearn.neighbors.DistanceMetric.html

12.1.3 Analyze

Band Ratio

Description:
Divide each pixel in one band by those in another band.

\[ BR = \frac{\text{Numerator}}{\text{Denominator}} \]

Usage:
\textit{Numerator Wavelength}: The wavelength of the band in the numerator of the ratio.
\textit{Denominator Wavelength}: The wavelength of the band in the denominator of the ratio.

Outputs:
• A new single band floating point datacube from the ratio.

Custom Normalized Difference Index

Description:
Calculate a custom normalized difference index for each pixel, defined as:

\[ NDI = \frac{B_1 - B_2}{B_1 + B_2} \]

If the denominator is zero for any pixels, the return value of those pixels will be set to zero.

Usage:
\textit{First Band}: The wavelength of the first band.
\textit{Second Band}: The wavelength of the second band.

Outputs:
• A floating point cube of the normalized difference.
First Derivative

Description:
Compute the first derivative (difference) along the spectral axis.

Outputs:
• A floating point datacube of the first derivative.

Minimum Noise Fraction

Description:
The Minimum Noise Fraction is a linear transformation of two separate principal component analysis (PCA) rotations. The first rotation utilizes the principal components of the noise covariance matrix to ensure the noise has unit variance and are spectrally uncorrelated. The second PCA rotation is applied to the original data after it has been noise whitened by the first.

Usage:
Bands To Return: The number of rotated bands to retain after the transformation.
Standardize: If this is checked, data is normalized band-wise prior to the transformation by subtracting the mean and dividing by the standard deviation. Unless the input datacube has already been standardized, this should usually be set to True because PCA is sensitive to scale variations across bands.
Est. Noise from Dark Field: If checked, the user will be prompted for a dark noise cube to estimate noise from. If unchecked, the noise will be estimated from the input data.

Outputs:
• A new floating point cube containing the MNF scores of the transformed data.
• The scree plot of eigenvalues is plotted in the User Plot window.

References:

PCA

Description:
Principal Component Analysis is an orthogonal linear transformation of data such that the new coordinate system orders bands by decreasing magnitude of explained total variance. The transformation is determined by the eigen decomposition of the covariance matrix. Typically, a relatively small subset of the resulting bands are retained after the coordinate rotation to reduce the dimensionality of the dataset.

Usage:
Bands To Return: The number of rotated bands to retain after the transformation.
Standardize: If this is checked, data is normalized band-wise prior to the transformation by subtracting the mean and dividing by the standard deviation. Unless the input datacube has already been standardized, this should usually be set to True because PCA is sensitive to scale variations across bands.

Save Transformation Matrix: If this is checked, a transformation matrix is saved to disk, which can can be used to apply the same transformation to other cubes using the PCA from Prior Transform plugin.
Outputs:

- A new floating point cube containing the PCA scores of the transformed data.
- The scree plot of eigenvalues is plotted in the User Plot window.

References:

1. https://en.wikipedia.org/wiki/Principal_component_analysis

PCA from Prior Transform

Description:

Uses a previously saved PCA transformation (from the PCA plugin) to transform other data.

Usage:

Bands To Return: The number of rotated bands to retain after the transformation.

Standardize: If this is checked, data is normalized bandwise prior to the transformation by subtracting the mean and dividing by the standard deviation of the data used to originally fit the transform. Unless the input datacube has already been standardized, this should usually be set to True because PCA is sensitive to scale variations across bands.

PCA Transformation Matrix: Once OK has been pressed, the software will prompt for the previously saved transform.

Outputs:

- A new floating point cube containing the PCA scores of the transformed data.
- The scree plot of eigenvalues is plotted in the User Plot window.

References:

1. https://en.wikipedia.org/wiki/Principal_component_analysis

Regression

Description:

A collection of different statistical methods for predicting a dependent variable (or outcome variable), from one or more independent variables, typically bands in the context of hyperspectral imaging.

Usage:

Method: Can be one of Support Vector (Linear), Support Vector (Radial Basis Function), Ordinary Least Squares, Partial Least Squares, Lasso, Ridge, Ridge (with Cross Validation). Partial Least Squares is a good place to start. See references for more information on these methods.

Alpha: This parameter is used in Ridge and Lasso as the regularization strength parameter. Larger values increase regularization and reduce variance of the estimate.

Auto Gamma: Uncheck this box to specify gamma manually.

Gamma: This parameter is used in SVR RBF to set how far the influence of a single training example reaches, low values meaning greater influence. It can be thought of as the inverse of the radius of influence of model-selected support vectors.

Penalty: This parameter is the regularization parameter, C, used by SVR. The strength of the regularization is inversely proportional to C.
Fit Intercept: Used in OLS, Lasso, Ridge, and RidgeCV to determine whether to calculate the intercept. If data has already been centered, fitting an intercept is not necessary.

Number of Components To Keep: Used by Partial Least Squares as the number of components to use after features have been transformed. With too few components, predictive power may be lost. With too many, the model may be prone to overfitting.

Classes:
Classes are the discrete training datacubes used to build the predictive model. Enter the number of Classes to be used for training, and that number will be reflected in the number of Class groupings below. For each Class grouping, select the datacube in the Class field and the associated dependent value in the Value field.

Outputs:
• A floating point datacube containing the predicted dependent variable.

References:
2. https://scikit-learn.org/0.21/modules/generated/sklearn.linear_model.LinearRegression.html
3. https://scikit-learn.org/0.21/modules/generated/sklearn.cross_decomposition.PLSRegression.html

Second Derivative

Description:
Compute the second derivative (difference) along the spectral axis.

Outputs:
• A floating point datacube of the second derivative.

Thinfilm Thickness

Description:
This algorithm finds the distance between peaks and valleys of interference spectra and uses this distance, along with the known Index of Refraction and angle of incidence, to compute film thickness in microns. The sample must be clean and have a mirror finish.

Usage:
Angle of Incidence: Angle between incident light and normal. Imager should be setup at the same angle on the other side of normal.

Index of Refraction: Measured or previously determined IOR of film.

Outputs:
• A floating point datacube of the film thickness in microns.

References:
Total Radiance

Description:
Sums all bands of a datacube in units of radiance to produce total radiance. May also be used to sum any band data into a single band datacube.

Outputs:
• A single band floating point datacube.

12.1.4 Correct

Back Out Raw Data from Reflectivity

Description:
Convert reflectance data back to raw data from a datacube containing the original white reference and optional dark current datacube.

Usage:
The input datacube must be in units of reflectance.

100% Reflectivity Scaled To: The scaling of the input reflectance datacube.
Correction Cube: The white reference datacube used to originally convert the data to reflectance.
Dark Noise Cube: The optional dark noise datacube used to originally covert data to reflectance.

Outputs:
• A datacube of uncorrected (raw) data.

Correct Optical Distortion

Description:
A Brown-Conrady radial distortion model in the across track dimension to correct for distortions. No tangential distortion coefficients used in this model.

Usage:

K1 through K4: These are the radial distortion coefficients. Contact the objective lens manufacturer or Resonon to obtain these coefficients.

Outputs:
• A corrected datacube.

References:
Georectify Airborne Datacube

Description:
Creates georectified output products based on GPS/IMU data.

Usage:

* Use Flat Earth Altitude:* Check to use flat earth altitude below, uncheck to use DEM file.

* Flat Earth Alt (m):* The altitude of the ground in the area the datacube was collected.

* Field of View (deg):* The field of view of the imager/lens combination used.

* Calculated Resolution (m):* An estimate of the cross and along track resolutions using the FOV and average above ground height for cross track, and distance flown and frame rate for along track. The recommended resolution is twice the largest estimate, to account for platform instabilities.

* Map Resolution (m):* Desired resolution of output products, in meters.

* Save Products in Source Folder?:* Check to save all products (beside Full Datacube) in original folder. Uncheck to specify destination.

* Generate Full Datacube?:* Creates a georectified datacube and adds it to the Resource Tree in Spectronon. It is not saved automatically like the other output products.

* Generate KML of Image?:* Check to generate a KML file of the selected Image.

* Generate GeoTiff of Image?:* Check to generate a GeoTiff of the selected Image.

* Generate Swath Outline?:* Check to generate an outline of the FOV down the flight path, useful for debugging.

* Interpolate Image?:* If checked, the missing pixels of the GeoTiff/KML images will be interpolated.

* Interpolate Datacube?:* If checked, the missing pixels of the datacube will be interpolated.

* Image To Use in KML/GeoTiff:* Select which Image to use in KML/GeoTiff (order follows Resource Tree)

* Stretch:* Apply the specified stretch to GeoTiff/KML image. Absolute is recommended.

* Low/High:* Absolute stretch values.

* Sync Offset (s):* Time delay between image data and GPS/IMU.

* Imager Roll Offset (deg):* Angular offset in the roll direction between imager nadir and IMU nadir.

* Imager Pitch Offset (deg):* Angular offset in the pitch direction between imager nadir and IMU nadir.

* Imager Heading Offset (deg):* Angular offset between imager and IMU in the heading direction.

* Crop Cube to Start/End Lines?:* Used to crop beginning and or ending of datacube in the geocorrection process.

Outputs:
All outputs are optional.

- KML and GeoTiff of Image.
- A georectified datacube.
- Outline KML of imager FOV along flight path.
- Mask of interpolated pixels
- Interpolated LCF file

References:

12.1. Cube Plugins 115
Radiance From Raw Data

Description:
Converts raw data to units of radiance (microflicks) based on the instrument’s radiometric calibration file (.icp). This process is outlined below:

1. The .icp file contains multiple calibration files, one gain cube that contains the pixel-by-pixel transfer from digital number to calibrated radiance and typically many offset cubes, which are dark noise cubes taken at different integration times and gains. This plugin automatically finds the best suited offset file to use (Auto Remove Dark) unless the user decides to provide one.

2. The gain and offset files are scaled to match the binning factor used in the datacube. Note that because the gain file represents the inverse instrument response, it gets scaled inversely.

3. The gain and offset data are averaged both spatially and spectrally to match the frame size of the datacube.

4. The gain and offset data are flipped spatially depending on the flip radiometric calibration header item in the datacube. (This header item is used to track left-right edges in airborne data.)

5. The gain data is scaled by the integration and gain ratios between it and the datacube. Note that gain in Resonon headers is 20 log, not 10 log.

6. The offset data is subtracted from the datacube. The result is multiplied by the gain file.

The resulting data is in units of microflicks.

\[
\text{microflicks} = \frac{(\mu W)}{sr \times cm^2 \times \mu m}
\]

Usage:
The datacube must be collected with a Resonon instrument with a supplied Imager Calibration Pack (.icp).

Auto Remove Dark Noise: Select to use the most appropriate dark noise file in the ICP file. Uncheck for user supplied dark noise datacube.

Return floating point: By default, the result is returned as a two byte integer because solar illuminated scenes in units of radiance will never exceed \(2^{16}\) uFlicks. For very bright light sources or very dimly illuminated scenes, floating point numbers are necessary.

Outputs:
- A datacube containing calibrated radiance units of microflicks.

References:
Reflectance from Radiance Data and Downwelling Irradiance Spectrum

**Description:**
Converts radiance data to reflectivity based on a downwelling irradiance spectrum collected with a single point spectrometer and cosine corrector. This approach assumes the surfaced imaged is Lambertian. Reflectance is then found by:

\[ R = \pi \frac{L}{E} \]

where \( L \) is the at-sensor radiance from the imager and \( E \) is the downwelling irradiance.

**Usage:**
- **Downwelling Type:** Choose either Resonon supplied or other vendor calibrated unit.
- **Use Downwelling Spectrum in Source Folder:** Check to automatically use spectrum in same folder as the source code. Uncheck for user supplied downwelling spectrum (open in Spectronon)
- **Downwelling Calibration:** Resonon supplied downwelling calibration file (.dcp)
- **Auto Remove Downwelling Dark Noise:** Check to use previously measured dark noise (inside DCP file). Uncheck for user supplied dark noise spectrum.
- **Scale 100% Reflectivity To:** Scaling factor for output datacube.
- **Use Correlation Coefficients:** Use previously computed Correlation Coefficients to align downwelling data to ground truth.

**Outputs:**
- Datacube containing reflectivity values.

**References:**

Reflectance from Radiance Data and Measured Reference Spectrum

**Description:**
This approach to converting data to reflectance assumes the input datacube has been corrected spatially, typically by converting to radianse but other approaches would be acceptable. A spectrum of measured reference material is used to correct the datacube spectrally. This spectrum should be collected with the same instrument under the same conditions as the datacube to be collected. The measured reflectance should be a tab, space, or comma delimited file with the first column containing wavelength in nanometers and the second column containing reflectance, scaled to either 0-1 or 0-100%.

**Usage:**
The input datacube should be in units of radiance (or spatially corrected by other means). The input spectrum is typically created with a region-of-interest tool of reference material in a datacube collected with the same imager and under the same conditions.

- **Measured Reflectivity:** Click this button to select the tab, space, or comma delimited file containing the measured reflectance spectrum of the reference material.
- **Measured Reflectivity in Percentage?** Check this box if measured reflectivity is scaled to 0-100. If unchecked, a scale of 0-1 is assumed.
- **ROI Spec:** Spectrum of measured reference material.
- **Scale 100% Reflectivity To:** Scaling factor for output datacube.
Outputs:

- A floating point datacube of reflectance data, scaled to the selected factor.

Reflectance from Radiance Data and Spectrally Flat Reference Spectrum

Description:

This approach assumes the input datacube has been corrected spatially, typically by converting to radiance but other approaches would be acceptable. A spectrum of spectrally flat reference material (typically Spectralon™ or Fluorilon™) is used to correct the datacube spectrally. This spectrum should be collected with the same instrument under the same conditions as the datacube to be corrected.

Usage:

The input datacube should be in units of radiance (or spatially corrected by other means). The input spectrum is typically created with a region-of-interest tool of spectrally flat material in a datacube collected with the same imager and under the same conditions.

ROI Spec: Spectrum of spectrally flat reference material.

Flat Reflectivity (0-1): Reflectivity of reference material on a scale of 0-1 (0-100%).

Scale 100% Reflectivity To: Scaling factor for output datacube.

Outputs:

- A floating point datacube of reflectance data, scaled to the user’s choice.

Reflectance from Raw Data and Downwelling Irradiance Spectrum

Description:

Converts raw data to reflectivity based on a downwelling irradiance spectrum collected with a single point spectrometer and cosine corrector. This approach assumes the surfaced imaged is Lambertian. Reflectance is then found by:

\[ R = \pi * \frac{L}{E} \]

where L is the at-sensor radiance from the imager and E is the downwelling irradiance.

Usage:

Imager Calibration: Resonon supplied imager calibration (.icp) file

Auto Remove Dark Noise: Check to use previously measured dark noise (inside ICP file). Uncheck for user supplied dark noise cube.

Downwelling Type: Choose either Resonon supplied or other vendor calibrated unit.

Use Downwelling Spectrum in Source Folder: Check to automatically use spectrum in same folder as the source code. Uncheck for user supplied downwelling spectrum (open in Spectronon)

Downwelling Calibration: Resonon supplied downwelling calibration file (.dcp)

Auto Remove Downwelling Dark Noise: Check to use previously measured dark noise (inside DCP file). Uncheck for user supplied dark noise spectrum.

Scale 100% Reflectivity To: Scaling factor for output datacube.

Use Correlation Coefficients: Use previously computed Correlation Coefficients to align downwelling data to ground truth.
Outputs:

- Datacube containing reflectivity values.

References:


Reflectance from Raw Data and Measured Reference Cube

Description:

Convert raw data to reflectance based on input datacube of measured reference material. This datacube needs to have the reference material fill the entire FOV of the instrument and be collected with the same instrument under the same conditions as the datacube to be corrected. The measured reflectance should be a tab, space, or comma delimited file with the first column containing wavelength in nanometers and the second column containing reflectance, scaled to either 0-1 or 0-100%.

Usage:

The input datacube should be in units of radiance (or spatially corrected by other means). The input spectrum is typically created with a region-of-interest tool of reference material in a datacube collected with the same imager and under the same conditions.

Measured Reflectivity: Click this button to select the tab, space, or comma delimited file containing the measured reflectance spectrum of the reference material.

Measured Reflectivity in Percentage?: Check this box if measured reflectivity is scaled to 0-100. If unchecked, a scale of 0-1 is assumed.

ROI Spec: Spectrum of measured reference material.

Scale 100% Reflectivity To: Scaling factor for output datacube.

Outputs:

- A floating point datacube of reflectance data, scaled to the user’s choice.

Reflectance from Raw Data and Spectrally Flat Reference Cube

Description:

Converts raw data to reflectivity based on another datacube of a spectrally flat reference material. This datacube needs to have the reference material fill the entire FOV of the instrument, and be collected with the same instrument under the same conditions as the datacube to be corrected.

Usage:

Correction Cube: Datacube of spectrally flat reference material.

Flat Reflectivity (0-1): Reflectivity of reference material on a scale of 0-1 (0-100%).

Scale 100% Reflectivity To: Scaling factor for output datacube.

Outputs:

- A floating point datacube of reflectance data, scaled to the user’s choice.
Remove Dark Noise From Cube

**Description:**
Subtracts a frame of dark noise from a datacube. This frame can be a single frame of data, or a datacube of dark noise. If a datacube is used, its lines are averaged to create a single frame.

**Usage:**

*Dark Noise Cube:* Datacube of recorded dark noise.

**Outputs:**
- A datacube with dark noise removed.

12.1.5 Anomaly

**Elliptic Envelope Anomaly**

**Description:** Anomalies are detected by estimating a robust covariance of the dataset, fitting an ellipse to the central data points, and ignoring those outside. Mahalanobis distances are used to determine a measure of a point’s likelihood of being an outlier.

**Usage:**
The method assumes a Gaussian distributed dataset. For other distributions, other algorithms may perform better, such as SVM Anomaly with the RBF kernel.

Outlier detection from covariance estimation may break or not perform well in high-dimensional settings. In particular, one will always take care to work with \( n_{samples} > n_{features}^2 \). Feature reduction should be performed prior in order to accelerate run times.

*Clutter:* Datacube containing only the statistical normal background, without any outliers.

**Outputs:** Floating point datacube containing 1 for inliers and -1 for outliers.

**References:**
1. [https://scikit-learn.org/0.21/modules/modules/generated/sklearn.svm.OneClassSVM.html](https://scikit-learn.org/0.21/modules/modules/generated/sklearn.svm.OneClassSVM.html)
2. [https://scikit-learn.org/0.21/modules/outlier_detection.html](https://scikit-learn.org/0.21/modules/outlier_detection.html)

**Least Squares Anomaly**

**Description:** Probabilistic method for the detection of anomalous outliers based on a squared-loss function.

**Usage:**

*Clutter:* Datacube containing only the statistical normal background, without any outliers.

*Probability:* If checked, the probability of a pixel being anomalous is returned, scaled 0-1. If unchecked, a classification is returned, with 0 being inliers and 1 for outliers.

**Outputs:** If Probability is checked, a floating point datacube containing a 0-1 probability is returned. If not, a floating point datacube of 0 for inliers and 1 for outliers is returned.

**References:**
**RX Anomaly Detector**

**Description:**
The Reed-Xiaoli (RX) anomaly detector detects pixels determined to be spectrally anomalous relative to a user-defined background or neighboring region. It assumes that the background can be modeled by a multivariate Gaussian distribution. The Mahalanobis distance between a given pixel and the background model is calculated and compared to a user-defined threshold, determining whether the pixel belongs to the background or is an anomaly.

\[
\delta_{RX}(r) = (r - \mu)^T K_{LxL}^{-1} (r - \mu)
\]

**Usage:**
- **Global:** Compares each pixel to the entire datacube.
- **Local (Lines):** Compares each pixel to the image area within a user-defined number of lines of the pixel under test.
- **Local (Squares):** Compares each pixel to a square image area of the user-defined size adjacent to the pixel under test.
- **User Selected:** Compares each pixel to a user supplied datacube that defines the expected background (also known as clutter).

**Outputs:**
A floating point datacube containing the Mahalanobis distance to the selected background area. The default Image for this cube is the Threshold To Colormap, which allows the user to threshold the results to better highlight anomalies.

**References:**

**SVM Anomaly**

**Description:** Anomalies are detected using a One Class Support Vector Machine with a user-supplied background (or clutter) datacube. This cube should contain only statistical normal pixels (without anomalies).

**Usage:**
As with other Support Vector Machines, feature reduction should be performed prior in order to accelerate run times.

**Method:** Can be one of Linear, Radial Basis Function (RBF), or Polynomial. See references for more information on these methods.

**Clutter:** Datacube containing only the statistical normal background, without any outliers.

**Nu:** Upper bound on the fraction of training errors and a lower bound of the fraction of support vectors.

**Auto Gamma:** If selected, the gamma kernel coefficient is set to 1/(number of features).

**Gamma:** Kernel coefficient for RBF and polynomial functions.

**Degree:** Degree of the polynomial kernel function.

**Outputs:** Floating point datacube containing +1 for inliers and -1 for outliers.
12.1.6 Color

CIE Colorspace Conversion

Description:
The CIE colorspace is a color coordinate system based on the human eye’s response to light and color. It is a mathematical generalization of human color vision, allowing one to define and reproduce colors in an objective way.

Usage:
The input datacube must be in units of reflectance with a reflectance scale factor header item, ideally with a spectral range of 390-700 nm.

Illuminant: The theoretical source of light in order to compare colors across different lighting.

Standard Observer: The observer function is the model of human vision produced from color matching experiments. The response of the eye is a function of the field of view, standard values being 10 and 2 degrees.

XYZ: The CIE XYZ color space is the extrapolation of RGB to avoid negative numbers. The Y parameter is a measure of the luminance, with X and Z specifying the chromaticity, with Z somewhat equal to blue and X is a mix of cone response chosen to be orthogonal to luminance.

xyY: In CIE xyY, Y is the luminance and x and y represents the chrominance values derived from the tristimulus values X, Y and Z in the CIE XYZ color space.

LAB: LAB is a color model and space in which L is brightness and a and b are chrominance components. It contains color values far more than the human gamut, but designed to be device independent.

Outputs:
- A floating point datacube containing XYZ, xyY, and/or LAB values.

References:

Delta E*

Description:
The distance metric of Delta E* is measure of color difference in LAB space. It is meant that a Delta E* of one is a just noticeable difference across the entire gamut. In practice, a Delta E* of about 2.3 is just noticeable. It is defined as:

\[ \Delta E_{ab}^* = \sqrt{(L_2^* - L_1^*)^2 + (a_2^* - a_1^*)^2 + (b_2^* - b_1^*)^2} \]

Usage:
The input datacube must have LAB bands named L (L*A*B*), A (L*A*B*), B (L*A*B*).
**XYZ to RGB Colorspace Conversion**

**Description:**
Convert CIE XYZ colorspace to a RGB colorspace.

**Usage:**
The input datacube must have bands containing XYZ data, named X (XYZ), Y (XYZ), and Z (XYZ).

**RGB Working Space:** Desired RGB space to output.

**Companding:** The desired non-linear transformation of intensity values.

**Result datatype:** The desired output datatype and bit-depth.

**Outputs:**
- A floating point datacube containing RGB values.

**References:**

### 12.1.7 Agricultural

**Anthocyanin Reflectance Index 1**

**Description:**
Index sensitive to relative concentrations of anthocyanin versus chlorophyll pigments. As plants are stressed the leaves contain higher concentrations of anthocyanin pigments. New leaves also have high anthocyanin concentrations. Data should be in units of reflectance.

\[
AR1 = \frac{1}{Green} - \frac{1}{Red}
\]

The wavelengths of Green and Red are 550 and 700 nm respectively.

**Outputs:**
- A floating point datacube of CRI data.

**References:**
Anthocyanin Reflectance Index 2

**Description:**
Index sensitive to relative concentrations of anthocyanin versus chlorophyll pigments. As plants are stressed the leaves contain higher concentrations of anthocyanin pigments. New leaves also have high anthocyanin concentrations. Data should be in units of reflectance.

\[ ARI2 = NIR \times \left( \frac{1}{Green} - \frac{1}{Red} \right) \]

The wavelengths of Green, Red, and NIR are 550, 700, and 800 nm respectively.

**Outputs:**
- A floating point datacube of ARI2 data.

**References:**

Atmospherically Resistant Vegetation Index

**Description:**
Atmospherically Resistant Vegetation Index robust to atmospheric aerosols and topographic effects.

\[ ARVI = \frac{NIR - 2 \times Red + Blue}{NIR + 2 \times Red + Blue} \]

The wavelengths of the NIR, Red, and Blue bands are 800, 680, and 450 nm respectively.

**Outputs:**
- A floating point datacube of ARVI data.

**References:**

Carotenoid Reflectance Index 1

**Description:**
Index sensitive to carotenoid versus chlorophyll pigments. As plants are stressed the leaves contain higher concentrations of carotenoid pigments. Values for green vegetation range from 1 to 12. Data should be in units of reflectance.

\[ CRI = \frac{1}{Green1} - \frac{1}{Green2} \]

The wavelengths of Green1 and Green2 are 510 and 550 nm respectively.

**Outputs:**
- A floating point datacube of CRI data.

**References:**
Carotenoid Reflectance Index 2

Description:
Index sensitive to carotenoid versus chlorophyll pigments. As plants are stressed the leaves contain higher concentrations of carotenoid pigments. Values for green vegetation range from 1 to 11. Data should be in units of reflectance.

\[ CRI2 = \frac{1}{Green} - \frac{1}{Red} \]

The wavelengths of Green and NIR are 510 and 700 nm respectively.

Outputs:
- A floating point datacube of CRI data.

References:

Enhanced Vegetation Index

Description:
Enhanced Vegetation Index is optimized for vegetation signal and reduced atmospheric influence.

\[ EVI = 2.5 \times \frac{NIR - Red}{NIR + 6 \times Red - 7.5 \times Blue + 1} \]

The wavelengths of the NIR, Red, and Blue bands are 800, 680, and 450 nm respectively.

Outputs:
- A floating point datacube of EVI data.

References:

Modified Chlorophyll Absorption Ratio Index Improved (MCARI2)

Description:
Index sensitive to the relative abundance of chlorophyll. An improvement of MCARI for better prediction of Leaf Area Index.

\[ MCARI2 = \frac{1.5 \times (2.5 \times (NIR - Red) - 1.3 \times (NIR - Green))}{\sqrt{(2 \times NIR + 1)^2 - \left(6 \times NIR - 5 \times \sqrt{Red}\right)^2}} - .5 \]

The wavelengths of Green, Red, and NIR are 550, 670 and 800 nm respectively.

Outputs:
- A floating point datacube of MCARI2 data.

References:
Modified Chlorophyll Absorption Reflectance Index

Description:
Index sensitive to the relative abundance of chlorophyll.

\[ MCARI = ((NIR - Red) - .2 * (NIR - Green)) \times \frac{NIR}{Red} \]

The wavelengths of Green, Red, and NIR are 550, 670 and 700 nm respectively.

Outputs:
- A floating point datacube of MCARI data.

References:

Modified Red Edge Normalized Vegetation Index

Description:
Narrowband vegetation index robust to leaf specular reflections.

\[ MRENDVI = \frac{NIR - Red}{NIR + Red - 2 \times Blue} \]

The wavelengths of the NIR, Red, and Blue bands are 750, 705, and 445 nm respectively. Vegetation has values typically between .2 and .7.

Outputs:
- A floating point datacube of MRESRI data.

References:
1. https://www.researchgate.net/publication/323723081_Vegetation_Indices_Combining_the_Red_and_Red-Edge_Spectral_Information_for_Leaf_Area_Index_Retrieval

Modified Red Edge Simple Ratio Index

Description:
Narrowband vegetation index robust to leaf specular reflections.

\[ MRESRI = \frac{NIR - Blue}{NIR + Red - Blue} \]

The wavelengths of the NIR, Red, and Blue bands are 750, 705, and 445 nm respectively. Vegetation has values typically between 2 and 8.

Outputs:
- A floating point datacube of MRESRI data.

References:
1. https://www.researchgate.net/publication/323723081_Vegetation_Indices_Combining_the_Red_and_Red-Edge_Spectral_Information_for_Leaf_Area_Index_Retrieval
Normalized Difference Vegetation Index (NDVI)

**Description:**
NDVI is an indicator of live green vegetation, primarily qualitatively but also useful as a quantitative tool. It serves as an estimate of the contrast of the red-edge feature of chlorophyll. It is defined as:

\[
NDVI = \frac{NIR - Red}{NIR + Red}
\]

The wavelengths of the NIR and Red bands are 800 and 680 nm respectively. NDVI returns a value between -1 and 1. Green vegetation is typically between .2 and .8.

**Outputs:**
- A floating point datacube of NDVI data ranging between -1 and 1.

**References:** 1. [https://en.wikipedia.org/wiki/Normalized_difference_vegetation_index](https://en.wikipedia.org/wiki/Normalized_difference_vegetation_index)

Photochemical Reflectance Index

**Description:**
Vegetation index sensitive to carotenoid pigments in live foliage.

\[
PRI = \frac{Green - Yellow}{Green + Yellow}
\]

The wavelengths of the Green and Yellow are 570 and 531 nm respectively.

**Outputs:**
- A floating point datacube of PRI data.

**References:**
1. [https://en.wikipedia.org/wiki/Photochemical_Reflectance_Index](https://en.wikipedia.org/wiki/Photochemical_Reflectance_Index)

Plant Senescence Reflectance Index

**Description:**
Plant Senescence index sensitive to ratio of carotenoid pigments to chlorophyll.

\[
PSRI = \frac{Red - Green}{NIR}
\]

The wavelengths of the NIR, Green and Red are 750, 500 and 680 nm respectively.

**Outputs:**
- A floating point datacube of PSRI data.

**References:**
1. [https://www.indexdatabase.de/db/i-single.php?id=69](https://www.indexdatabase.de/db/i-single.php?id=69)
Red Edge Normalized Difference Vegetation Index

Description:
Similar to NDVI but using the middle of the Red Edge for the red band.

\[ RENDVI = \frac{NIR - Red}{NIR + Red} \]

The wavelengths of the NIR and Red bands are 750 and 705 nm respectively. NDVI returns a value between -1 and 1. Green vegetation is typically between .2 and .9.

Outputs:
• A floating point datacube of RENDVI data.

References:

Simple Ratio Index

Description:
Simple Ratio is a quick way to distinguish green plants from a background. It is defined as:

\[ SR = \frac{NIR}{Red} \]

The wavelengths of the NIR and Red bands are 850 and 675 nm respectively. Green vegetation has a resulting ratio much greater than one and most other objects close to one.

Outputs:
• A floating point datacube of SR data.

References:
1. https://www.hiphen-plant.com/blog/vegetation-indices/

Structure Insensitive Pigment Index

Description:
Vegetation index sensitive to carotenoid pigments while minimizing impact of canopy structure.

\[ SIPI = \frac{NIR - Blue}{NIR + Red} \]

The wavelengths of the NIR, Blue and Red are 800, 445 and 680 nm respectively.

Outputs:
• A floating point datacube of SIPI data.

References:
**Transformed Chlorophyll Absorption Reflectance Index**

**Description:**
Index sensitive to the relative abundance of chlorophyll, impacted by signal from soil if the leaf area index is low.

\[
TCARI = 3 \times \left( (NIR - Red) - 0.2 \times (NIR - Green) \right) \times \left( \frac{NIR}{Red} \right)
\]

The wavelengths of Green, Red, and NIR are 550, 670 and 700 nm respectively.

**Outputs:**
- A floating point datacube of TCARI data.

**References:**

**Vogelmann Red Edge Index 1**

**Description:**
Narrowband vegetation index sensitive to chlorophyll concentration, leaf area, and water content.

\[
VREI_1 = \frac{NIR}{Red}
\]

The wavelengths of the NIR and Red are 740 and 720 nm respectively. Vegetation has values typically between 4 and 8.

**Outputs:**
- A floating point datacube of VREI1 data.

**References:**

**Vogelmann Red Edge Index 2**

**Description:**
Narrowband vegetation index sensitive to chlorophyll concentration, leaf area, and water content.

\[
VREI_2 = \frac{NIR_2 - NIR_1}{NIR_4 + NIR_5}
\]

The wavelengths of the NIR1, NIR2, NIR3, and NIR5 are 747, 734, 726, and 715 nm respectively. Vegetation has values typically between 4 and 8.

**Outputs:**
- A floating point datacube of VREI2 data.

**References:**
Vogelmann Red Edge Index 3

**Description:**
Narrowband vegetation index sensitive to chlorophyll concentration, leaf area, and water content.

\[
V_{REI3} = \frac{NIR2 - NIR1}{NIR4 + NIR3}
\]

The wavelengths of the NIR1, NIR2, NIR3, and NIR5 are 747, 734, 720, and 715 nm respectively. Vegetation has values typically between 4 and 8.

**Outputs:**
- A floating point datacube of VREI3 data.

**References:**

Water Band Index

**Description:**
Index sensitive to water concentration in vegetation canopies. Smaller numbers mean higher water content.

\[
WBI = \frac{NIR2}{NIR1}
\]

The wavelengths of NIR1 and NIR2 are 900 and 970 nm respectively.

**Outputs:**
- A floating point datacube of WBI data.

**References:**

12.1.8 Clustering

HDBSCAN Clustering

**Description:**
HDBSCAN is a clustering tol that allows varying density clusters and does not force outliers into clusters. This is an important feature when working with noisy data. It is a multi-step algorithm summarized in 5 steps below:

1. Transform the space according to the density estimate.
2. Build the minimum spanning tree of the distance weighted graph via Prim’s algorithm.
3. Construct a cluster hierarchy of connected components.
4. Condense the cluster hierarchy based on minimum cluster size.
5. Extract the stable clusters from the condensed tree.
Usage:

*Note:* This algorithm can require long run times. Dimensionality of features should be reduced first.

*Minimum Cluster Size:* Sets smallest size grouping to be considered a cluster.

*Minimum Samples:* Sets a measure of how conservative the clustering should be. The larger the number, the more points will be declared noise.

**Outputs:**

- A floating point datacube containing integer cluster numbers with -1 representing noise.

**References:**

2. https://en.wikipedia.org/wiki/Prim%27s_algorithm

### K-Means Clustering

**Description:**

Partitioning algorithm that groups each sample to the cluster with the nearest mean. Iterations then successively recompute cluster means and distances until convergence or the maximum number of iterations has been met.

**Usage:**

*Note:* As with most clustering algorithms, dimensionality of features should be reduced first if possible.

*Clusters:* Number of desired clusters.

*Max Iterations:* Maximum iterations to be used. The higher the number the slower the run time but more accurate the results. If convergence has been met the algorithm will stop before the maximum number of iterations has occurred.

**Outputs:**

- A floating point datacube containing integer cluster numbers.

**References:**


### 12.1.9 Mask

**Apply Mask**

**Description:**

Overlays the supplied mask on the datacube. Pixels with mask values of one get passed through where mask pixels with zeros are set to a user supplied number or cropped. If cropped, spatial relationships of the datacube will likely be lost. This is useful for creating training datacubes for other classification tools.

Masks must be the same spatial dimensions as the datacube they operate on.

**Usage:**

*New Name:* New name for resulting datacube.
**Technique:** For Set Unmasked to Value, masked pixels will be set to the Mask Value. If set to Crop Masked, the pixels will be cropped from the datacube, likely losing the spatial relationship of the pixels as a result. This can be used to build training datacubes for use with classification tools.

**Mask Value:** Value to set unmasked pixels to.

**Mask Cube:** Mask datacube.

**Outputs:** Datacube with mask applied.

---

**Build Mask**

**Description:**
Builds a mask to apply with the Apply Mask tool. This tool uses a threshold to partition pixels into ones or zeros. In the Apply Mask tool, the mask is overlayed on a datacube and pixels with ones get passed through where pixels with zeros are set to a user supplied number or cropped. Typically, the Build Mask tool is applied to a previously classified datacube which can be partitioned with a threshold.

**Usage:**

**Threshold:** Threshold to partition data with.

**Invert:** If checked, pixels above the threshold are set to zero and pixels below the threshold are set to one. If unchecked, the opposite is true.

**Outputs:**
- A mask datacube containing ones and zeros.

---

### 12.2 Render Plugins

#### 12.2.1 Band Average

**Description:**
Averages all of the bands and displays the greyscale result.

#### 12.2.2 Classification to Colormap

**Description:**
Display a unique color for each class, where pixel membership is given by the band number in which the maximum or minimum value occurs.

**Usage:**

**Evaluate by:** Choose whether class membership is evaluated by minimum value or maximum.

**Colors:** Click on class color button to choose color.
12.2.3 Foreground Object Detection

**Description:**

This powerful tool is useful for extracting objects from their backgrounds for creating training sets, making masks, or for visualization purposes.

The tool has two distinct steps. First, input the Background cube and the known options below. It will return a preview of the results, which can be used to fine tune the output. (Use the Auto Update checkbox to streamline the fine tuning process.) Once satisfied, press the Run Now button in the Export Results box.

**Usage:**

*Background Cube:* Select a datacube representative of the background to remove.

*Threshold:* Threshold to fine tune the separation of foreground from background.

*Minimum Object Size:* Smallest size for foreground object to extract. Can be used to filter out spatial noise.

*Rectangular Cube:* Check to preserve the spatial representation of the image and return a square cube. The background pixels needed to make a square image will be padded with zeros, which will contaminate training cubes. If Unchecked, spatial positioning will be scrambled but padding pixels are not necessary. Use this option for creating training cubes. The outline showing the foreground datacubes will disappear if this option is unchecked.

*Combine Objects:* Check to combine individual objects into a single datacube or mean spectrum. Uncheck to return all individuals.

*Outline color:* Press to select a different color to outline the results.

*Morphological Opening:* Erodes then dilates an image, useful for removing small objects.

*Morphological Closing:* Dilates then erodes an image, useful for filling small holes.

*Border Pixel Erosion:* Shrinks the foreground by expanding the background, removing unwanted edge pixels.

*Structuring Element Size:* Size of the structuring element to use in Opening, Closing, and Erosion. The larger the element, the greater the effect.

*Save To Disk:* Check cube and/or spectra to save them to disk. Specify directory below. Spectra are the mean spectra of the foreground objects, or mean of the combined foreground objects if chosen.

*Spectrum Save Format:* Select to save spectra as .spec, .txt, or .csv.

*Save Directory:* Directory to save results to.

*Add To Workbench:* Check to add datacubes, spectra, and/or mask cubes to the Spectronon Resource Tree (Workbench).

*Run Now:* Press button to execute.

**Outputs:**

- Datacubes and/or spectra saved to disk (optionally)
- Datacubes, spectra, and or mask cubes added to Workbench (optionally)
12.2.4 Greyscale

Description:
View a greyscale image by wavelength, band number, or band name.

Mode: Choose to view by wavelength, band number, or band name, if available.

Wavelength/Band/Band Name selector: Choose wavelength/band to view.

12.2.5 Greyscale to Single Color

Description:
Assigns a color to a single band datacube.

Usage:

Color: Select Red, Green, or Blue.

Invert: Check to make lower pixel values the brightest.

Clip Negatives?: Check to clip negative values to zero for display purposes.

12.2.6 Last Frame

Description:
View the last raw data frame of the datacube. A frame is a 2D image with a spatial dimension on one axis and spectral on the other.

12.2.7 RGB

Description:
View three bands of a datacube with a composite Red, Green, and Blue (RGB) image.

Usage:

Mode: Specify band selection by wavelength or band number.

Adjustments (Red, Green, and Blue): Select wavelength or band number for each color.

Presets: Depending on the wavelength range of the datacube, the Preset buttons will specify the above wavelengths for Red, Green, and Blue to standard wavelengths. In the VNIR, the options are True Color (normal color image) or Color Infrared (uses Green, Red, and Infrared wavelengths).

12.2.8 Raw Camera Data

Description:
View raw data frame from a datacube. A frame is a 2D image with a spatial dimension on one axis and spectral on the other.

Usage:

Frame Number: Select the frame number (line number) to view. -1 is the last frame, 0 is the first.
12.2.9 Scalar to Colormap

**Description:**
Assigns a unique color to each scalar value in a chosen band.

**Usage:**
- **Colormap:** Choose colormap to use.
- **Normalize:** Normalize band before assigning color.
- **Mode:** Choose band selection by number or name.
- **Band:** Band to visualize.

**References:**

12.2.10 Stack

**Description:**
Stack multiple images on top of one another with transparency. This can be useful to show classification results on top of the original RGB image, or show multiple classification results simultaneously.

**Usage:**
- **Stack Height:** Number of images to stack.
- **Image:** Select image to show.
- **Alpha:** Sets transparency of the image. 1 is no transparency.
- **Make zero values transparent:** Useful for classified or mask datacubes.

12.2.11 Threshold To Colormap

**Description:**
Assign a unique color to thresholded band value of the pixels in a datacube. Useful for datacubes containing a continuous range of values (such as spectral angles or probability results) that must be thresholded for final classification.

If multiple bands of a pixel meet the threshold requirement, color is assigned by band values furthest away from threshold.

**Usage:**
- **Classification type:** Select Distance or Probability. Distance assigns a color to pixels below the threshold, Probability above.
- **Value:** Threshold value.
- **Color:** Color to assign.
12.3 Filter Plugins

12.3.1 Colormap Stretch

Description:
Assigns a color to individual scalars in a single band image. Similar to the Scalar to Colormap Image tool.

Usage:
*Use Contrast Enhancement:* Check to stretch image by below method (Linear Percentage, Linear Absolute, or Histogram).

*Colormap:* Choose colormap to use.

*Method:* Choose how to stretch the data before applying colormap.

*Parameters:* Depending on Stretch method chosen, different parameters are available:

*Low/High Cut:* Choose high and low cut values to stretch the remainder through. Use Preset buttons go to default presets.

*Stretch Bands Individually:* Not used in this tool.

*Inverse:* Invert the high/low colormap scheme.

*Bins:* Number of histogram bins to use.

References:

12.3.2 Contrast

Description:
Apply a linear percentile, linear absolute, or histogram stretch to the image.

Usage:
*Use Contrast Enhancement:* Check to stretch image by below method (Linear Percentage, Linear Absolute, or Histogram).

*Colormap:* Choose colormap to use.

*Method:* Choose how to stretch the data before applying colormap.

*Parameters:* Depending on Stretch method chosen, different parameters are available:

*Low/High Cut:* Choose high and low cut values to stretch the remainder through. Use Preset buttons go to default presets.

*Stretch Bands Individually:* Check to stretch each band by statistics of each band, uncheck to stretch all bands by statistics of all.

*Inverse:* Invert the high/low colormap scheme.

*Bins:* Number of histogram bins to use with the histogram stretch.
12.3.3 Local Modal Filter

**Description:**
Assign each pixel to the value most frequently occurring in a local spatial window around the pixel.

**Usage:**
*Use Modal Filter:* Turns the filter on or off.
*Window Size:* Size of the window to evaluate most frequent value around given pixel, in pixels.
*Window Shape:* Shape of the window.

12.3.4 No Filter

**Description:**
No filter is used. Depending on the datacube values, the resulting image might appear completely dark, white, or have very little contrast on the screen.

12.3.5 Threshold

**Description:**
Apply a threshold to the image. Options include a simple binary threshold based on a single number, thresholding below minimum and above maximum values, and thresholding values between the minimum and maximum. Inversion of the results is also an option.

**Usage:**
*Min Threshold:*
Values below or equal to this threshold are set to white, unless thresholds are locked. If True Between is checked, values below this threshold are set to black.

*Max Threshold:*
Values above or equal to this threshold are set to white. If True Between is checked, values above this are set to black.

*Lock Min-Max:*
Locks Min and Max together. Values above or equal to this value are set to white, values below are set to black.

*Inverse:*
Inverts black and white.

*True Between:*
Sets values below Min Threshold to black, values above Max Threshold to black, and values between (including equal to) to white.
12.4 Select Plugins

12.4.1 Create Cube From Selection

Description:
Create a new cube from selection. If the selected area is rectangular, the dimensions of the selected area will be preserved. If the selection is not rectangular, the spatial relationship will be scrambled in order to make a rectangular datacube.

12.4.2 Mean Spectrum

Description:
Create and plot a spectrum from the mean of the selected region. Wavelength metadata must be present for this tool to be available.

The standard deviation will also be plotted, which can be turned on or off by right clicking on the Spectrum object in the Resource Tree and selecting Show/Hide Standard Deviation.

12.4.3 Show Mean of Selection

Description:
Show a dialog box with the mean of each band in the selected region. Useful for datacubes with few bands, bands that don’t contain spectral data, and/or datacubes containing disparate data.

12.4.4 Mean Z profile

Description:
Create and plot the mean profile of the selected region in the Z (band) dimension. If wavelength metadata is present this is equivalent to the Mean Spectrum tool.

The standard deviation will also be plotted, which can be turned on or off by right clicking on the Spectrum object in the Resource Tree and selecting Show/Hide Standard Deviation.

12.4.5 Mean First Derivative

Description:
Plot the mean first derivative (discrete difference) of the selected area.
12.4.6 Create Correlation Coefficients

**Description:**
A correlation coefficient aligns reflectance data created with downwelling irradiance data to a ground truth target such as a tarp or reflectance standard. Once created, these coefficients are used in the Reflectance Conversion with Downwelling Irradiance plugins to improve the accuracy of the result.

**Usage:**
Start by creating a region of interest of the ground truth target.

*Use Measured Reflectance:*
Check to use a previously measured reflectance spectra for the reference object the region of interest contains. Uncheck to assume a flat reflectivity.

*Measured Reflectance File:*
Select the previously measured reflectance spectrum. The first column should be wavelength in nanometers, the second column is reflectance from 0-1 or 0-100%. File should be tab, comma, or space delimited.

*Flat Reflectivity:*
Select flat reflectivity of ground truth target if measured spectrum is not available or desired.

*Measured Reflectivity in Percentage:*
Check if measured reflectance spectrum is on the scale 0-100. Uncheck if 0-1.

**Outputs:**
- Spectrum of correlation coefficients.

12.4.7 Find Closest RAL Classic Color

**Description:**
Uses spectral information to determine the closest RAL Classic color to selected region. Wavelength and reflectance scale factor metadata must be present. Useful for color matching.

**References:**

12.4.8 Median Spectrum

**Description:**
Plot the median spectrum within selected area.
12.4.9 Mean Nth Derivative

Description:
Compute and plot the mean Nth derivative (discrete difference) of the selected region.

12.4.10 Save random spectra to text file

Description:
Randomly selects pixels within the region of interest and exports their spectra (or bands) to a text file.

Usage:

Number of spectra to export:
Number of randomly selected spectra to export.

Add Wavelengths Col:
Option to add the wavelength metadata as the first column of exported data.

12.4.11 Send To Clipboard

Copy Mean as Text

Description:
Copies mean spectrum as text to clipboard for pasting into other applications.

Copy All as Transposed Text

Description:
Copies mean spectrum as transposed text to clipboard for pasting into other applications.

All Spectra as Text

Description:
Copies all spectra in region as text to clipboard for pasting into other applications.

Copy as Image

Description:
Copies image of region to clipboard for pasting into other applications.
GLOSSARY OF HYPERSPECTRAL IMAGER TERMINOLOGY

**Datacube** The images taken by hyperspectral imagers are stored as datacubes. Similar to RGB images, datacubes have two spatial dimensions and one spectral dimension, except that the spectral dimension can have hundreds of “colors”. Datacubes are often accompanied by metadata that store additional info about the datacube (e.g., the size of each dimension and the spectral wavelengths).

**Spectral Range** The range of electromagnetic wavelengths (e.g. light) over which the hyperspectral imager collects signal. For reference, visible wavelengths span from approximately 400 to 700 nanometers (nm).

**Spectral Channels** The number of wavelength bands that the hyperspectral imager measures across the Spectral Range.

**Spectral Resolution** A measure of the narrowest spectral feature (in nm) that you can measure with a hyperspectral imager. Alternatively, it is the width of each spectral channel of the imager in nm.

**Spectral Pixels** The number of pixels across which the spectral signal is imaged. Line-scan (aka. push-broom) hyperspectral imagers distribute spectral data along one-axis of the two-axis camera focal plane array and spatial data along the other axis.

**Spectral Sampling** The Spectral Sampling is the Spectral Range divided by the Spectral Pixels. The Spectral Sampling is often narrower than the Spectral Resolution.

**Spatial Channels** Resonon’s hyperspectral imagers are line-scan or push-broom imagers. The Spatial Channels are the number of pixels along this line.

**Max. Frame Rate** The maximum line-scan rate of the hyperspectral imager. (This is NOT the 2-dimensional image acquisition frame rate, which is slower and depends on the number of lines in the image.)

**Bit Depth** The resolution of the data recorded for each pixel. For example, a Bit Depth of 12 means that each signal acquisition is stored as one of $2^{12} = 4096$ discrete values.

**f/#**

f-number f/# is a measure of the lens speed, and is a quantity that is needed to determine the optical system’s radiometric performance. f/# for a hyperspectral imager means the same as it does for a conventional camera.

**Average RMS Spot Radius** This is a measure of the resolution of the hyperspectral imager and is averaged across the Spectral Range of the imager.

**Smile (peak to peak)** Smile is an optical distortion associated with hyperspectral imagers that manifests when light from a single wavelength falls across different spectral columns of the focal plane array for different spatial channels. The figures show the signal that a “perfect” hyperspectral imager would obtain for a uniformly lit sample with narrow-band red, green, and blue signal, along with an exaggeration of the signal that a “real-life” hyperspectral imager obtains. Peak-to-peak Smile for the red signal is the distance between the black vertical lines. All real hyperspectral imagers have some Smile, usually measured in pixels. Smile leads to spectral signatures that change slightly across different spatial channels.
Keystone (peak to peak)  Keystone is an optical distortion associated with hyperspectral imagers that manifests when light from a single spatial position falls across different spatial rows of the focal plane array for different spectral channels. The figures show how a “perfect” hyperspectral imager would map the signal from three spatial channels, one at the top of the field of view, one in the middle, and one at the bottom, along with an exaggeration of the signal that a “real-life” imager maps the same signal. The peak-to-peak Keystone is the distance between the dashed lines, measured in pixels. All real hyperspectral imagers have some Keystone, which causes the signal to be “mixed” between spatial channels.
“PERFECT” IMAGER

“REAL-LIFE” IMAGER (exaggerated)
CONTACT US FOR PRODUCT SUPPORT

Phone: 1.406.586.3356
Website: https://resonon.com/support
Email: inquiry@resonon.com
Copyright © 2016 - 2020 Resonon Inc. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or by any means (electronic or mechanical, including photocopying) for any purpose without written permission from Resonon. Resonon will not be responsible or liable for any accidental or inevitable damage that may result from unauthorized access or modifications.

This document may contain errors or inaccuracies, and it may be revised without advance notice. This manual is updated frequently.

Please see our software copyright and EULA at

Resonon welcomes any recommendations regarding this manual. Customer feedback is always welcome as it helps us to continuously improve upon the quality of our products.

Please contact us at Resonon with any questions or comments. We look forward to hearing from you.