# HypPy User Manual[1]

graphical user-interface (GUI)

Wim Bakker

October 9, 2018

# Contents

**Abstract**

This document is meant for users of the high-level applications in the Hyperspectral Python (HypPy)[1] package. The package was created by Wim Bakker.

**Contributors**

- Jelmer Oosthoek

**Prerequisites**

The programs described in this document were developed using the Python programming language version 2.7 [8].

The core of these programs forms the envi2 module (see the Programmers' Guide), which relies heavily on the array processing functionality of the Numerical Python (NumPy) package [4].

User interfaces were developed using Tkinter. The Tkinter module ("Tk interface") is the standard Python interface to the Tk GUI toolkit [11].

The continuum removal program (section 4.1) uses a modified version of the Quick Hull algorithm [3] for finding the convex hull.

Display programs (chapter 2) use the Image and ImageTk[2] modules from the Python Imaging Library [6] (now called Pillow) for displaying images, and Matplotlib (a.k.a. PyLab) for making 2D graphs of spectra [2].

The thermal correction (section 9.6) uses the least squares estimator from the Scientific Python (SciPy) package [10].

The WMS get program (section 3.14) uses the OWSlib [5], package for client programming with OGC™ web services. On Linux it can easily be installed with the command 'easy_install OWSlib'.

The View Shapefile (section **??**) uses the shapelib and dbflib modules. These modules are shipped with the Matplotlib Basemap module [1].

[1] hyppy = the state of being both insanely happy and hyper at the same time.
[2] On Linux, Image and ImageTk are two separate modules!

# Chapter 1

# The Top-level Menu

The top-level menu of the image toolbox (figure 1.1) has the following sub-menus:

- Viewers — contains viewers for image data, vector data and spectral libraries.
- Conversion — contains conversion tools for manipulating the bands of and image (sorting, splitting and merging bands, importing).
- Tools — contains tools that operate on the spectral information.
- Filters — contains filters that use the full spatio-spectral data content of the image.
- Classification — contains the spectral mapper and rule image classifier.
- Segmentation — contains tools for generating edge maps and image segmentation.
- Mars Tools — contains tools for OMEGA data.
- Help — contains the user guide, programmers' manual and GNU license.

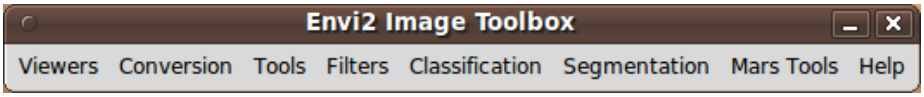


Figure 1.1: The HypPy Image Toolkit main menu.

The menu can be re-configured using the menu configuration file, see A.

The user interfaces for the processing tools were created using Tkinter, Python's de-facto standard GUI (Graphical User Interface) package. Tkinter is a thin object-oriented layer on top of Tcl/Tk. All the names of the tools start with 'tk' to indicate that these are GUIs.

Most of the tools are split into two parts, one for the GUI and one for the actual processing. Most tools are simple interfaces for setting input and

output parameters. Input and output files can be picked using the [···] buttons. The 'Run' button starts the process, while the text box shows the progress of the running process. In its simplest form, an interface may require only input and output files to be set. In addition, some programs may require additional options or parameters to be set.

In many of the programs you can determine whether you want to sort the input band on wavelength and whether you want to skip bad bands. Just check or uncheck the appropriate check box.

Note: some of the figures in this document show old interfaces or the old display programs. The functionality, whoever, remains the same and may have improved in a number of cases.

# Chapter 2

# Viewers

This chapter describes the display tools that were developed for visualizing bands, color composites, spectral libraries and shapefiles.

## 2.1 The Display program

This section describes the Display program (figure 2.4). This viewer has multiple stretch functions, can show a band in pseudo-color, can save pixel data and can show x, y and z profiles. If the input image is an ENVI Classification image and has a color table in the form of a 'class lookup', then the image will be shown in color. The Display can show one band in black & white or pseudo-color and 3 bands in RGB color.

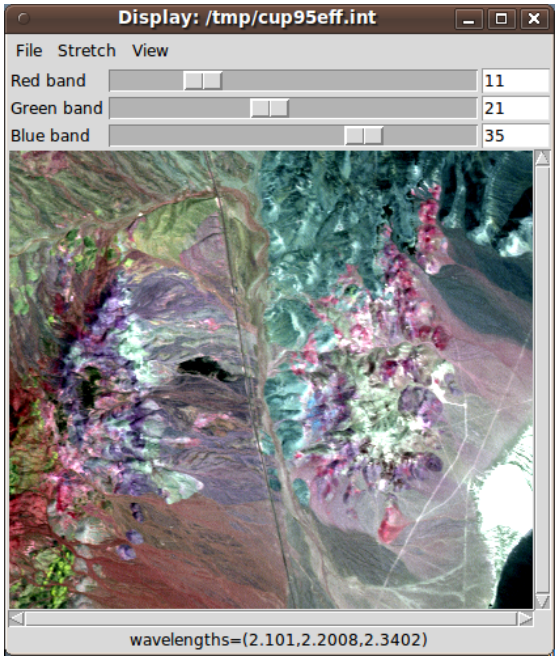


Figure 2.1: The Display program.

The menu bar contains the following menu items: file, stretch and view. Below, the functions of the various menu buttons will be discussed.

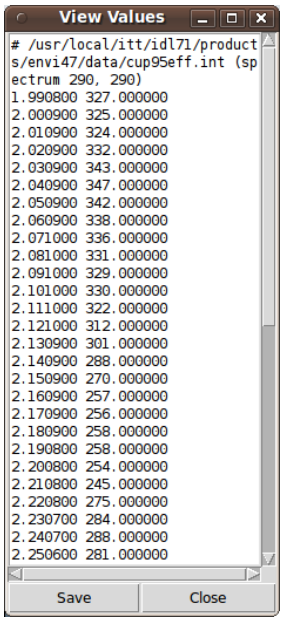


Figure 2.2: The View Values window.

- Sort by Wavelength — Check this button if you want the bands sorted on wavelength. This is useful if you want to plot spectra of images that have the bands not in wavelength order. Set this option before opening the image.
- Use BBL — If your images contain bad bands and these are marked in the bad band list (BBL) of the ENVI header then you can check this button to exclude these bands. Set this option before opening the image.
- Open — Load an image from disk.
- Exit — Quit the application.
- min-max — Stretch between minimum and maximum value.
- 1%–99% — Stretch between the 1% and 99% histogram values.
- 2 std. dev. — Stretch between the mean minus two standard deviations and the mean plus two standard deviations.
- Inverted — Show the grey-values inverted. Can be used together with the stretch options mentioned above.
- Pseudo-color off — No pseudo-color.
- Rainbow (bright) — Use the Rainbow pseudo-color.

- Rainbow (dark→bright) — Use the Rainbow pseudo-color. This particular palette is dark for low values and bright for high values.
- Blue-Red — pseudo-color table that ranges from blue to magenta to red.
- Green-Red — pseudo-color table that ranges from green to yellow to red.
- Random — Use random colors. The colors are really random and change every time you load an image.
- Color Display — Switch to three-channel RGB mode.
- Values Window — Opens the values window.
- Plot Window — Opens the plot window (will only open after clicking on the image).
- X profile — Show x-profile or horizontal cross-section.
- Y profile — Show y-profile or vertical cross-section.
- Z profile — Show z-profile or spectrum.
- single profile — Show a single profile. The extent for x and y is the entire image.
- mean profile — Show an average profile. The extent for x and y is set to the extent of the display. This can be useful for quickly retrieving an average spectrum from a small area and then saving it using View→View Values.

## The View Values window

The View Values window can be used for viewing and saving pixel values.

If you save a number of spectra in one directory, then these spectra can be used as an ASCII spectral library later, for instance in the spectral library viewer or in the Spectral Mapper. The files should be saved as a .txt, .asc or .dat file (figure 2.3). Make only one collection of spectra per directory.

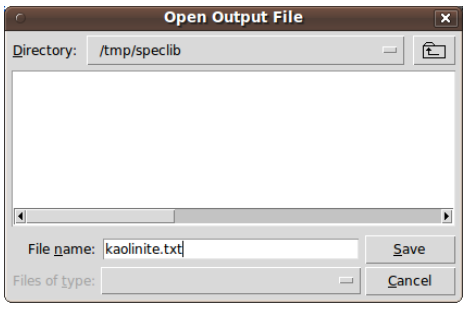


Figure 2.3: Save file window.

## Single band display

In this mode the Display pogram is a simple black & white viewer for ENVI image data. Images can be loaded by going to the menu bar and select File → Open. The slider can be used to quickly jump to another band. A band number can also be entered directly next to the slider. The band numbering starts at 0, which is different from ENVI, which starts at 1. When either the 'sort_wavelengths' or 'use_bbl' option is checked the bands may be numbered quite different from the way they are numbered in ENVI! Three stretch methods are currently implemented, the min-max stretch, 1%–99% (default) and the two standard deviation stretch.

When a different band is selected, the status bar shows the wavelength of this band. If no wavelength information is present in the header of the image, a band name will be shown. If no band names are present then it will simply show the band number.

When the mouse is moved over the image, the status bar will show the current location and value. When clicking on the image a plot window will pop up showing the spectrum on the current location. Multiple spectra can be selected by clicking on different locations in the image. The legend of the plot window shows from which locations the spectra where taken. To start a fresh plot window simply throw away the old plot window first by pressing the 'X' button of the window. The next click on the image will start a new plot window.

Figure 2.4 shows the spectrum of location 290, 290 in the image. It shows a typical spectral feature of a clay mineral around 2.2 $\mu$m. Figure 2.5 shows a plot window with multiple spectra from different locations in the image.

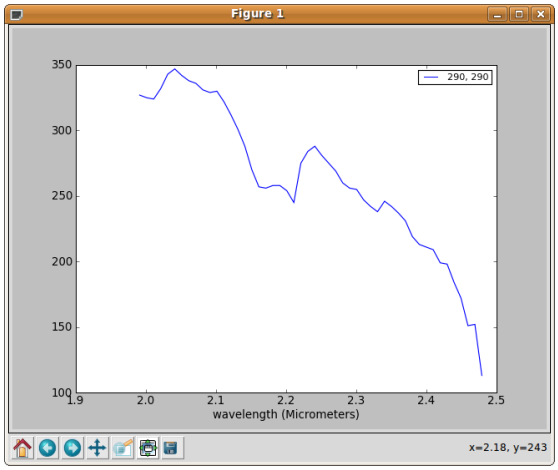


Figure 2.4: A spectrum shown in a plot window.

## Color composite display

In this mode the Display display program is very similar to the previous mode. The only difference is that it is capable of showing color composites. The three

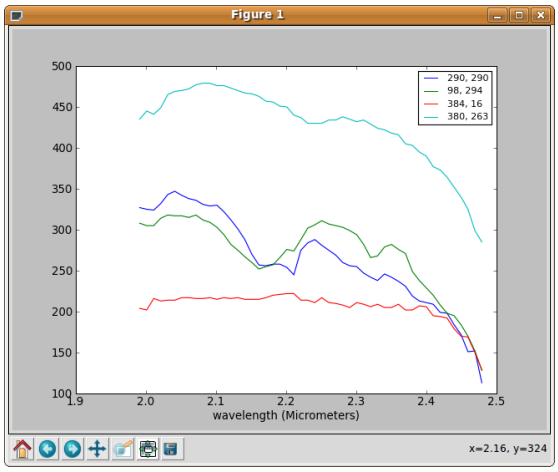


Figure 2.5: Multiple spectra shown in a plot window.

sliders can be used to select the bands for the red, green and blue channel on screen. The status bar will now show three values from the three bands instead of one. By clicking on the image a plot window will appear with spectra from the image. Refer to the previous section for more information.

See figure 7.8 for an example of the various modes of the display program. Left is the RGB color mode, middle is the greylevel mode and right is a classification image.

### Zooming and Panning

The latest version of the image display can zoom and pan. Next to the zoom entry in the menu bar, the mouse buttons can be used.

The mouse buttons have the following functions:

- click left button: plot profile
- rollwheel: zoom in/out
- click middle button: reset zoom to 1
- drag right button: panning

## 2.2 Spectral library viewer

The Spectral Library Viewer is a viewer for ENVI spectral libraries. After selecting the input spectral library, the names of the spectra inside the spectral library are shown in the list box. By clicking on a name a plot window will pop up showing the spectrum of the selected mineral or material. Clicking multiple spectra will plot more spectra in the plot window, which will enable the user to compare the spectra (figure 2.7). Deleting the plot window (by pressing 'X') and selecting a new spectrum will restart the whole process of selecting spectra. The legend of the plot will show the names of the selected spectra.

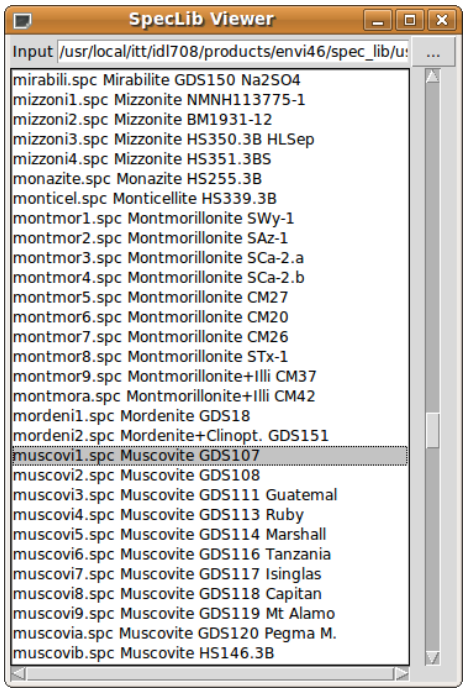


Figure 2.6: The Spectral Library Viewer user interface.

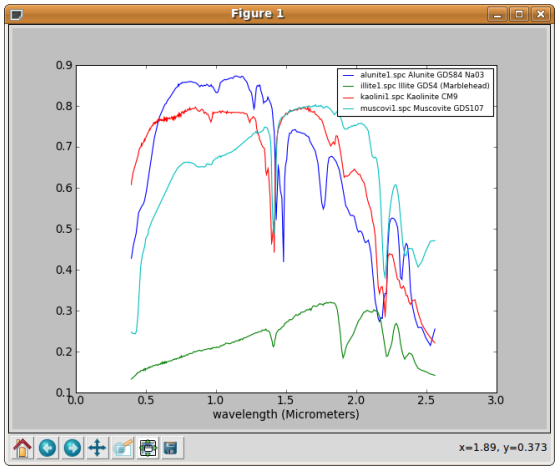


Figure 2.7: Multiple spectra in a plot window.

The spectral library viewer can also read ASCII files or a whole folder containing ASCII files in various formats. In case of a folder, the files should have the extension txt, asc, dat, or tab. All files from the folder will be read and listed in the listbox.

Using the option 'ASCII directory recursive' can be used to read an entire directory tree in search of ascii spectra files.

The viewer can also perform spectral math on the spectra. Check the HypPy

Spectral Math Manual for details and examples.

TODO

## 2.3 Scatterplot Viewer

Create a scatter plot of two bands. The bands can come from two different images. If the image for the Y-axis is left empty then the same image from the X-axis is taken for the Y-axis.

# Chapter 3

# Conversion

## 3.1 Subset/Convert

The Subset/Convert program can be used for making spatial and spectral subsets of hyperspectral images. It is also capable of changing the interleave (bip, bil, bsq) and the data type. The spatial selection determines which samples and lines will be selected for the output image. The Select Bands button opens the Band Selector window and can be used for selecting a set of bands for the output image. The output format bip, bil or bsq determines the output format of the output image. The Output Data Type can be set to force a specific output data type. Currently, these output data types are supported: uint8, int16, int32, int64, uint16, uint32, uint64, float32, float64. Data type conversion does not check for overflow, so make sure your input data fits into the specified output data type! Default format and data type are obtained from the header of the input image.

Summarizing, the Subset/Convert program can be used for:

- making a spatial subset
- making a spectral subset
- conversion to bip, bil or bsq
- conversion from integer to float or vice versa
- conversion to more or less bits per pixel
- any combination of the above

The Subset/Convert interface is shown in figure 3.1.

## 3.2 Sorting Channels

The Sort Channels program reads the input image and creates an output image in which the bands are sorted according to wavelength. If no wavelength information is present in the input image then the program does nothing. Bad bands are skipped in output.

The Sort Channels interface is shown in figure 3.2.

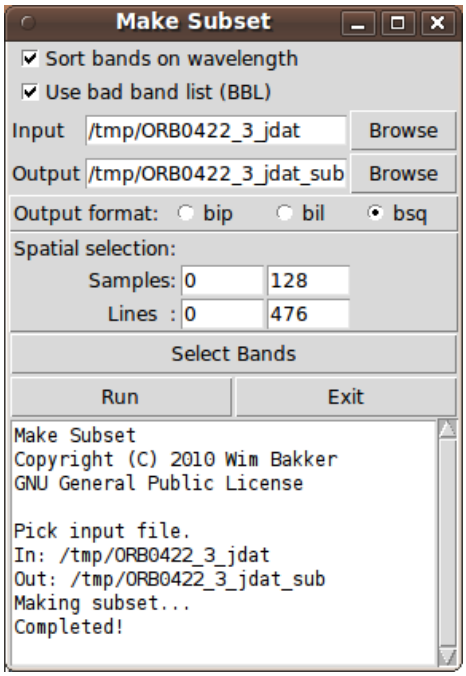


Figure 3.1: The Subset/Convert user interface.

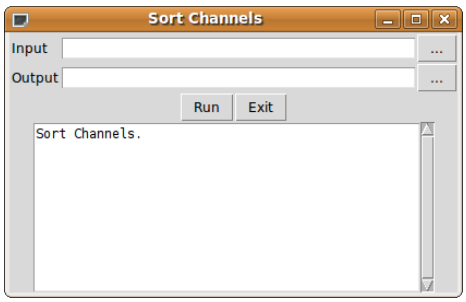


Figure 3.2: The Sort Channels user interface.

## 3.3 Stack ENVI Images

This program can be used to stack a (large) number of separate images and bands into one big ENVI image. This may be useful for analyzing time series, for instance. If the 'sort bands' option is checked then all the bands are sorted before the output image is created.

The buttons below the listbox have the following function:

- Add — Add more images to the list
- Delete — Delete selected images from the list
- Up — Move selected images up in the list
- Down — Move selected images down in the list

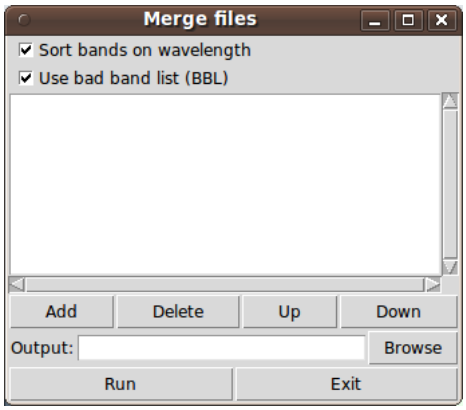


Figure 3.3: The Stack ENVI Images user interface.

## 3.4 Split into Separate Bands

This program splits a multi-band ENVI image into a number of one-band images. The output format can be selected from ENVI, JPEG, TIFF and PNG.

If JPEG, TIFF or PNG is selected for output then the images are stretched first using the two standard deviation stretch.

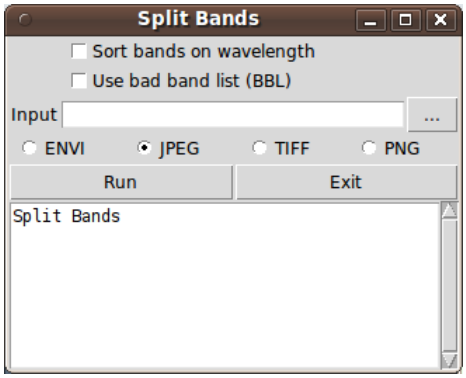


Figure 3.4: The Split into Bands user interface.

## 3.5 Generate KML

The Generate KML program is used to convert an image to KML. The KML file can then be viewed in Google Earth or Google Mars. The input data must have (pseudo-)coordinates. Currently, the program only supports coordinates that are defined in the ENVI header by the keyword 'geo points'. The Geocorrection programs (sections 4.14 and 9.2) generate these pseudo-coordinates.

The Generate KML program generates three files: a PNG file, a PGW file and the KML file. The PNG file is a normal RGB image. In fact, the PNG image contains a fourth band, which is the so-called alpha channel. The alpha

channel is used to control the transparency of certain areas of the PNG file. The PGW file is a so-called world file and contains the (pseudo-)coordinates of the image. The coordinates are copied from the original image file. The PNG file and the PGW file together can be used as georeferenced files in ArcMAP for instance. The KML file together with the PNG file can be used as georeferenced files in Google Earth or Google Mars.

The Generate KML user interface is shown in figure 3.5.

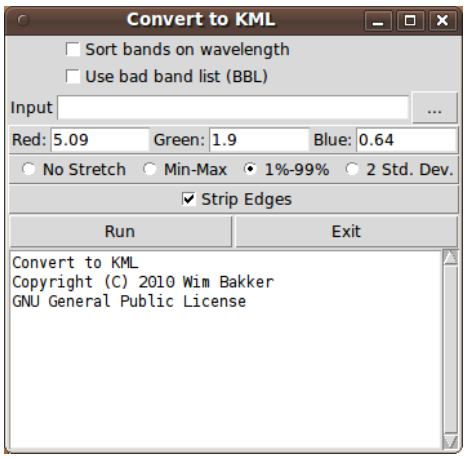


Figure 3.5: The Generate KML user interface.

The red, green and blue entries will determine which bands of the image will be taken for the red, green and blue bands in the PNG image. If the input image has wavelengths defined for the bands then the wavelengths of the bands must be entered. If the image does not have wavelengths then the numbers are taken as band numbers. Please, be aware that the band numbering starts at zero!

The RGB bands can be stretched. There are 4 options:

- no stretch, use the original values of the input image. These should fall within the range of 0 to 255
- min-max stretch, stretching using the minimum and maximum values
- 1%–99%, stretching using 1% and 99% values of the cumulative histogram
- 2 Std. Dev., stretching using the mean minus and plus 2 standard deviation values

If the input image was already stretched using your own stretching algorithm, then use the 'no stretch' option.

If the 'strip edges' check box is ticked then the NaN values of the input image will appear as transparent values in the output image. This option is useful for geocorrected images that have no data values (NaN's) on the sides of the image data. If this option is not selected then the edges will come out as black and may mask any underlying images.

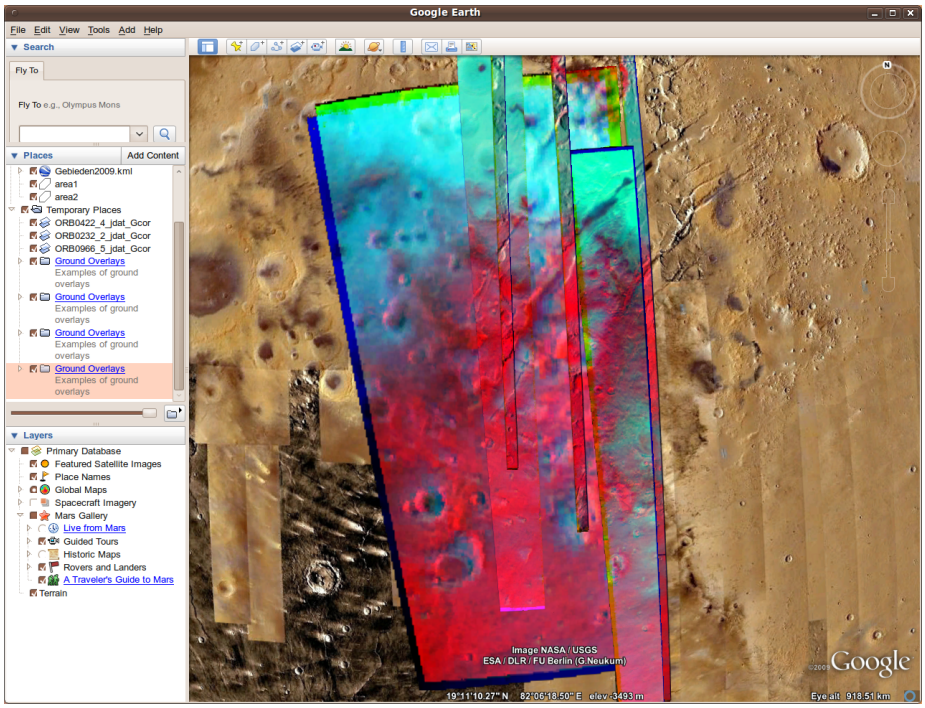


Figure 3.6: OMEGA images viewed in Google Mars.

Figure 3.6 shows a number of OMEGA images in Google Mars.
Figure 3.7 shows a mineral map in Google Mars.

## 3.6 Import Image

This program can be used to convert an image in one of the popular formats into the ENVI format. Only those formats are supported that are supported by the Python Imaging Library (PIL). These formats are currently supported: BMP, GIF, IM, JPEG, MSP, PCD, PCX, PIXAR, PNG, PPM, PSD, SGI, SPIDER, TGA, TIFF, XBM, XPM. This is not the full list. Check the PIL documentation [7] for a complete overview.

The output ENVI image will have 1, 3 or 4 bands.

## 3.7 Fix Nodata

This program can be used for converting unwanted values—no data values—to IEEE not a number (NaN). The advantage is that NaN's are not taken into account in statistics and other operations. The disadvantage is that the image is converted to float data (only floats can contain NaN's), which might be more bulky than the original image.

The Nodata field can be specified as a list of space-separated simple expressions. The list consists of:

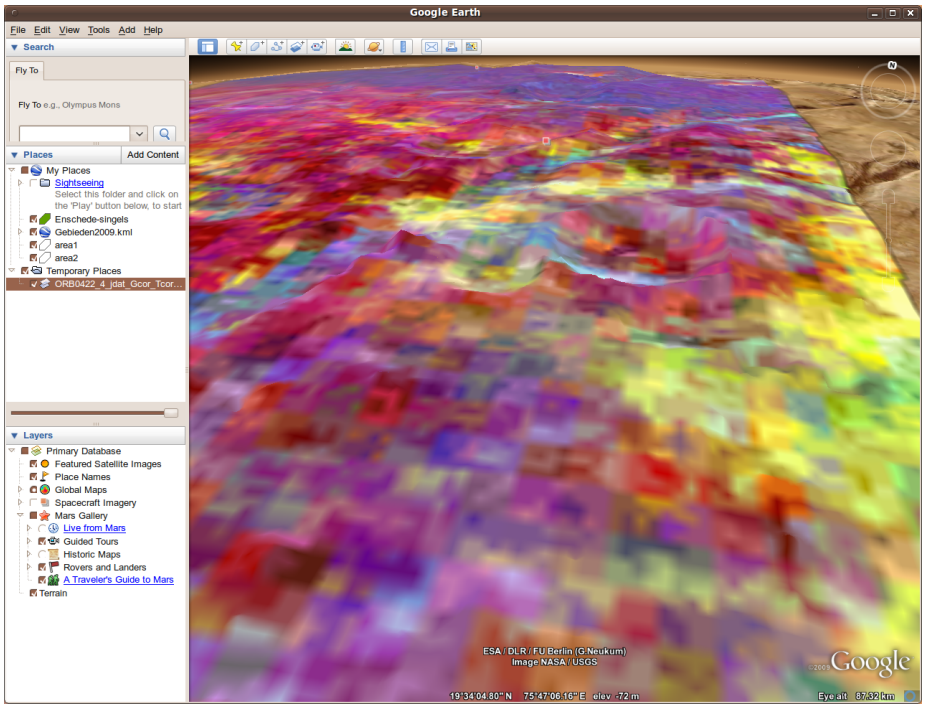


Figure 3.7: A mineral map draped on top of Mars using Google Mars.

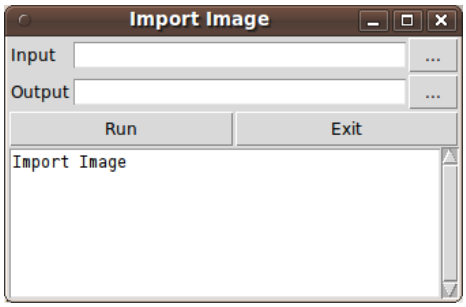


Figure 3.8: The Import Image user interface.

- single numbers.
- a number preceded by a greater sign '>' for all the values greater than or equal to the number.
- a number preceded by a less than sign '<' for all the values less than or equal to the number.
- two numbers separated by a dash '−' for the range of values between and equal to the two numbers.

Example nodata list: 0 >30000. This will turn the values 0 and 30000 and above to NaN.

## 3.8 Replace Values

With this tool values or a range of values can be replaced by a single new data value. This is useful for replacing, for instance, a range of values to no data values (NaN's).

Enter a list of no data values separated by spaces.

Use a number for a single value, e.g. 65535.

Use >number for everything greater (or equal) than number, e.g. >30000.

Use <number for everything less (or equal) than number, e.g. <10.

Use number−number for a range of values, e.g. 500−1000. The last example will set all values between 500 and 1000 to the new value, including the values 500 and 1000.

## 3.9 Convert OPUS files

This program can be used for converting, plotting and analyzing (sets of) OPUS files. OPUS files contain spectra and additional information created by the OPUS IR and Raman spectroscopy software of the company Bruker Optics.

Input can be one file or a pattern of files. Only one dataset can be analyzed at the time. Run the program multiple times if you need to analyze multiple datasets. The following types of datasets are supported:

- Refl — reflectance spectrum
- AB — absorbance spectrum
- TR — transmittance spectrum
- IgSm — sample, interferogram
- PhSm — sample, phase spectrum
- ScSm — sample, single channel spectrum
- IgRf — reference, interferogram
- ScRf — reference, single channel spectrum

The following options are supported:

- Average Spectra by Basename — files with the same basename (the file name with the extension stripped off) are averaged prior to plotting and conversion
- Plot Spectra — the spectra are plotted in a separate plot window
- Plot Stdev of Spectra — plots the standard deviation
- Write Data to Text File — convert the opus data to text. The basename of the text file is the same as the OPUS file or the basename of the set of OPUS files. The added extension of the text file reflects the type of the dataset (.Refl, .IgSm etc.). The output is two-column text with wavenumber/value pairs.

If a file name is supplied for the Speclib (spectral library) then the data will be converted to an ENVI Spectral Library file as well. Use the description box to enter a meaningful description for the spectral library. Tick the 'Enforce Micrometers in Speclib' checkbox if you want micrometers rather than wavenumbers in the spectral library.

Pressing the 'Save Logfile' button saves all the text from the textbox into a file. This can be useful for storing and analyzing the metadata of the OPUS files.

The OPUS Converter user interface is shown in figure 3.9.

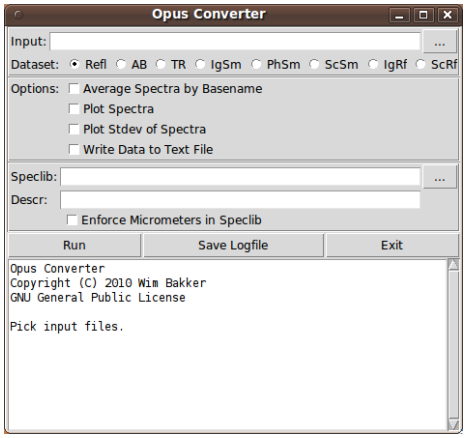


Figure 3.9: The OPUS Converter user interface.

This program was created on request of Chris Hecker.

## 3.10 Convert Meteosat files

This program converts a set of Meteosat GeoTIFF's into one multiband ENVI image. Byte images are calibrated to temperature using separate calibration files. Only the calibration files for the years 1999 to 2004 are currently available.

Files are entered as a pattern of file names. The pattern can contain shell-style wildcards. Use a question mark '?' for any character. Use an asterisk '*' for any number of characters.

The acquisition time of the images (taken from the file name) is converted to proleptic Gregorian time, which is the number of days since the beginning of the first day of 1 AD (01-01-0001), plus the fraction of the day. The proleptic Gregorian time is stored in the wavelength list of the ENVI image. The original time stamp is stored in the band names list of the ENVI image.

The georeference is copied from the first GeoTIFF of the stack. The program has proven that it can easily stack more than 40,000 Meteosat images.

The Meteosat Converter user interface is shown in figure 3.10.
This program was created for the MSc thesis work of Nadira Khan [20].

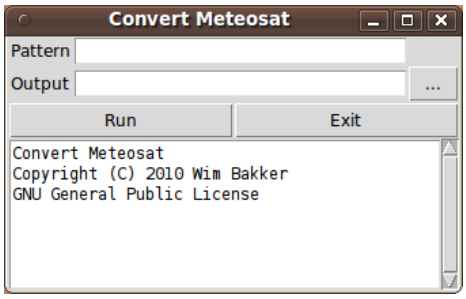


Figure 3.10: The Meteosat Converter user interface.

## 3.11 Covert SIT files

Convert files from a thermal camera.

## 3.12 Convert EOS HDF Files

Convert EOS HDF files into something more useable.

## 3.13 Convert ASTER Files

Convert ASTER HDF format files into ENVI files.

## 3.14 WMS get

This program can be used to obtain data from a Web Map Server (WMS). The options should be selected from top to bottom, because some options depend on earlier options.

# Chapter 4

# Tools

## 4.1 Continuum removal

The Continuum Removal program does a convex hull removal. For each pixel at calculates the convex hull (the continuum) and removes it either by division or by subtraction. Subtraction is safe to use if the spectra are true reflectance spectra, that is all the values are between 0 and 1. If that is not the case, it is safer to use the 'Divide' option. The convex hull is calculated and removed pixel by pixel, which makes it a rather slow process. Basically, the continuum

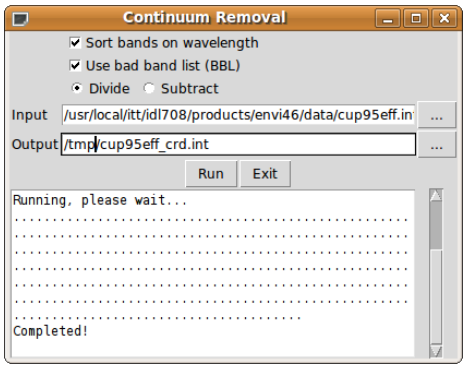


Figure 4.1: The Convex Hull Removal user interface.

removal normilizes a spectrum to value 1, expect in absorption features where the value will be less than one.

Figure 4.2 shows a color composite of a continuum removed image.

The continuum removal program uses a modified version of the Quick Hull algorithm described in [9].

## 4.2 Band Normalization

As a result of the Band Normalize program the output bands will all have a mean of 0 and a standard deviation of 1.

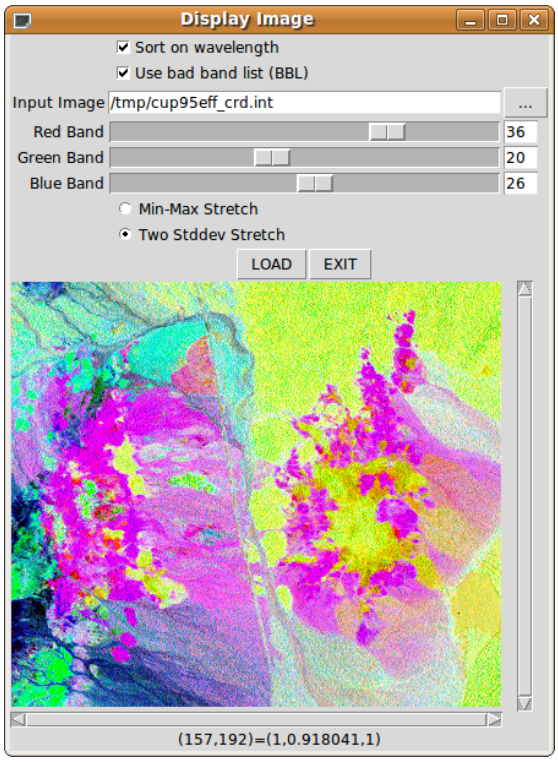


Figure 4.2: A continuum removed image shown in the Color Display.

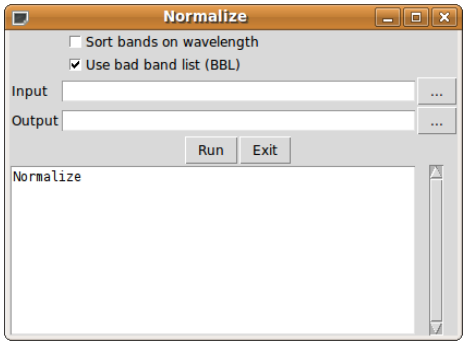


Figure 4.3: The Band Normalize user interface.

The following code does the trick.

```
for i in range(im.bands):
    band = im[i]
    im2[i] = ( band - band.mean() ) / band.std()
```

The image object *im* is the input image, *im2* is the output image. The variable *band* is an array that holds one band of data at the time. Numpy arrays have methods for calculating the mean and standard deviation.

## 4.3 Log/Kwik Residuals

This section describes the operation of the Log Residuals program. The log residuals method is a two-step normalization method in which the spectra of the pixel are normalized first and then pixel are divided by the mean spectrum of the whole image.

There are different ways to do the double normalization [26]. The first methods calculated a geometric mean rather than an arithmetic mean. The geometric mean can be calculated efficiently in log space, hence the name of the method 'log residuals'. It seems that in general the difference between the arithmetic and geometric means is not dramatic and quick (kwik) methods use the arithmetic mean rather than the geometric mean. However, for most application the Log Residuals seems to work best. This program calculates arithmetic means if the option 'KWIK Residuals' is selected (default). This program geometric arithmetic means if the option 'LOG Residuals' is selected.

To make the mean spectrum of the entire image less sensitive to noise a statistical method is used to determine this spectrum. The statistical least upper bound (SLUB) spectrum is determined as follows (per channel):

$$SLUB_i = min((scene\_avg_i + N * scene\_stdev_i), scene\_max_i) \tag{4.1}$$

In which:

- $scene\_avg_i$ is the scene's averag value for band $i$.
- $scene\_stdev_i$ is the scene's standard deviation for band $i$.
- $scene\_max_i$ is the scene's maximum value for channel $i$. In fact, to make the algorithm even less sensitive to noise, to top 1% of the values are ignored.
- $N$ is the user-supplied factor, typically something like 3.0 for 3 standard deviations. In a normal distribution only 1% of the values would be larger than the mean plus 3 times the standard deviation.

Loosely put, the SLUB spectrum is a spectrum constructed from "pretty bright" values in the image.

Setting $N$ to 0 sets the SLUB spectrum to the scene average spectrum.

The purpose of the algorithm is to remove albedo effects (caused by topography etc.) and to remove atmospheric effects. The latter only works if the atmosphere is constant over the entire scene. At the same time, it also assumes that the image is rather heterogeneous. If the image contains a spectral feature that is present in the entire image then it will *not* be visible in the resulting image anymore.

Figure 4.4 shows the interface to the log residuals program.

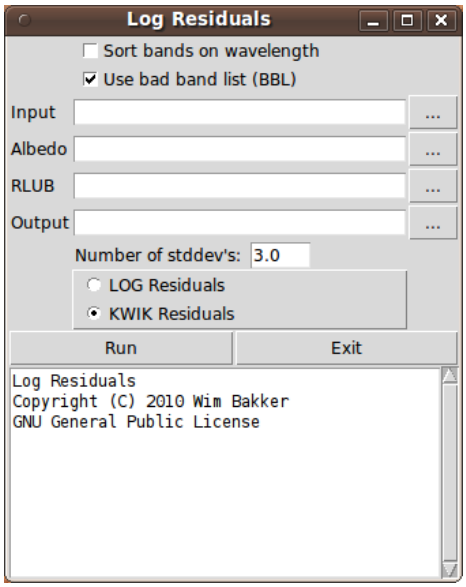


Figure 4.4: The Kwik Residuals user interface.

## 4.4 Band Ratio

The Band Ratio program (not to be confused with the next program called Band Ratios) calculates the ratio of two bands. If the image contains a wavelength list then bands must be specified as wavelengths. If the image does not contain a wavelength list then the two numbers are taken as band numbers. Note: band numbers start counting at 0.

The result is a one-band float image which is the ratio of the two bands.

## 4.5 Band Ratios

The Band Ratios program calculates the ratios of all the adjacent bands in the input image. This can be very useful for quickly obtaining an idea of which absorption features are present in the image. The 'Delta' parameter sets the distance between two 'adjacent' bands. If delta is set to three than the output image will contain the bands b1/b4, b2/b5, and so on. The output image will have less bands than the input image, (bands-delta) to be precise.

Output file names will automatically be appended by '_rat' plus the delta used.

Figure 4.6 shows one ratio band around the 2.2 $\mu$m absorption feature of clay minerals.

## 4.6 Absorption features

The Band Depths program is similar to the Band Ratios program, but uses 3 bands at the time instead of 2 bands for the calculation. The parameter 'Delta' can be set to higher values to look for broader absorption features. By

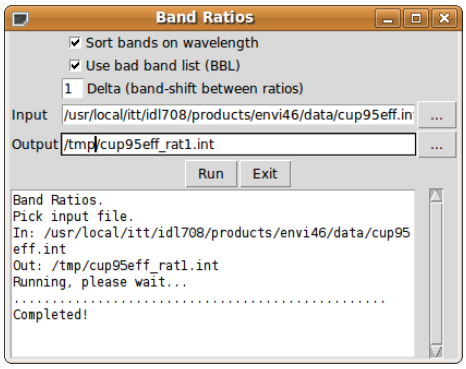


Figure 4.5: The Band Ratios user interface.

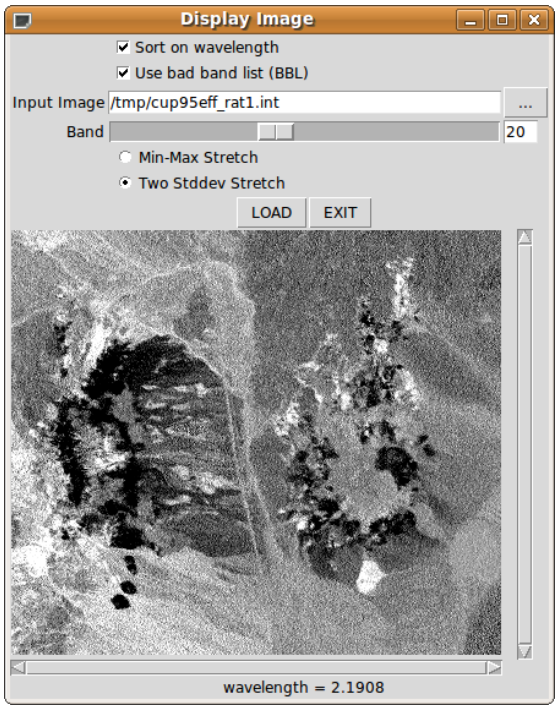


Figure 4.6: A ratio band shown in the Display.

default, the output file name is appended with '_spikes' plus the value of the delta parameter.

Figure 4.8 shows one output band around the 2.2 $\mu$m absorption feature of clay minerals.

## 4.7 Band Math

This tool can be used for applying a formula on a number of input bands.

Inputs are a number of files. In the formula the files are numbered from i1

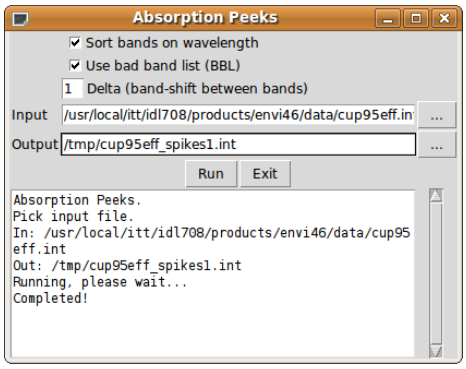


Figure 4.7: The Band Depths user interface.

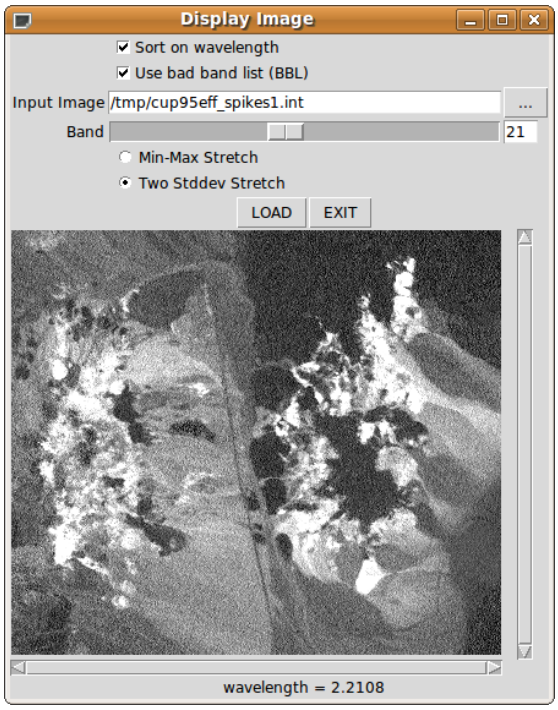


Figure 4.8: An absorption band shown in the Display.

to i20. Bands are indicated either by band number or by wavelength. Band numbers must be given between square brackets, wavelengths must be given between round brackets. Please note that band numbers start at number 0 (zero), which my differ from the convention of other software packages!

In case a wavelength is used, the closest wavelength in the image is used. There is no guarantee that the requested wavelength is actually used. It is the responsibility of the user to make sure that the right band is selected. Make sure that the units in the formula correspond to the units of the image (micrometer, nanometer or whatever).

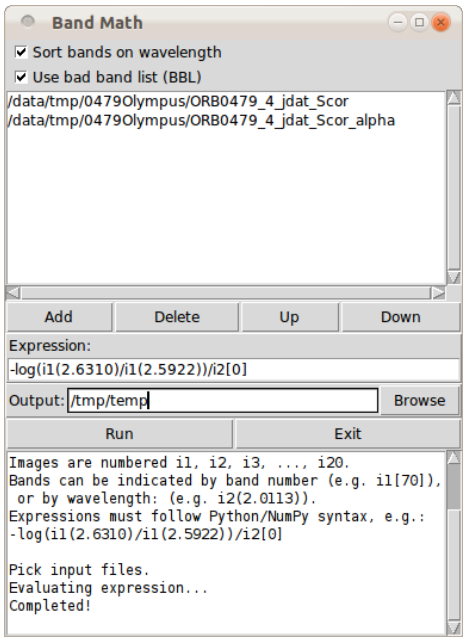


Figure 4.9: The Band Math user interface.

The input bands must all have the same number of samples and lines.

Expressions must be valid Python/Numpy expressions. Image bands are presented as 2-dimensional numpy arrays, which means that the functions used must be able to handle arrays. The output is also assumed to be a numpy array. The output will be saved as a 1-band ENVI file.

Example expression: $-log(i1(2.6310)/i1(2.5922))/i2[0]$

Using the numpy 'roll' function convolutions can be done. Please note that x-direction has axis=1, and y-direction has axis=0! The following example does an edge detection in x-direction:

Example filter: $roll(i1(2.6310), 1, 1) - roll(i1(2.6310), -1, 1)$

Here's another example, showing the least 4 bits of an image:

Example: $i1[51]$ & $0b1111$

Calculate the minimum of two bands, minimum of band 100 and band 101:

Example: $array((i1[100], i1[101])).min(axis = 0)$

## 4.8 Spectral Math

TODO

## 4.9 Wavelength of Minimum

This program determines the wavelength of the smallest value of the spectrum. If a range is defined using 'start' and 'end' then the minimum is determined

in this wavelength range. The resulting image is a 1-band image in which the values denote 'wavelength'.

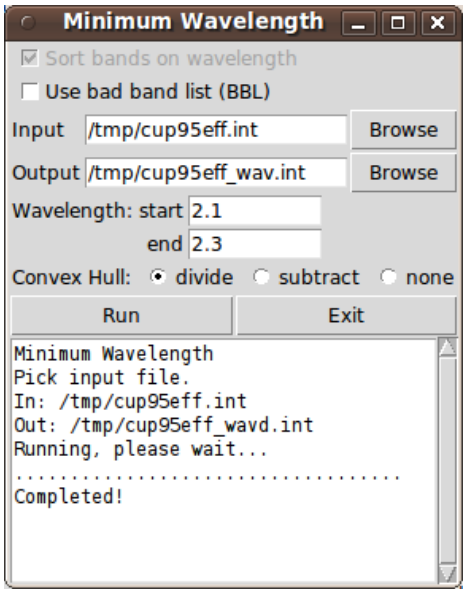


Figure 4.10: The Minimum Wavelength user interface.

The program can do a continuum removal first. You can chose to 'divide by' or 'subtract' the convex hull. If set to 'none' the continuum removal is skipped. An example can be found in figure 4.11.

The following output bands are generated:[1]

1. minimum wavelength—the wavelength of the minimum value found in the spectral range
2. interpolated min. wav.—the wavelength of the minimum of the interpolated parabola
3. interpolated depth—depth of the parabola at the minimum
4. interpolated narrowness—narrowness of the interpolated parabola
5. wavelength of 1st local min—wavelength of the first local minimum. In rare cases this may differ from the wavelength of the minimum.
6. depth of 1st local min—depth at the first local minimum
7. wavelength of 2nd local min—wavelength of the second local minimum. This may indicate a second absorption peak in the spectral range. Use this with care, not every local minimum is an absorption peak.
8. depth of 2nd local min—depth at the second local minimum

[1] note: current version has an extended set of 19 bands!

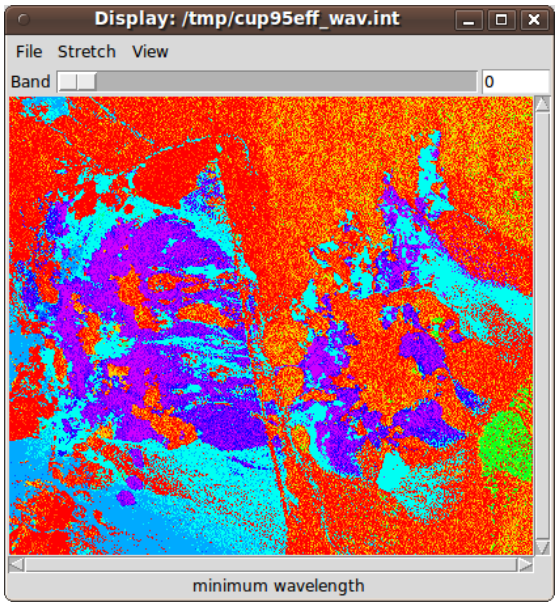


Figure 4.11: A pseudo-color image of the Minimum Wavelength of the Cuprite image. Wavelength range used was 2.1 $\mu$m–2.3 $\mu$m.

9. wavelength of 3rd local min—wavelength of the third local minimum. Very often these last two bands will contain garbage. On clear spectra this may indicate a third absorption peak in the spectral range.

10. depth of 3rd local min—depth at the third local minimum

The wavelength of (local) minimum combined with the depth at the (local) minimum appear to be very useful for producing mineral maps.

The current version supports the use of a mask image. The mask image can be used to focus the calculations only at certain areas of interest, this may considerably speed up the process.

The mask image must have the same number of lines and samples as the input image. The values 0 and $NaN$ are taken as $False$, all other values are taken as $True$. If your mask image contains more than one band then only the first band is taken into account.

In order to be able to compare the image with the generated legend, the output RGB image should not be stretched in a viewer. In the ENVI header the default stretch is set to '0 255 linear'.

## 4.10 Wavelength Mapping

This program takes the output of the previous program and produces a color map by using a color table on the interpolated wavelength of minimum and HSV merges it with the interpolated depth of the feature.

The resulting map can be used the mineral interpretation of the area. The map is particularly useful for areas with alteration zones. See figure 4.13 for an example wavelength map of the Cuprite area.

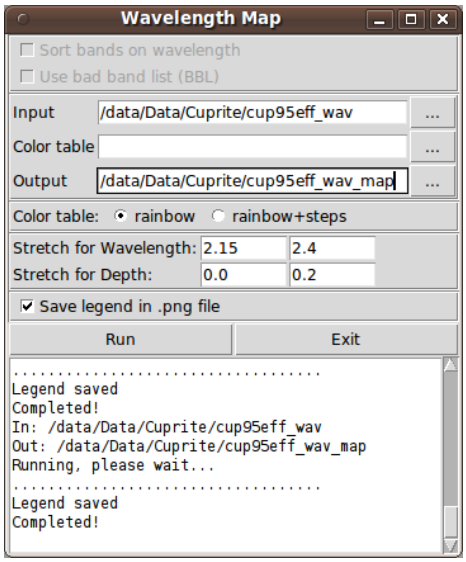


Figure 4.12: The Wavelength Mapping user interface.

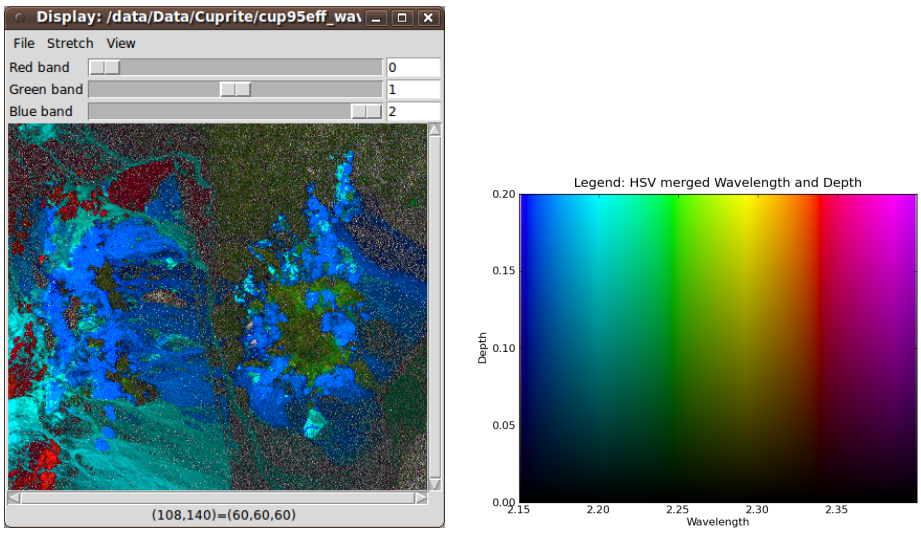


Figure 4.13: Left: Wavelength Map of the Cuprite image. Wavelength range used was 2.15 $\mu$m–2.40 $\mu$m. Right: legend of the Wavelength Map of the Cuprite image.

If no name for color table file is given, then the internal rainbow or rainbow plus brightness steps is used. Color table files should have 256 lines with each line containing a triplet for the R, G and B values ranging from 0 to 255.

Check the "Save legend" checkbox for a legend. The legend will have the same base name as the output image file.

## 4.11 Colorize

This program is derived from the Wavelength Mapping tools described above, but offers more flexibility regarding the files and bands used.

## 4.12 Destriping

The Destriping program is used for removing striping patterns in images. It can remove horizontal striping and vertical striping. Use the 'subtract' option for removing striping patterns caused by additive effects. Use the 'divide' option for removing striping patterns caused by multiplicative effects.

Figure 4.14 shows the interface of the Destriping program.

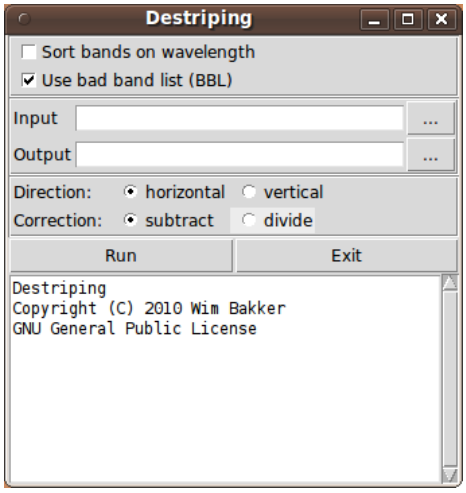


Figure 4.14: The Destriping user interface.

Figure 4.15 shows the dramatic effect of destriping performed on an old Landsat MSS scene.

## 4.13 Mask noisy bands

This program uses a signal to local busyness ratio (SLBR) for detecting noisy bands. The signal to local busyness ratio appeared to perform better than the signal to noise ratio (SNR). The bad bands are recorded in the bad band list (BBL) of the image. Only the header information is changed; nothing is changed on the image data itself!

The original BBL will be remembered. If you're not satisfied with the result then you can redo the noisy band masking with a different threshold.

Setting the threshold to zero will restore the original BBL.

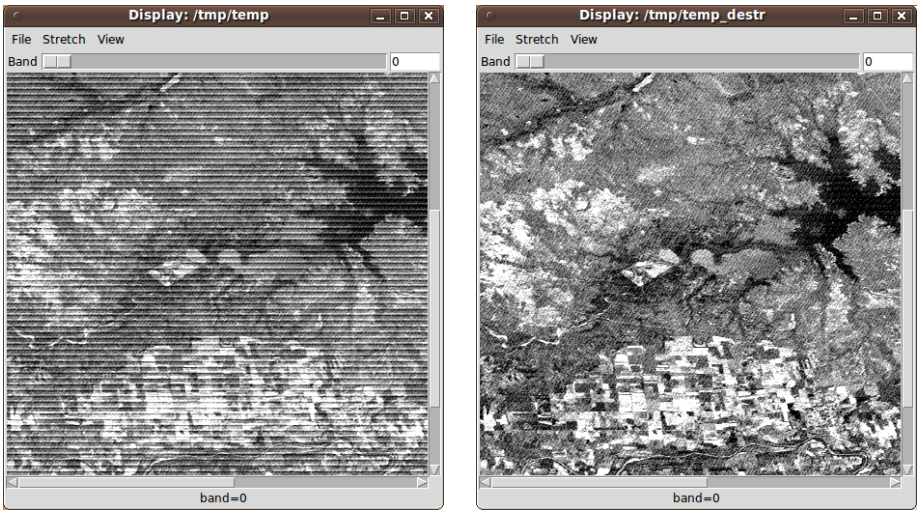


Figure 4.15: Before and after destriping of a Landsat MSS band.

A higher threshold will result in more bands being marked as bad.

Figure 4.16 shows the interface of the Mask Noisy Bands program.

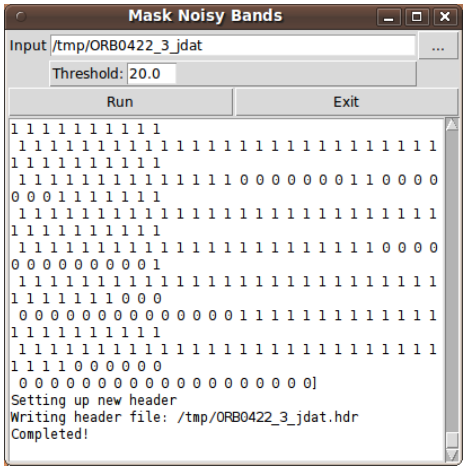


Figure 4.16: The Mask Noisy Bands user interface.

The program will plot the SLBR for all bands. See figure 4.17. The red line indicates the used threshold. The plot can be used to quickly assess which bands will be marked as bad (the bands below the threshold). If you're not satisfied simply run the program again with a different threshold.

## 4.14 Geocorrection using Lat/Lon

This program is used for geocoding images using a so-called latitude/longitude file. The lat/lon file must contain the coordinates for every pixel in the input

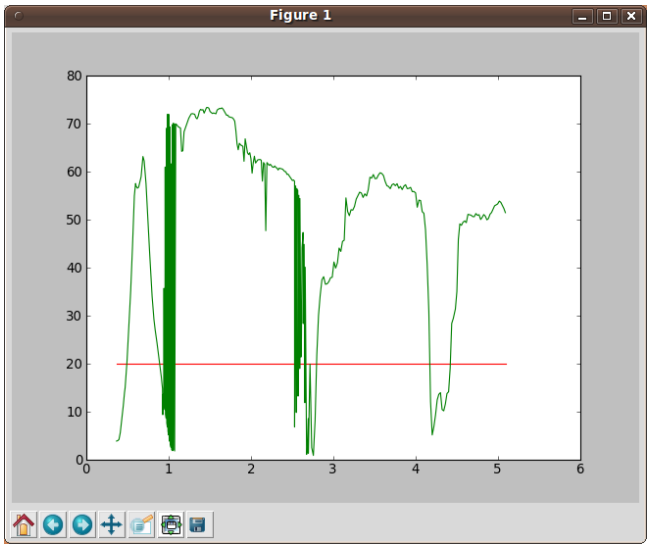


Figure 4.17: The Mask Noisy Bands user interface.

image. Hence, both images must have the same number of lines and samples. The lat/lon file at least must have two bands with the band names 'Latitude' and 'Longitude'.

In order to avoid gaps in the output, the program takes output by output pixel and searches for the nearest input pixels. This has the advantage that it avoids having to use a gap-filling algorithm afterwards. The disadvantage is that the searching is quite slow. So, be patient!

No-data pixels in output are set to the NaN (not a number) value.

Figure 4.18 shows the interface of the Geometric Correction program.

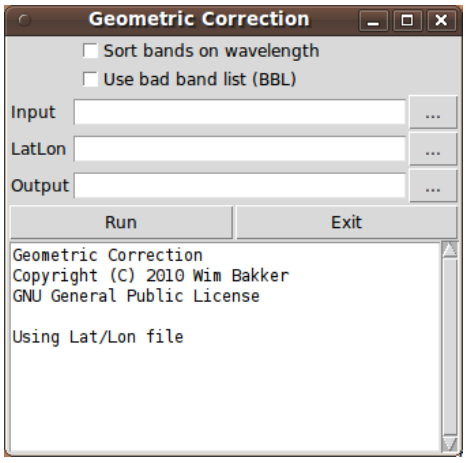


Figure 4.18: The Geometric Correction user interface.

See also also the OMEGA Geometric Correction program in section 9.2,

which uses a GeoCube instead of a lat/lon file.

## 4.15 Geocorrection using GLT

A GLT file contains coordinates of pixels can be used to do a geocorrection on the image.

This program is capable of doing a forward as well as a backward GLT correction. Reversing the GLT may be useful for correcting image artefacts that remained uncorrected before doing the GLT geocorrection.

## 4.16 Resample image

Resample image by a certain step size.

## 4.17 Flip image in X, Y or Z

Flip image along a certain axis. Can only be done at one axis at the time.

# Chapter 5

# Transform

## 5.1 Principal Component Analysis

This program does a Principal Component Analysis using a fast SVD algorithm [17]. The SVD decomposes $X$ into three matrices $USV$, in which $U$ are the scores, the diagonal $s$ of $S$ contains the singular values and $V$ contains the Eigenvectors. The square of $s$ gives the Eigenvalues. The scores $U$ are returned as and output image. The singular values $s$ are plotted in a diagram. The PCA is performed by subtracting the mean first and then do a fast SVD. The mean values Xm, the singular values s, the Eigenvectors V and the wavelengths of the original image are recorded in a separate stats file.

Figure 5.1 shows the interface of the Principal Component Analysis program.

Figure 5.2 shows the singular values.

If a subset is selected then the analysis is done on the subset. The scores $U$, however, will be returned for the entire image. This is useful if the user is only interested in a particular part of the image, or if the image is too large for the SVD algorithm.

The user can also exclude bands from the analysis by making a selection first using the Select Bands. The resulting scores will have as many bands as there are bands selected.

If you image contains NaNs then you should use the Exclude NaNs option. The program will first analyse the image and kick out the bad spectra that contain NaNs.

Figure 5.3 shows the first 10 principal components of an image. Note how the main component of this image—the albedo—ends up in the first component of the pca.

## 5.2 Inverse Principal Component Analysis

This program does an inverse PCA transform. As input it takes the resulting scores plus the statistics file generated the PCA program. The user can select components using the Components Selector. Useful options include leaving out

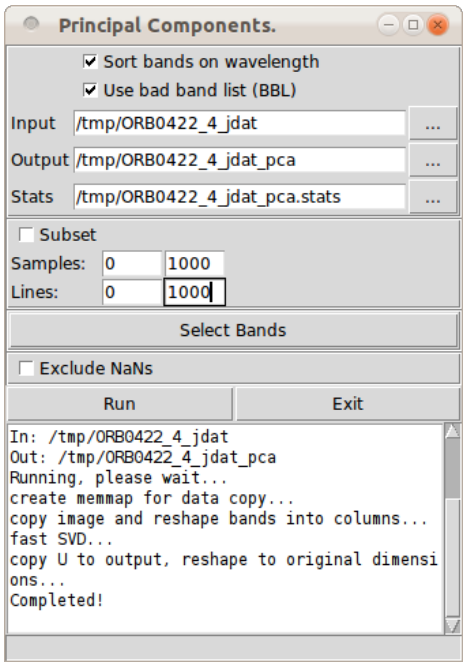


Figure 5.1: The Principal Component Analysis user interface.

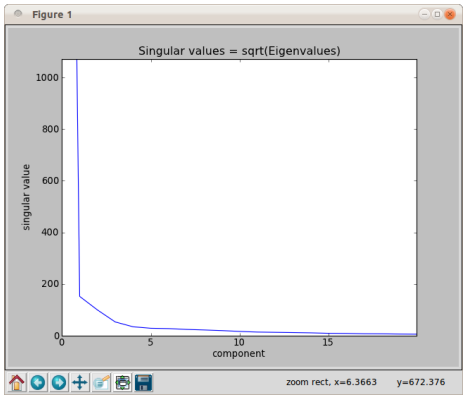


Figure 5.2: Singular Values.

the first component (if it contains the albedo) or leaving out components that contain only noise or unwanted patterns.

Figure 5.4 shows the Inverse PCA user-interface.

Figure 5.5 shows the Components Selector.

Figure 5.6 shows the result of an inverse PCA transformed image leaving out the first component that appeared to contain the albedo. Subtle difference in reflection stand out more clearly in this result.

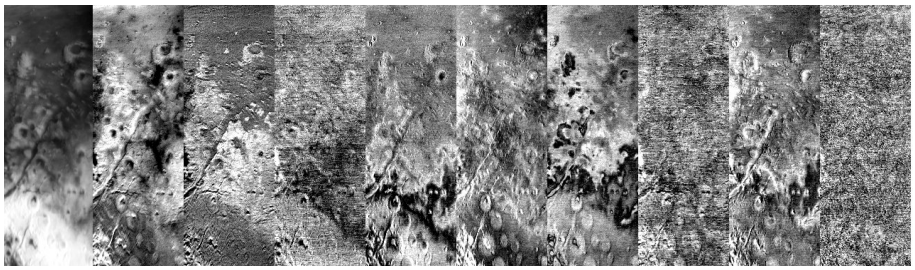

Figure 5.3: First 10 principal components of an image.

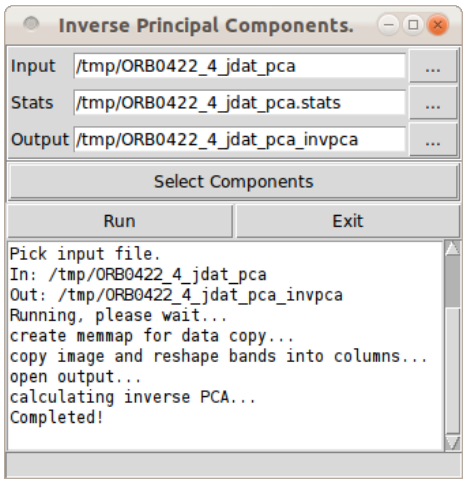


Figure 5.4: The user-interface of the Inverse PCA program.

## 5.3 Hough transform for circles

Program for calculating the Hough transform for circles.

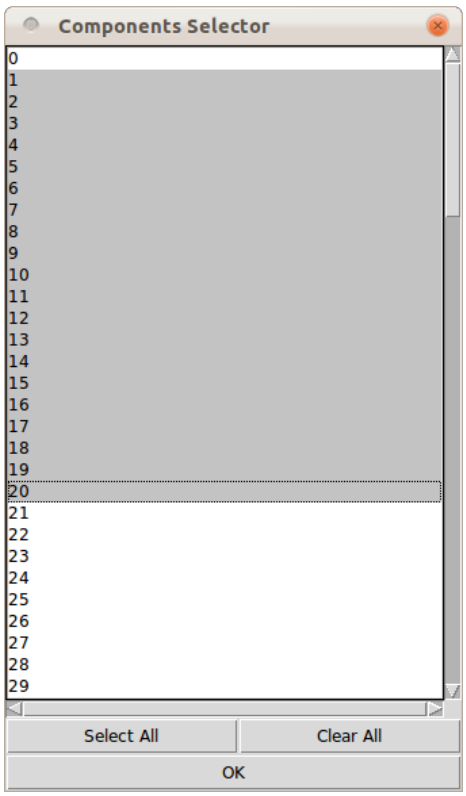


Figure 5.5: The Components Selector.

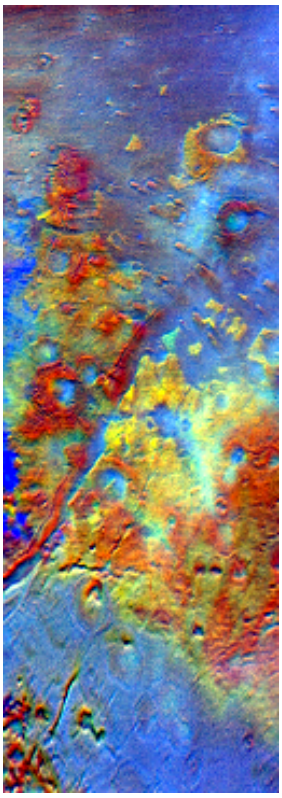

Figure 5.6: Color composite of an inverse PCA transformed image of Mars.

# Chapter 6

# Filters

## 6.1 Linear Filter 3x3

This offers a standard linear convolution filter of 3x3 filters. The chosen filter operates on all bands of the image.

Standard output format is float. However, choose a bias and offset if you want the output to be within a certain range.

Example kernels can be found in tabel 6.1.

## 6.2 Spectral Gradient Filters

This program (figure 6.1) is an extension of the Edgy program and can calculate up to 9 different hyperspectral gradients using five different spectral distance measures.

The following gradients are available:

- X gradient — calculates the gradient in x-direction.
- Y gradient — calculates the gradient in y-direction.
- X+Y — sum of x and y gradient.
- Up diagonal — calculates the gradient diagonally from the lower-left to the upper-right.
- Down diagonal — calculates the gradient diagonally from the upper-left to the lower-right.

| | | |
|---|---|---|
| 1 | 1 | 1 |
| 1 | 1 | 1 |
| 1 | 1 | 1 |

(a) mean

| | | |
|---|---|---|
| 0 | -1 | 0 |
| -1 | 4 | -1 |
| 0 | -1 | 0 |

(b) laplace

| | | |
|---|---|---|
| 0 | 0 | 0 |
| -1 | 0 | 1 |
| 0 | 0 | 0 |

(c) gradient x

| | | |
|---|---|---|
| 0 | -1 | 0 |
| 0 | 0 | 0 |
| 0 | 1 | 0 |

(d) gradient y

| | | |
|---|---|---|
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

(e) sobel x

Table 6.1: Several filter kernels

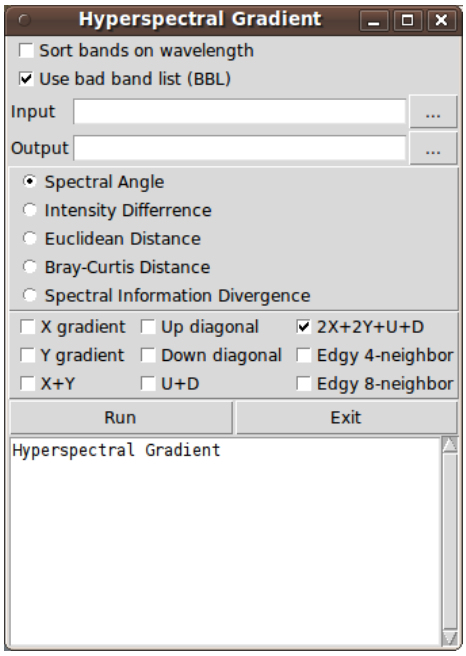


Figure 6.1: The Hyperspectral Gradients user interface.

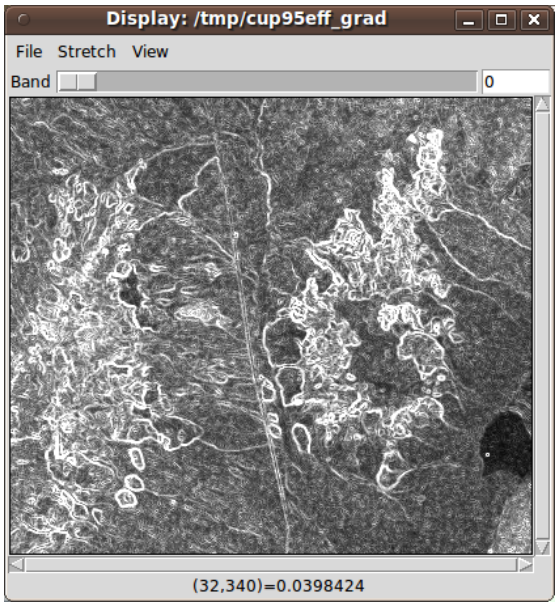


Figure 6.2: The 2X+2Y+U+D Gradient.

- U+D — sum of Up and Down gradients
- 2X+2Y+U+D — weighted average of the x, y, up and down gradients. X and y gradients are weighted heavier than the diagonals.

- Edgy 4-neighbor — sum of distance between the central pixel and its 4 closest neighbors (left, right, up and down).
- Edgy 8-neighbor — weighted sum of distance between the central pixel and all its 8 neighbor pixels. The four closest pixels are weighted double.

The edge map generated by the 2X+2Y+U+D gradient appears to be a good input for hyperspectral image segmentation and is therefore the default (figure 6.2).

If only one gradient is selected then the output will have one band. Selecting more than one option will result in a multi-band output image. The band names reflect the gradients that were selected.

By default the output name will have '_grad' appended to it.

## 6.3 Hyperspectral edge detection

The program Spectral Edge Filter is used for hyperspectral edge detection filtering. The method was described in [16]. Basically, the method calculates local spectral variability. The interface looks like figure 6.3. First you should chose

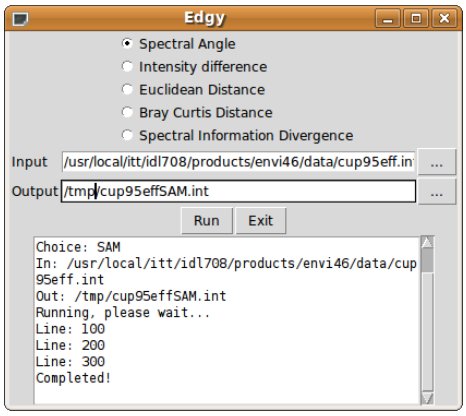


Figure 6.3: The Spectral Edge Filter user interface.

the right filtering method. The choices offered are:

- spectral angle, a.k.a. SA
- intensity difference
- Euclidean distance
- Bray Curtis distance
- spectral information divergence, a.k.a. SID

Every filtering method has its advantages and disadvantages. Simply put, the spectral angle looks at the 'color' variation in the scene, while the intensity difference looks at brightness variation. The Eclidean distance gives the variation

in feature space and behaves like a combination of the spectral angle and the intensity difference.

The Bray-Curtis distance and spectral information divergence are two other distance measures that are widely used in statistics and image processing.

The input image must be a hyperspectral image in ENVI format. The entire spectral information is used to calculate the single-band result, the edge map. The edge map can be used for visual interpretation or in further processing. High values indicate areas with a high spectral variability, while low values indicate spectrally homogeneous areas.

When you pick the input file, by pressing the ... button, an output file name is automatically generated by appending a postfix to the name (SA, ID, ED, BC or SID). This output name can be overridden by changing the name in the text box or by picking a new output file by pressing the ... button next to the text box.

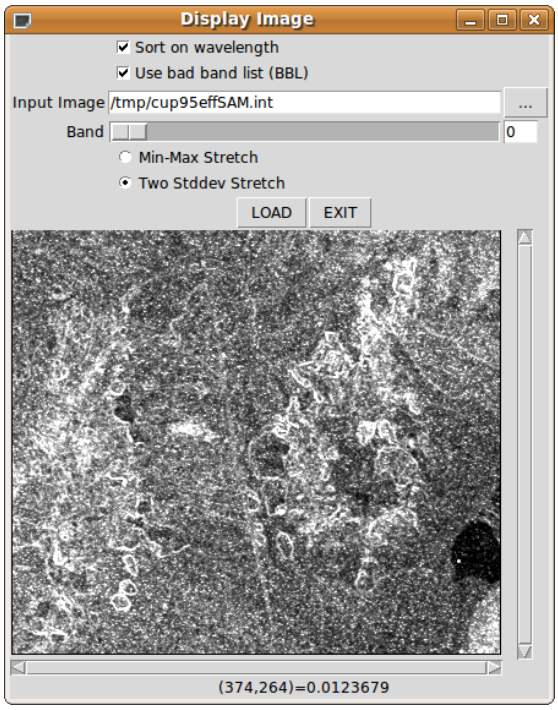


Figure 6.4: An hyperspectral edge map shown in the Display.

## 6.4 Hyperspectral Median and Mean filtering

The Spectral Median Filter program smooths the input image by applying a 3-dimensional 3x3x3 median filter. It offers two options. The first option takes into account the full 27 value neighborhood. The second option takes only values into account, the values above, below, left, right, in front and behind the central value.

To a certain extent, median filtering is more edge-preserving than averaging. It is also less sensitive to outlier values.

Because median filtering requires sorting, it is rather time-consuming.

The edges of the input are copied to output to prevent errors in further processing.

The interface of Spectral Median Filter is shown in figure 6.5.

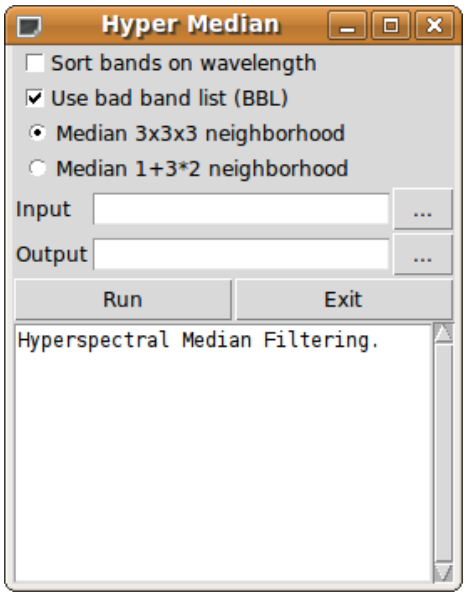


Figure 6.5: The Spectral Median Filter user interface.

# Chapter 7

# Classification & Segmentation

## 7.1 Spectral Mapper

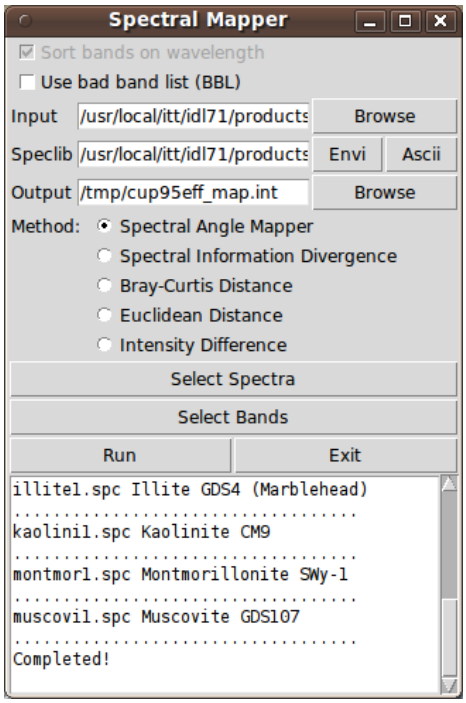


Figure 7.1: The Spectral Mapper user interface.

The Spectral Mapper is sort of a clone of the ENVI Spectral Angle Mapper but offers more choice for the distance measure used. In addition to the traditional spectral angle also the euclidean distance, intensity difference, Bray-Curtis distance and Spectral Information Divergence can be used.

For the end-member collection an ENVI or an ASCII spectral library must be used. A subset of the collection can be made using the Select Spectra button

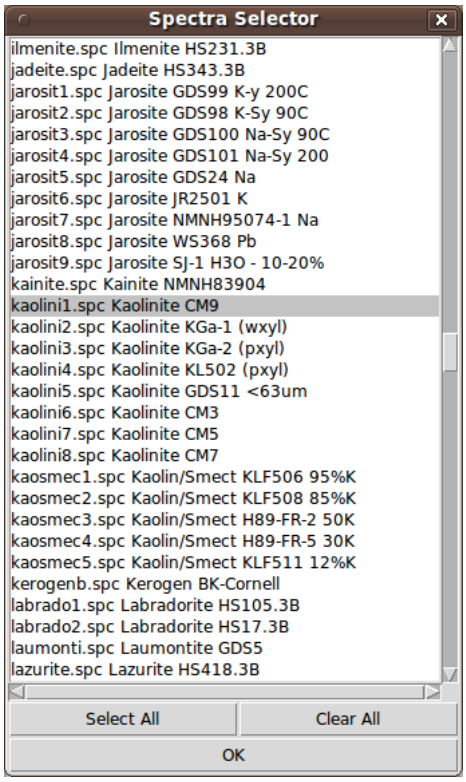


Figure 7.2: The Select Spectra window.

(figure 7.2). If no selection is made then the whole end-member collection is used. The spectra from the spectral library are resampled on the fly into the wavelength bands of the input image. This means that it is not necessary to resample the spectral library into the wavelengths of the image in advance of using the Spectral Mapper. This is also the reason why the Sort on Wavelengths checkbutton is disabled, because for resampling process to work the wavelengths must be ordered.

For every selected end-member a rule image will be created as a separate band in the output. The band names of the output reflect the names of the end-members used. The Display program can be used for collecting end-member spectra. Look for the View Values window.

A specific set of wavelengths can be selected by using the Select Bands button (figure 7.3). Wavelengths that are not selected will not be taken into account in the classification process. By default all the wavelengths are selected.

The rule image automatically has '_map' in its name. This can be changed. Figure 7.4 shows an example of the rule image for the end-member Kaolinite.

The rule images can be classified used the Rule Image Classifier.

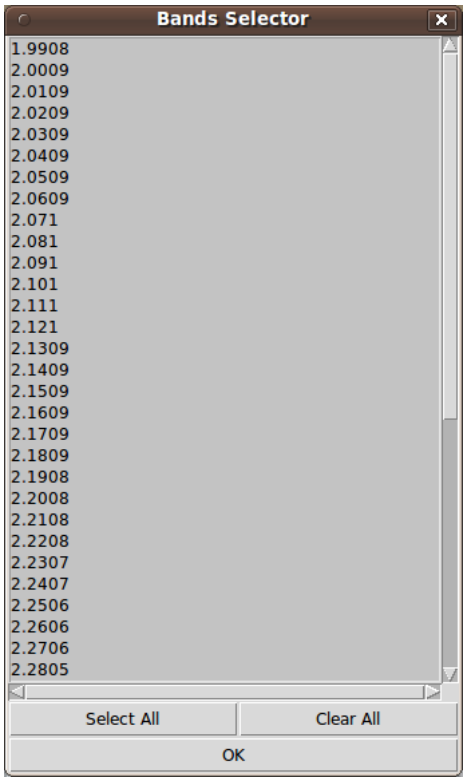


Figure 7.3: The Select Bands window.

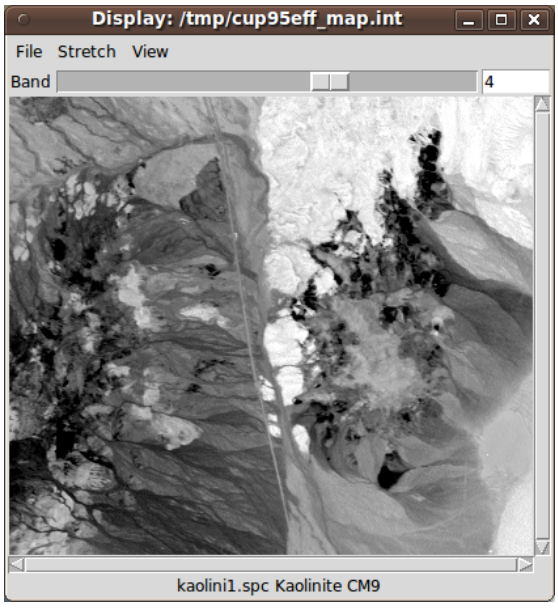


Figure 7.4: The rule image for Kaolinite.

## 7.2 The Rule Image Classifier

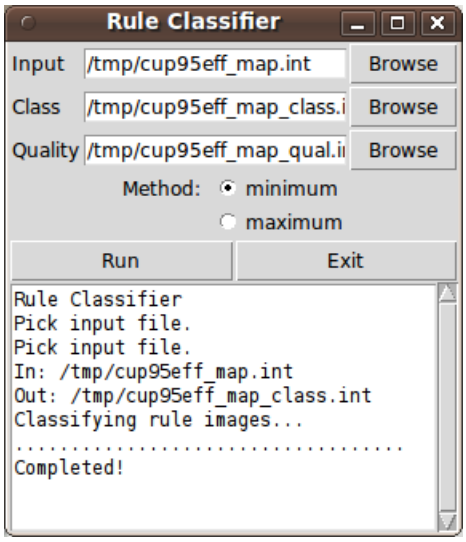


Figure 7.5: The Rule Image Classifier user interface.

The Rule Image Classifier (figure 7.5) classifies the rule images by per pixel selecting the class with the minimum distance. The output is an ENVI Classification image. Random colors are assigned to the classes. Class names reflect the names of the original end-members used in the Spectral Mapper. Figure 7.6 shows the result of a classification.

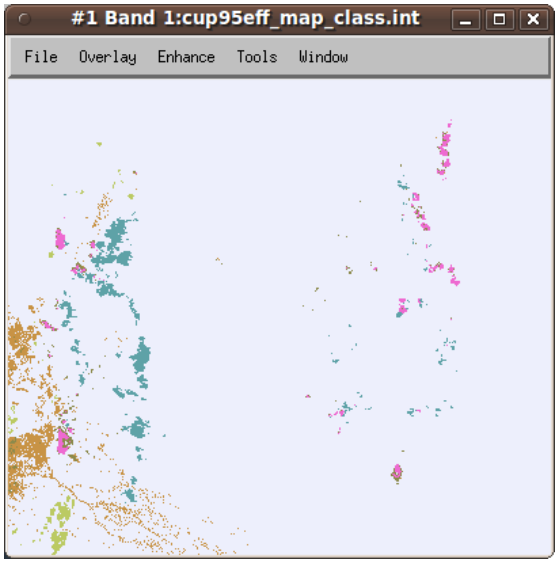


Figure 7.6: The result of the classification shown in an ENVI display window. Pink is Kaolinite, brownish is Illite.

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

## Edge Map

Before we can run the split & merge segmentation using quad trees [18, 19, 21, 15, 14] we have to create the (hyperspectral) edge image (edge map) first. The edge map can be generated using the Spectral Edge Filter or the Spectral Gradient Filter. The edge map must have one band only. An important choice for the generation of the edge map is the choice of the distance function. Usually this will be 'spectral angle', which is the default.

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

Steps:

- Start the Display program.
- Go to File → Open and open the edge map.
- Go to View → x-profile.
- Click somewhere in the image. The x-profile plot will pop up.

What is the lowest level? Going slightly above this value will give you an indication for the choice of the split level in the next step.

## Split & Merge

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

Steps:

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

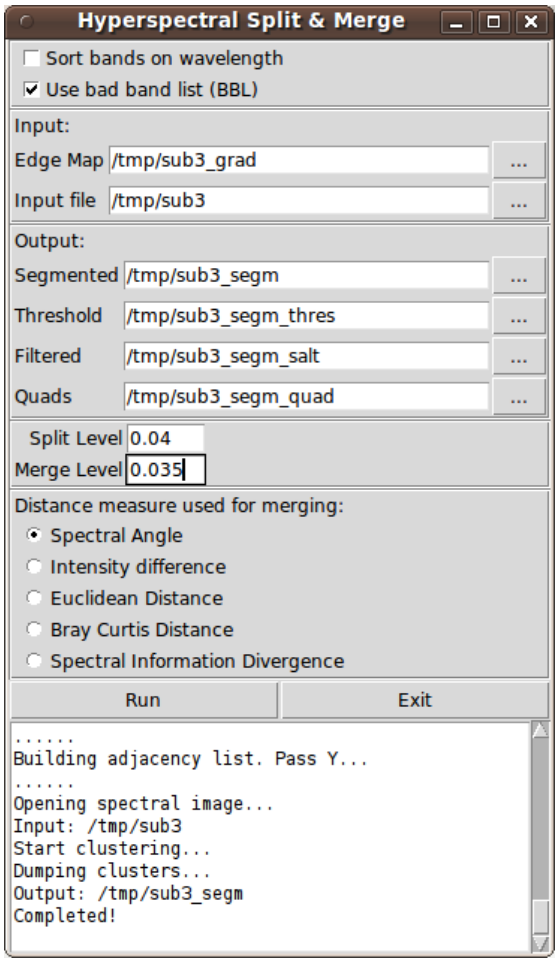


Figure 7.7: The user-interface of the Segmentation program.

- Press 'Run'.

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

## Generated files

1. filename+'_segm_thres' — The split level is used the threshold the edge map. This file contains the result. Black is considered 'homogeneous', white is considered 'heterogeneous'.
2. filename+'_segm_salt' — The script does a pepper & salt filtering on the edge image. This image contains the result of that filtering.
3. filename+'_segm_quad' — This file contains the quads as they are generated by the split phase of the segmentation.

4. filename+'_segm' — This file is the final result of the segmentation and contains the merged clusters.

Try to modify the split level and the merge level to see what effect these parameters have on the split phase and the merge phase. Inspect intermediate images. The '_quad' and '_segm' images are ENVI Classification images and contain a random color table for the classes.


Figure 7.8: Left: the input image, agriculture near Wichita, Kansas. Middle: the gradient edge map. Right: result of the segmentation.

Figure 7.8 show the result of the segmentation of an image of an agricultural area near Wichita, Kansas [27].

# Chapter 8

# Special Tools

## 8.1 Dark & White Reference

Calculates a reflectance image from raw DN values using a dark current and a white reference image. If a Specim 'manifest.xml' file is selected then the input and output files are automatically selected.

If the option 'robust' is selected then only the 50% brightest pixels of the white reference image are used for the correction. This is useful if the white reference plate contains specs of dust. This is currently the default option. Make sure that there are enough white reference lines taken during the recording process. A number of 200 lines for the white reference is recommended.

For the output image four output data types can be chosen: 16-bit signed integer, 16-bit unsigned integer, 32-bit floating point and 64-bit floating point.

## 8.2 Destriping Filter

Experimental filter for removing noise by a sort of recalibration of the detector chip. Use this with care; the result may be worse than the original.

The correction is done by calculating the mean and standard deviation of a detector cell, compare this to its neighbouring cells, and apply a correction if needed.

Really bad detector cells are replaced by NaNs (not-a-number) and can be replaced with the data from neighboring cells.

## 8.3 Fix SWIR 8th pixel problem

Corrects for bad columns in the data from the Specim SWIR3 camera. Every eighth column is replace by the average of its neighbours.

## 8.4 Keystone

This program tries to estimate the keystone from an image. The image should be homogeneous in Y and Z direction but have a pattern in X direction. A

simple sheet with printed vertical black lines is sufficient.

The program works by scaling in X direction and a matching technique using normalized cross-correlation. This process is rather sensitive and is able to detect sub-pixel differences between different bands.

Usually the chromatic aberration of the lens will have a larger effect than the keystone of the spectrograph. Therefore, this program can be used to find to closest optimal focus distance of your lens. It is advised to keep the keystone / aberration less than one pixel.

## 8.5 Smile

This program tries to estimate the smile of a hyperspectral camera by analysing an image. The image should be homogeneous in X and Y direction but have a pattern in Z (spectral) direction. In the visible range this could be an image of a white paper or reference plate illuminated by a fluorescent tube. The spectral lines from the tube can be used for finding the optimal stretch and bias in spectra. For the optimization it uses a normalized cross-correlation. This process is sensitive and is able to detect a smile that is well below one pixel.

The smile of your camera cannot be changed by external influences. The smile of a well-designed camera will be in the order of maximal 0.2 pixels.

# Chapter 9

# Mars Tools

Figure 9.1 shows the suggested processing chain for OMEGA data.

## 9.1 Mask noisy bands

This program is described in section 4.13.

## 9.2 Geocorrection using GeoCube

The Geocorrection program uses the GeoCube generated by the Soft05 OMEGA calibration software to geocode the OMEGA images. All the 352 original OMEGA bands must be present. See also section 4.14.

## 9.3 Mars Solar Correction

The Solar Correction program corrects for the solar irradiance curve by dividing the radiance data by a standard solar irradiance curve (figure 9.2). The Martian solar irradiance curve is available in the wavelength.dat and specsol_0403.dat data files. These files should be present for the program to work. This program only corrects for solar irradiation *not* for topographic effects. Which means that there may still be albedo effects present caused by the topography of the terrain.

Furthermore, it does *not* take into account the Mars-Sun distance. This distance varies between 204.52 million km and 246.28 million km, which means that the real irradiance may be $\pm 16\%$ off. The Mars-Sun distance of a particular image can be found in the header of the cube file.

The program does not account for thermal effects, therefore the program should not be trusted above the 3 $\mu$m range without doing a thermal correction first.

Figure 9.3 shows the user interface to the Solar Correction program. For the program to work correctly the bands must be sorted on wavelength. The irradiance spectrum is resampled to the wavelengths of the image bands and then used for the correction.

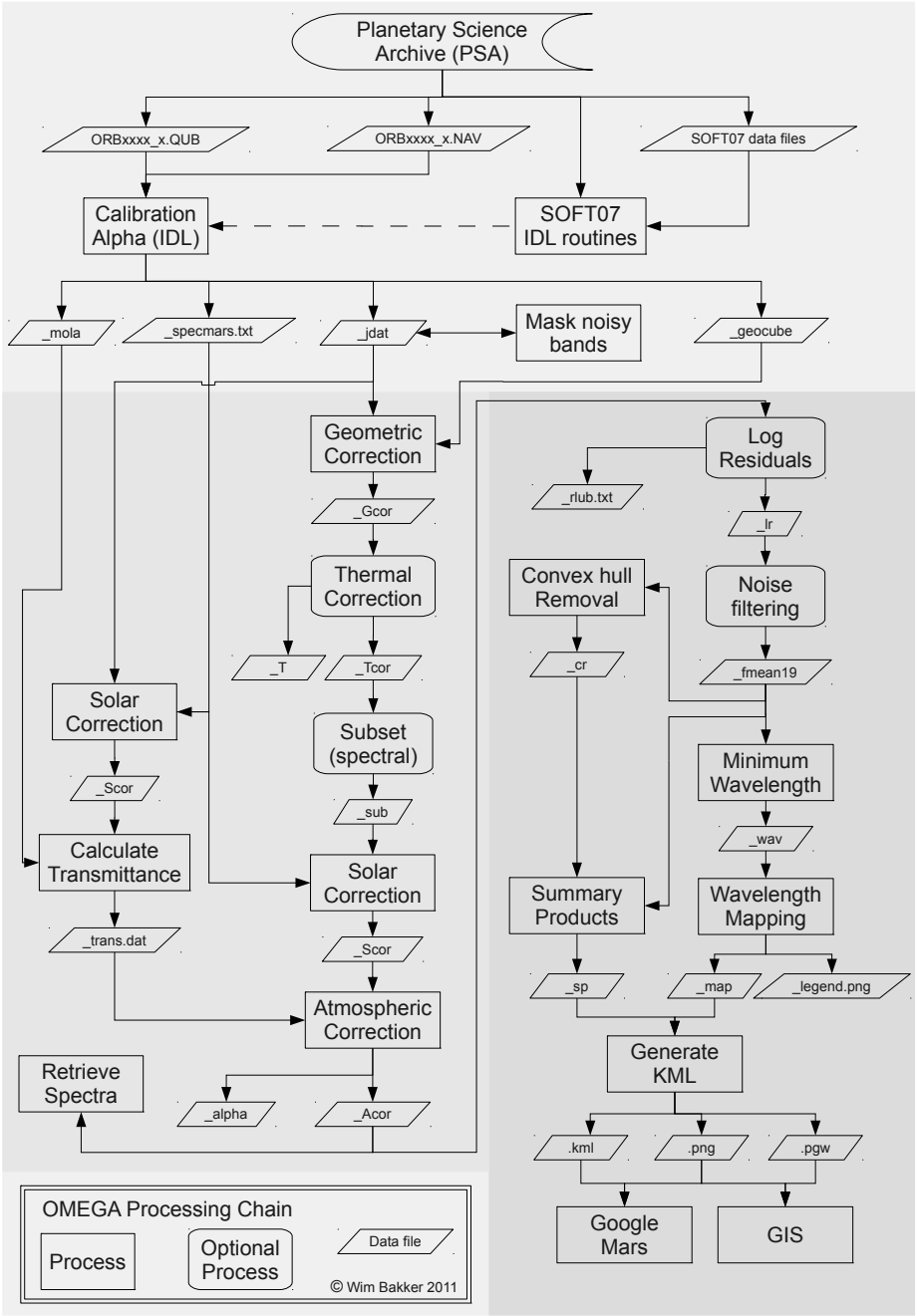


Figure 9.1: OMEGA Processing Chain.

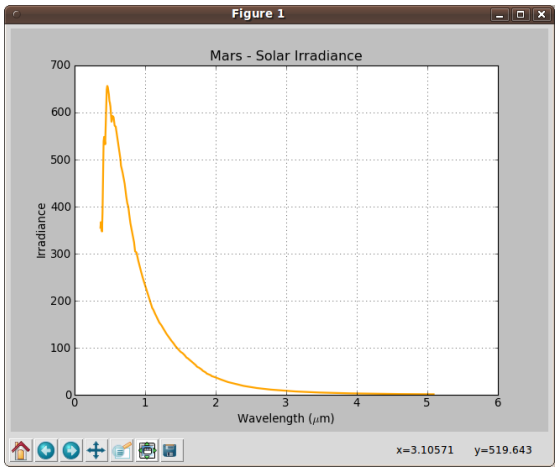


Figure 9.2: Solar Irradiance at Mars.

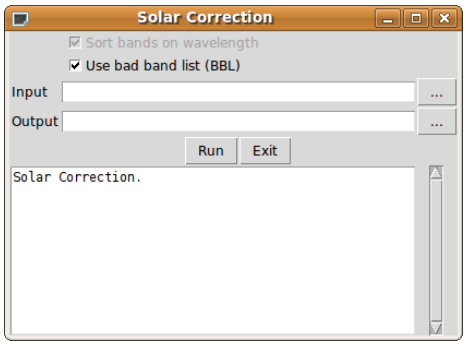


Figure 9.3: The Solar Correction user interface.

## 9.4 Calculate Transmittance

Use this tool for obtaining a transmittance model. As input it takes a solar corrected radiance image and a height model (MOLA). The output is an estimation of the transmittance at elevation 0 for the current image.

Transmittance $t$ can be expressed as a function of the optical depth[1] $\tau$ [28, 24]:

$$t = e^{-\tau} \tag{9.1}$$

It should be noted that the optical depth $\tau$ itself is an exponential funtion of the height $z$, which makes the $t$ a double exponential function of the elevation $z$ [24].

The optical depth is proportional to the elevation $z$:

$$\tau \sim e^{-\frac{z}{H}} \tag{9.2}$$

The scale height $H$ is a distance over which a quantity, in this case the optical depth $\tau$, decreases by a factor of $e$.

[1] "Optical path" would be a better term, because we model the *entire* path through the atmosphere.

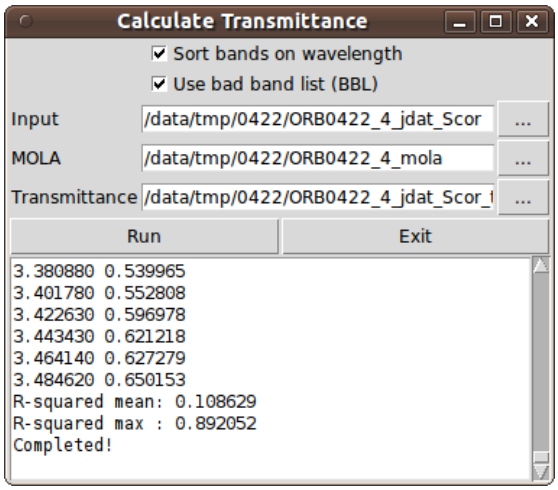


Figure 9.4: The user-interface of the Get Transmittance program.

The transmittance $t_0$ at zero elevation ($z = 0$) is then:

$$t_0 = e^{-\tau_0} \tag{9.3}$$

where $\tau_0$ is the optical depth at zero elevation.

It can be shown that:

$$t = {t_0}^{\alpha} \tag{9.4}$$

Filling in the equations for $t$ and $t_0$ we get:

$$e^{-\tau} = \left(e^{-\tau_0}\right)^{\alpha} = e^{-\tau_0 \alpha} \tag{9.5}$$

Taking the log on both sides gives:

$$-\tau = -\tau_0 \alpha \tag{9.6}$$

This gives us:

$$\alpha = \frac{\tau}{\tau_0} \tag{9.7}$$

We call $\alpha$ the relative optical depth at elevation $z$.

The observed intensity $I$ can then be expressed by the intensity without atmosphere $I_0$ times the transmittance $t$,

$$I = I_0 t \tag{9.8}$$

which gives us the following relation:

$$I = I_0 {t_0}^{\alpha} \tag{9.9}$$

If the transmittance at zero elevation is known and the intensity without atmosphere would be known, then the intensity $I$ is only a function of the relative optical depth $\alpha$.

Unfortunately, both $I_0$ and $t_0$ are unknown. But not all is lost. In the following we will show that with a few assumptions regarding $I_0$ in fact $t_0$ can be estimated from an image and its elevation model.

The first observation we make is that, for the spectral range between 1 $\mu$m and 2.5 $\mu$m the spectrum is rather flat and that the bands are highly correlated (see figure 9.5). This leads us to the first assumption, that for $I_0$ we might as well take any band within the range of 1 $\mu$m to 2.5 $\mu$m that falls within an atmospheric window. Inside the atmospheric window we may assume that the transmittance is 1, or at least very close to 1.

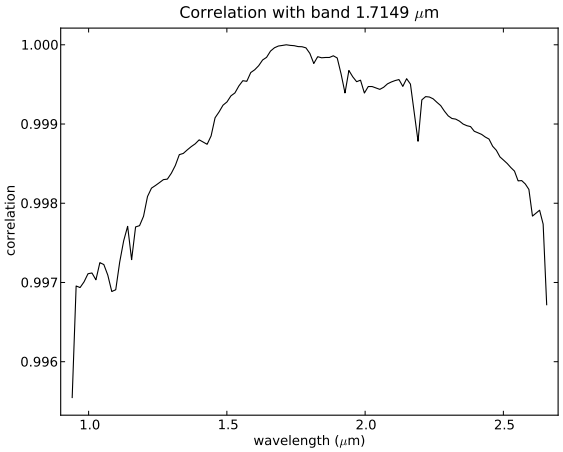


Figure 9.5: Correlation of band 1.7149 $\mu$m with the other bands. Note that the y-axis runs roughly from 0.996 to 1.000. The correlation between these bands is very high. Graph produced from image ORB0422_4, SWIR1, corrected for geometry, solar curve and atmosphere.

The atmosphere of Mars is composed of 95% carbon dioxide, 3% nitrogen, a tiny fraction of water vapor ($\pm$0.03%), and 2% other trace gases [12]. Carbon dioxide and water vapor have strong absorption features between 1 $\mu$m and 2.5 $\mu$m. Absorption by nitrogen and the trace gases does not play a role in this spectral region. For the $I_0$ we have selected the OMEGA band of 1.714 $\mu$m, because we think this band offers the best compromise of all the factors involved. The band is well within an atmospheric window of carbon dioxide (1.66 $\mu$m—1.76 $\mu$m) [25], it's at the shoulder of the broad water absorption feature (1.72 $\mu$m—2.03 $\mu$m). See figure 9.6. At the same time this band is roughly in the middle of the spectral range of 1 $\mu$m to 2.5 $\mu$m.

Unfortunately, the irradiation depends on the wavelength and therefore we must use solar corrected data.

This means that as an approximation for $I_0$ we can use $I_{1.7149}$, which is the solar-corrected radiance data ($I$) of the 1.7149 $\mu$m band of the OMEGA scene.

$$I_0 = I_{1.7149} \tag{9.10}$$

It turns out that if we plot the log-minus-log of the ratio of a particular band $I_\lambda$ and $I_{1.7149}$ against the elevation $z$, then we almost obtain a perfect straight line. In figure 9.7 all the log-log relative intensity values are plotted against altitude. This particular figure was generated from an OMEGA scene of Olympus Mons (ORB4604_5), the highest volcano in the solar system. The

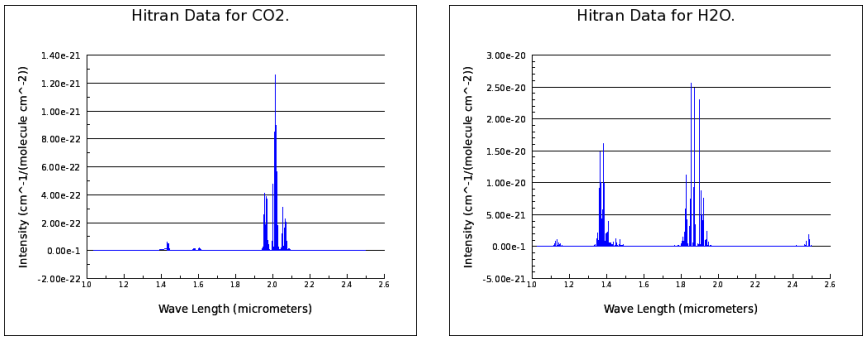


Figure 9.6: HITRAN absorption lines of $CO_2$ and $H_2O$ between 1.0 $\mu$m and 2.5 $\mu$m. Data generated using the online tool at http://savi.weber.edu/hi_plot/.

elevation of the terrain ranges from minus 655 m to plus 21,185 m, which is an elevation difference of almost 22 kilometer! For the figure, the band with most atmospheric absorption (2.0133 $\mu$m) is divided by the band with least atmospheric absorption (1.7149 $\mu$m). The 45,696 values almost form a perfect straight line. The regression using a straight line fit has an R-squared of 0.996. This shows that atmospheric transmission can be modeled using a linear fit on the double-logarithm of the relative intensity against the elevation.

Of course, this image (ORB4604_5) and this band (2.0133 $\mu$m) presents the best possible case with a large elevation difference and the deepest absorption. However, the method works surprisingly well for the other bands and other images as well. A clear advantage is that the atmospheric transmission can be estimated from the image itself. This avoids all sorts of problems regarding calibration, bad bands, different solar elevation angle and look angle, and a varying atmosphere. The only condition for a good estimate of the transmission spectrum is to have enough elevation difference in the scene, and the scene should be relatively homogeneous with a relative flat spectrum in the spectral range between 1 $\mu$m and 2.5 $\mu$m.

In other words:

$$\ln\left(-\ln\left(\frac{I_\lambda}{I_{1.7149}}\right)\right) = c_0 z + c_1 \tag{9.11}$$

The coefficients $c_0$ and $c_1$ can be determined using linear regression from the pixel values $I_\lambda$ against their elevations $z$.

It turns out that the transmittance $t_0$ at zero elevation can be derived from $c_1$.

$$-\ln\left(\frac{I_\lambda}{I_{1.7149}}\right) = e^{c_0 z + c_1} = e^{c_1} e^{c_0 z} \tag{9.12}$$

or:

$$\frac{I_\lambda}{I_{1.7149}} = e^{-e^{c_1} e^{c_0 z}} \tag{9.13}$$

looking at equation 9.9 we see that this can be split into two parts:

$$\alpha = e^{c_0 z} \tag{9.14}$$

$$t_0 = e^{-e^{c_1}} \tag{9.15}$$

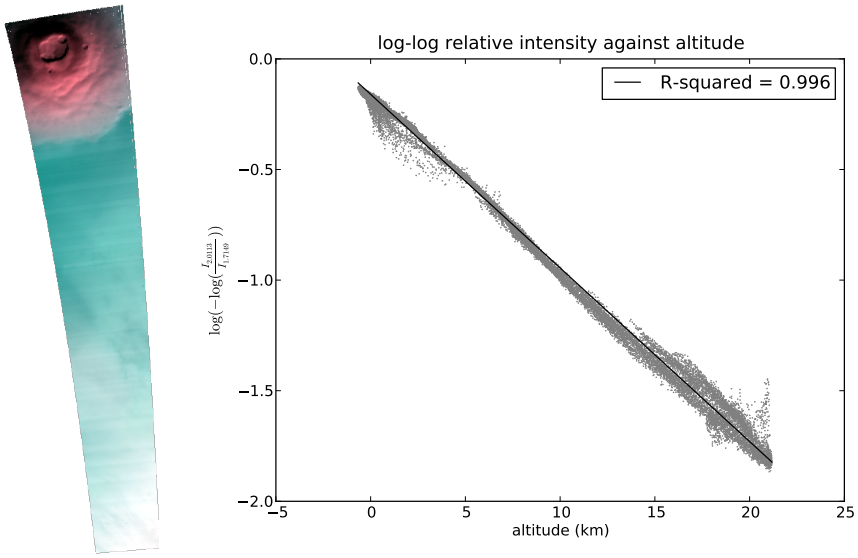


Figure 9.7: Log-log plot of the relative intensity against altitude. Image ORB4604_5 (Olympus Mons, left figure), band 2.0113 $\mu$m. For $I_0$ the band 1.7149 $\mu$m was used. The linear fit on the data has an R-squared of 0.996.

Thus the transmittance at elevation zero $t_0$ can be derived from the linear regression constant $c_1$.

Looking at equation 9.2 it is clear that the scale height $H$ can be derived from the coefficient $c_0$:

$$H = -\frac{1}{c_0} \tag{9.16}$$

The scale height only depends on the temperature and the molecular weight of the gas (equation 9.17), and does not depend on the wavelength. The scale height of the Martian atmosphere is around 11 km [12]. This means that the first 33 km contains 95% of the total atmosphere.

A second version of the Get Transmission program operates in two steps. In step one the $c_0$ is determined from the deepest absorption feature between 1.8 $\mu$m and 2.2 $\mu$m. In step two $c_1$ for every wavelength is determined by keeping $c_0$ constant.

$$H = \frac{kT}{Mg} \tag{9.17}$$

with:

- $k$ = Boltzmann constant = $1.38 \cdot 10^{23}$ J·K$^{-1}$
- $T$ = mean planetary surface temperature in Kelvin
- $M$ = mean molecular mass of dry air (kg)
- $g$ = acceleration due to gravity on planetary surface (m/s$^2$)

It must be noted that, because of the different molecular weight, the scale height for water vapor is different and that the transmission spectrum generated

is only valid for $CO_2$. In fact, because the $H_2O$ molecule is much lighter than $CO_2$, the scale height for water vapor is 2.44 (18/44) times larger than carbon dioxide.

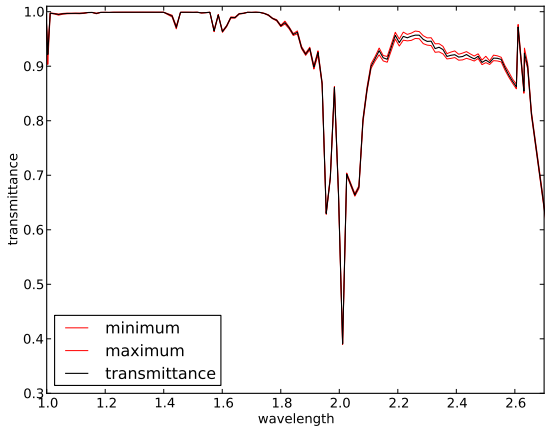


Figure 9.8: Derived transmittance spectrum at zero altitude for the OMEGA scene ORB0422_4. The minimum and maximum are an indication for the quality of the transmittance spectrum.

### Sensitivity analysis

To get an impression of the sensitivity of the derived transmittance spectrum, the image is divided into 100 random subsets and the minimum and maximum transmittance are determined from these 100 subsets. This minimum and maximum are plotted together with the derived transmittance of the entire image. The amount of variation between the minimum and the maximum gives an indication of the quality of the transmittance spectrum (see figure 9.8).

## 9.5 Atmospheric correction

The ESA uses a transmission spectrum derived by measuring the transmittance at the base and the summit of Olympus Mons. This method is used for the CRISM instrument of the Mars Reconnaissance Orbiter (MRO) [29]. This method can also be applied to the OMEGA instrument. A scaling factor is derived from the depth of the 2 $\mu$m $CO_2$ feature by applying a band ratio of two bands, one in the absorption feature and one on the shoulder of the feature. Unfortunately, this method suffers from the fact that the OMEGA instrument has developed bad bands in the 2 $\mu$m region. As a consequence the method is forced to use different band combinations. A careful recalibration must be done in order to produce comparable results to the old band combinations.

In our processing chain we developed a method that uses the transmittance either derived from Olympus Mons or derived from the image itself (see previous section 9.4), but calculates the scaling in a different way by calculating the local

busyness for the bands between 1.8 $\mu$m and 2.2 $\mu$m. The local busyness is the sum of the absolute differences between adjacent bands. This method is insensitive to bad bands, because a bad band only adds a constant to the local busyness, but does not change the shape of the curve. Furthermore, the local busyness curve as a function of $\alpha$ appears to be a smooth curve, which means that a simple minimum finding algorithm suffices to find the $\alpha$ for which the local busyness is minimal.

The optical depth $\tau$ can be expressed in terms of the incoming and outgoing radiation, respectively $I_0$ and $I$ [23]:

$$\tau = -\ln\left(\frac{I}{I_0}\right) \tag{9.18}$$

The transmission $t$ is equal to the ratio of $I$ and $I_0$:

$$t = \frac{I}{I_0} \tag{9.19}$$

We define a relative optical depth $\alpha$ with respect to a standard (average) Martian atmosphere. The relationship between the standard optical depth $\tau$, the relative optical depth $\alpha$ and the standard transmittance $t_0$ then becomes:

$$e^{-\alpha\tau} = {t_0}^{\alpha} \tag{9.20}$$

The relative optical depth $\alpha$ can be estimated from spectra by assuming that the spectrum of a pixel must be smooth in the spectral region between 1.8 $\mu$m and 2.2 $\mu$m. On the other hand, the $CO_2$ absorption feature around 2.0 $\mu$m is not smooth at all and has a triplet with three distinct absorption peaks. This triplet can be used to estimate the effect of the atmosphere. The transmittance $t_0$ can be scaled by modifying the relative optical depth $\alpha$ in such a way that the resulting spectrum is as smooth as possible. We use a local busyness operator for expressing smoothness.

The local busyness is minimized by multiplying the spectrum with a scaled version of the transmission spectrum. In other words, minimize the relative optical depth $\alpha$ such that the following expression is minimal:

$$localbusyness = \sum_{1.8\mu\mathrm{m}}^{2.2\mu\mathrm{m}} \left| \frac{I_j}{{t_0}_j^{\alpha}} - \frac{I_{j+1}}{{t_0}_{j+1}^{\alpha}} \right| \tag{9.21}$$

Where $t_0$ is the standard transmission spectrum of the Martian atmosphere and $I$ is the observed intensity. The index $j$ denotes the spectral channels of the Omega sensor.

It must be noted that light travels through the atmosphere twice and that the $\alpha$ models the entire path through the atmosphere.

Once the factor $\alpha$ is known the atmospheric correction can be done by dividing $I$ by ${t_0}^{\alpha}$:

$$I_{corr} = \frac{I}{{t_0}^{\alpha}} \tag{9.22}$$

The method works well for wavelengths up to 3.5 $\mu$m. After that, the influence of thermal radiation becomes noticeable and the correction model becomes much more complex.

As mentioned before, the factor $\alpha$ is the relative optical depth of the Martian atmosphere. The optical depth is, amongst others, related to the elevation of the Martian surface. The factor $\alpha$, recorded for every pixel, and can be compared to the elevation model of the MOLA to check the consistency. In general, the image containing the factor $\alpha$ is more noisy than MOLA. However, in some locations, like in deep fossae or deep craters, MOLA looks over-generalized, whereas the image with the factors $\alpha$ show more details in these areas.

In fact the elevation $z$ of the terrain can be derived from the scale height $H$ and $\alpha$ (see equations 9.14 and 9.16):

$$z = -H \ln(\alpha) \tag{9.23}$$

To summarize, the advantages of this method are:

- the local busyness operator is simple and fast. The curve on which the minimum must be found is smooth and a simple algorithm suffices to find this minimum.
- the method is robust with respect to bad bands. The method does not need to be changed when a different band, or, worse, when yet another band goes bad.
- the method does not need an elevation model. The advantage of this that it is not prone to errors or generalizations introduced in MOLA and that it is not sensitive to misregistration between the image data and the elevation data. The latter makes all areas with relatively steep slopes suspect.
- with a proper model of the built-up of the Martian atmosphere, the elevation could in fact be derived from the relative optical depths $\alpha$.

Figure 9.9 shows the user interface of the Atmosperic Correction program. The Atm. trans. is an input file that should contain a transmission model of the atmospere. The data is expected to be a text file with two columns. The first column is the wavelength and the second column contains the transmissivity.

## 9.6 Mars Thermal Correction

The Thermal Correction program uses the method for temperature assessment that was described in [13]. The method uses a double blackbody curve for estimating the solar irradiance and the surface tempurature. Because the system of equations is underdetermined the solution is based on a number of assumptions:

- The reflection at 5 $\mu$m is half the value of the reflection at 2.5 $\mu$m.
- The emmisivity is assumed to be constant 0.85.

The solar blackbody curve is estimated from the wavlength range between 2.2 $\mu$m and 2.5 $\mu$m. The thermal contrinbution is estimated from the wavelengths above 5 $\mu$m. Both wavelength ranges are hardly affected by the Martian atmosphere.

The second output of the program is one image that contains the reflection values for the entire wavelength range, and the estimation of the temperature.

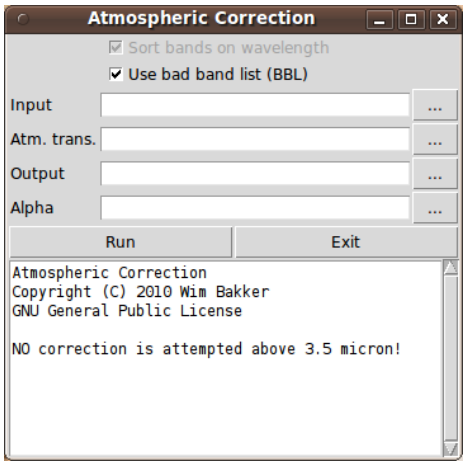


Figure 9.9: The Atmospheric Correction user interface.

The model is a gross simplification, however due to the fact that the blackbody curve is steep on the higher wavelength side the temperature image is rather accurate, probably to within -1K and +4K [13]. One effect that is not accounted for is the slight cooling effect of the Martian atmosphere, which may give a slight systematic shift towards lower temperatures.

Because the method uses wavlength ranges for the estimation, the temperature image is surprisingly clear and contains less noise than the individual channels. On the other hand, the reflectance values above 3 $\mu$m are rather noisy and should not be trusted, because of the many assumptions used in the model and because in the 5 $\mu$m range reflectance effects are small compared to the thermal effects.

The output image is the original image minus the estimated thermal contribution. To get the reflectance image from this it should be further corrected for topographic effects, atmosphere and solar irradiance.

Figure 9.10 shows the user interface of the thermal correction program.

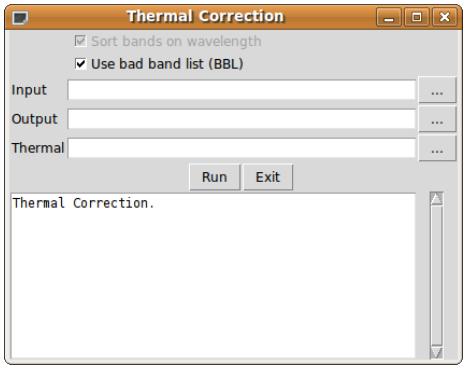


Figure 9.10: The Thermal Correction user interface.

## 9.7 Log residuals

See section 4.3.

## 9.8 Spectral median filter

See section 6.4.

## 9.9 Convex hull removal

See section 4.1.

## 9.10 Summary Products

The Summary Products program calculates a number of spectral parameters, such as ratios and band depths. The summary products are based on [22].

Table 9.1 lists the summary products that are implemented.

Figure 9.11 shows the program interface. The program needs two inputs, the original data and the continuum removed data. The continuum removed is needed for the DROP2300 and DROP2400 products. Output is one file containing all the summary products. A log file is created with the same name as the output plus the extension '.log'. Consult the log file for the real wavelengths that are used by the program. Not all wavelengths may be present in the image because of bad bands. Furthermore, the summary products are developed for the CRISM instrument, which may have different wavelength characteristics than the OMEGA instrument. The program always tries to find the closest wavelength available in the image, which may give you wavelengths that you don't want or don't expect. Use the program and the log file with care.

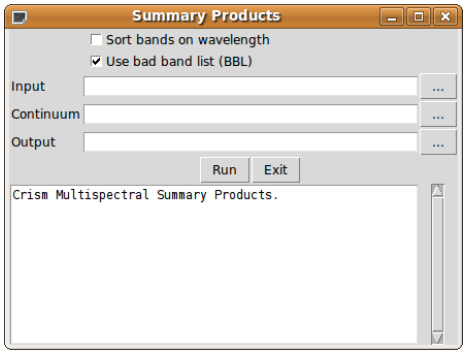


Figure 9.11: The Summary Products user interface.

Regarding correct interpretation of these summary products the user is urged to consult [22].

## 9.11 Generate KML

See section 3.5.

| **Summary Product** | **Description** |
|---|---|
| BD0530 | crystalline ferric minerals |
| BD0640 | select ferric minerals |
| BD0860 | select ferric minerals |
| BD1435 | $CO_2$ ice |
| BD1500 | $H_2O$ ice |
| BD1750 | gypsum |
| BD2210 | Al-OH minerals |
| BD2290 | Mg,Fe-OH minerals |
| BD3100 | $H_2O$ ice |
| BD3200 | $CO_2$ ice |
| BD2000 | atmospheric $CO_2$ |
| BD2600 | $H_2O$ vapor |
| OLINDEX | Olivine index |
| OLINDEX2 | Olivine index 2 |
| LCPINDEX | pyroxene index |
| HCPINDEX | pyroxene index |
| ISLOPE1 | ferric coating on dark rock |
| DROP2300 | hydrated min. particularly phyllosilicates |
| DROP2400 | hydrated min. particularly phyllosilicates |
| BD3400 | carbonates, organics |
| CINDEX | carbonates |
| RBR | rock / dust, red/blue ration |
| SH600 | select ferric minerals |
| ICER1 | $CO_2$, $H_2O$ ice mixtures |
| BD1900 | $H_2O$ |
| BD2100 | monohydrated minerals |
| ICER2 | $CO_2$ ice will be $\gg$1, $H_2O$ ice and soil will be $\sim$1 |
| BDCARB | carbonate overtones, 2.33 and 2.53 band depth |
| R410 | clouds/haze |
| R770 | rock/dust |
| IRA | IR albedo |
| IRR1 | clouds / dust |
| IRR2 | clouds / dust |
| IRR3 | clouds / dust |
| BD1270O2 | $O_2$ emission, signature of ozone |
| BD3000 | $H_2O$ |
| BD1400H2O | $H_2O$ vapor |
| R2700 | high aerosols |
| BD2700 | $CO_2$, atmospheric structure |
| D2300 | drop 2300 redefined |
| VAR | spectral variance, olivine and pyroxene will have high values |

Table 9.1: List of implemented Summary Products.

# Chapter 10

# Help

## 10.1 User Manual

This document.

## 10.2 Programmers´ Manual

Manual for programmers.

## 10.3 Scripting Manual

Manual to the command-line interface (CLI) of HypPy.

## 10.4 Spectral Math Manual

Manual for the Spectral Math functions and syntax.

## 10.5 GNU GPL License

The GNU General Public License (GPL) of the software.

## 10.6 Restore to factory defaults

Actually this doesn't restore anything at all; it simply removes the configuration file in which all the options are remembered. Are you sure this is what you want? If you want you can also edit certain sections in the file by hand. See also appendix B.

# Bibliography


[1] Basemap. http://matplotlib.sourceforge.net/basemap/doc/html/. Last visited December 2009.

[2] Matplotlib Library for 2D Plots. http://matplotlib.sourceforge.net/. Last visited June 2009.

[3] Modified Quick Hull—Python Code. http://members.home.nl/wim.h.bakker-/python/quickhull2d.py. Last visited June 2009.

[4] NumPy Scientific Computing. http://numpy.scipy.org/. Last visited June 2009.

[5] OWSlib. http://trac.gispython.org/lab/wiki/OwsLib. Last visited February 2011.

[6] Python Imaging Library. http://www.pythonware.com/products/pil/. Last visited July 2009.

[7] Python Imaging Library Handbook. http://www.pythonware.com/library/pil/handbook/index.htm. Last visited December 2009.

[8] Python Programming Language Official Website. http://python.org/. Last visited June 2009.

[9] Quick Hull Algorithm. http://en.literateprograms.org/Quickhull_(Python,-_arrays). Last visited June 2009.

[10] SciPy—Scientific Tools for Python. http://scipy.org/. Last visited June 2009.

[11] TkInter Wiki. http://wiki.python.org/moin/TkInter. Last visited June 2009.

[12] M.H. Carr. *The surface of Mars*. Cambridge planetary science series. Cambridge University Press, 2006.

[13] Jouglet D., F. Poulet, R. E. Milliken, J. F. Mustard, J.-P. Bibring, Y. Langevin, B. Gondet, and C. Gomez. Hydration state of the martian surface as seen by mars express omega: 1. analysis of the 3 mm hydration feature. *J. Geophys. Res.*, 112, 2007.

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

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

[16] Bakker W. H. and K. S. Schmidt. Hyperspectral edge filtering for measuring homogeneity of surface cover types. *ISPRS Journal of Photogrammetry and Remote Sensing*, 56(4):246–256, 2002.

[17] N. Halko, P.-G. Martinsson, and J. A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *ArXiv e-prints*, September 2009. http://arxiv.org/abs/0909.4061v2.

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

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

[20] N.A. Khan. Thermal infrared time series analysis for earthquake precursory detection. Master's thesis, University of Twente, 2010.

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

[22] Pelkey S. M., J. F. Mustard, S. Murchie, R. T. Clancy, M. Wolff, M. Smith, R. Milliken, J.-P. Bibring, A. Gendrin, F. Poulet, Y. Langevin, and B. Gondet. Crism multispectral summary products: Parameterizing mineral diversity on mars from reflectance. *J. Geophys. Res.*, 112, 2007.

[23] R. E. Milliken, J. F. Mustard, F. Poulet, D. Jouglet, J.-P. Bibring, B. Gondet, and Y. Langevin. Hydration state of the martian surface as seen by Mars Express OMEGA: 2. $H_2O$ content of the surface. *Journal of Geophysical Research*, 112:15, 2007.

[24] G.W. Petty. *A First Course In Atmospheric Radiation (2nd Ed.).* Sundog Publishing, Madison, Wisconsin, March 2006.

[25] C. Prabhakara and Joseph S. Hogan. Ozone and carbon dioxide heating in the martian atmosphere. *Journal of the Atmospheric Sciences*, 22(2):97–109, 1965.

[26] unknown. *The MMTG Manual?*, chapter Log / Kwik / LUB Residuals, pages 46–57. CSIRO, 2002.

[27] Earth observing 1 (eo-1), agricultural samples. http://eo1.usgs.gov/-sampleagriculture.php, 2006. eo1.usgs.gov.

[28] J.M. Wallace and P.V. Hobbs. *Atmospheric science: an introductory survey.* International geophysics series. Elsevier Academic Press, 2006.

[29] S. M. Wiseman, R. E. Arvidson, F. Morgan, M. J. Wolff, R. V. Morris, P. C. McGuire, S. L. Murchie, J. F. Mustard, F. P. Seelos, and M. D. Smith. Radiative transfer modeling of the empirical 'volcano scan' atmospheric correction: discussion and artifacts. Lunar and Planetary Institute, 2010. $41^{st}$ Lunar and Planetary Science Conference.

# Appendix A

# The menu configuration file

The ordinary user can skip this. If you want to copy the software and configure the top-level menu for your own purposes, continue reading.

The menu can be configured using the hyppymenu.cfg file. Every line in the configuration file contains the name of the menu entry and the function it should perform. Name and function must be separated by a semi-colon. A (sub-)menu must start with a line containing 'Menu' as a function. Every (sub-)menu *must* end with an empty line! If a menu contains a sub-menu as a last item than it must be followed by two empty lines, and so on.

Functions can be Python scripts or PDF files. Python scripts are executed by Python. PDF file are delegated to a PDF reader. The functions are executed as separate processes, which means that multiple functions can be executed at the same time.

Lines starting with '#' are ignored and can be used for comment.

The configuration file may look like this:

```
#
# HyPy menu configuration file
# every ((sub-)sub-)menu MUST end with an empty line!!!
# WHB 20091117
#
Viewers;Menu
  Display B&W (NEW!);tkDisplay2.py
  Display B&W;tkDisplay.py
  Display Color;tkColorDisplay.py
  Display Shapefile;tkViewShapefile.py
  ;Separator
  Spectral Library Viewer;tkSpecLib.py

Band Tools;Menu
  Sort Bands by Wavelength;tkSortChannels.py
  Stack ENVI Images;tkMerge.py
  Split into separate Bands;tkSplit.py
```

```
Spectral Tools;Menu
  Convex Hull Removal;tkConvexHull.py
  Normalize;tkNormalize.py
  Kwik Residuals;tkLogResiduals.py
  Band Ratios;tkRatios.py
  Band Depths;tkSpijkers.py
  Wavelength of Minimum;tkMinWavelength.py

Spectral Filters;Menu
  Spectral Edge Filter;tkEdgy.py
  Spectral Gradient Filters;tkGradient.py
  Spectral Median Filter;tkMedian.py

Classification;Menu
  Spectral Mapper;tkSAM.py
  Rule Image Classifier;tkClassify.py

Segmentation;Menu
  Generate Edge Map;tkGradient.py
  Segmentation;tkSegmentation.py

Mars Tools;Menu
  Thermal Correction;tkThermalCorrection.py
  Solar Correction;tkSolarCorrection.py
  Summary Products;tkSummaryProducts.py

Help;Menu
  User Manual;Docs/user-manual.pdf
  Programmers' Manual;Docs/programmers-manual.pdf
  ;Separator
  License;Docs/gpl-3.0.pdf
```

The lines are indented for clarity, but this has no specific meaning for the configuration.

# Appendix B

# The configuration file

This is the file that remembers all the settings you have been using. It resides somewhere in your home directory. The home directory depends on the operating system and is typically stored in an environment variable called $HOME or %HOMEPATH% or something similar.

The file follows the structure similar to what you would find on Microsoft Windows INI files. In HypPy, every script has its own section that carries the basename of the script. The section header in square brackets is followed by the option-value pairs that look like option = value, or option:value. Sections are separated by an empty line.

If you want to reset a certain option of a particular script, simply close the script, remove the option-value pair that gives you the headache, save the file, and restart the script. The option names used should be self-explanatory.