Remote sensing image preprocessing

...

Mini Project 1

Introduction to the topic

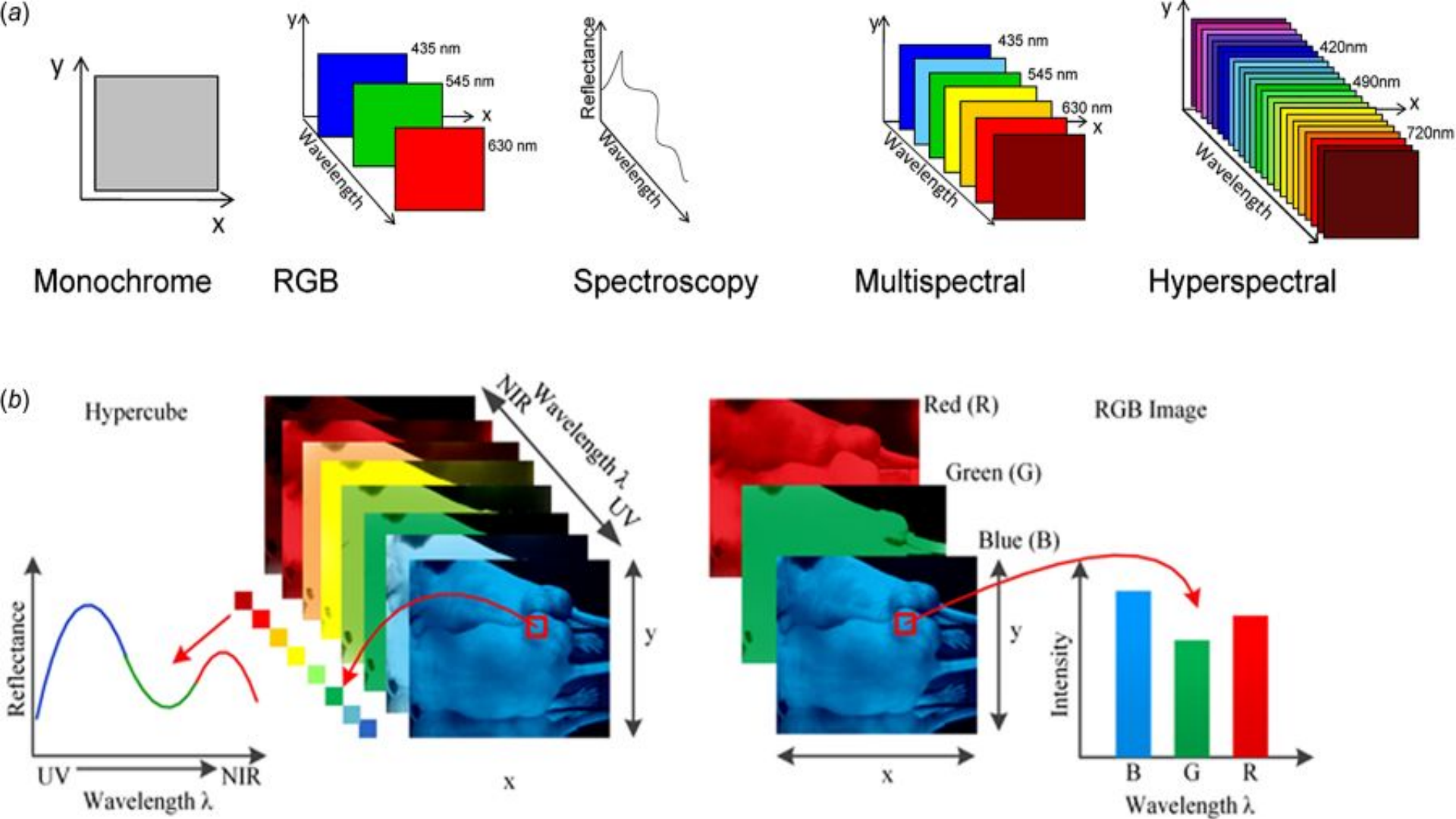
- Remote sensing images contains lot of data which can be used for many purposes:
  - Agriculture
  - Weather forecasting
  - Environment study
  - Natural hazards study
  - Resource exploration
  - Land use mapping
  - Temperature and global warming extent detection

Content

Idea

- ➔ In this project we are basically going to design a GUI to load and process hyperspectral images (.envi files).
- ➔ These images have a header (.hdr files) file without which the images cannot be accessed.
- ➔ We would be able to access any bands and would be able to search any place on the image.
- ➔ We would also remove the unnecessary parts and would focus on the vestige of the image we need.

Concepts of Hyperspectral Images


(a)
y
x
Monochrome
y
435 nm
545 nm
630 nm
x
Wavelength
RGB
Reflectance
Wavelength
Spectroscopy
y
435 nm
545 nm
630 nm
x
Wavelength
Multispectral
y
420nm
490nm
720nm
x
Wavelength
Hyperspectral
(b)
Hypercube
NIR Wavelength λ UV
y
x
Reflectance
UV
NIR
Wavelength λ
Red (R)
Green (G)
Blue (B)
RGB Image
y
x
Intensity
B
G
R
Wavelength λ

Introduction

- ➔ Hyperspectral images collect and processes information from across the electromagnetic spectrum.
- ➔ The goal of hyperspectral imaging is to obtain the spectrum of each pixel in the image of the scene, with the objective of finding objects, identifying materials and understanding processes.
- ➔ The spectral imaging, unlike human eye which sees 3-band images (RGB images), converts the images into many more bands (220, 425, 825, etcetera).

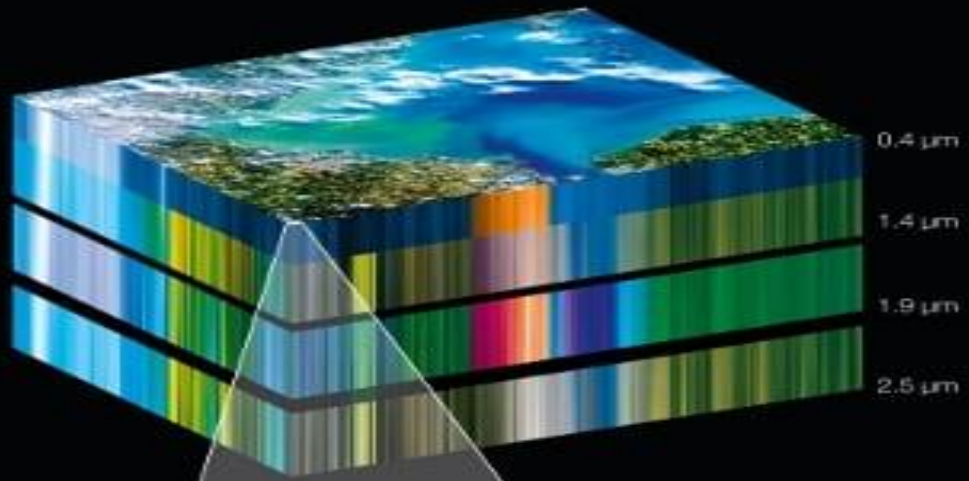
Wavelengths

- ➔ Hyperspectral images records the spectra of fine wavelength resolution and cover a wide range of wavelengths.
- ➔ The wavelengths may vary from any positive value to any other positive value.

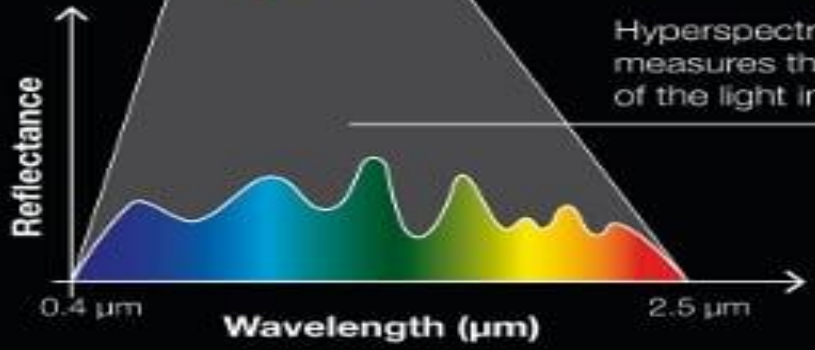
Reflectance values

- ➔ The reflectance values are the values reflected by each substances with different radiations and varies with different values of wavelength.
- ➔ Spectral reflectance coefficients represented in a function of wavelength, known as spectral characteristic describes the dependence of the reflectance coefficients of an object's surface from the wavelength of the incident radiation.
- ➔ This relationship is plotted on a graph of reflectance values vs wavelength.
- ➔ The negative reflectance values are of no use to identification of objects and properties of the images of the scene.

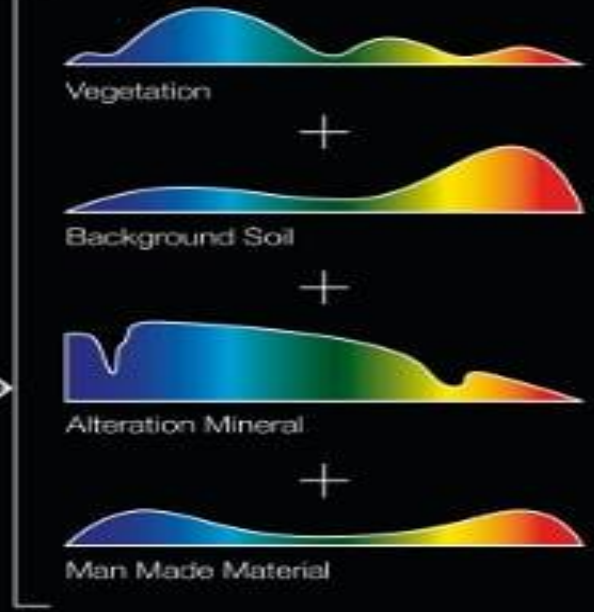
Hyperspectral Imaging Technology



Hyperspectral imaging measures the spectrum of the light in each pixel.

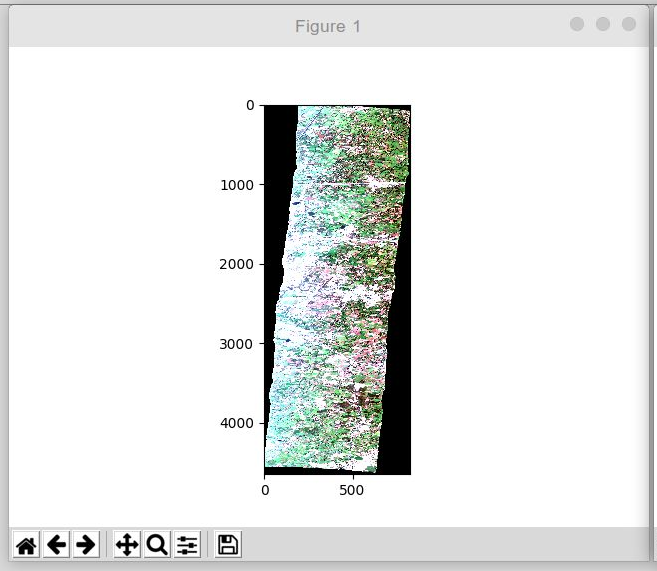


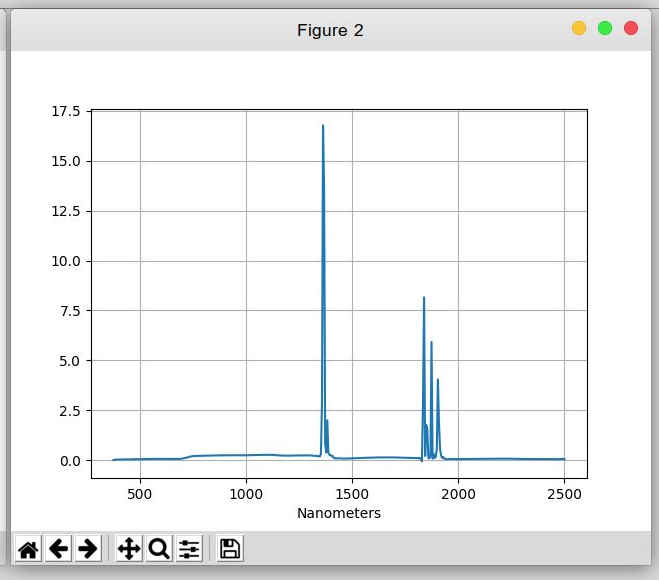
Components of Spectrum



Mini Project 1

File Edit View





Above graph is of reflectance value vs wavelength

Interleave

- There are basically 3 types of interleaves:
  - Band Interleaved by Pixel (BIP)
  - Band Interleaved by Line (BIL)
  - Band SeQuential (BSQ)
- These interleave processes are used for image encoding for multiband images.

1. Band Interleaved by Pixels (BIP)

- ➔ The BIP data organisation can handle any number of bands, and thus accommodates black and white, grayscale, pseudo colour, true colour and multispectral image data.
- ➔ Example Raster image file produced during by remote sensing systems and spectrometers.

2. Band Interleaved by Lines

- ➔ The BIL is not in itself and image format, but is a scheme for storing the actual pixel values of an image in a file band by band for each line or row of the image.
- ➔ If there ‘n’ bands in an image, all ‘n’ bands are written for row 1, row2 and so on.
- ➔ The BIL encoding is a compromise format, allowing fairly easy access to both spatial and spectral information.

3. Band SeQuential (BSQ)

- ➔ BSQ format is a very simple format, where each line of the data is followed immediately by the next line in the same spectral band.
- ➔ This format is optimal for spatial (x,y) access of any part of a single spectral band.

Data Ignore Value

- A Data Ignore Value is designated pixel value that ENVI (Environment for Visualisation Images) should ignore when processing an image or computing statistics.
- The Data Ignore value field is available in the Save File As Parameters Dialog when you save an image to disk.


Mini Project 1
File
Edit
View
GUI

- ➔ The GUI, presented above is made in python using Tkinter library.
- ➔ This library contains functions like Button, Menu, Canvas, Frame, etcetera.
- ➔ I also used classes like filedialog. This class contains functions used to open a dialog box to navigate through computer drives and select the file.

- Here I first created a frame in which we will be making other features like buttons or menu or even loading the photo.
- After that I created a navigation menu bar which contains options like “File” and “Edit” which contains options like:
  - File : “Open”, “Save” and “Exit”.
  - Edit : “Bands” and “Scale”.

```
47  top = Tk()
48  top.title("Mini Project 1")
49  top.geometry("3900x1200")
50  top.configure(bg="white")
51
52  canvas = Canvas(top, width=3900, height=1200)
53  canvas.pack()
54
55  # Creating the navigation bar
56
57  menubar = Menu(canvas)
58
59  filemenu = Menu(menubar, tearoff=0)
60  filemenu.add_command(label="Open", command=osf.open_file)
61  filemenu.add_command(label="Save", command=osf.save)
62  menubar.add_cascade(label="File", menu=filemenu)
63
64  editmenu = Menu(menubar, tearoff=0)
65  editmenu.add_command(label="Bands", command=bands)
66  menubar.add_cascade(label="Edit", menu=editmenu)
67
68  exitmenu = Menu(menubar, tearoff=0)
69  exitmenu.add_command(label="Quit", command=quit)
70  menubar.add_cascade(label="Quit", menu=exitmenu)
71
72  top.config(menu=menubar)
73
74  top.mainloop()
```

- The “File” option:
  - The “Open” option will open a dialog box which will be used to select the “.ENVI” file which then will be used to load the hyperspectral image.
  - The “Save” option will save the current modified image as “.ENVI” file or any other as per user’s choice.
  - The “Exit” option will close the window and the program will shut down.

- ➔ The “Open” option opens a dialog box and allows us to select the image to open.
- ➔ For this I used the “spectral” library of python from which I used “io” class and “envi” function.

```
def open_file():
    open_file.has_been_called = True
    filedialog.askopenfilename.has_been_called = True
    file_name = filedialog.askopenfilename(initialdir=os.path.expanduser("/"), filetypes=(("ENVI","*.envi"),("All files","*")))
    if str(file_name).endswith(".envi"):
        img = envi.open(file_name + ".hdr", file_name + ".envi")
    else:
        img = envi.open(file_name + ".hdr", file_name)
    file_header = file_name + ".hdr"
    file_save = file_name.split('/')
    file_save = file_save[:-1]
    save_dir = ""
    for string in file_save:
        save_dir = save_dir + string + "/"
    print(save_dir)
    filedialog.askopenfilename.has_been_called = False
    header_dict = envi_header.read_hdr_file(file_name + ".hdr")
    wavelengths = header_dict['wavelength'].split(',')
    wavelengths = [float(l) for l in wavelengths]
    bands = header_dict['bands'].split(',')
    bands = [float(l) for l in bands]
    imshow.has_been_called = True
    imshow(img, bands=(55, 32, 20), aspect=0.45, stretch=0.25)
    print(img)
    print(file_name)
    image = hy.read_image(file_header)
    print('Number of bands :', image.img_bands)
    print('Image height :', image.img_height)
    print('Image width :', image.img_width)
    noisy_bands_info = hy.noise_removal(image, min_threshold=0, max_threshold=0.55)
    noisy_bands_info.reflectance_plot()
    print("--------- List of noisy bands ---------")
    noisy_bands_info.show_noisy_bands_with_min_max()
    x = noisy_bands_info.show_noisy_bands()
    print(len(x), " : ", x)
    pre = hy.preprocessing(img_path=file_header, save_directory=save_dir, available_memory_gb=12)
    pre.perform(ndvi_threshold=125, data_ignore_value=-9999.0, NIR=90, RED=55, min_threshold=0, max_threshold=0.55, noisy_bands=x)
    file2_header = file_name + ".hdr"
    pre_image = hy.read_image(file2_header)
    imshow(pre_image, bands=(55, 32, 20), aspect=0.45, stretch=0.25)
    imshow.has_been_called = True
    print('Number of bands :', pre_image.img_bands)
    print('Image height :', pre_image.img_height)
```

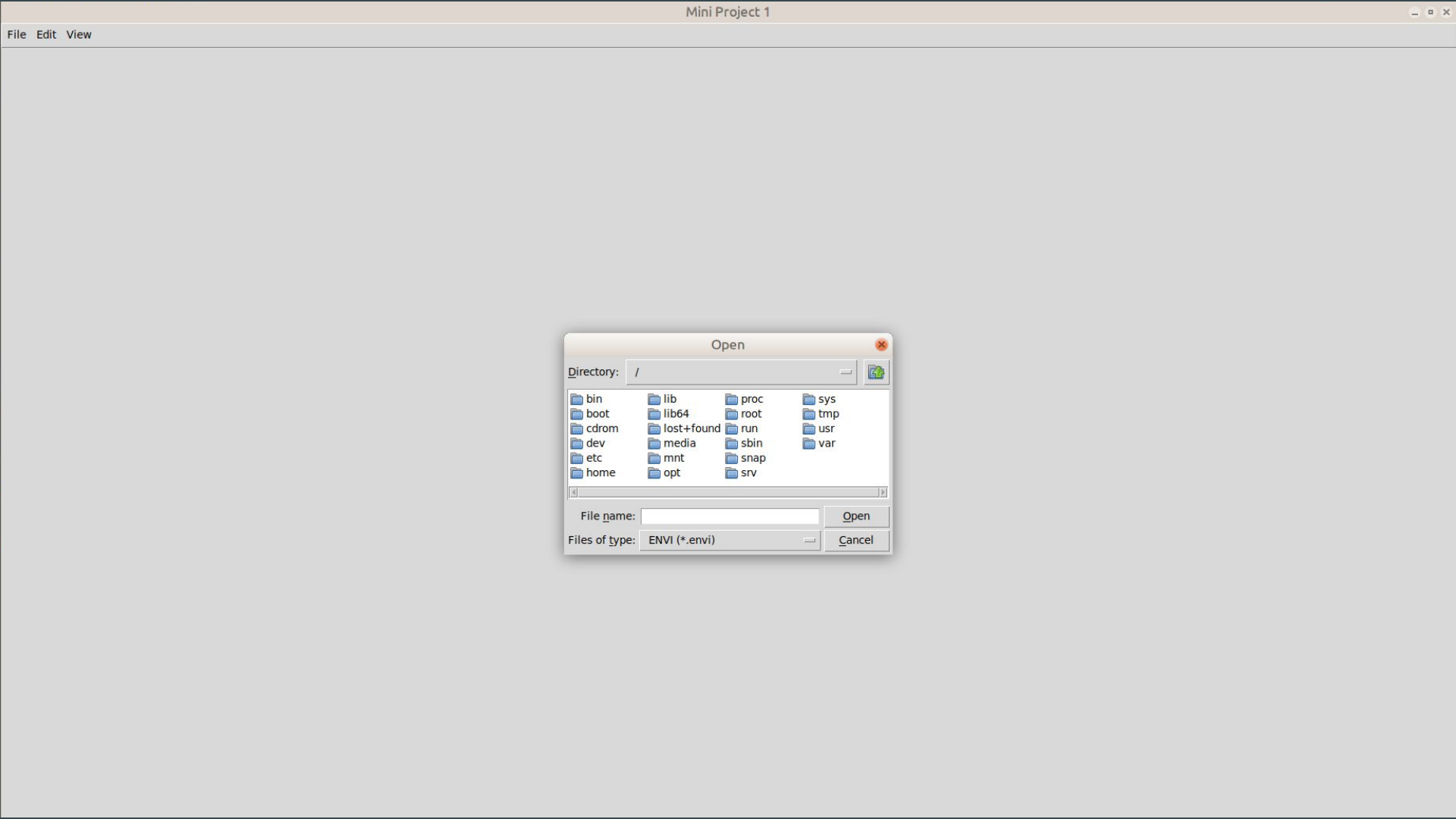
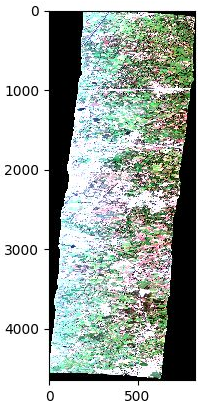

Mini Project 1
File
Edit
View
Open
Directory:
/
bin
boot
cdrom
dev
etc
home
lib
lib64
lost+found
media
mnt
opt
proc
root
run
sbin
snap
srv
sys
tmp
usr
var
File name:
Open
Files of type:
ENVI (*.envi)
Cancel

Figure 1



Preprocessing

- ➔ As we know the bands having negative reflectance values are not used, we process the image to remove it.
- ➔ For this I have used a python module named “Hyspeclib”. This library has functions to remove the unnecessary bands and noise.
- ➔ It also gives us the number of the bands which are not necessary.

List of noisy bands with the minimum and maximum value of reflectance

In [19]:
```
print("--------- List of noisy bands ---------")
noisy_bands_info.show_noisy_bands_with_min_max()
```

```
--------- List of noisy bands ---------
```

Out[19]:
```
[(0, -0.004925476, 0.023974443),
 (1, -0.00092434185, 0.025517834),
 (2, -0.00044369636, 0.054534975),
 (196, 1.0006491, 6.2307205),
 (197, 3.2283175, 28.862068),
 (198, 3.2342486, 25.140608),
 (199, 0.31904712, 1.5330484),
 (200, 0.15408023, 0.7062405),
 (201, 0.45762303, 3.9798782),
 (202, 0.13401343, 0.60886395),
 (203, 0.11217423, 0.61088246),
 (290, -0.14219935, 1.0380087),
 (291, -4.9735317, 18.169424),
 (292, -26.933197, 47.18488),
 (293, -1.634299, 3.5738058),
 (294, -2.5168257, 7.248437),
 (295, -3.633431, 14.122553),
 (296, -2.4177022, 6.4896426),
 (297, -0.3969328, 1.8057275),
 (298, -2.5399668, 4.6726427),
 (299, -6.389768, 17.949532),
 (300, -0.40353727, 1.2678614),
 (301, -0.25679904, 0.70010936),
 (302, -0.109293625, 0.56055313),
 (303, -0.20577697, 0.5249394),
 (304, -0.42758185, 1.1891016),
 (305, -8.138784, 27.858128),
 (306, -2.4245825, 7.083447),
 (307, -0.43808395, 1.6269228),
 (308, -0.1394753, 0.5329536),
 (309, -0.08115421, 0.31171006),
 (310, -0.086150914, 0.41048175),
 (311, -0.018062774, 0.27391008),
 (312, -0.010429804, 0.28637552)]
```

In [20]:
```
x = noisy_bands_info.show_noisy_bands()
print(len(x)," : ",x)
```

```
34  :  [0, 1, 2, 196, 197, 198, 199, 200, 201, 202, 203, 290, 291, 292, 293, 294, 295, 296, 297, 298, 299, 300, 30
1, 302, 303, 304, 305, 306, 307, 308, 309, 310, 311, 312]
```

Step 2 - Removal of noisy bands, zero fills and non-vegetation pixels

Preprocessing continued...

- ➔ It also removes these bands and stores the new image at a place so that we can use it later.
- ➔ Also this image is divided into 2 parts:
  - ◆ Having only noisy bands
  - ◆ Having no noisy bands
- ➔ The reflectance values are transformed from large values to values between 0 and 1.

Step 2 - Removal of noisy bands, zero fills and non-vegetation pixels

- **available_memory_gb** : you can specify the maximum available memory to load image into RAM at a time in GB

In [16]:
```
pre = hy.preprocessing(img_path = our_image_path, save_directory = pre_save_dir, available_memory_gb=0.12)
```

```
Image will be saved in 2 partitions.
```

- **ndvi_threshold** : maximum value of scaled NDVI for non-vegetation pixels.
- **data_ignore_value=** : Value used for out of scene pixels. It is specified in header file
- **NIR** : Band number to be used as NIR chennel for NDVI calculation
- **RED** : Band number to be used as RED chennel for NDVI calculation
- **noisy_bands** : Manually specify the list of bands to be removed. If not specified then noisy bands will be automatically identified bansed on min_threshold and max_threshold

In [21]:
```
pre.perform(ndvi_threshold=125,
            data_ignore_value=-9999.0,
            NIR=90,
            RED=55,
            min_threshold=0,
            max_threshold=0.55,
            noisy_bands=x)
```

```
--------------- Performing Preprocessing ---------------

Partition : 1 / 2 running...

Saving /Users/hetulpatel/Documents/major/code12062018/anand/Preprocessed_image/unprocessed_image_part_1

Partition : 2 / 2 running...

Saving /Users/hetulpatel/Documents/major/code12062018/anand/Preprocessed_image/unprocessed_image_part_2

Preprocessing completed. Output directory : anand/Preprocessed_image/


------------------------------------------------------------
```

Open and check preprocessed image

In [26]:
```
pre_image = hy.read_image(pre_save_dir+'unprocessed_image_part_1.hdr')
```

In [27]:
```
print('Number of bands :',pre_image.img_bands)
print('Image height :',pre_image.img_height)
print('Image width :',pre_image.img_width)
```

```
Number of bands : 391
Image height : 179
Image width : 393
```

References

- https://www.rsipvision.com/image-processing-for-precise-agriculture/
- https://dialnet.unitoja.es/descarga/articulo/5178334.pdf
- https://gsp.humboldt.edu/OLM/Courses/GSP_216_Online/lesson2-1/reflectance.html
- http://www.spectralpython.net/
- https://github.com/hetul-patel/hyspeclib/

Thank You!!!

17BIT034 (Kandarp Kakkad)