

```
# Install necessary libraries
!pip install rasterio earthpy matplotlib numpy
```

Collecting rasterio
 Downloading rasterio-1.4.3-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (9.1 kB)
 Collecting earthpy
 Downloading earthpy-0.9.4-py3-none-any.whl.metadata (9.2 kB)
 Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages (3.8.0)
 Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (1.26.4)
 Collecting affine (from rasterio)
 Downloading affine-2.4.0-py3-none-any.whl.metadata (4.0 kB)
 Requirement already satisfied: attrs in /usr/local/lib/python3.10/dist-packages (from rasterio) (24.3.0)
 Requirement already satisfied: certifi in /usr/local/lib/python3.10/dist-packages (from rasterio) (2024.12.14)
 Requirement already satisfied: click>=4.0 in /usr/local/lib/python3.10/dist-packages (from rasterio) (8.1.7)
 Collecting cligj>=0.5 (from rasterio)
 Downloading cligj-0.7.2-py3-none-any.whl.metadata (5.0 kB)
 Collecting click-plugins (from rasterio)
 Downloading click_plugins-1.1.1-py2.py3-none-any.whl.metadata (6.4 kB)
 Requirement already satisfied: pyparsing in /usr/local/lib/python3.10/dist-packages (from rasterio) (3.2.0)
 Requirement already satisfied: geopandas in /usr/local/lib/python3.10/dist-packages (from earthpy) (1.0.1)
 Requirement already satisfied: scikit-image in /usr/local/lib/python3.10/dist-packages (from earthpy) (0.25.0)
 Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from earthpy) (2.32.3)
 Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (1.3.1)
 Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (0.12.1)
 Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (4.55.3)
 Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (1.4.7)
 Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (24.2)
 Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (11.0.0)
 Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (2.8.2)
 Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.7->matplotlib) (1.17.0)
 Requirement already satisfied: pyogrio>=0.7.2 in /usr/local/lib/python3.10/dist-packages (from geopandas->earthpy) (0.10.0)
 Requirement already satisfied: pandas>=1.4.0 in /usr/local/lib/python3.10/dist-packages (from geopandas->earthpy) (2.2.2)
 Requirement already satisfied: pyproj>=3.3.0 in /usr/local/lib/python3.10/dist-packages (from geopandas->earthpy) (3.7.0)
 Requirement already satisfied: shapely>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from geopandas->earthpy) (2.0.6)
 Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests->earthpy) (3.4.0)
 Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests->earthpy) (3.10)
 Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests->earthpy) (2.2.3)
 Requirement already satisfied: scipy>=1.11.2 in /usr/local/lib/python3.10/dist-packages (from scikit-image->earthpy) (1.13.1)
 Requirement already satisfied: networkx>=3.0 in /usr/local/lib/python3.10/dist-packages (from scikit-image->earthpy) (3.4.2)
 Requirement already satisfied: imageio!=2.35.0,>=2.33 in /usr/local/lib/python3.10/dist-packages (from scikit-image->earthpy) (2.36)
 Requirement already satisfied: tifffile>=2022.8.12 in /usr/local/lib/python3.10/dist-packages (from scikit-image->earthpy) (2024.12)
 Requirement already satisfied: lazy-loader>=0.4 in /usr/local/lib/python3.10/dist-packages (from scikit-image->earthpy) (0.4)
 Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.4.0->geopandas->earthpy) (2024.12)
 Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.4.0->geopandas->earthpy) (2024.12)
 Downloading rasterio-1.4.3-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (22.2 MB)
 22.2/22.2 MB 26.8 MB/s eta 0:00:00
 Downloading earthpy-0.9.4-py3-none-any.whl (1.4 MB)
 1.4/1.4 MB 55.0 MB/s eta 0:00:00
 Downloading cligj-0.7.2-py3-none-any.whl (7.1 kB)
 Downloading affine-2.4.0-py3-none-any.whl (15 kB)
 Downloading click_plugins-1.1.1-py2.py3-none-any.whl (7.5 kB)
 Installing collected packages: cligj, click-plugins, affine, rasterio, earthpy
 Successfully installed affine-2.4.0 click-plugins-1.1.1 cligj-0.7.2 earthpy-0.9.4 rasterio-1.4.3

```
# Import required libraries
import glob
import numpy as np
import rasterio
import matplotlib.pyplot as plt
import earthpy.plot as ep
from rasterio.plot import show
from google.colab import drive
```

```
# Mount Google Drive
drive.mount('/content/drive')
```

Show hidden output


```
# Define the path to your images
image_paths = sorted(glob.glob('/content/drive/MyDrive/Btech/RS_Lab/EXP_2/*.tif'))

# Step 3: Read bands, display min/max values, and shape
bands = []
for path in image_paths:
    with rasterio.open(path) as src:
        band = src.read(1)
        bands.append(band)
    print(f"{path.split('/')[-1]}: Shape = {band.shape}, Min = {band.min()}, Max = {band.max()}")
```

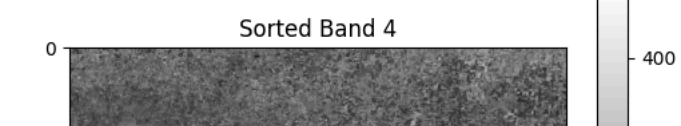
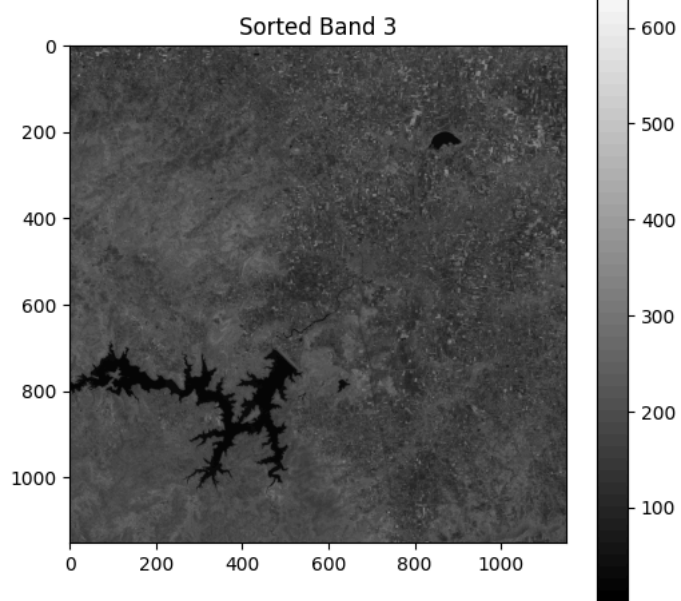
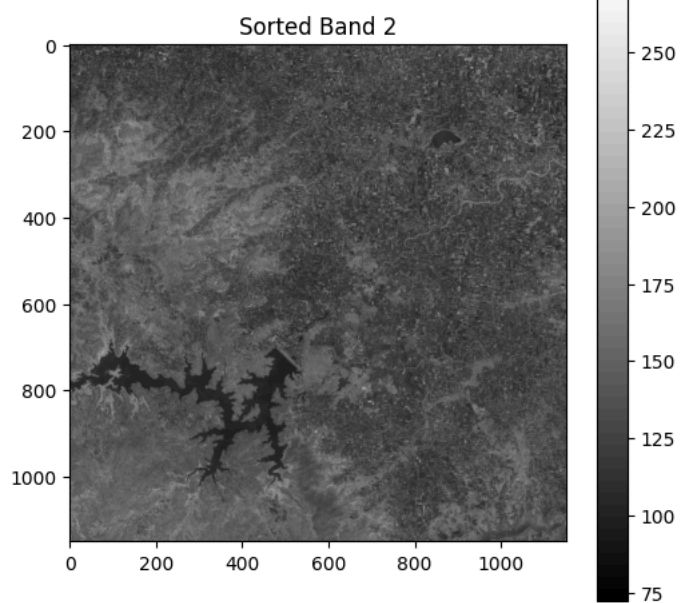
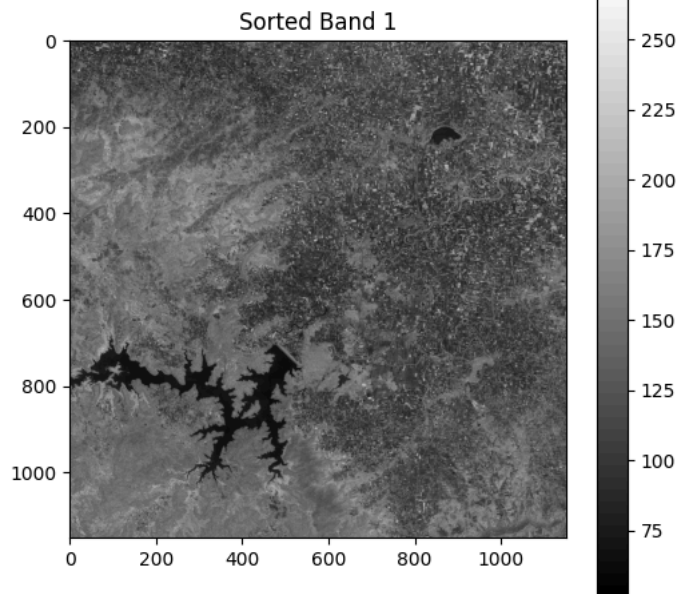
L3-NE43C11-096-059-05Apr19-BAND2.tif: Shape = (1151, 1151), Min = 72, Max = 272
 L3-NE43C11-096-059-05Apr19-BAND3.tif: Shape = (1151, 1151), Min = 51, Max = 271

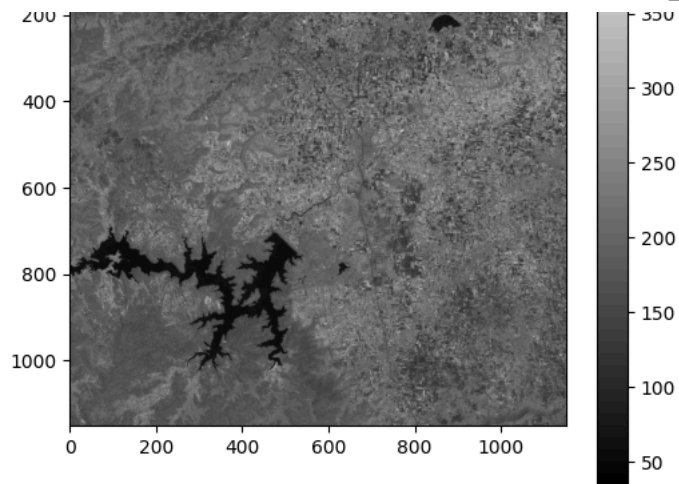
```
L3-NE43C11-096-059-05Apr19-BAND4.tif: Shape = (1151, 1151), Min = 34, Max = 447
L3-NE43C11-096-059-05Apr19-BAND5.tif: Shape = (1151, 1151), Min = 1, Max = 644
```

```
# Step 4: Sort the bands by mean intensity
sorted_indices = np.argsort([band.mean() for band in bands])
sorted_bands = [bands[i] for i in sorted_indices]
print("Bands sorted based on mean intensity.")
```

 Bands sorted based on mean intensity.

```
# Step 5: Open images using rasterio and plot them
for i, band in enumerate(sorted_bands):
    plt.figure(figsize=(6, 6))
    plt.imshow(band, cmap='gray')
    plt.colorbar()
    plt.title(f"Sorted Band {i+1}")
    plt.show()
```



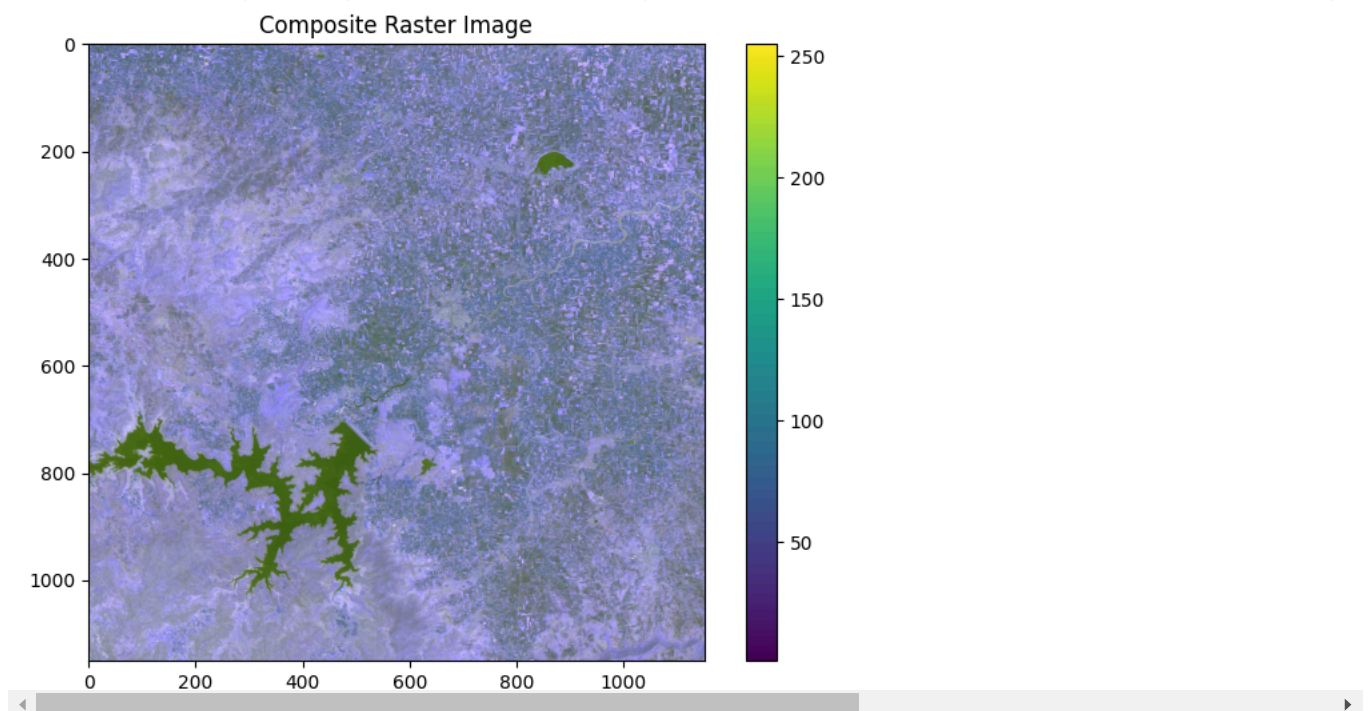


```
# Step 6: Create a composite raster
composite_raster = np.stack(sorted_bands, axis=-1)
print("Composite raster shape:", composite_raster.shape)
print("Composite raster min/max:", composite_raster.min(), composite_raster.max())
```

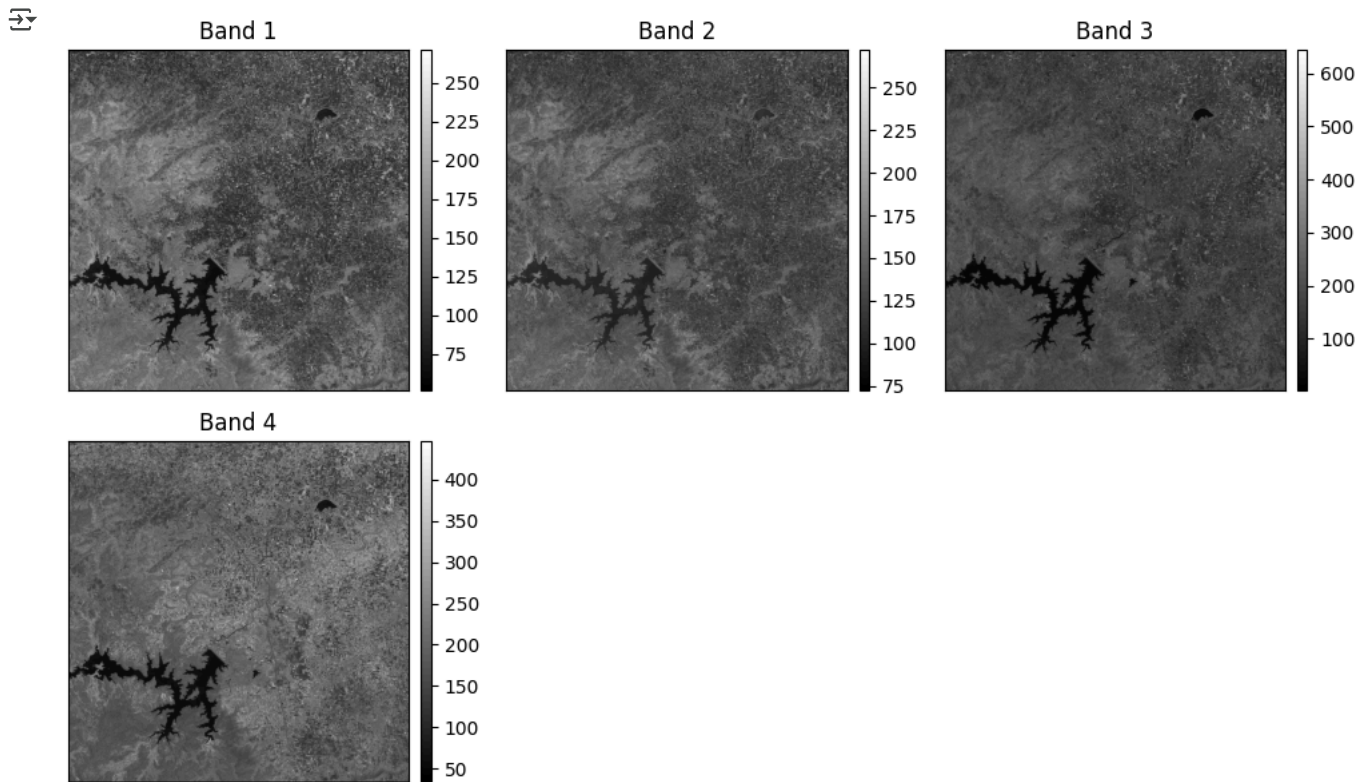
```
↗ Composite raster shape: (1151, 1151, 4)
Composite raster min/max: 1 644
```

```
# Display the composite raster
plt.figure(figsize=(8, 6))
plt.imshow(composite_raster[:, :, :3]) # Using first 3 bands
plt.title("Composite Raster Image")
plt.colorbar()
plt.show()
```

```
↗ WARNING:matplotlib.image:Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers)
```



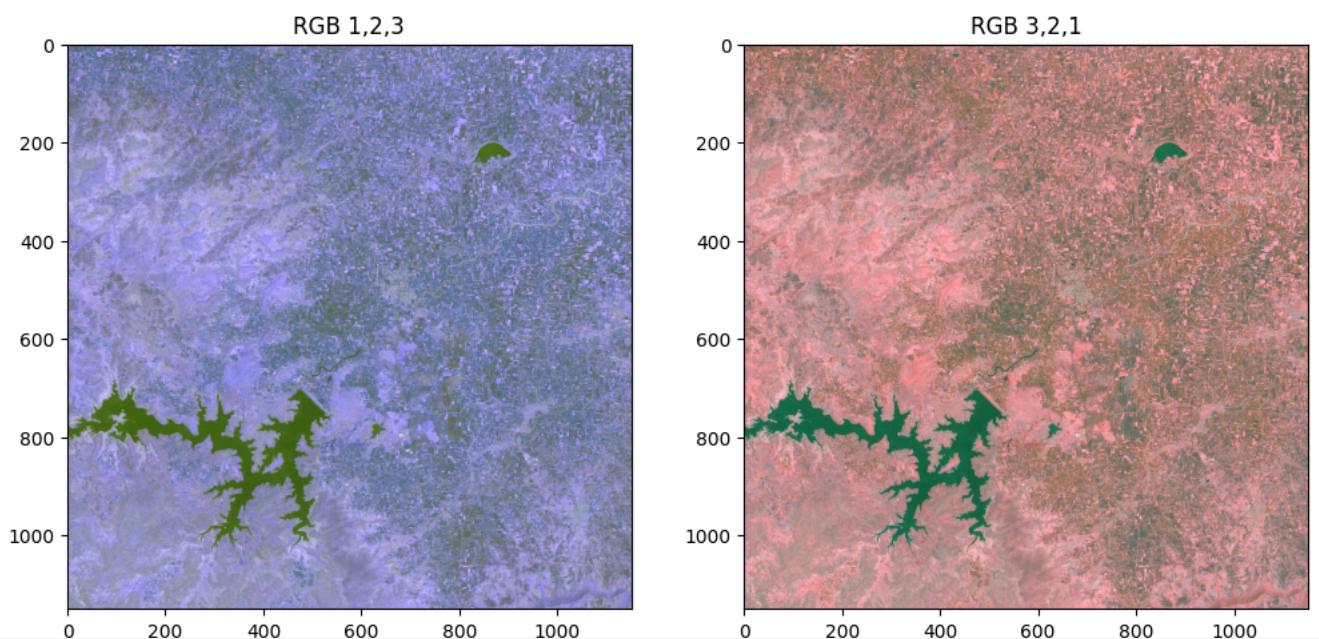
```
# Step 7: Visualize all bands using earthpy
ep.plot_bands(np.array(sorted_bands), cmap='gray', figsize=(10, 6))
```

```
array([[<Axes: title='{center': 'Band 1'}>,
       <Axes: title='{center': 'Band 2'}>,
       <Axes: title='{center': 'Band 3'}>],
      [<Axes: title='{center': 'Band 4'}>, <Axes: >, <Axes: >]],
      'image')
```

```
# Step 8: Plot RGB images in different band combinations
fig, ax = plt.subplots(1, 2, figsize=(12, 6))
ax[0].imshow(composite_raster[:, :, [0, 1, 2]]) # RGB 1,2,3
ax[0].set_title("RGB 1,2,3")
ax[1].imshow(composite_raster[:, :, [2, 1, 0]]) # RGB 3,2,1
ax[1].set_title("RGB 3,2,1")
plt.show()
```

WARNING:matplotlib.image:Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers)
 WARNING:matplotlib.image:Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers)



```
# Step 9: Plot histograms for individual bands
for i, band in enumerate(sorted_bands):
    plt.figure(figsize=(8, 6))
    plt.hist(band.flatten(), bins=50, color='skyblue', edgecolor='black')
    plt.xlabel("Pixel Value")
    plt.ylabel("Frequency")
    plt.title(f"Histogram of Sorted Band {i+1}")
    plt.show()
```

