Pre-Institute Week 3 RK

July 3, 2018

1 Pre-Institute Week 3

Reza Khatami

```
In [1]: import sys
        sys.version
```

```
Out[1]: '3.5.5 |Anaconda custom (64-bit)| (default, Apr  7 2018, 04:52:34) [MSC v.1900 64 bit
```

2 Importing Libraries

```
In [ ]: import gdal, osr
```

```
In [4]: import numpy as np
        import matplotlib.pyplot as plt
        %matplotlib inline
        import warnings
        warnings.filterwarnings('ignore')
```

3 The function to read RGB images

```
In [5]: def RGBraster2array(RGB_geotif):
            """RGBraster2array reads in a NEON AOP geotif file and returns
            a numpy array, and header containing associated metadata with spatial information.
            --------
            Parameters
                RGB_geotif -- full or relative path and name of reflectance hdf5 file
            --------
            Returns
            --------
            array:
                numpy array of geotif values
            metadata:
                dictionary containing the following metadata (all strings):
                    array_rows
                    array_cols
                    bands
```

```
        driver
        projection
        geotransform
        pixelWidth
        pixelHeight
        extent
        noDataValue
        scaleFactor
--------
Example Execution:
--------
RGB_geotif = '2017_SERC_2_368000_4306000_image.tif'
RGBcam_array, RGBcam_metadata = RGBraster2array(RGB_geotif) """

metadata = {}
dataset = gdal.Open(RGB_geotif)
metadata['array_rows'] = dataset.RasterYSize
metadata['array_cols'] = dataset.RasterXSize
metadata['bands'] = dataset.RasterCount
metadata['driver'] = dataset.GetDriver().LongName
metadata['projection'] = dataset.GetProjection()
metadata['geotransform'] = dataset.GetGeoTransform()

mapinfo = dataset.GetGeoTransform()
metadata['pixelWidth'] = mapinfo[1]
metadata['pixelHeight'] = mapinfo[5]

metadata['ext_dict'] = {}
metadata['ext_dict']['xMin'] = mapinfo[0]
metadata['ext_dict']['xMax'] = mapinfo[0] + dataset.RasterXSize/mapinfo[1]
metadata['ext_dict']['yMin'] = mapinfo[3] + dataset.RasterYSize/mapinfo[5]
metadata['ext_dict']['yMax'] = mapinfo[3]

metadata['extent'] = (metadata['ext_dict']['xMin'],metadata['ext_dict']['xMax'],
                      metadata['ext_dict']['yMin'],metadata['ext_dict']['yMax'])

raster = dataset.GetRasterBand(1)
array_shape = raster.ReadAsArray(0,0,metadata['array_cols'],metadata['array_rows'])
metadata['noDataValue'] = raster.GetNoDataValue()
metadata['scaleFactor'] = raster.GetScale()

array = np.zeros((array_shape[0],array_shape[1],dataset.RasterCount),'uint8') #pre
for i in range(1, dataset.RasterCount+1):
    band = dataset.GetRasterBand(i).ReadAsArray(0,0,metadata['array_cols'],metadata
    band[band==metadata['noDataValue']]=np.nan
    band = band/metadata['scaleFactor']
    array[...,i-1] = band
```

```
            return array, metadata
```

```
In [8]: RGB_geotif = './2017_SERC_2_368000_4306000_image.tif'
        SERC_RGBcam_array, SERC_RGBcam_metadata = RGBraster2array(RGB_geotif)
```

```
In [9]: SERC_RGBcam_array.shape
```

```
Out[9]: (10000, 10000, 3)
```

```
In [10]: #Display information stored in header
         for key in sorted(SERC_RGBcam_metadata.keys()):
           print(key)
```

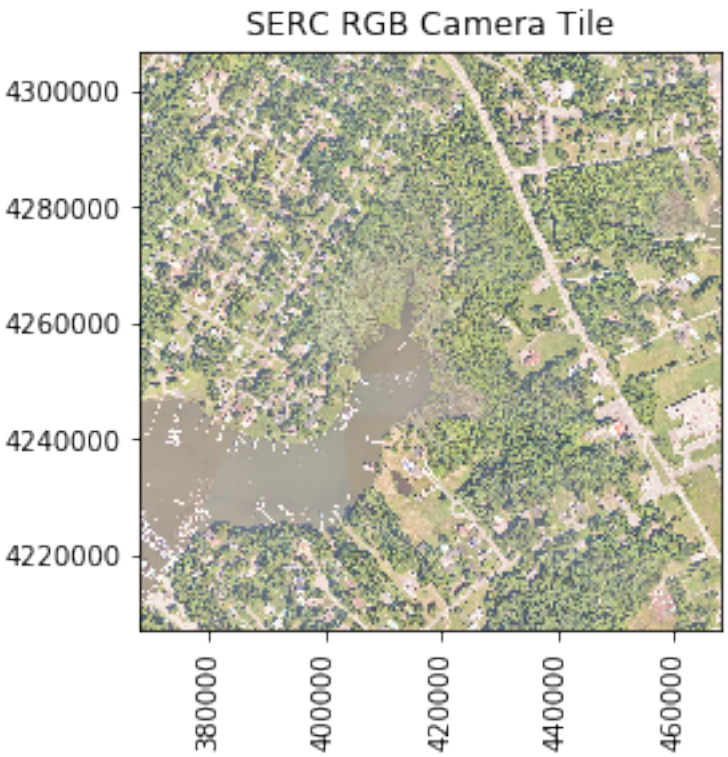
```
array_cols
array_rows
bands
driver
ext_dict
extent
geotransform
noDataValue
pixelHeight
pixelWidth
projection
scaleFactor
```

4 The function to plot images

```
In [ ]: def plot_band_array(band_array,
                            refl_extent,
                            colorlimit,
                            ax=plt.gca(),
                            title='',
                            cbar ='on',
                            cmap_title='',
                            colormap='spectral'):

            '''plot_band_array reads in and plots a single band or an rgb band combination of
            --------
            Parameters
            --------
                band_array: flightline array of reflectance values, created from h5refl2array
                refl_extent: extent of reflectance data to be plotted (xMin, xMax, yMin, yMax)
                colorlimit: range of values to plot (min,max). Best to look at the histogram o
                ax: optional, default = current axis
                title: string, optional; plot title
                cmap_title: string, optional; colorbar title
```

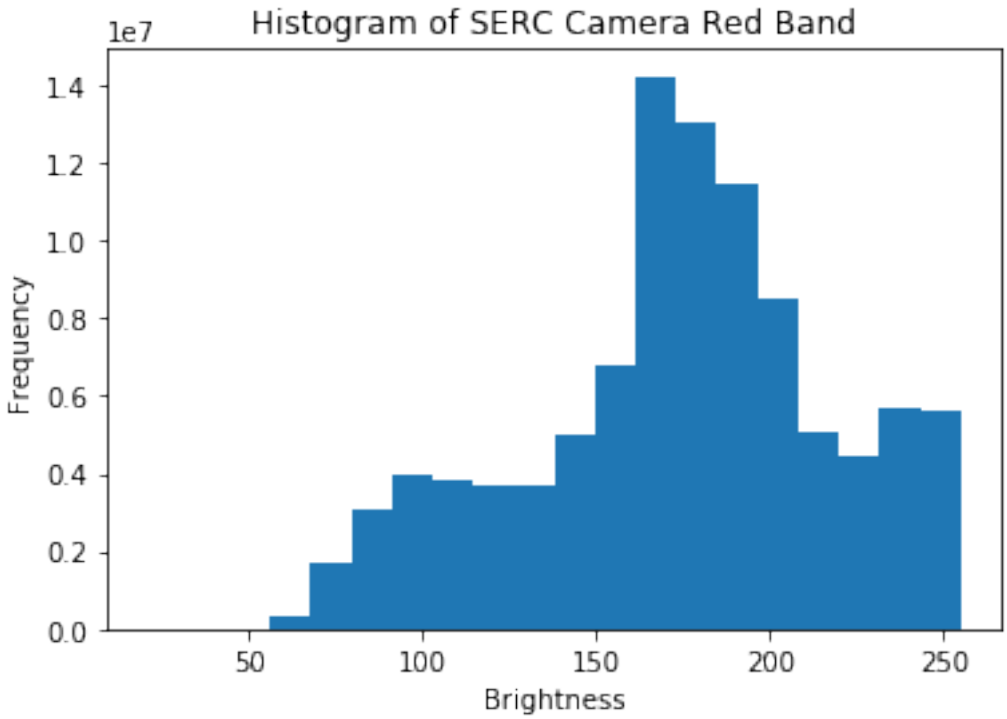
```
        colormap: string, optional; see https://matplotlib.org/examples/color/colormap
    --------
    Returns
        plots array of single band or RGB if given a 3-band
    --------
    Example:
    --------
    plot_band_array(SERC_RGBcam_array,
                    SERC_RGBcam_metadata['extent'],
                    (1,255),
                    title='SERC RGB Camera Tile',
                    cbar='off')'''

    plot = plt.imshow(band_array,extent=refl_extent,clim=colorlimit);
    if cbar == 'on':
        cbar = plt.colorbar(plot,aspect=40); plt.set_cmap(colormap);
        cbar.set_label(cmap_title,rotation=90,labelpad=20)
    plt.title(title); ax = plt.gca();
    ax.ticklabel_format(useOffset=False, style='plain'); #do not use scientific notati
    rotatexlabels = plt.setp(ax.get_xticklabels(),rotation=90); #rotate x tick labels
```

In [12]:
```
plot_band_array(SERC_RGBcam_array,
                SERC_RGBcam_metadata['extent'],
                (1,255),
                title='SERC RGB Camera Tile',
                cbar='off')
```



```
In [14]: plt.hist(np.ravel(SERC_RGBcam_array[:,:,0]),20);
         plt.title('Histogram of SERC Camera Red Band')
         plt.xlabel('Brightness'); plt.ylabel('Frequency')
```

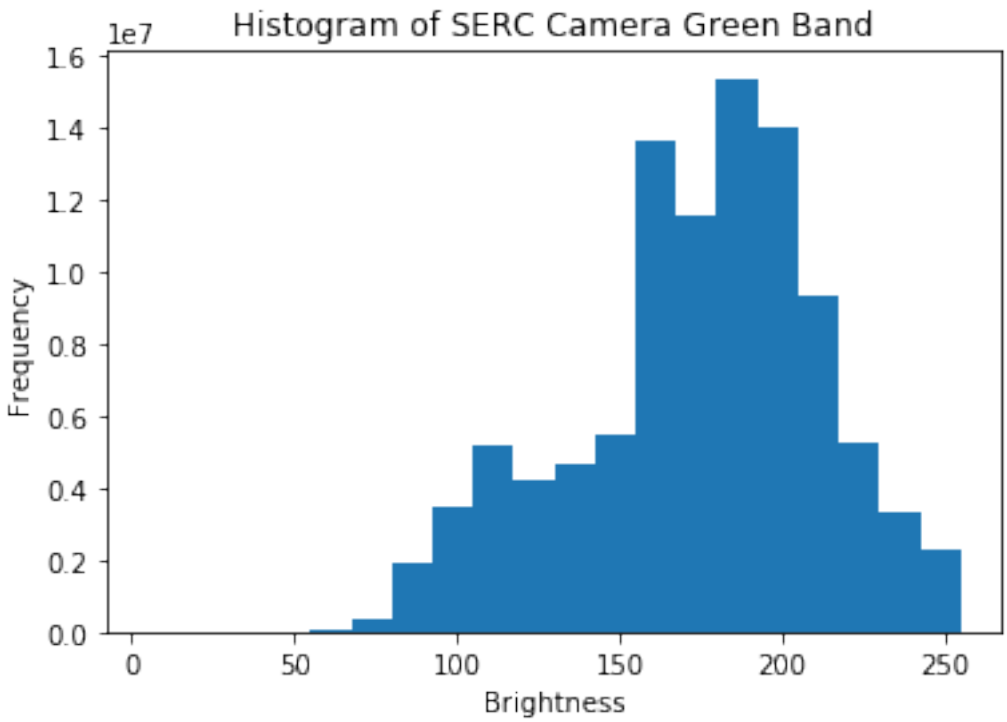
```
Out[14]: Text(0,0.5,'Frequency')
```



5 Challenge Exercises

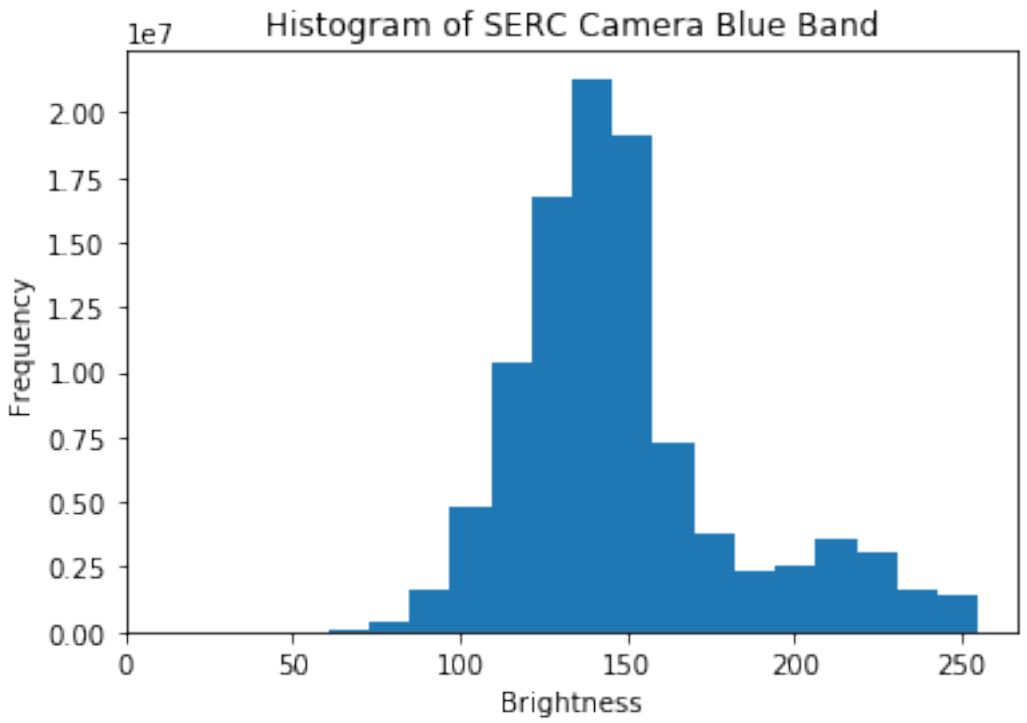
```
In [15]: plt.hist(np.ravel(SERC_RGBcam_array[:,:,1]),20);
         plt.title('Histogram of SERC Camera Green Band')
         plt.xlabel('Brightness'); plt.ylabel('Frequency')
```

```
Out[15]: Text(0,0.5,'Frequency')
```




```
In [16]: plt.hist(np.ravel(SERC_RGBcam_array[:,:,2]),20);
         plt.title('Histogram of SERC Camera Blue Band')
         plt.xlabel('Brightness'); plt.ylabel('Frequency')
```

```
Out[16]: Text(0,0.5,'Frequency')
```



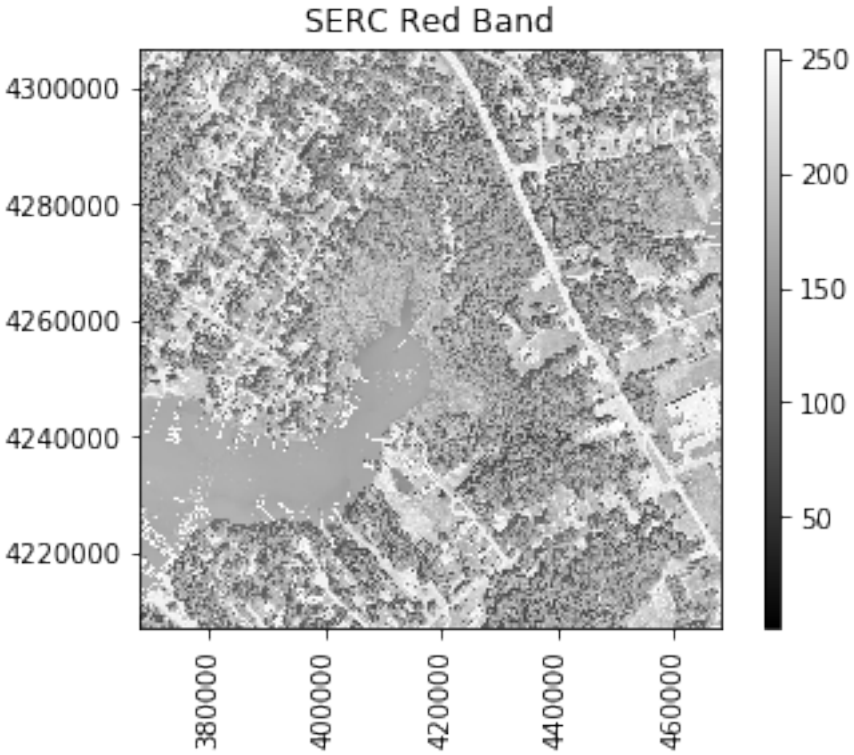

```
In [24]: min_red=np.amin(SERC_RGBcam_array[:,:,0])
         max_red=np.amax(SERC_RGBcam_array[:,:,0])
         min_green=np.amin(SERC_RGBcam_array[:,:,1])
         max_green=np.amax(SERC_RGBcam_array[:,:,1])
         min_blue=np.amin(SERC_RGBcam_array[:,:,2])
         max_blue=np.amax(SERC_RGBcam_array[:,:,2])
         print('min red:', min_red)
         print('max red:', max_red)
         print('min green:', min_green)
         print('max green:', max_green)
         print('min blue:', min_blue)
         print('max blue:', max_blue)
```

```
min red: 21
max red: 255
min green: 5
max green: 255
min blue: 12
max blue: 255
```

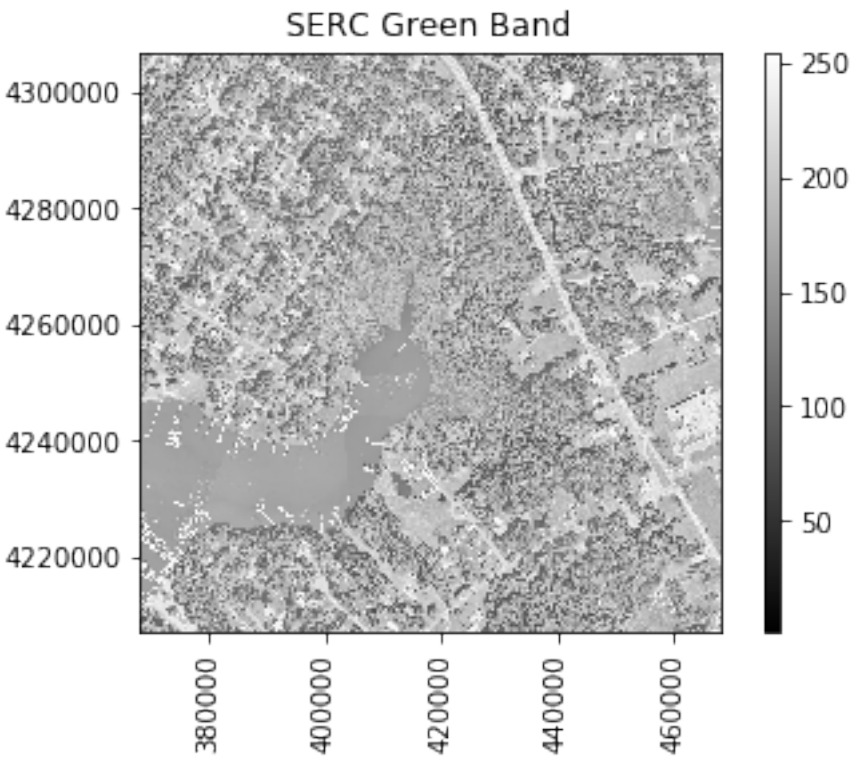
```
In [29]: src=osr.SpatialReference(wkt=SERC_RGBcam_metadata['projection'])
         print(src.GetAttrValue('PROJCS'))
```

```
WGS 84 / UTM zone 18N
```

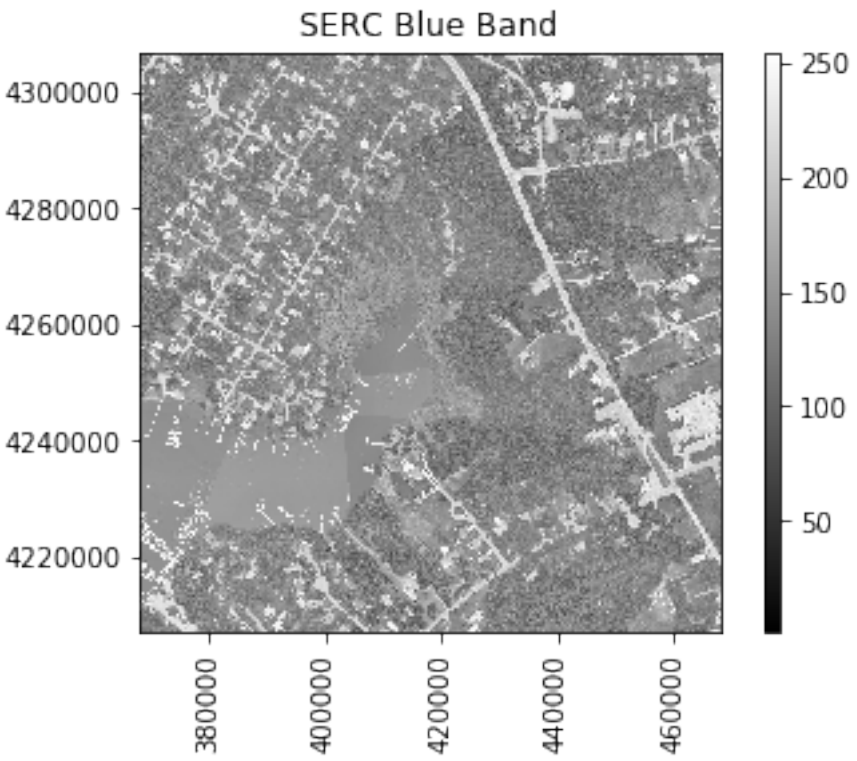
```
In [38]: plot_band_array(SERC_RGBcam_array[:,:,0],
                         SERC_RGBcam_metadata['extent'],
                         (1,255),
                         title='SERC Red Band',
                         cbar='on',
                        colormap='gray')
```




```
In [42]: plot_band_array(SERC_RGBcam_array[:,:,1],
                         SERC_RGBcam_metadata['extent'],
                         (1,255),
                         title='SERC Green Band',
                         cbar='on',
                         colormap='gray')
```




```
In [41]: plot_band_array(SERC_RGBcam_array[:,:,2],
                         SERC_RGBcam_metadata['extent'],
                         (1,255),
                         title='SERC Blue Band',
                         cbar='on',
                         colormap='gray')
```


SERC Blue Band
4300000
4280000
4260000
4240000
4220000
380000
400000
420000
440000
460000
250
200
150
100
50