NEON_RGB_Wk3_Lamping

July 6, 2018

1 Plotting RGB Imagery in Python

Created 6/30/18 by James Lamping

I had started this assignment on another notebopok, however, I had a lot of issues getting the RGB image to run through the function. *This is a second attempt with a fresh start*.

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

Out[1]: '3.5.5 |Anaconda custom (64-bit)| (default, Apr 26 2018, 08:11:22) \n[GCC 4.2.1 Compat
```

Problem I cant seem to get the right version of python to run while running the created NEON kernal. I am able to get the right version to run on the standard "Python 3" kernal and am using that to run this code.

Update I remade the p35 NEON kernal and am running the code on that.

```
In [2]: import gdal
```

Read in RGB Camera Image

```
In [3]: import numpy as np
        import matplotlib.pyplot as plt
        %matplotlib inline
        import warnings
        warnings.filterwarnings('ignore')

In [4]: 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'],metadat
                band[band==metadata['noDataValue']]=np.nan
                band = band/metadata['scaleFactor']
                array[...,i-1] = band

            return array, metadata
```

Load image and convert to array

```
In [5]: RGB_geotif = './2017_SERC_2_368000_4306000_image.tif'
        SERC_RGBcam_array, SERC_RGBcam_metadata = RGBraster2array(RGB_geotif)
```

1.1 FIXED IT!

```
In [6]: SERC_RGBcam_array.shape
```

```
Out[6]: (10000, 10000, 3)
```

Display information stored in header

```
In [7]: for key in sorted(SERC_RGBcam_metadata.keys()):
          print(key)
```

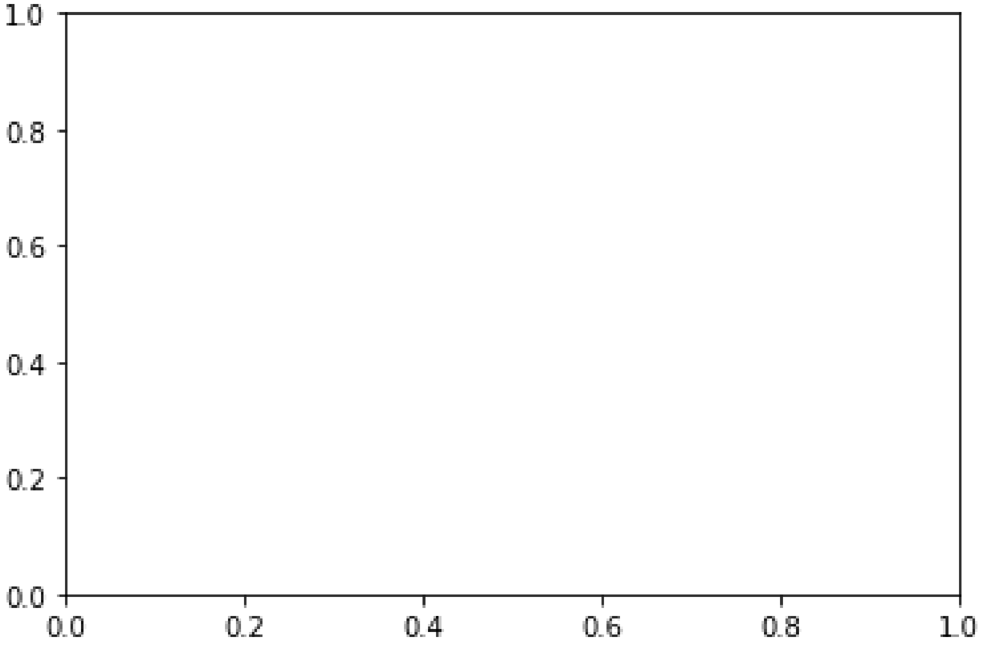
```
array_cols
array_rows
bands
driver
ext_dict
extent
geotransform
noDataValue
pixelHeight
pixelWidth
projection
scaleFactor
```

```
In [8]: 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
            --------
```

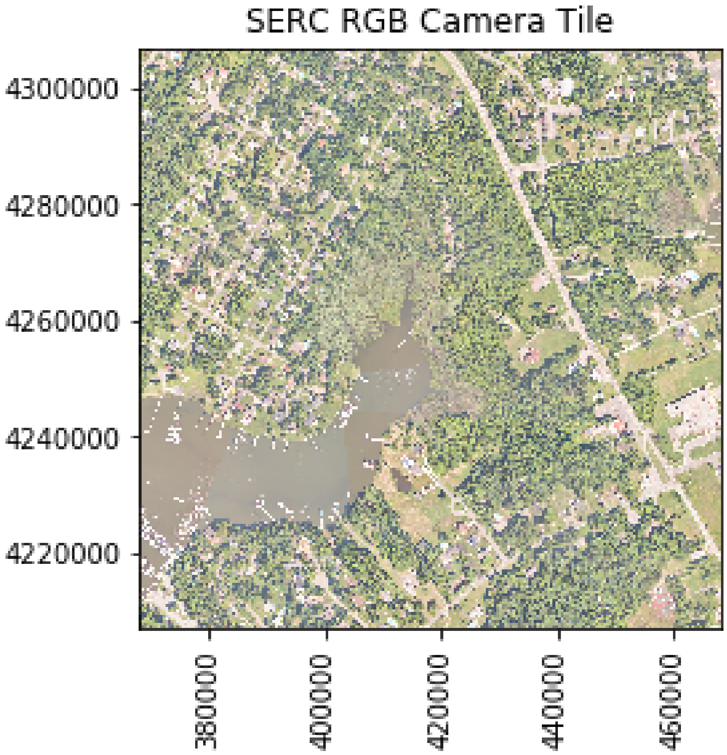
```
Parameters
--------
    band_array: flightline array of reflectance values, created from h5refl2array
    refl_extent: extent of reflectance data to be plotted (xMin, xMax, yMin, yMax)
    colorlimit: range of values to plot (min,max). Best to look at the histogram o
    ax: optional, default = current 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
```



Plotting image

```
In [9]: plot_band_array(SERC_RGBcam_array,
                        SERC_RGBcam_metadata['extent'],
                        (1,255),
                        title='SERC RGB Camera Tile',
                        cbar='off')
```

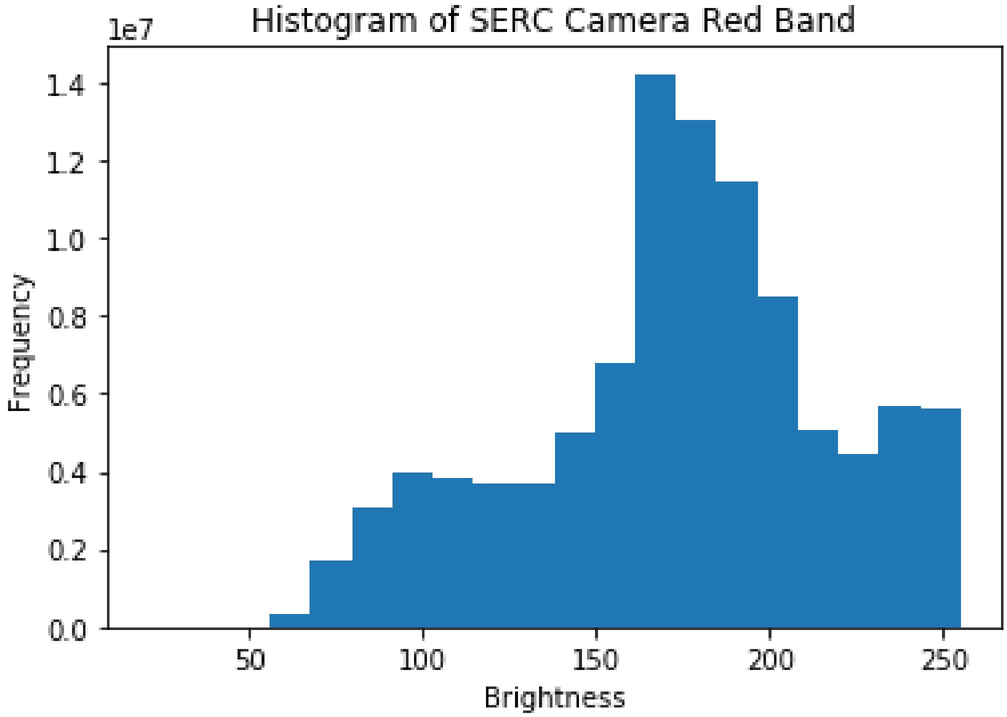


Creating red, green and blue images

Creating histogram of red band

```
In [10]: plt.hist(np.ravel(SERC_RGBcam_array[:,:,0]),20);
         plt.title('Histogram of SERC Camera Red Band')
         plt.xlabel('Brightness'); plt.ylabel('Frequency')

Out[10]: Text(0,0.5,'Frequency')
```

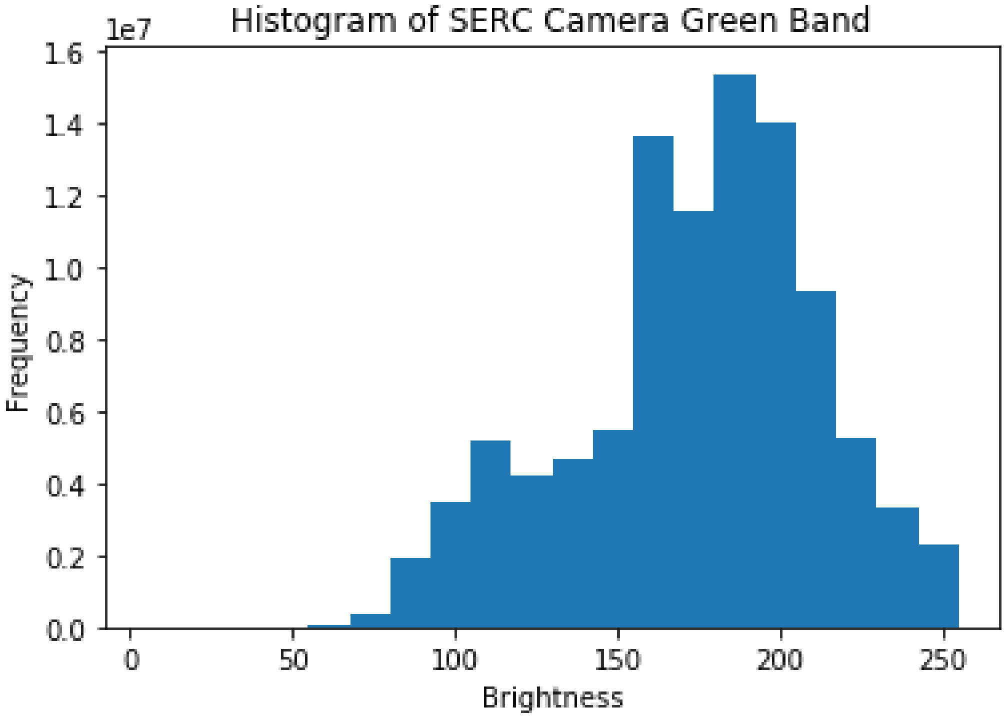


1.2 Challenge Exercises

1. Plot Histograms

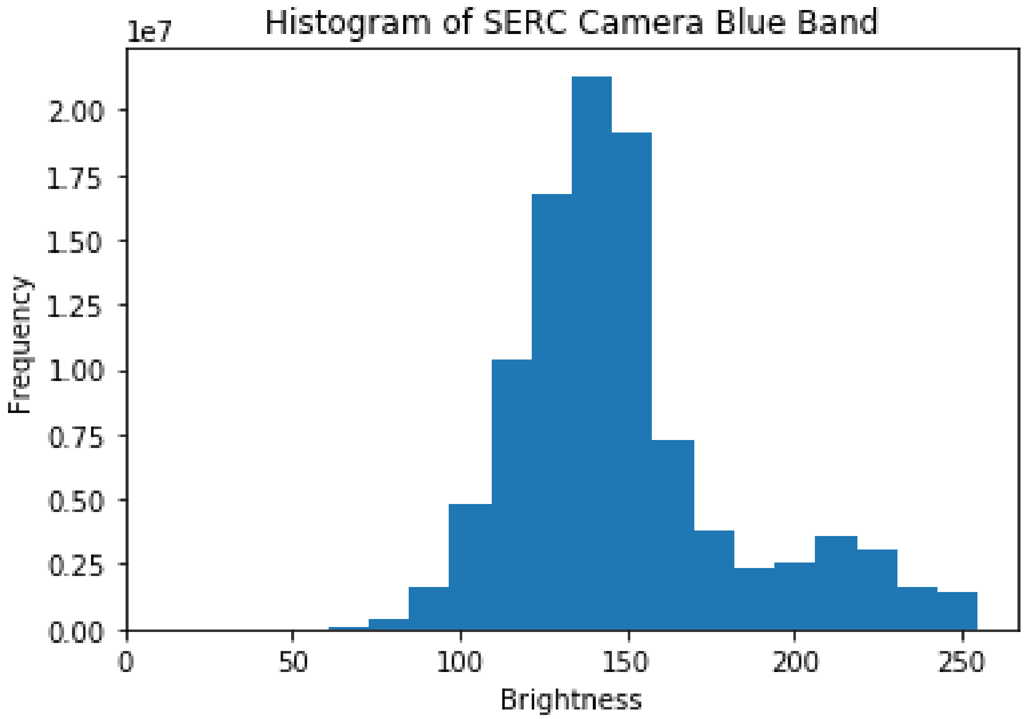
```
In [11]: plt.hist(np.ravel(SERC_RGBcam_array[:,:,1]),20);
         plt.title('Histogram of SERC Camera Green Band')
         plt.xlabel('Brightness'); plt.ylabel('Frequency')

Out[11]: Text(0,0.5,'Frequency')
```



```
In [12]: plt.hist(np.ravel(SERC_RGBcam_array[:,:,2]),20);
         plt.title('Histogram of SERC Camera Blue Band')
         plt.xlabel('Brightness'); plt.ylabel('Frequency')

Out[12]: Text(0,0.5,'Frequency')
```



Red Band min and max

```
In [13]: np.amin(SERC_RGBcam_array[:,:,0])

Out[13]: 21

In [14]: np.amax(SERC_RGBcam_array[:,:,0])

Out[14]: 255
```

Green Band min and max

```
In [15]: np.amin(SERC_RGBcam_array[:,:,1])

Out[15]: 5

In [16]: np.amax(SERC_RGBcam_array[0,:,1])

Out[16]: 255
```

Blue Band min and max

```
In [17]: np.amin(SERC_RGBcam_array[:,:,2])

Out[17]: 12

In [18]: np.amax(SERC_RGBcam_array[:,:,2])

Out[18]: 255
```

Finding the UTM Zone

```
In [19]: SERC_RGBcam_metadata['projection']
```

```
Out[19]: 'PROJCS["WGS 84 / UTM zone 18N",GEOGCS["WGS 84",DATUM["WGS_1984",SPHEROID["WGS 84",63
```

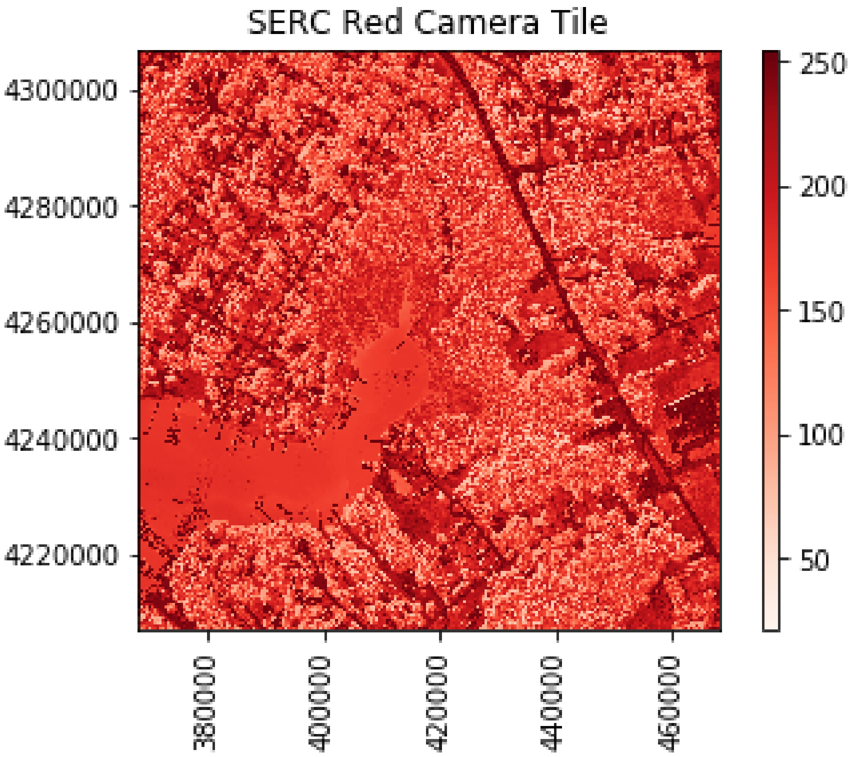
Looks like we are in UTM zone 18N!

```
In [20]: SERC_RGBcam_metadata
```

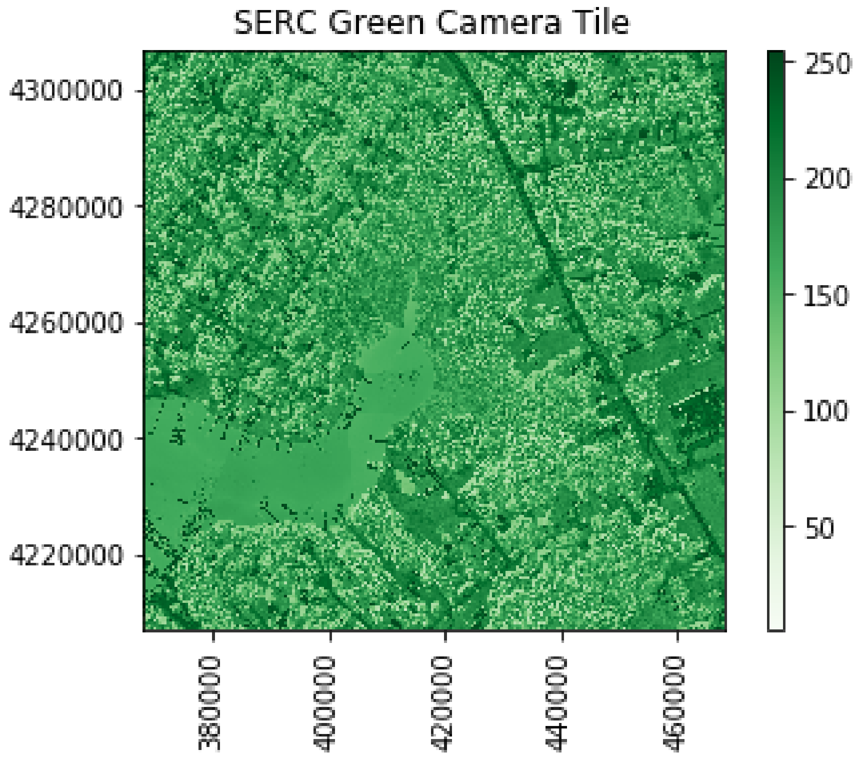
```
Out[20]: {'array_cols': 10000,
          'array_rows': 10000,
          'bands': 3,
          'driver': 'GeoTIFF',
          'ext_dict': {'xMax': 468000.0,
           'xMin': 368000.0,
           'yMax': 4307000.0,
           'yMin': 4207000.0},
          'extent': (368000.0, 468000.0, 4207000.0, 4307000.0),
          'geotransform': (368000.0, 0.1, 0.0, 4307000.0, 0.0, -0.1),
          'noDataValue': None,
          'pixelHeight': -0.1,
          'pixelWidth': 0.1,
          'projection': 'PROJCS["WGS 84 / UTM zone 18N",GEOGCS["WGS 84",DATUM["WGS_1984",SPHER
          'scaleFactor': 1.0}
```

Plotting the green and blue bands

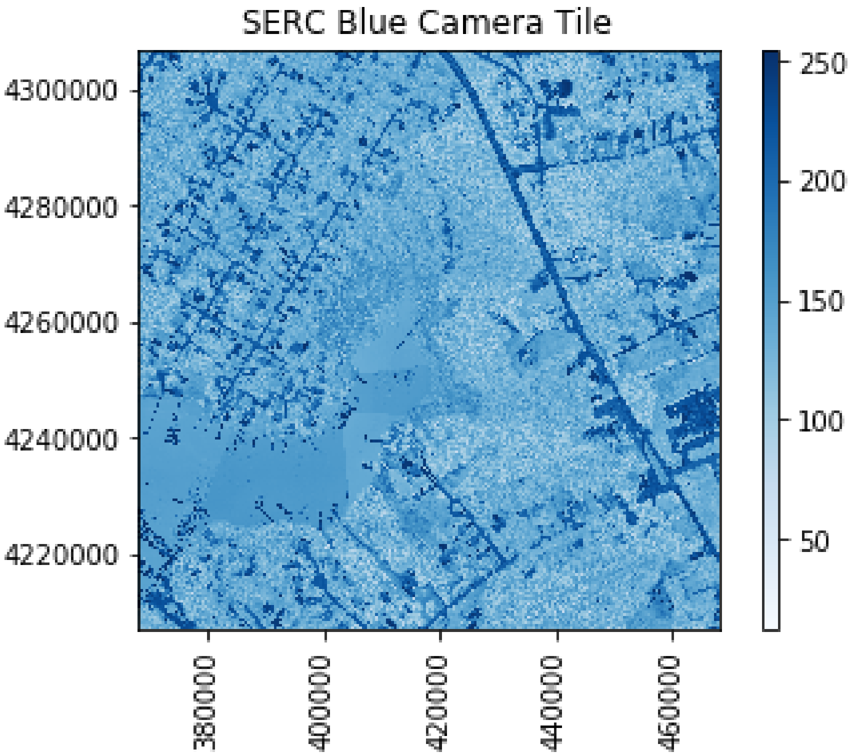
```
In [21]: plot_band_array(SERC_RGBcam_array[:,:,0],
                         SERC_RGBcam_metadata['extent'],
                         (21,255),
                         title='SERC Red Camera Tile',
                         cbar='on',
                        colormap='Reds')
```



```
In [22]: plot_band_array(SERC_RGBcam_array[:,:,1],
                          SERC_RGBcam_metadata['extent'],
                          (5,255),
                          title='SERC Green Camera Tile',
                          cbar='on',
                         colormap="Greens")
```



```
In [23]: plot_band_array(SERC_RGBcam_array[:,:,2],
                          SERC_RGBcam_metadata['extent'],
                          (12,255),
                          title='SERC Blue Camera Tile',
                          cbar='on',
                         colormap='Blues')
```


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