vscholl_week3

June 27, 2018

In [1]:
```
# check python version
import sys
sys.version
```

Out[1]:
```
'3.6.5 |Anaconda, Inc.| (default, Apr 26 2018, 08:42:37) \n[GCC 4.2.1 Compatible Clang
```

In [2]:
```
# import necessary libraries: gdal, numpy, matplotlib
#!conda install gdal
import gdal
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
```

In [3]:
```
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
```

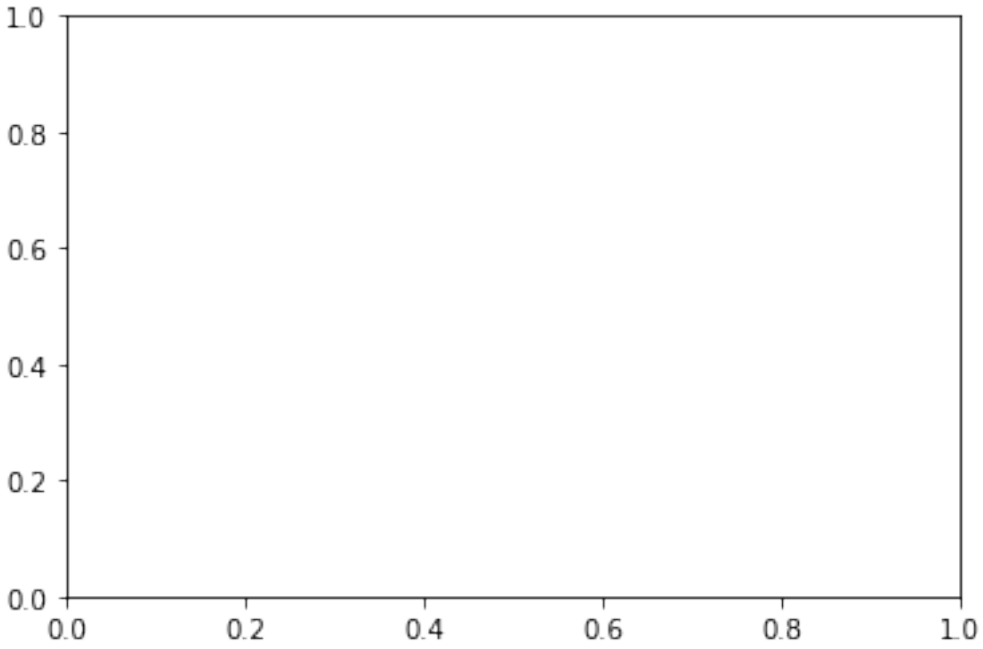
```
In [4]: RGB_geotif = './2017_SERC_2_368000_4306000_image.tif'
        SERC_RGBcam_array, SERC_RGBcam_metadata = RGBraster2array(RGB_geotif)

In [5]: SERC_RGBcam_array.shape

Out[5]: (10000, 10000, 3)
```

```
In [6]: #Display information stored in header
        for key in sorted(SERC_RGBcam_metadata.keys()):
          print(key)
```

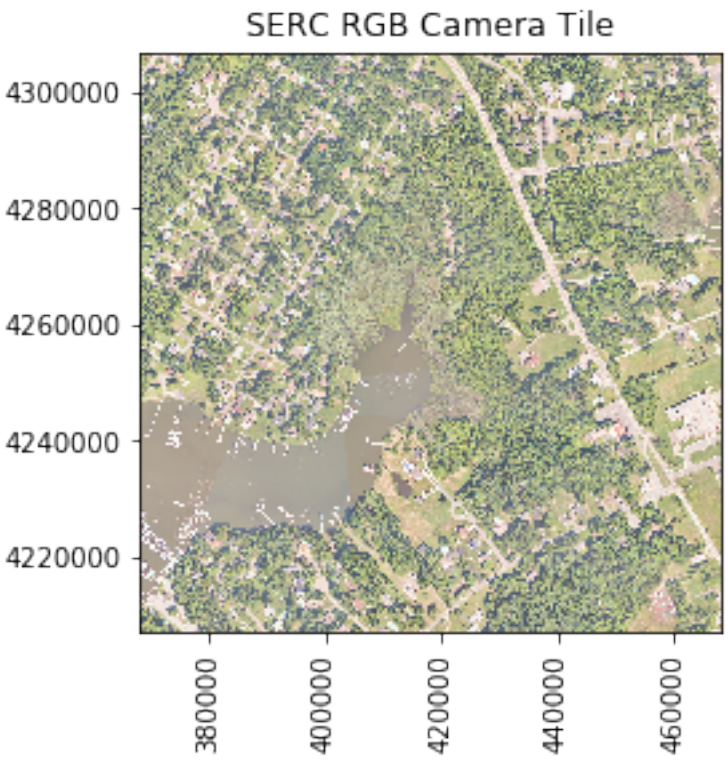
```
array_cols
array_rows
bands
driver
ext_dict
extent
geotransform
noDataValue
pixelHeight
pixelWidth
projection
scaleFactor
```

```
In [7]: 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',
```

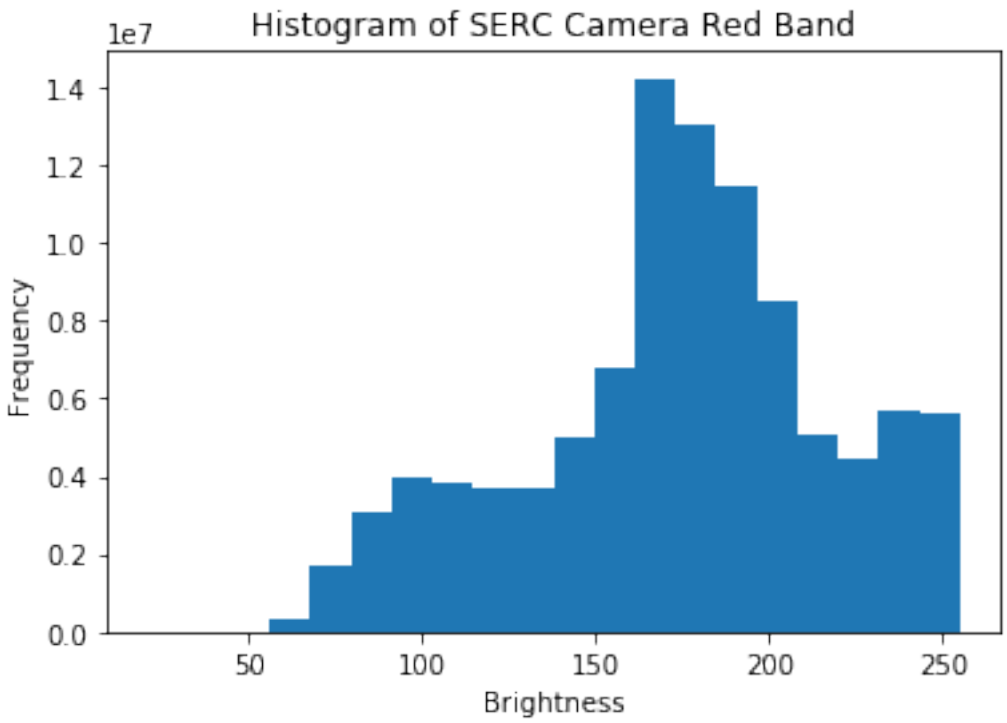
```
                    cbar='off')'''

    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 [8]:
```
plot_band_array(SERC_RGBcam_array,
                SERC_RGBcam_metadata['extent'],
                (1,255),
                title='SERC RGB Camera Tile',
                cbar='off')
```

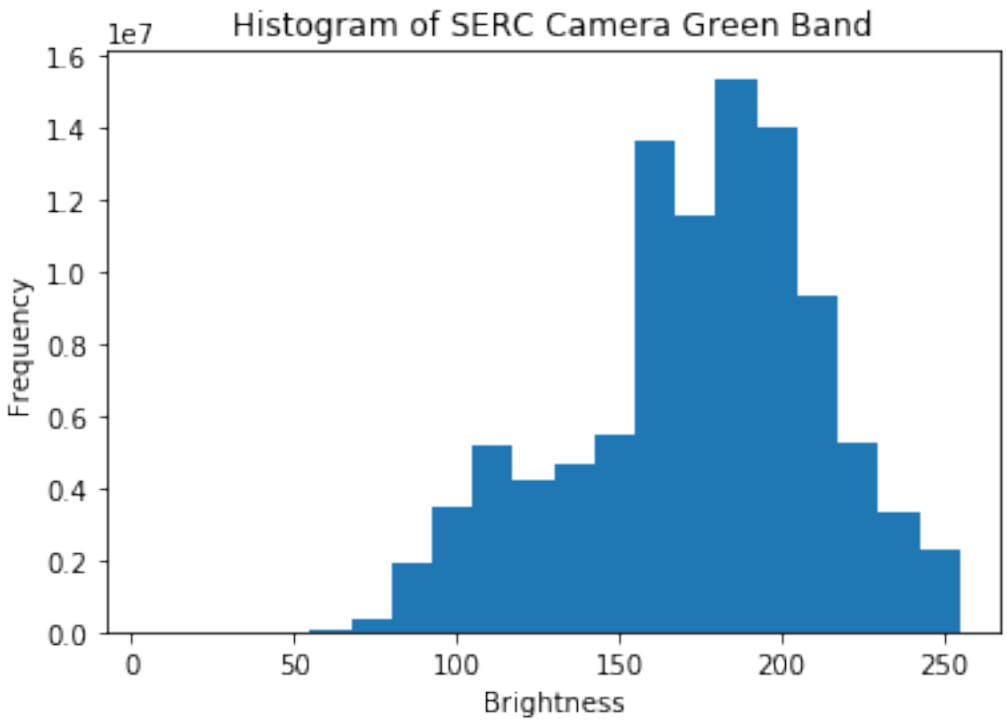



```
In [10]: plt.hist(np.ravel(SERC_RGBcam_array[:,:,0]),20);
         plt.title('Histogram of SERC Camera Red Band')
         plt.xlabel('Brightness'); plt.ylabel('Frequency')

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



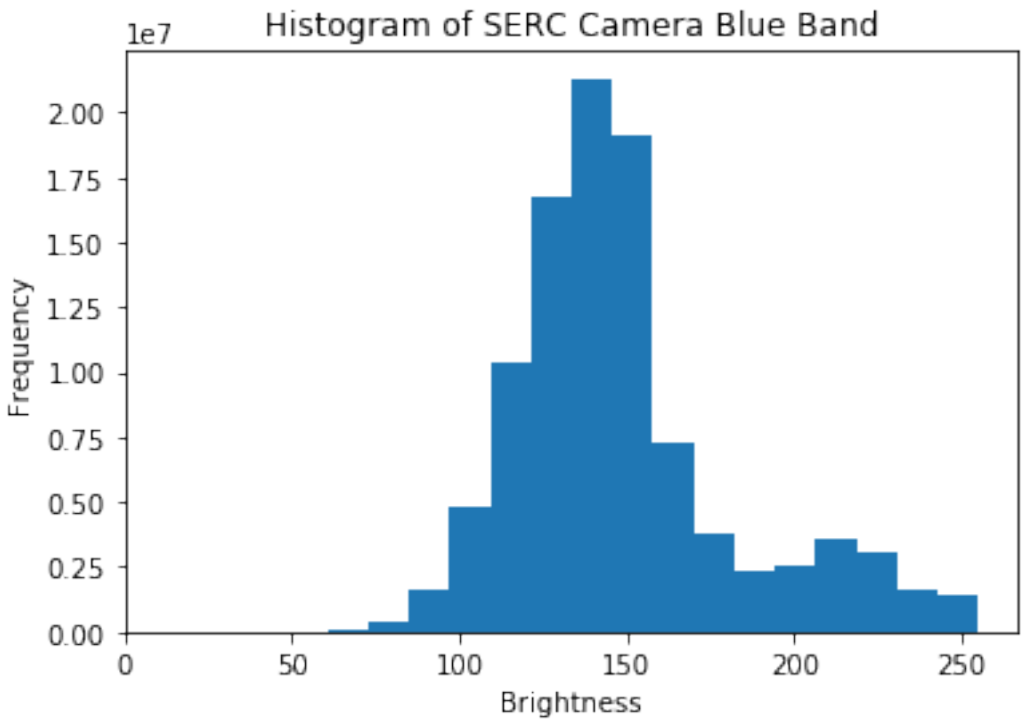
1. Plot histograms of the green and blue bands.

```
In [12]: # Histogram of green band
         plt.hist(np.ravel(SERC_RGBcam_array[:,:,1]),20);
         plt.title('Histogram of SERC Camera Green Band')
         plt.xlabel('Brightness'); plt.ylabel('Frequency')

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




```
In [13]: # Histogram of blue band
         plt.hist(np.ravel(SERC_RGBcam_array[:,:,2]),20);
         plt.title('Histogram of SERC Camera Blue Band')
         plt.xlabel('Brightness'); plt.ylabel('Frequency')
```

```
Out[13]: Text(0,0.5,'Frequency')
```



2. Explore the data

```
In [18]: # Determine the minimum and maximum reflectance for each band.
         # Print these values with a print statement.
         # HINT: Use the numpy functions np.amin() and np.amax()

         # red
         print('Min reflectance in red band: ', str(np.amin(np.ravel(SERC_RGBcam_array[:,:,0])))
         print('Max reflectance in red band: ', str(np.amax(np.ravel(SERC_RGBcam_array[:,:,0])))

         # green
         print('Min reflectance in green band: ', str(np.amin(np.ravel(SERC_RGBcam_array[:,:,1]
         print('Max reflectance in green band: ', str(np.amax(np.ravel(SERC_RGBcam_array[:,:,1]

         # blue
         print('Min reflectance in blue band: ', str(np.amin(np.ravel(SERC_RGBcam_array[:,:,2]]
         print('Max reflectance in blue band: ', str(np.amax(np.ravel(SERC_RGBcam_array[:,:,2]]
```

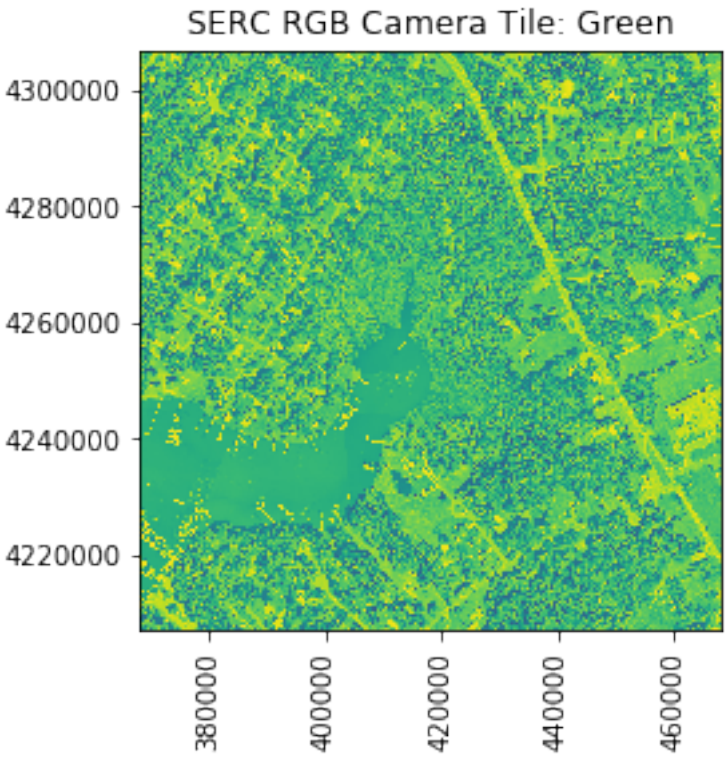
```
Min reflectance in red band:  21
Max reflectance in red band:  255
Min reflectance in green band:  5
Max reflectance in green band:  255
Min reflectance in blue band:  12
Max reflectance in blue band:  255
```

```
In [22]: # b. What UTM zone is this data in?
         # Hint: Print out SERC_RGBcam_metadata['projection']

         print(SERC_RGBcam_metadata['projection'])
```

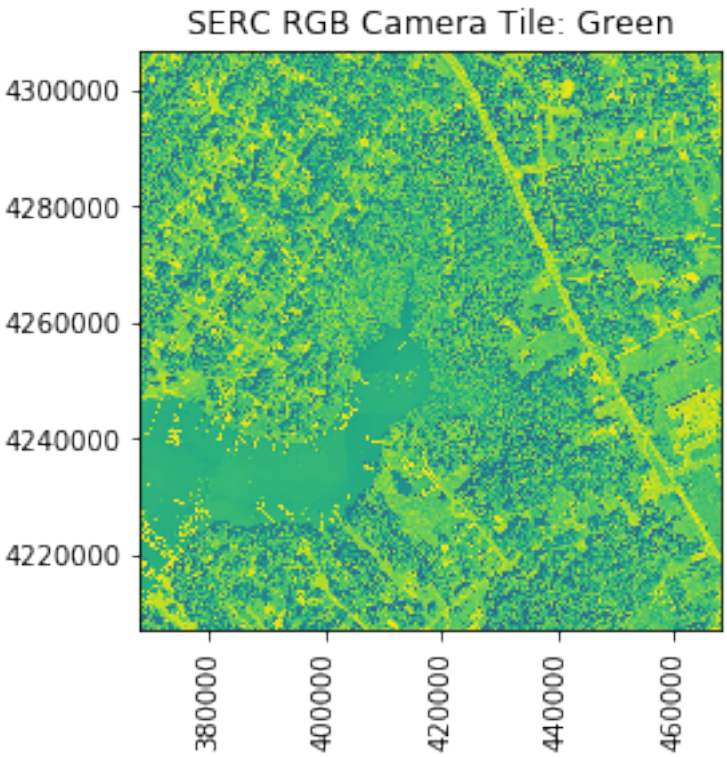
```
PROJCS["WGS 84 / UTM zone 18N",GEOGCS["WGS 84",DATUM["WGS_1984",SPHEROID["WGS 84",6378137,298.2
```

b. This data is in UTM zone 18N

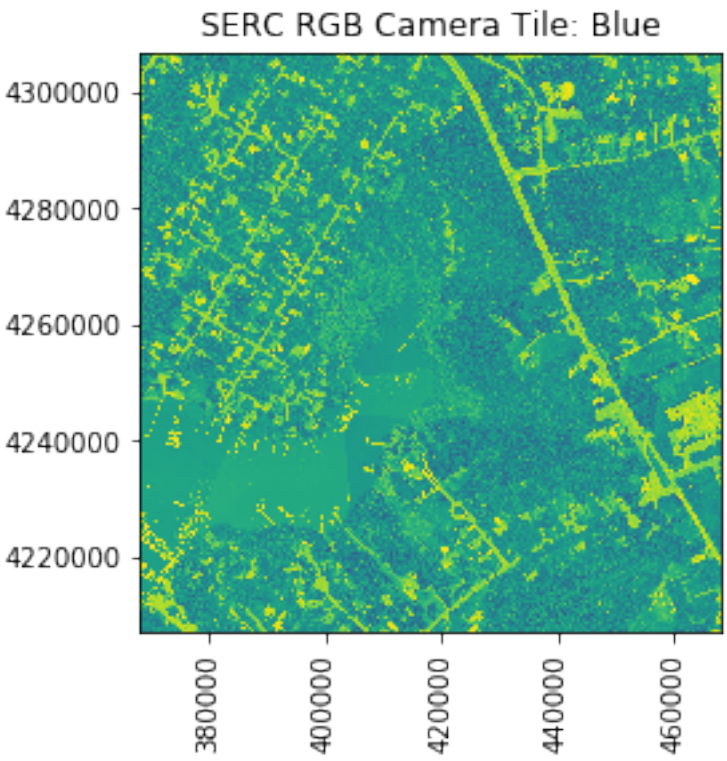
```
In [26]: # c. Use the plot_band_array function to plot each band of the camera image separatel
         # HINT: Use splicing to extract each band (e.g., SERC_RGBcam_array[:,:,0]).

         red = SERC_RGBcam_array[:,:,0]
         green = SERC_RGBcam_array[:,:,1]
         blue = SERC_RGBcam_array[:,:,2]

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



```
In [27]: plot_band_array(green,
                         SERC_RGBcam_metadata['extent'],
                         (1,255),
                         title='SERC RGB Camera Tile: Green',
                         cbar='off')
```




```
In [28]: plot_band_array(blue,
                         SERC_RGBcam_metadata['extent'],
                         (1,255),
                         title='SERC RGB Camera Tile: Blue',
                         cbar='off')
```


SERC RGB Camera Tile: Blue
4300000
4280000
4260000
4240000
4220000
380000
400000
420000
440000
460000