**Exercise 7 – Importing and applying Quality Control information**

Prerequisite

* Python: <http://www.python.org/doc/>
* GDAL: <http://www.gdal.org/>
* Geospatial Data Abstraction Library
* Raster data access
* Used by commercial software like ArcGIS
* C++ library, but Python bindings exist
* Matplotlib: <http://matplotlib.org/>

Note: Outputs (e.g., figure, statistics, metainfo) might be different by input

In this exercise you will import MODIS Leaf Area Index (LAI) and corresponding Quality Control (QC) for western Europe. The image will be imported in Python/Numpy/GDAL/matplotlib, scaled to physical LAI units and screened using the quality control data.

1. **Import and play around with MODIS data**

* Start Python and import libraries
* Open the MODIS “LAI\_1km” TIFF file of day 2002-­‐225 using GDAL and convert GDAL object as Numpy array
* Plot the LAI layer

import gdal

import matplotlib.pyplot as plt

import numpy as np

import os

## set your work directory

os.chdir('/Users/Shared/GE529-2020/data/')

# open dataset

lai = gdal.Open('MOD15A2H.A2002225.h18v04.006.2015150063613.tif')

# read raster as numpy array

laiArray = lai.ReadAsArray()

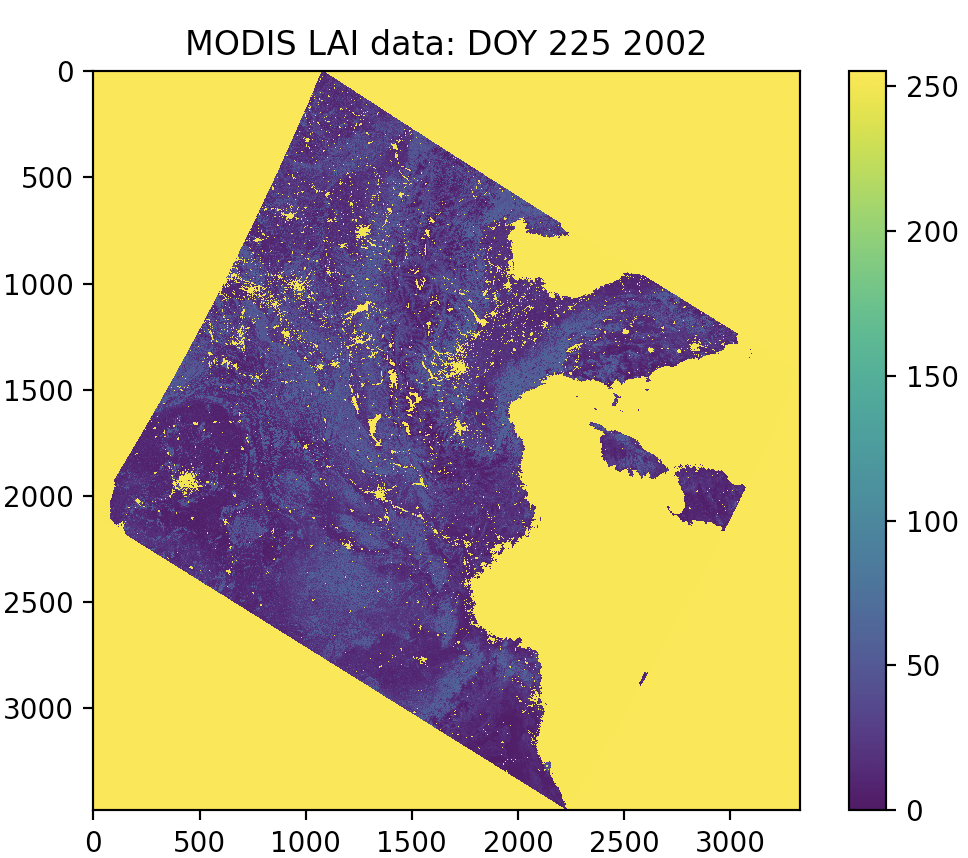
# plot the LAI layer

plt.imshow(laiArray, interpolation='none')

plt.title('MODIS LAI data: DOY 225 2002')

plt.colorbar()

plt.show()



**Figure 1**. Overview of MODIS LAI layer

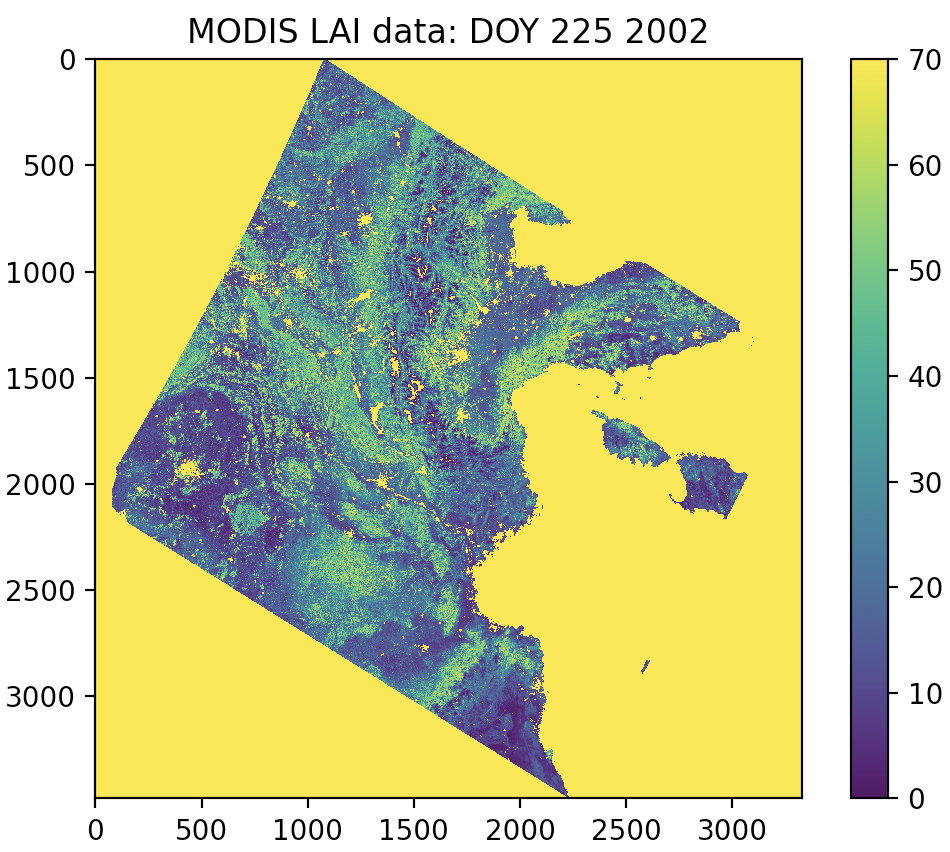
* To improve the contrast by applying minimum and maximum value, the matplotlib figure will now show more contrast

plt.imshow(laiArray, interpolation='none', vmin=0, vmax=7)

plt.title('MODIS LAI data: DOY 225 2002')

plt.colorbar()

plt.show()



**Figure 2**. Improving image contrast

1. **Apply Quality Control**

* Open MODIS “FparLai\_QC” TIFF of day 2002-­‐225 using GDAL, and covert it to Numpy array

From the Documentation and the HDF file it can be derived that bits {5,6,7} of every QC pixel contain quality control information that is produced by the MODIS LAI/Fpar algorithm (see MOD 15 product guide for the exact interpretation of all QC bits).

* In order to extract the QC information we need to applying Python bit wise operation to the QC array.
* See <https://lpdaac.usgs.gov/documents/2/mod15_user_guide.pdf> for more information about the quality control bits in MOD15.
* Because we are only interested in the bits {5,6,7} we first set all other bits to value '0' by applying a bitwise AND (Python: &) operation to each pixel. This can be easily done by using decimal value '224' which corresponds to binary value '11100000'.
* Next we want to shift the remaining bits {5,6,7} five positions to the right in order to have them on bit positions {0,1,2}. This could be carried out using a bit-­‐shift operation (Python: >>).
* Use np.unique(qcArray) to check the extracted QC information

qc = gdal.Open('MOD15A2H.A2002225.h18v04.FparLai\_QC.tif')

qcArray = qc.ReadAsArray()

# & Binary AND, 224 == 11100000, get 5,6,7 bit from qc layer

qcArray = qcArray & 224

# >> Binary Right Shift, right shift 5,6,7 bit, 11100000 will become 111

qcArray = qcArray >>5

# check the output array, you will find that only the {5,6,7} bits were extracted from original pixel values.

np.unique(qcArray)

# you will get: array([0, 1, 3, 4, 7], dtype=uint8)

* Now we will use the MOD15 QC info to create a mask that contains all good quality pixels. Follow the following steps:
* Create a mask layer like LAI

mask = qcArray==0

# Select a black/white colormap

cmap = plt.cm.gray

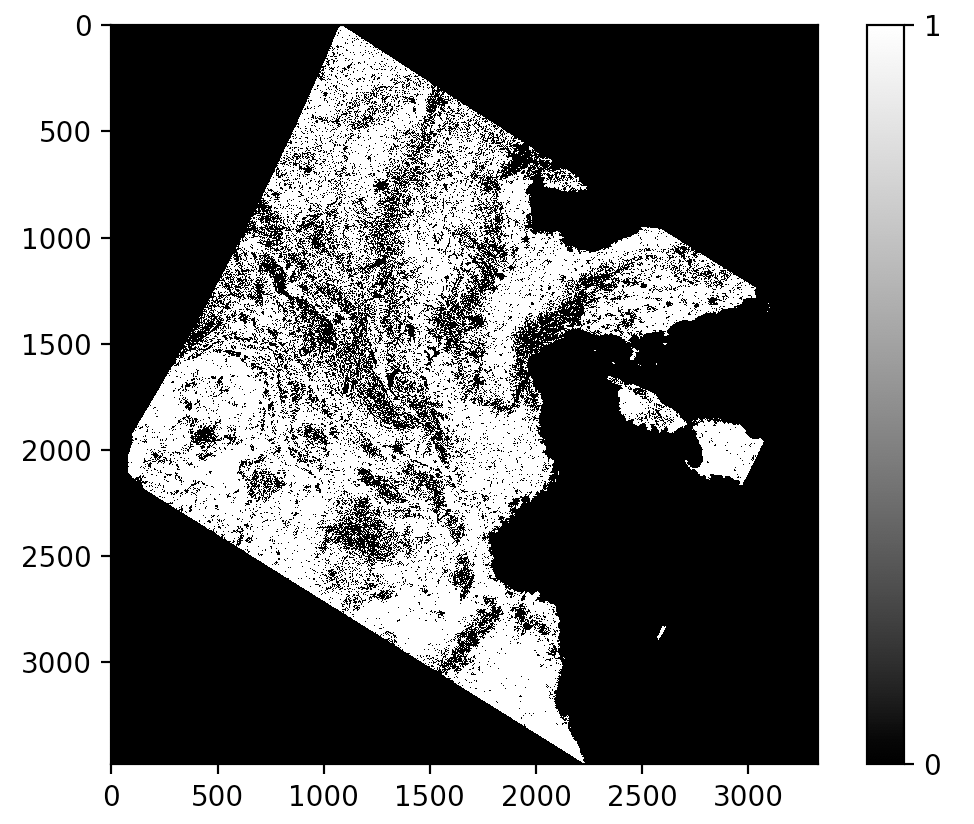
# plot the data

plt.imshow(mask, interpolation='nearest', cmap=cmap)

# add a colorbar

plt.colorbar(ticks=[0,1])

plt.show()



**Figure 3**. QC mask. 0 is bad quality, 1 is good quality.

* To plot LAI only for the valid pixels (mask==True). We can use masked array for this. Masked array is like normal array, but has an associated mask. Remember that the mask in a masked array should be False for good data, so we can directly use QC to define plot areas.
* Also, We know form using LAI documentation that the conversion factor for converting DN values into physical LAI values is 0.1. Therefore we will apply Numpy function to apply this conversion factor.

laiArray = laiArray \* 0.1

laim = np.ma.array(laiArray, mask=qcArray)

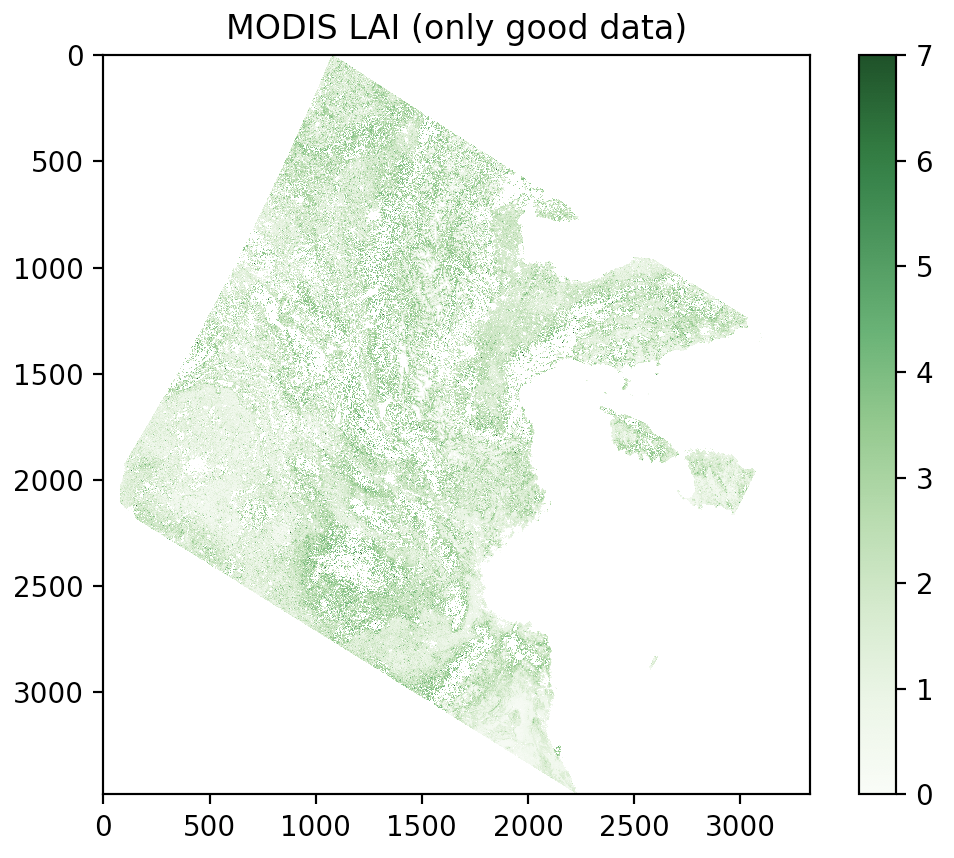
cmap=plt.cm.Greens

plt.imshow(laim, cmap=cmap, interpolation='nearest',vmin=0, vmax=7)

plt.title('MODIS LAI (only good data)')

plt.colorbar()

plt.show()



**Figure 4**. Overview of LAI layer –only good pixels

**End of Exercise 7**