THE UNIVERSITY OF ARIZONA

BE/BAT 485/585

Remote Sensing Data and Methods

Lab - 6

Instructor: Kamel Didan[1,2]

Helpers: Dr. Armando Barreto[1,2]

Mr. Truman Combs[1,2]

[1]BE Dept., University of Arizona,

[2]VIP Lab.

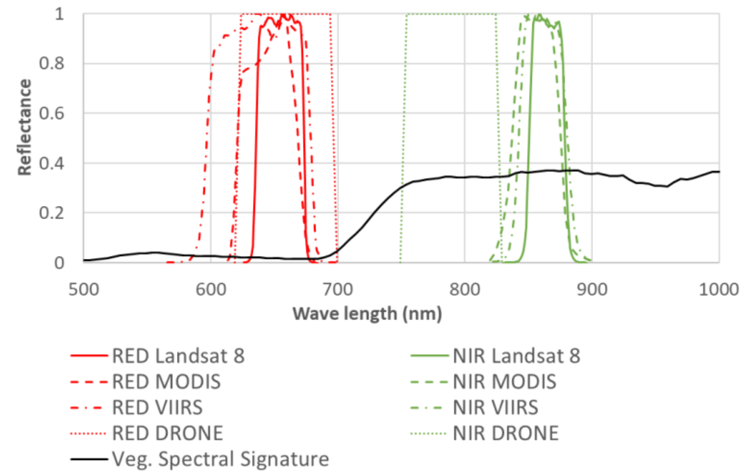
vip.arizona.edu

vegetation index & phenology Lab.

...Understanding a piece of the Earth system

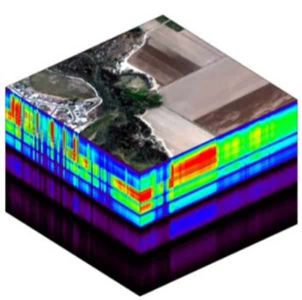
Spectral/Spatial convolution

- Convolution uses the Relative Spectral Responses (RSR) to simulate any sensor from a higher definition one. This makes it possible to simulate any sensor from hyperspectral data (ex: MODIS, VIIRS or Landsat from NEON-AOP or AVIRIS/N hyperspectral data).
- The RSR defines how the sensor collects and records the ‘EMR’ data and how sensitive.
- There are many kinds of convolutions, ex:
  - Spectral convolution
    - Aggregates bands (from finer ones)
  - Spatial convolution
    - Aggregates pixels (from finer obs)



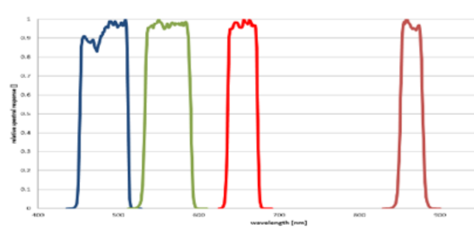
Simulated sensors

NEON hyperspectral

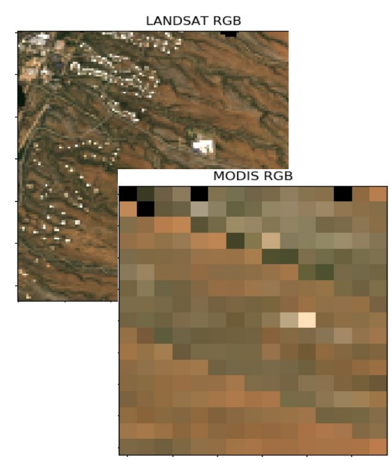


RSR

$$L_b = \frac{\int_{\lambda 1}^{\lambda 2} L_h \mathrm{SRF}\, d\lambda}{\int_{\lambda 1}^{\lambda 2} \mathrm{SRF}\, d\lambda},$$

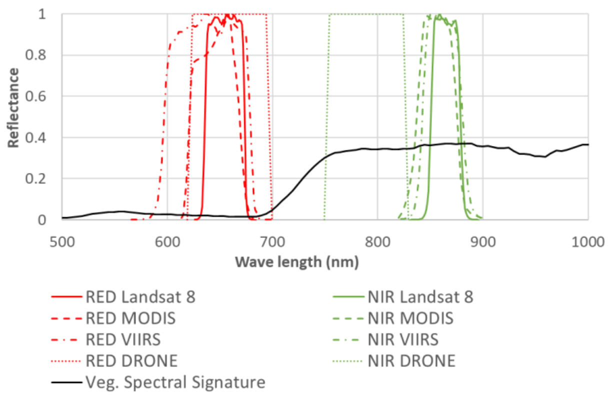


Convolution



Exercise #1: Convolution

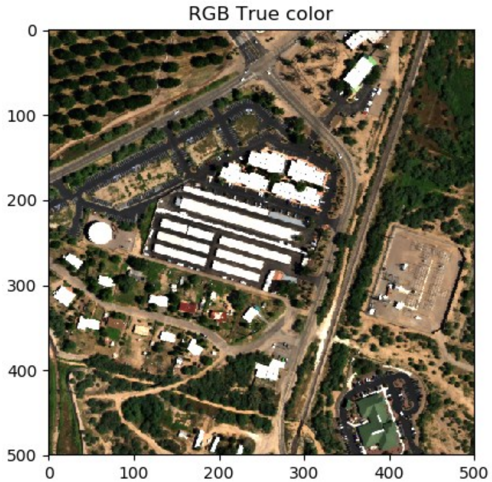
- **Data Input**
  - We will work with HDF5 file format (a modern and highly specialized data format)
  - You can use text file with NEON wavelength values (or read straight from the HDF5 file)
  - RSR for each sensor (MODIS, VIIRS, Landsat 5/8, Sentinel2)
- **You will learn how to**
  - Read any data layer/band from the HDF 5 file
  - Perform a Spectral convolution & Spatial resampling
  - Compare how the multispectral differs from the hyperspectral original data
    - RGB, FCC, statistical analysis, etc.
- **Homework**
  - Use the provided example (code) for Sentinel 2 (ESA platform)
  - Simulate the following sensors from NEON:
    - Landsat 8 OLI, MODIS/VIIRS [but at 50m – We do not have enough coverage to simulate the actual 250m/300m data]
  - General a figure plot with all RSRs for all bands and all sensors
  - Extract different (objects) Spectral Signatures and plot from original and simulated sensor data

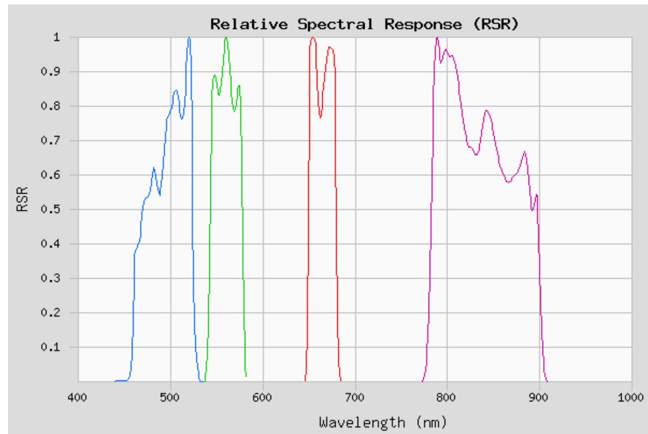


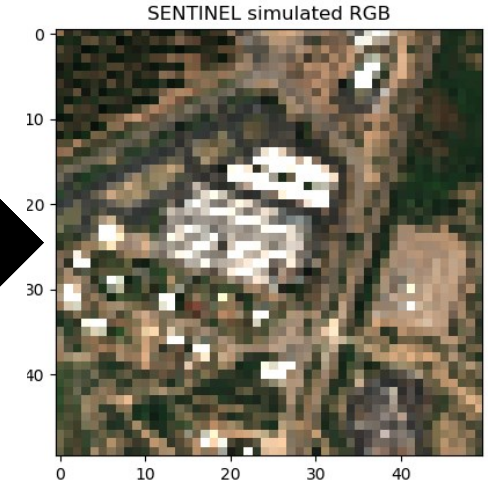
Instructions:

- Acquire the following from D2L:
  - viplab_lib3.py (new library)
  - viplab_convolution.py (new library)
  - PickImageToWorkWith.h5
  - NEON_wavelength_values.txt (as before)
  - NEON_Landsat8_RSR.csv
  - NEON_MODIS_RSR.csv
  - NEON_VIIRS_RSR.csv
  - NEON_Sentinel2A_RSR.csv
  - **BE485_Lab6_Ex1.ipynb**

Examples from NEON

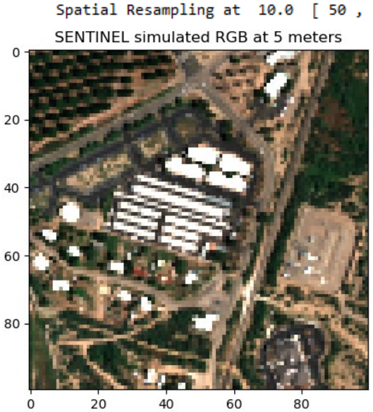




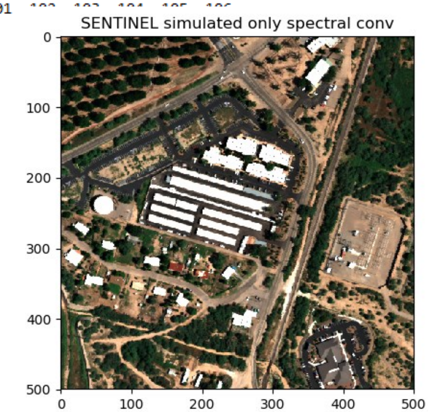



```
Reading RSR: NEON_Sentinel2A_RSR.csv
Convolution:
Sensor: SENTINEL2A
Resolution: 10.0 meters
Bands: ['BLUE', 'GREEN', 'RED', 'NIR']
Processing convolution...
 band: BLUE
 min= 440.0 max= 535.0
  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31 .
 band: GREEN
 min= 537.0 max= 582.0
  32  33  34  35  36  37  38  39  40  41 .
 band: RED
 min= 646.0 max= 684.0
  53  54  55  56  57  58  59  60  61 .
 band: NIR
 min= 773.0 max= 908.0
  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101
Spatial Resampling at  10.0  [ 50 , 50 ]
```

Stats and output during program run



Spectral convolution + spatial at 5 meters



Only Spectral convolution
No spatial (retains native spatial resolution)

The following is similar to Machine Learning

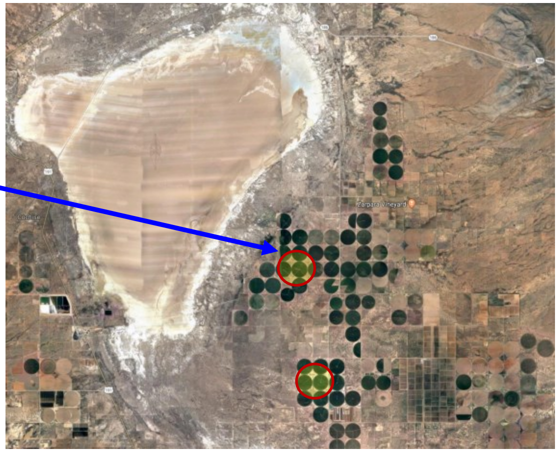
What is Masking

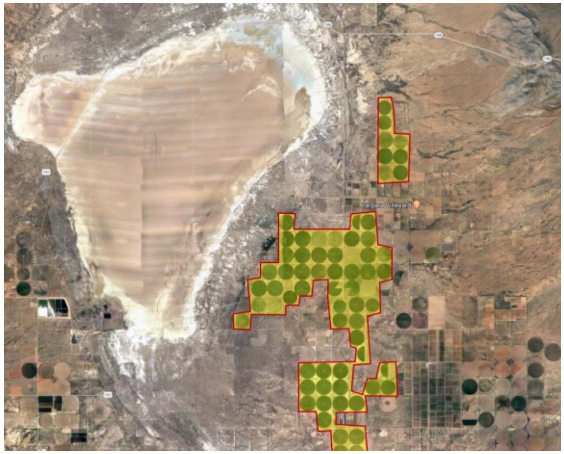
- Let us say we are interested in a feature in an image
  - We can run a spectral signature or any other analysis for one or few pixels over the visually identified feature to learn how it is supposed to look like in the spectral domain
- But what if we want to identify the 'full' spatial extent of this feature
- This is called masking, which basically creates a True/False or 0/1 image that indicates where the feature is
- We can use this mask to analyze the full image for our feature of interest and ignore everything else
- Generating mask can be done

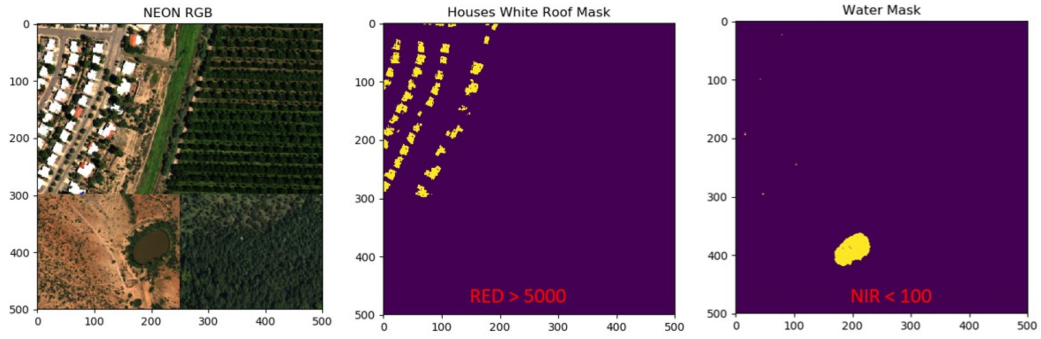
  •Visually or Manually

  •Using machine learning based on :

  •Threshold technique (like a machine learning decision tree algorithm)
  - You tell the computer what to look for based on training and the computer does the rest
  - For example: We know that a water feature must have an NDVI < -0.5 (from training). Then the machine will look over the image for any pixel with NDVI < -0.5 and assigns it to water
  - You can use other metrics too (Red, NIR, Green, Thermal, etc…) any things that defines and separates the features well could be used (hence signature)
  - Try to experiment in this exercise and you will notice that you can improve the classification performance if you combine/include more than one test
  - Look at the provided examples and develop your own







Exercise #2: Data Masking

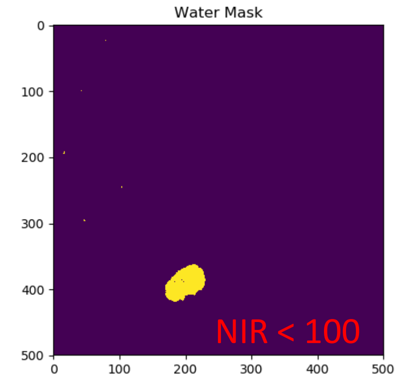
- **Read NEON data cube (We will use a composite cube in BSQ format)**
- Extract spectral signatures from Cube
  - By Pixels from an object (you already did this)
  - By a window, 3x3, 5x5, etc (use slicing – it should be easy)
  - Extract using a spatial Mask (a mask is again a 0/1 image that defines the extent of the feature(s))
    - Use the provided predefined mask(s)
- **Homework**:
  - Follow the exercise code and,
  - Compute the stats for all bands over the predefined masks (BLUE, GREEN, RED, NIR, SWIR1)
    - Average signal over the masked feature for a single band
  - Modify the code to mask through the full cube and generate population (masked) spectra
    - Average signal over mask feature for all bands (426) and generate an average plot of spectral signatures of the feature
  - Generate your own mask(s) using what you have learned so far in terms of spectral signatures (and/or NDVI)
    - Combine more than one thresholds/tests to generate a robust more useful mask

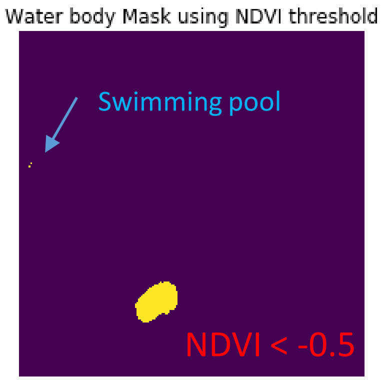
Instructions:

- Download from D2L files:
  - NEON_Composed_Img.bsq
  - NEON_wavelength_values.txt
  - NEON_mask.bsq
  - viplab_lib3.py
  - **BE485_Lab6_Ex2.ipynb**

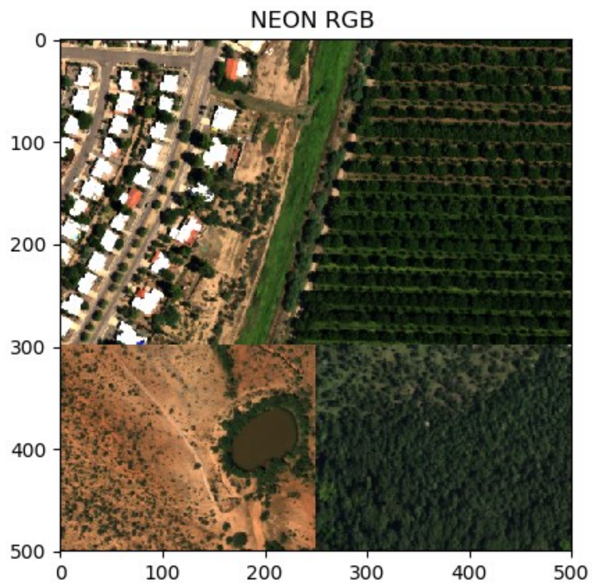
FYI

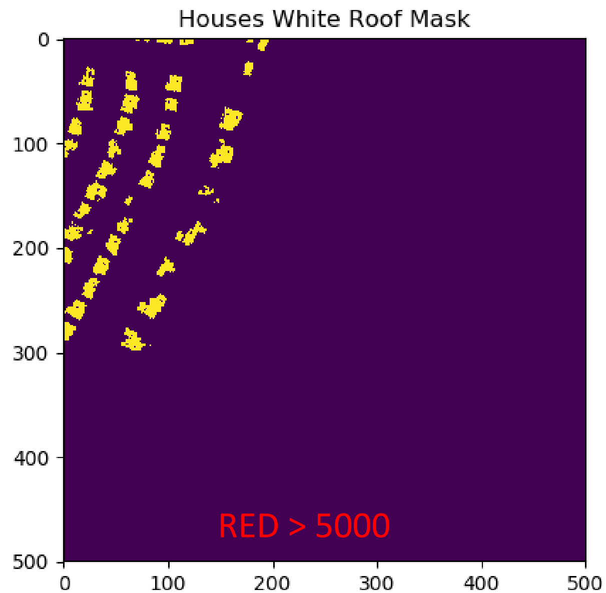
- What to do with noise in a mask (features)
  - Ignore it if not too much
  - Analyze the spectral signatures
  - Learn what the noise is and if it has any behavior that sets it apart, then enhance your (mask, features extraction) algorithms to avoid that noise
  - The Examples to the right are:
    - Masking water using NIR < 100 only
    - Masking Water using NDVI < -0.5 only

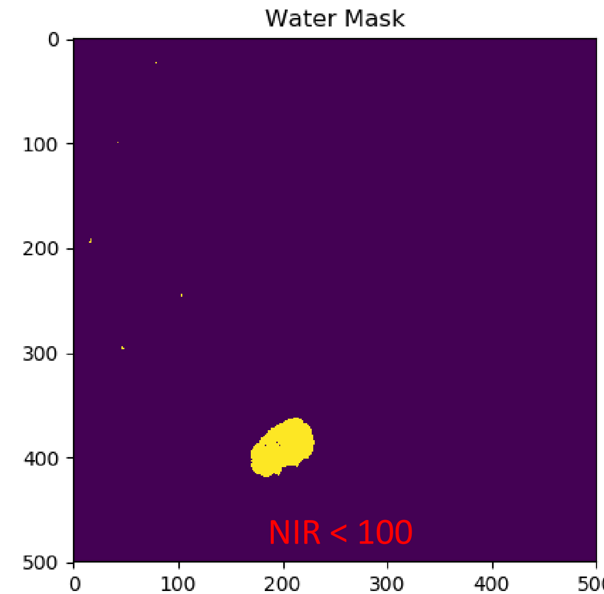


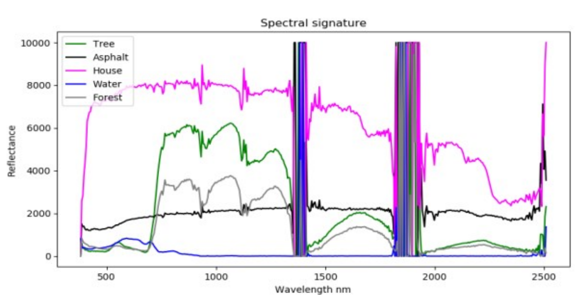


How to Design a Mask



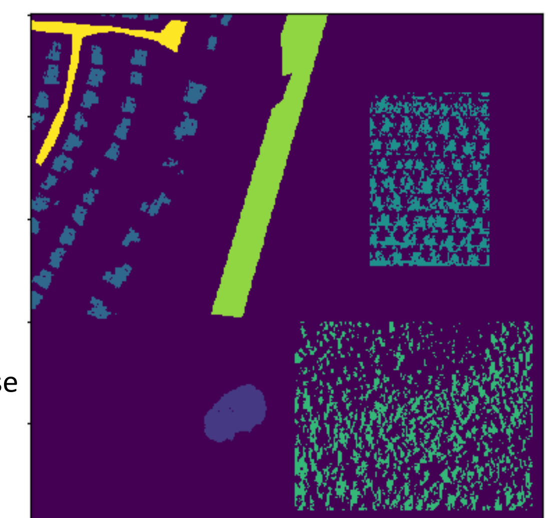






Layer 0 (Band 0)

Mask keys:
1 Water
2 White roofs
3 Walnut trees
4 Forest
5 Grass
6 Road
0 everything else



Layer 1 (Band 1)

TIPS

- Remember you need not write new code each time
- You simply reuse (from older lab or online) and modify code
  - Clever programmer never write code from scratch
- That way you do not waste time in creating new code each time
  - Just start from the notebooks we share with you and
    - Add, modify, and/or improve them

Reference information

Remember what colors correspond to what bands

Data and Specs

Blue

Green

Red

NIR

SWIR1

Index	Wavelength
0	381.27301
1	386.281097
2	391.289215
3	396.297302
4	401.305511
5	406.313599
6	411.321686
7	416.329895
8	421.338013
9	426.3461
10	431.354309
11	436.362396
12	441.370514
13	446.378601
14	451.38681
15	456.394897
16	461.403015
17	466.411194
18	471.419312
19	476.427399
20	481.435486
21	486.443695
22	491.451813
23	496.4599
24	501.468109
25	506.476196
26	511.484314
27	516.492493
28	521.50061
29	526.508728
30	531.516785
31	536.525024
32	541.533081
33	546.541199
34	551.549377
35	556.557495
36	561.565613
37	566.573792
38	571.581909
39	576.590027
40	581.598083
41	586.606323
42	591.61438
43	596.622498
44	601.630676
45	606.638794
46	611.646912
47	616.65509
48	621.663208
49	626.671326
50	631.679382
51	636.687622
52	641.695679
53	646.703796
54	651.711975
55	656.720093
56	661.72821
57	666.736328
58	671.744507
59	676.752625
60	681.760681
61	686.768921
62	691.776978
63	696.785095
64	701.793274
65	706.801392
66	711.809509
67	716.817627
68	721.825806
69	726.833923
70	731.84198
71	736.85022
72	741.858276
73	746.866394
74	751.874573
75	756.88269
76	761.890808
77	766.898926
78	771.907104
79	776.915222
80	781.923279
81	786.931519
82	791.939575
83	796.947693
84	801.955872
85	806.963989
86	811.972107
87	816.980225
88	821.988403
89	826.996521
90	832.004578
91	837.012817
92	842.020874
93	847.028992
94	852.03717
95	857.045288
96	862.053406
97	867.061523
98	872.069702
99	877.07782
100	882.085876
101	887.094116
102	892.102173
103	897.110291
104	902.118408
105	907.126587
106	912.134705
107	917.142822
108	922.151001
109	927.159119
110	932.167175
111	937.175415
112	942.183472
113	947.191589
114	952.199707
115	957.207886
116	962.216003
117	967.224121
118	972.2323
119	977.240417
120	982.248474
121	987.256714
122	992.264771
123	997.272888
124	1002.281006
125	1007.289185
126	1012.297302
127	1017.30542
128	1022.313599
129	1027.321655
130	1032.329834
131	1037.338013
132	1042.346069
133	1047.354248
134	1052.362305
135	1057.370483
136	1062.37854
137	1067.386719
138	1072.394897
139	1077.402954
140	1082.411133
141	1087.419312
142	1092.427368
143	1097.435547
144	1102.443604
145	1107.451782
146	1112.459961
147	1117.468018
148	1122.476196
149	1127.484253
150	1132.492432
151	1137.500488
152	1142.508667
153	1147.516846
154	1152.524902
155	1157.533081
156	1162.54126
157	1167.549316
158	1172.557495
159	1177.565552
160	1182.57373
161	1187.581787
162	1192.589966
163	1197.598145
164	1202.606201
165	1207.61438
166	1212.622559
167	1217.630615
168	1222.638794
169	1227.646851
170	1232.655029
171	1237.663086
172	1242.671265
173	1247.679443
174	1252.6875
175	1257.695679
176	1262.703857
177	1267.711914
178	1272.720093
179	1277.728149
180	1282.736328
181	1287.744385
182	1292.752563
183	1297.760742
184	1302.768799
185	1307.776978
186	1312.785156
187	1317.793213
188	1322.80127
189	1327.809448
190	1332.817627
191	1337.825684
192	1342.833862
193	1347.842041
194	1352.850098
195	1357.858276
196	1362.866455
197	1367.874512
198	1372.882568
199	1377.890747
200	1382.898926
201	1387.906982
202	1392.915161
203	1397.92334
204	1402.931396
205	1407.939575
206	1412.947754
207	1417.955811
208	1422.963867
209	1427.972046
210	1432.980225
211	1437.988281
212	1442.99646
213	1448.004639
214	1453.012695
215	1458.020874
216	1463.029053
217	1468.037109
218	1473.045166
219	1478.053345
220	1483.061523
221	1488.06958
222	1493.077759
223	1498.085938
224	1503.093994
225	1508.102173
226	1513.110352
227	1518.118408
228	1523.126465
229	1528.134644
230	1533.142822
231	1538.150879
232	1543.159058
233	1548.167236
234	1553.175293
235	1558.18335
236	1563.19165
237	1568.199707
238	1573.207764
239	1578.215942
240	1583.224121
241	1588.232178
242	1593.240356
243	1598.248535
244	1603.256592
245	1608.264648
246	1613.272949
247	1618.281006
248	1623.289063
249	1628.297241
250	1633.30542

These are the NEON-AOP images you can work with

Where & How to get the NEON AOP data

- https://vip.arizona.edu/classes/BE485/NEON_1KM/
  - NEON_D01_HARV_DP3_725000_4712000_reflectance.h5
    NEON_D09_WOOD_DP3_476000_5221000_reflectance.h5
    NEON_D14_JORN_DP3_316000_3613000_reflectance.h5
    NEON_D14_SRER_DP3_501000_3523000_reflectance.h5
    NEON_D14_SRER_DP3_502000_3523000_reflectance.h5
    NEON_D16_ABBY_DP3_551000_5070000_reflectance.h5

NEON_D01_HARV_DP3_725000_4712000_reflectance.h5

Massachusetts

Templeton

NEON_D09_WOOD_DP3_476000_5221000_reflectance.h5

North Dakota

Woodworth

NEON_D16_ABBY_DP3_551000_5070000_reflectance.h5

Washington

Vancouver

NEON_D14_JORN_DP3_31600_361300_reflectance.h5

New Mexico

Rincon

NEON_D14_SRER_DP3_502000_3523000_reflectance.h5

Arizona

Continental

NEON_D14_SRER_DP3_511000_3510000_reflectance.h5

Arizona

Madera Canyon

NEON_D14_SRER_DP3_502000_3524000_reflectance.h5

Arizona