Integrate eMODIS and NLCD to Update OK-Fire Fuel Model Map Using Python

JING WANG

OKLAHOMA STATE UNIVERSITY

DEPARTMENT OF GEOGRAPHY

Background

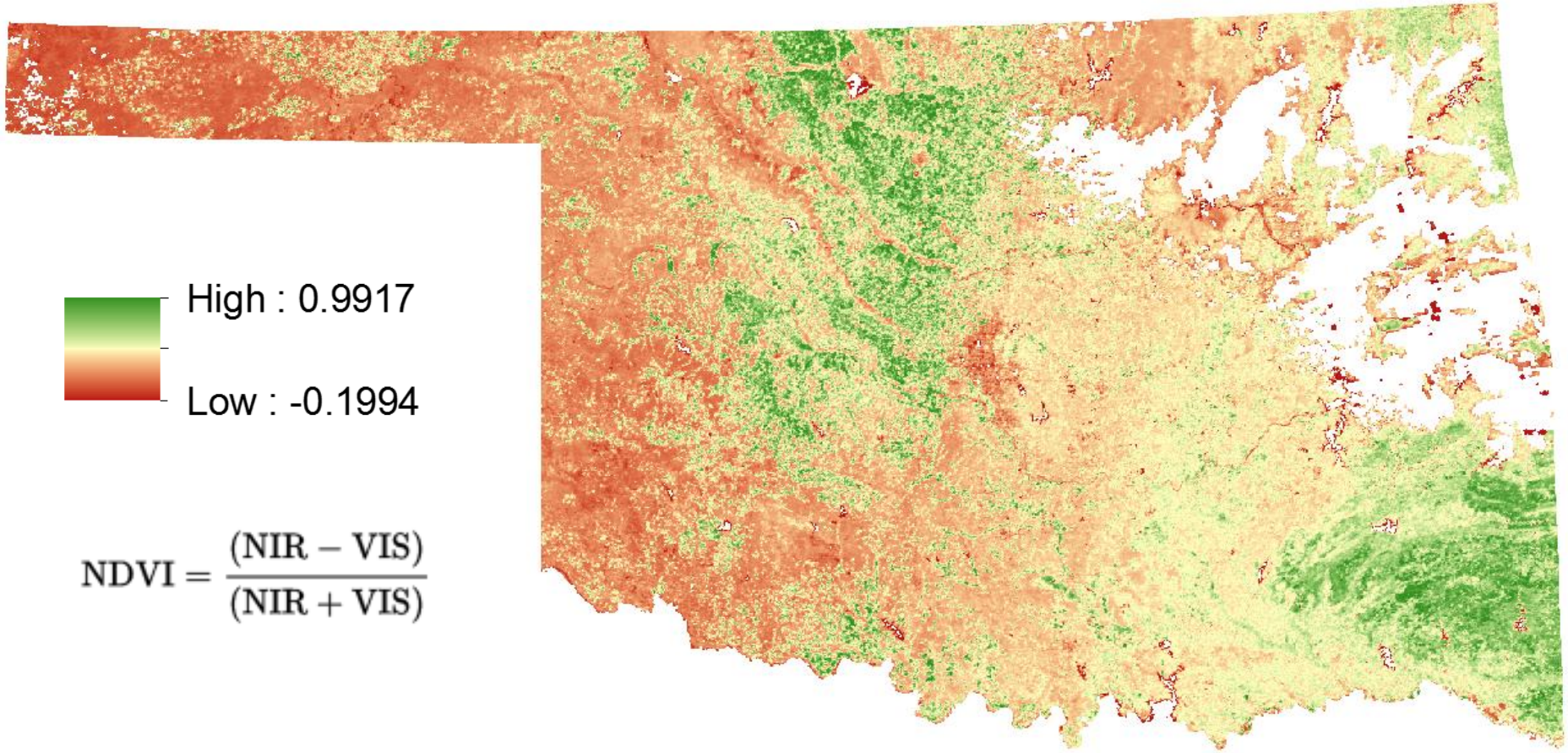
OK-FIRE
WEATHER
FIRE
SMOKE
SATELLITE
RADAR
AIR QUALITY
BURN SITE
LINKS
Stillwater
Wed 9/07/16
Weather
2:15 pm CDT
Temperature: 87°F
Heat Index: 96°F
Relative Humidity: 66%
10-m Wind: S 9 mph
24-h Rainfall: 0.00"
Dispersion: Good
Fire Danger
1:00 pm CDT
Current Fire Danger:
LOW
Burning Index: 10
Spread Component: 7
Ignition Component: 6%
NFDRS Fuel Model: T
1-hr Fuel Moisture: 10%
10-hr Fuel Moisture: 15%
KBDI: 370
Relative Greenness: 51%
Sunrise: 7:04 am
Sunset: 7:48 pm
CHOOSE A STATION
Weather-Based
Decision Support Products
for Wildland Fire Management
in Oklahoma
WeatherScope
required
The interactive features of this web site require the free WeatherScope software.
Click Here to download WeatherScope.
DOWNLOAD NOW
Fire
Prescription
Planner
OK-FIRE is a program of the Oklahoma Mesonet with leadership being provided by Oklahoma State University. Initial funding for OK-FIRE was provided from a grant from the Joint Fire Science Program (# 05-2-1-81). Copyright © 2006-2016 Board of Regents of the University of Oklahoma. All Rights Reserved. webmaster@mesonet.org
Terms of Use

Data

- eMODIS (EROS Moderate Resolution Imaging Spectroradiometer)
- NLCD (National Land Cover Database)

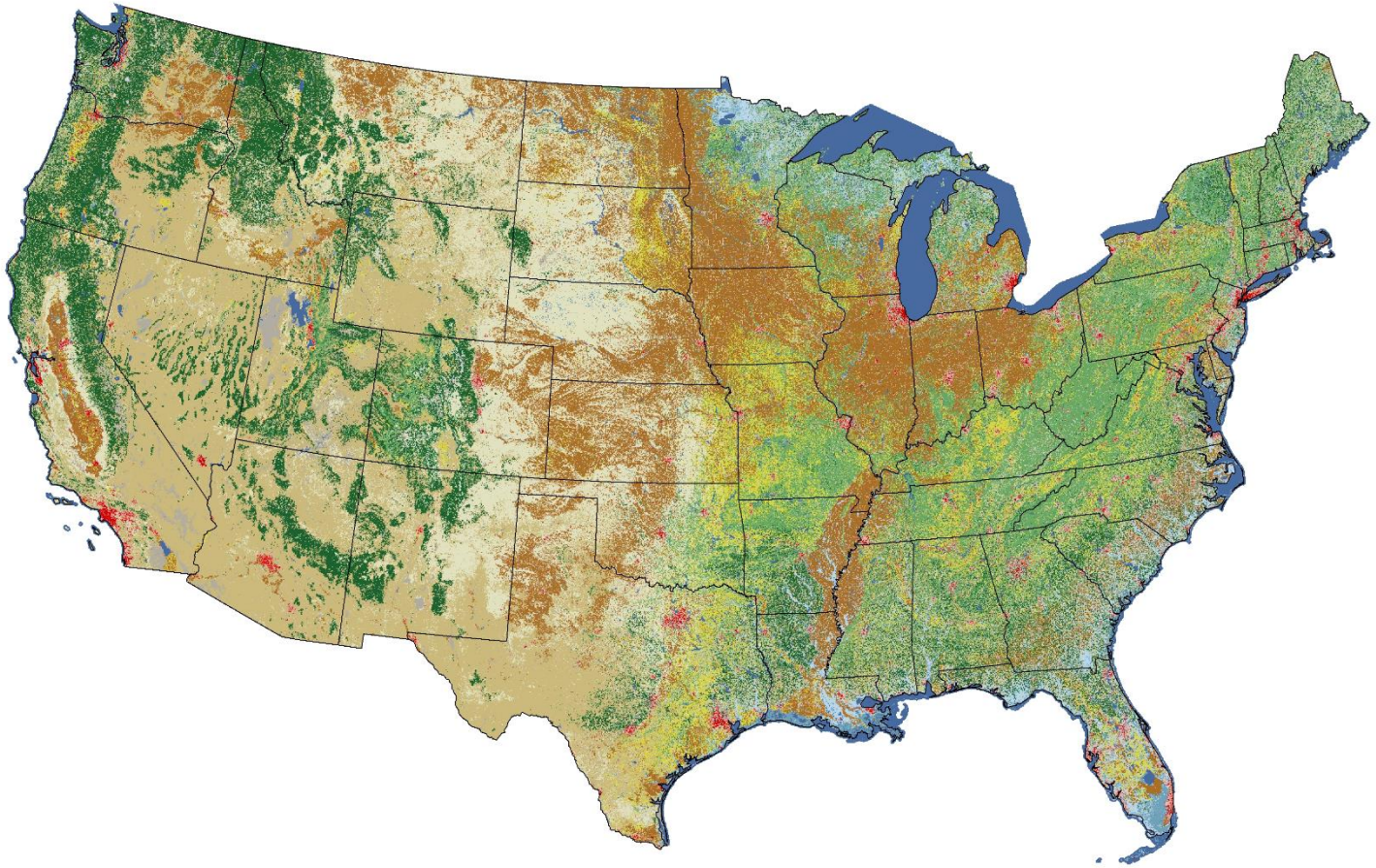
Name	Satellite	Data	Data Range	Resolution
MODIS	Terra & Aqua	NDVI	-1.0—1.0	500m
NLCD	Landsat	Land cover type	11-95	30m

MODIS NDVI(Normalized Difference Vegetation Index)



$$\mathrm{NDVI} = \frac{(\mathrm{NIR} - \mathrm{VIS})}{(\mathrm{NIR} + \mathrm{VIS})}$$

National Land Cover Database



NLCD Land Cover Classification Legend

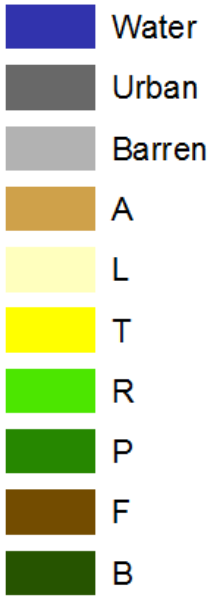
- 11 Open Water
- 12 Perennial Ice/ Snow
- 21 Developed, Open Space
- 22 Developed, Low Intensity
- 23 Developed, Medium Intensity
- 24 Developed, High Intensity
- 31 Barren Land (Rock/Sand/Clay)
- 41 Deciduous Forest
- 42 Evergreen Forest
- 43 Mixed Forest
- 51 Dwarf Scrub*
- 52 Shrub/Scrub
- 71 Grassland/Herbaceous
- 72 Sedge/Herbaceous*
- 73 Lichens*
- 74 Moss*
- 81 Pasture/Hay
- 82 Cultivated Crops
- 90 Woody Wetlands
- 95 Emergent Herbaceous Wetlands

* Alaska only

EPA NOAA USFS USGS BLM NASS NPS NASA USFWS US Army Corps

Analysis

- Reclass NLCD map into 30m fuel model map
- Use the 30m fuel model map as ancillary data to create the 500m fuel model map
  - OK Fire Fuel Models
    - Water
    - Urban
    - Barren
    - Grassy Models
      - A – Western annual grasses
      - L – Western perennial grasses
      - T – Tallgrass with brush
    - Forest Models
      - R – Hardwood forest
      - P – Southern pine forest
    - Brushy Models
      - F – Intermediate brush
      - B – Evergreen brush



Algorism

1) f(i) → >= 67%, then FM(J) = f(i)

2) WATER+URBAN+BARREN

3) L+T

IF 4) A+L+T

5) R+P

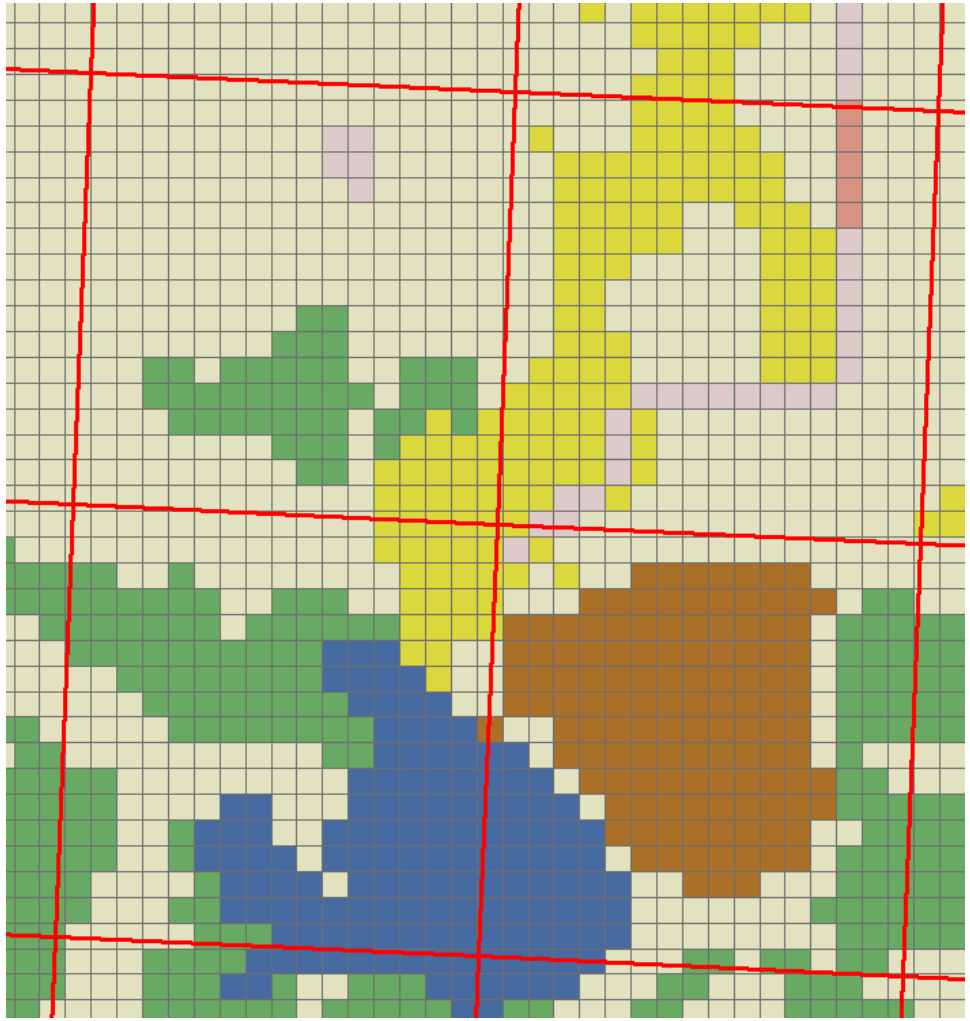
6) F+B

7) F+B >= 33% AND F+B+A+L+T

>= 67%, then FM(J) = Largest

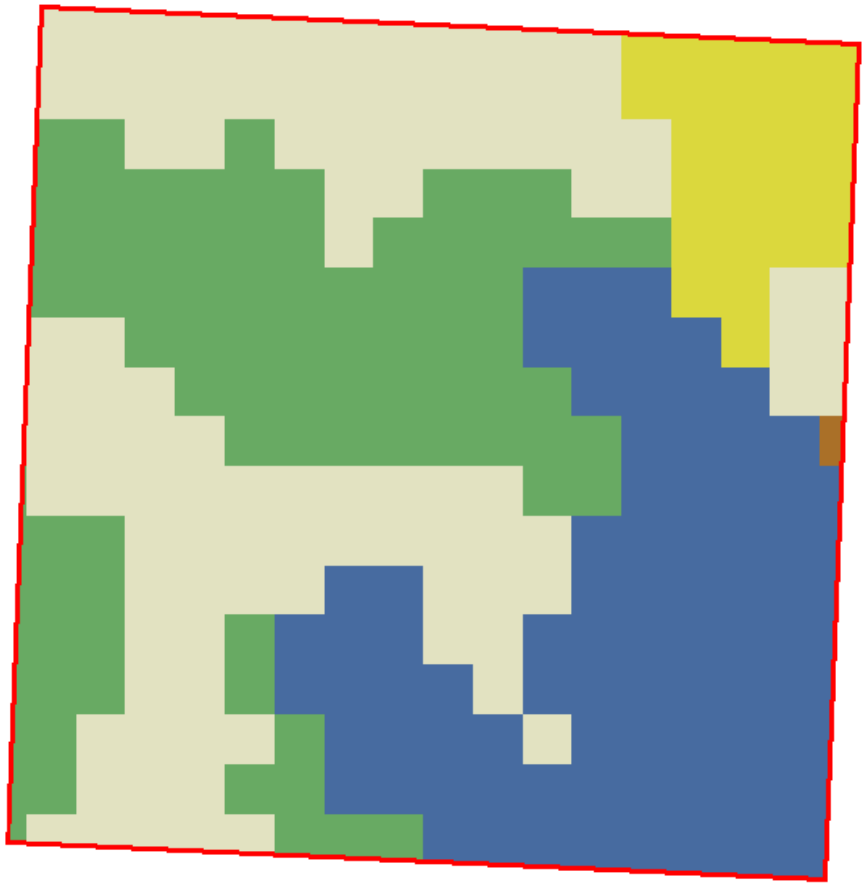
7) FM(J) =R

8) Set f(J) to largest of all f(i) in MODIS pixel



- For each 500m MODIS pixel I
  collects all 30m NLCD pixel i's

NLCD_fm	MODIS_ID	MODIS_AREA	area	%
WATER	171357	249970	73001.8	29.2
R	171357	249970	53344	21.3
R	171357	249970	1255.08	0.5
R	171357	249970	10548.2	4.2
R	171357	249970	4675.06	1.9
T	171357	249970	4143.52	1.6
T	171357	249970	615.687	0.2
T	171357	249970	82896.4	33.1
L	171357	249970	19084.9	7.6
A	171357	249970	405.586	0.2



Program Flow

- Create a list for all MODIS pixels
- For each MODIS pixel
  - create a list for the fm
  - create a list for %

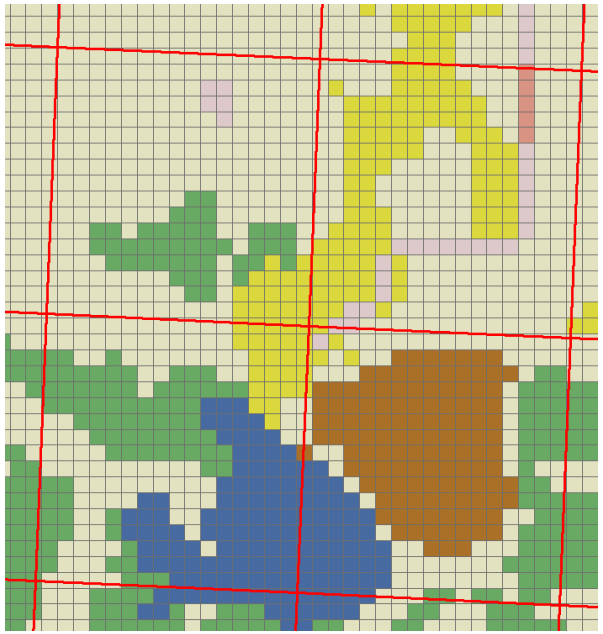
mk_order(fm, pt)

list_duplications_of(item,seq)

find_cat(p)

find_largest(in_lst)

NLCD_fm	MODIS_ID	MODIS_AREA	area	%
WATER	171357	249970	73001.8	29.2
R	171357	249970	53344	21.3
R	171357	249970	1255.08	0.5
R	171357	249970	10548.2	4.2
R	171357	249970	4675.06	1.9
T	171357	249970	4143.52	1.6
T	171357	249970	615.687	0.2
T	171357	249970	82896.4	33.1
L	171357	249970	19084.9	7.6
A	171357	249970	405.586	0.2



mk_order() function

Return the calculated fuel model value

1 For each fuel model type, find its area proportion

1.1> For non-exist fm, assign 0;

1.2> For exist fm, add up proportion for the same fm

2 Return the calculated fuel model value

```
def mk_order(fm, pt):
    lst_prop = []
    for k in lst_cat:
        if k not in fm:
            lst_prop.append(0)
        else:
            id_k = list_duplications_of(k,fm)
            k_pt = []
            for i_k in id_k:
                k_pt.append(pt[i_k])
            if sum(k_pt)>=67:
                return k
            else:
                lst_prop.append(sum(k_pt))

    return find_cat(lst_prop)
```

1.1>

1.2>

2

NLCD_fm	MODIS_ID	MODIS_AREA	area	%
WATER	171357	249970	73001.8	29.2
R	171357	249970	53344	21.3
R	171357	249970	1255.08	0.5
R	171357	249970	10548.2	4.2
R	171357	249970	4675.06	1.9
T	171357	249970	4143.52	1.6
T	171357	249970	615.687	0.2
T	171357	249970	82896.4	33.1
L	171357	249970	19084.9	7.6
A	171357	249970	405.586	0.2

Water	Urban	Barren	A	L	T	R	P	F	B
29.2	0	0	0.2	7.6	34.9	27.9	0	0	0

List_duplications_of()

1 Check if any of the existing fuel model have more than one record

2 Return a list of its index

Water	R	T	L	A
[0]	[1,2,3,4]	[5,6,7]	[8]	[9]

Index	NLCD_fm	MODIS_ID	MODIS_AREA	area	%
0	WATER	171357	249970	73001.8	29.2
1	R	171357	249970	53344	21.3
2	R	171357	249970	1255.08	0.5
3	R	171357	249970	10548.2	4.2
4	R	171357	249970	4675.06	1.9
5	T	171357	249970	4143.52	1.6
6	T	171357	249970	615.687	0.2
7	T	171357	249970	82896.4	33.1
8	L	171357	249970	19084.9	7.6
9	A	171357	249970	405.586	0.2

```
def list_duplications_of(item,seq):
    start_at = -1
    locs =[]
    while True:
        try:
            loc = seq.index(item,start_at+1)
            print"loc =",loc,"start_at+1 =", start_at+1
        except ValueError:
            break
        else:
            locs.append(loc)
            start_at = loc
    return locs
```

mk_order() function

Return the calculated fuel model value

1 For each fuel model type, find its area proportion

1.1> For non-exist fm, assign 0;

1.2> For exist fm, add up proportion for the same fm

2 Return the calculated fuel model value

NLCD_fm	MODIS_ID	MODIS_AREA	area	%
WATER	171357	249970	73001.8	29.2
R	171357	249970	53344	21.3
R	171357	249970	1255.08	0.5
R	171357	249970	10548.2	4.2
R	171357	249970	4675.06	1.9
T	171357	249970	4143.52	1.6
T	171357	249970	615.687	0.2
T	171357	249970	82896.4	33.1
L	171357	249970	19084.9	7.6
A	171357	249970	405.586	0.2

```
def mk_order(fm, pt):
    lst_prop = []
    for k in lst_cat:
        if k not in fm:
            lst_prop.append(0)
        else:
            id_k = list_duplications_of(k,fm)
            k_pt = []
            for i_k in id_k:
                k_pt.append(pt[i_k])
            if sum(k_pt)>=67:
                return k
            else:
                lst_prop.append(sum(k_pt))

    return find_cat(lst_prop)
```

1.1>

1.2>

2

Water	Urban	Barren	A	L	T	R	P	F	B
29.2	0	0	0.2	7.6	34.9	27.9	0	0	0

1) f(i) → >= 67%, then FM(J) = f(i)

2) WATER+URBAN+BARREN

3) L+T

IF 4) A+L+T

5) R+P

6) F+B

7) F+B >= 33% AND F+B+A+L+T

>= 67%, then FM(J) = Largest

7) FM(J) =R

8) Set f(J) to largest of all f(i) in MODIS pixel

find_cat()

- Evaluate the area percentage list and return the fuel model type

```
def find_cat(p_evaluate):
    if sum(p_evaluate[0:3])>=67:
        out_cat = find_largest(p_evaluate, p_evaluate[0:3])
    elif sum(p_evaluate[4:6])>= 67:
        out_cat = find_largest(p_evaluate,p_evaluate[4:6])
    elif sum(p_evaluate[3:6])>=67:
        out_cat = find_largest(p_evaluate, p_evaluate[3:6])
    elif sum(p_evaluate[6:8]) >= 67:
        out_cat = find_largest(p_evaluate, p_evaluate[6:8])
    elif sum(p_evaluate[8:10])>=67:
        out_cat = find_largest(p_evaluate, p_evaluate[8:10])
    elif sum(p_evaluate[8:10])>=33 and sum(p_evaluate[3:6])>=34:
        out_cat = 6
    else:
        out_cat = find_largest(p_evaluate)
    return out_cat
```

IF

1) f(i) → >= 67%, then FM(J) = f(i)

2) WATER+URBAN+BARREN

3) L+T

4) A+L+T

5) R+P

6) F+B

7) F+B >= 33% AND F+B+A+L+T

>= 67%, then FM(J) = Largest

7) FM(J) =R

8) Set f(J) to largest of all f(i) in MODIS pixel

Water	Urban	Barren	A	L	T	R	P	F	B
0	1	2	3	4	5	6	7	8	9

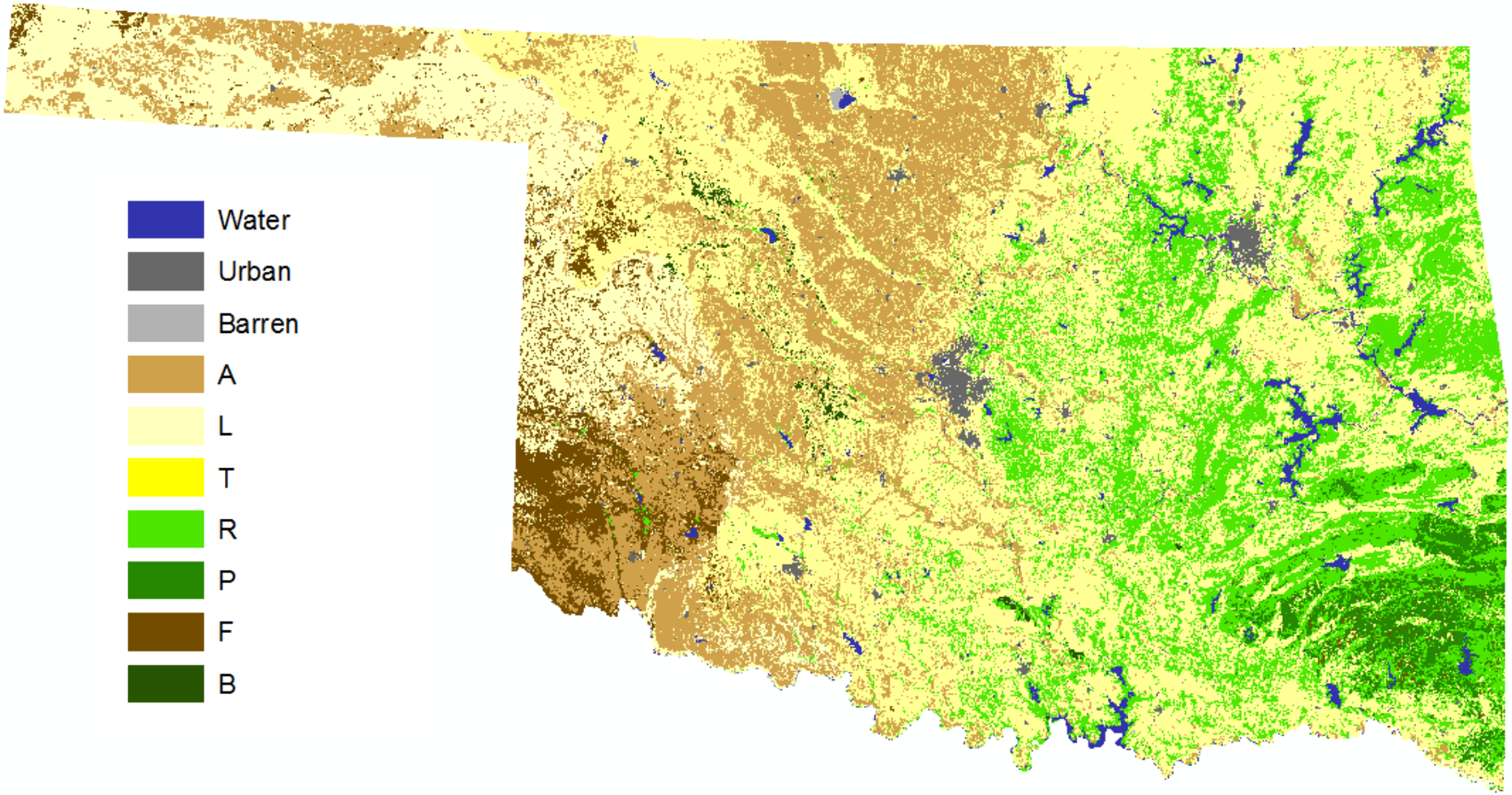
find_largest()

Find the largest area percentage in the proportion list, and use its index to find the corresponding fuel model type

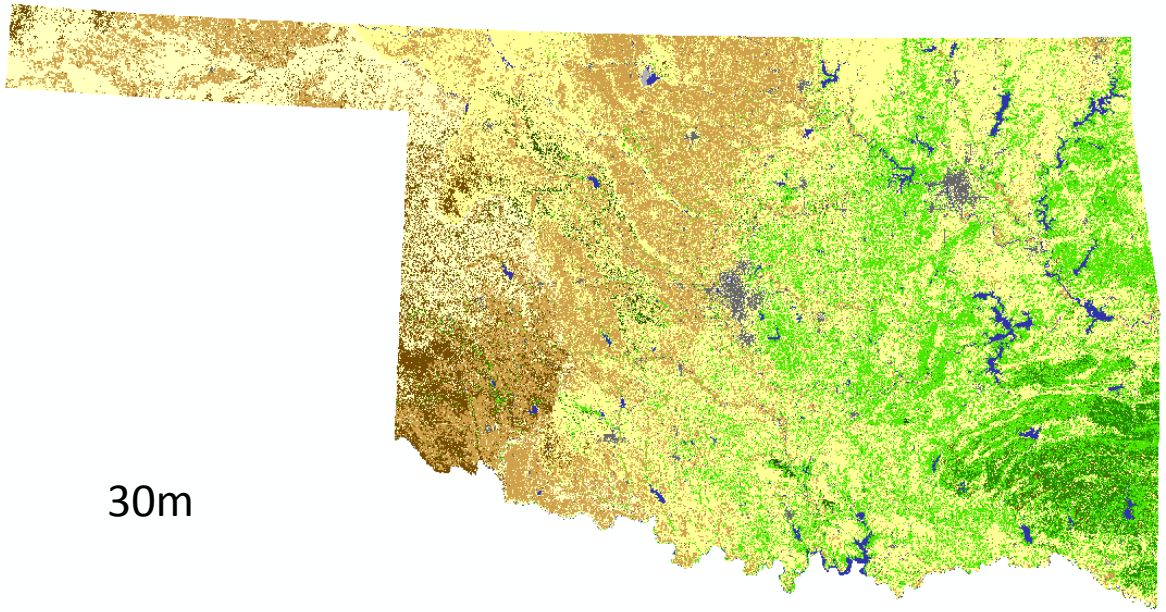
```
def find_largest(in_lst):
    out_prop = max(in_lst)
    out_id = in_lst.index(out_prop)
    return lst_cat[out_id]
```

Water	Urban	Barren	A	L	T	R	P	F	B
29.2	0	0	0.2	7.6	34.9	27.9	0	0	0

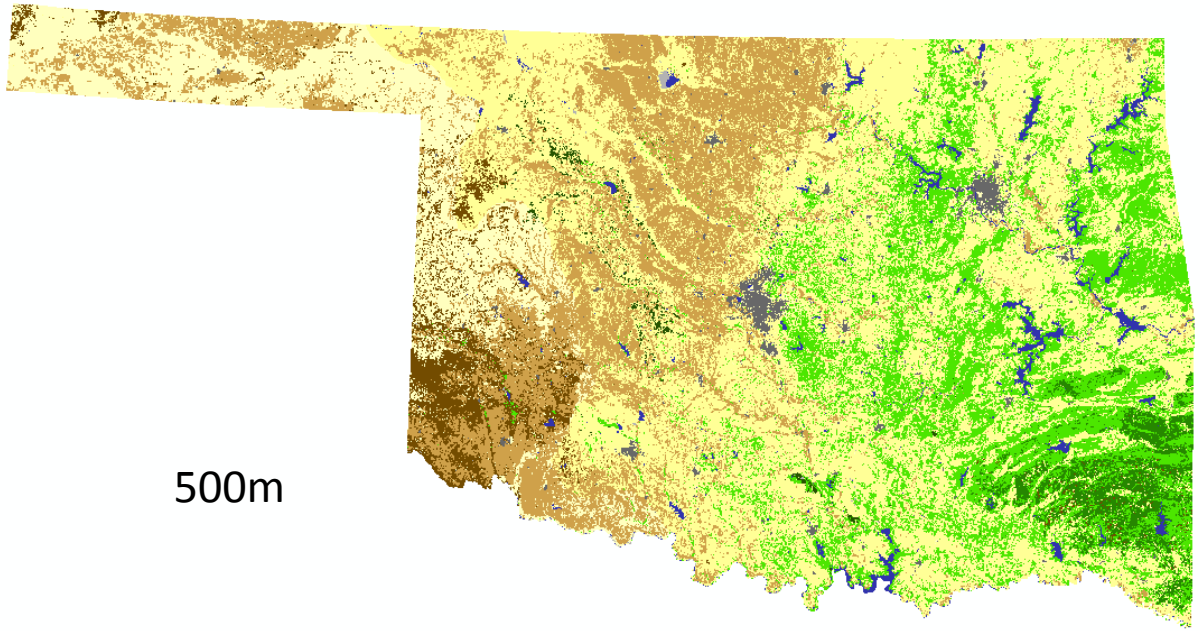
500 m fuel model map



30m vs. 500m



30m



500m

Acknowledgement

J.D. Carlson – Associate Researcher, OSU Biosystems and Agricultural Engineering

Amy Frazier – Assistant Professor, OSU Geography

Bruce Battles – former CARS Coordinator, OSU Geography