MSU Graduate Spatial Ecology Lab 3

Phoebe Zarnetske; plz@msu.edu

Sep 2015; rev. Sep 17, 2019, Sep 17, 2020

Lab 3: Patch Metrics & Spatial Design

RMarkdown version of this Lab

This lab has 2 Parts. Part 1 involves computing patch metrics. Part 2 involves applying patch metrics to design a sampling scheme. You will need to hand in a single PDF produced from your R Markdown file with code, plots, and answers to Questions 1-4, to D2L. When referring to the code below it may be more useful to use the .Rmd file linked above. You will need to be connected to the internet to complete this lab.

At the end of Lab 3 is Homework in preparation for Lab 4 (familiarizing yourself with GitHub).

Part 1: Patch Metrics

We will be using the R package, SDMTools to complete the analysis below. Note that another package, "landscapemetrics" also exists. Feel free to try it out: https://r-spatialecology.github.io/landscapemetrics/ and https://cran.r-project.org/web/packages/landscapemetrics/index.html. Both packages provide similar metrics based on Fragstats. See how they compare here:

Start working in R:

```
##### STARTING UP R
# Clear all existing data
rm(list=ls())

# Close graphics devices
graphics.off()

# Set the paths for your work
output_path<-("output")
# if this folder doesn't exist, create it
if(!dir.exists(output_path)){
  dir.create(output_path)
}

# Create the folders (directories) "data" and "lab2" - If they exist already, this
# command won't over-write them.
data_path<-(file.path("data","lab3"))
if(!dir.exists(data_path)){
  dir.create(data_path,recursive = TRUE)
}

# If you previously saved your workspace and want to load it here, do so this way
# load(file.path(output_path,"lab3.RData"))
```

```
## NOTES on R packages:
## (1) FedData should be installed from github and requires data.table package first.
#install.packages("data.table", type="binary")
#library(data.table)
#install.packages("remotes")
#remotes::install_github("ropensci/FedData")
#library(FedData)
## NOTE: If the installation for FedData doesn't work for you, I have included the NLCD data for Oregon

## (2) The packages landscapemetrics and landscapetools may require loading separately - uncomment belo
#install.packages("landscapemetrics")
#install.packages("landscapetools")
#library(landscapemetrics)
#library(landscapetools)
## If you run into errors, try this installation instead:
#devtools::install_github("r-spatialecology/landscapemetrics")
#library(landscapemetrics)

## (2) With R Markdown, it is helpful to install packages locally before knitting (copy this into your
#for (package in c("raster","sp","rgdal","maptools","rgeos")) {
#  if (!require(package, character.only=T, quietly=T)) {
#    install.packages(package)
#    library(package, character.only=T)
#  }
#}
# sometimes the above packages don't get required correctly
library(raster)
library(sp)
library(rgdal)
library(rgeos)
library(maptools)
library(landscapemetrics)
library(landscapetools)
library(FedData)
```

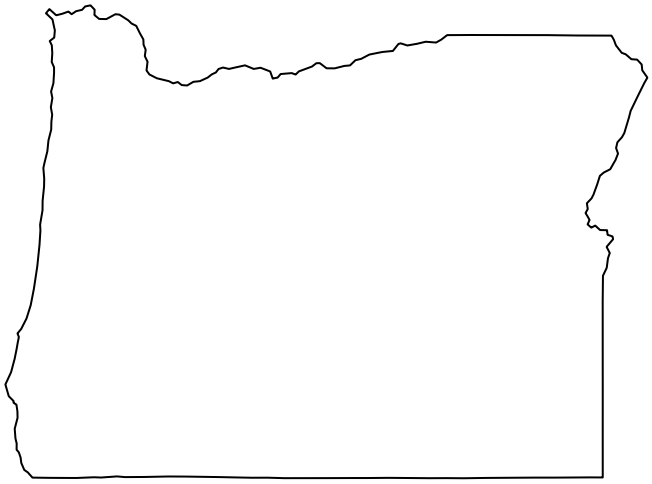
The `FedData` R package is a good way to pull in US environmental data from federal sources. See more here: https://docs.ropensci.org/FedData/. Here we are downloading the 2016 NLCD land cover product for the western half of Oregon. We want to work with NLCD data, so it's important to look up its projection. That can be found in the metadata for 2016 NLCD (navigate to https://www.mrlc.gov/data, scroll down to 2016 Land Cover for CONUS, click on the 3 dots to expand the menu and select "Metadata". Metadata contain data about data. Search for "proj" to find the projection information) https://www.mrlc.gov/downloads/sciweb1/shared/mrlc/metadata/NLCD_2016_Land_Cover_L48.xml. It's Albers Conical Equal Area, WGS84. Refer to this summary sheet (https://www.nceas.ucsb.edu/sites/default/files/2020-04/OverviewCoordinateReferenceSystems.pdf), Lab 2's links, or resources on D2L if you need a refresher on Coordinate Reference Systems.

NOTE: If the installation for FedData doesn't work for you, I have included the NLCD data for Oregon, linked below, so you can read it in directly from the Lab website.

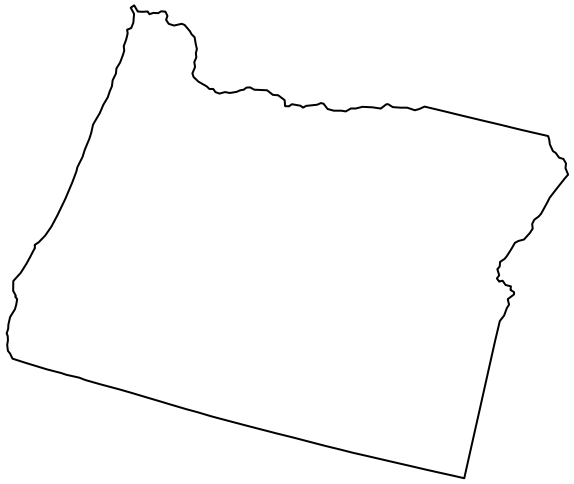
```
# Create a polygon the state of Oregon from the States of US shapefile
if(! file.exists(file.path(data_path,'uss.zip'))){
  download.file("http://www2.census.gov/geo/tiger/GENZ2014/shp/cb_2014_us_state_20m.zip", dest=file.patl
}
unzip (file.path(data_path,"uss.zip"), exdir = data_path)
```

```r
# Read in the shapefile with rgdal package - if you get an error try, library(rgdal) first
uss <- readOGR(file.path(data_path,"cb_2014_us_state_20m.shp"))

#Take a look at the shapefile info. It's a geographic coordinate system  https://epsg.io/42310.
summary(uss)
# What are the first 6 lines of uss?
head(uss)
# Plot it. See that it's showing the states and territories of the US.
plot(uss)
```

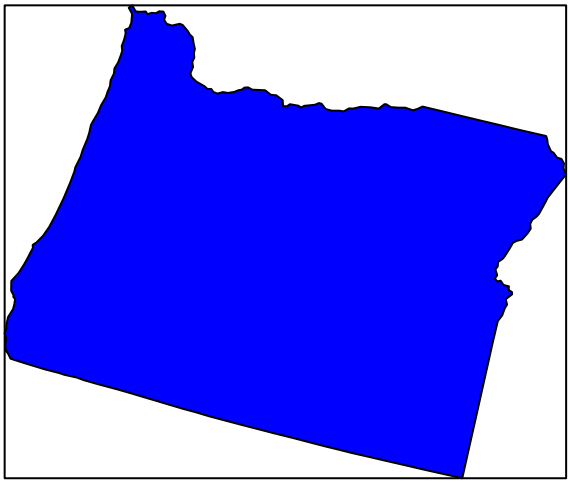
```r
# What are the unique entries in the field, NAME?
unique(uss$NAME)

# We can project these data to match the NLCD data coordinate reference system (Albers, also known as N
# Here's the CRS info for the CRS that FedData returns for NLCD (Pseudo Mercator: https://epsg.io/3857)
merc.proj <- CRS("+proj=merc +a=6378137 +b=6378137 +lat_ts=0 +lon_0=0 +x_0=0 +y_0=0 +k=1 +units=m +nadg
# Here's the CRS info for CONUS North America Albers Equal Area Conic (https://epsg.io/5070)
us.aea.proj <- CRS("+proj=aea +lat_1=29.5 +lat_2=45.5 +lat_0=23 +lon_0=-96 +x_0=0 +y_0=0 +ellps=GRS80 +

# First let's make a shapefile of just Oregon from the entire US shapefile:
or <- uss[uss$NAME =="Oregon",]

# Plot Oregon
plot(or)
```



```
# Practice projecting, and plot again
or1<-spTransform(or, us.aea.proj)
plot(or1)
```



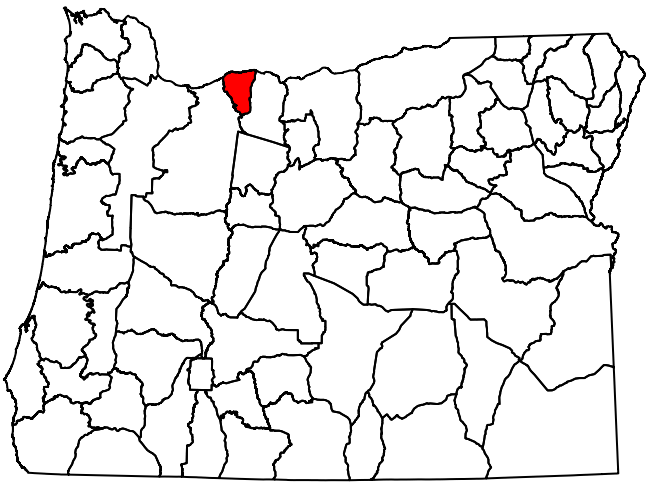
```
# Supposed you decide to use a rectangle instead of the Oregon polygon boundary.
# Refer to the extent (min and max x and y) of the shapefile, "or1"
summary(or1)
or2 <- polygon_from_extent(raster::extent(-2294351,-1584307,2301475,2899476), proj4string='+proj=aea +la

# See how the or2 outlines to the polygon, "or1"
plot(or2)
plot(or1,add=T, col="blue")
```



Depending on your computer, these polygons will still be too big to extract NLCD quickly (when I run it on my laptop it takes about 5 mins, so you could try it but it may get hung up if you are knitting to PDF). Let's try a smaller extent. If you wanted to subset out just a portion of Oregon, you could select 1 wildlife management unit. Here is a map of OR Fish & Wildlife Units: https://www.dfw.state.or.us/resources/hunting/big_game/units/bigmap.asp.

Repeat the same process as above with Oregon:

```
if(! file.exists('mgmt.zip')){
 download.file("https://nrimp.dfw.state.or.us/web%20stores/data%20libraries/files/ODFW/ODFW_805_5_wildli
}
unzip (file.path(data_path,"mgmt.zip"), exdir = data_path)
mgmt<-readOGR(file.path(data_path,"wmu24poly.shp"))

summary(mgmt) # Its CRS is lcc (Lambert Conformal Conic: https://proj.org/operations/projections/lcc.ht
lcc.proj<-CRS("+proj=lcc +lat_0=41.75 +lon_0=-120.5 +lat_1=43 +lat_2=45.5 +x_0=400000 +y_0=0 +datum=NAD8

# Project mgmt into the lcc projection from the Oregon NLCD 2011 (see below)
mgmt1<-spTransform(mgmt, lcc.proj)

head(mgmt1)
unique(mgmt1$UNIT_NAME)
# Select the HOOD Management Unit (where Mount Hood is).
hood <- mgmt1[mgmt$UNIT_NAME =="HOOD",]
## See where it's located. Plotting and rendering may take a minute because these polygons are complex.
plot(mgmt1)
plot(hood,add=T,col="red")
```



```
## Here is where you will import the Oregon NLCD directly (if the FedData package is not installing cor

## OPTION 1: Importing the Oregon 2011 NLCD data directly
## Download the 2011 Oregon NLCD (54 MB) - may take a few mins
## Download the Oregon NLCD (54 MB) - uncomment below
# if(! file.exists('or_nlcd.zip')){
# download.file("http://oe.oregonexplorer.info/ExternalContent/SpatialDataforDownload/OR_NLCD_2011.zip"
# }
# unzip (file.path(data_path,"or_nlcd.zip"), exdir = data_path)
# or_nlcd <- raster(file.path(data_path,"./OR_NLCD_2011/nlcd_or_20111" ))
# save.image("lab3.RData")
#
# # Take a look at or_nlcd from 2011 - it is projected in lcc
# or_nlcd
#
# # Crop or_nlcd by hood1
# hood_nlcd<-crop(or_nlcd, extent(hood))
# hood_nlcd <- mask(x = hood_nlcd, mask = hood)

# plot(hood_nlcd)
# plot(hood,add=TRUE, lwd=2, border='white')

## Uncomment to output a raster as a tif
?writeRaster
#writeRaster(raster, "raster.tif")
```

```
## OPTION 2 (uncomment below): Use the get_nlcd command from FedData to extract the NLCD data for this
# ?get_nlcd
#
# hood_nlcd<-get_nlcd(
#   template = hood,
#   label="Hood",
#   year = 2016,
#   dataset = "Land_Cover",
#   extraction.dir = paste0(data_path, "/FedData/"),
# )
# hood_nlcd

# plot(hood_nlcd)
# plot(hood,add=TRUE, lwd=2, border='white')
```

Now work with the `landscapemetrics` R package to compute different patch and landscape statistics.

```
## Make a plot of the Hood NLCD data - this renders a map of the NLCD raster. It may take a few mins to

#show_landscape(hood_nlcd)
#check_landscape(hood_nlcd)
```

You could proceed to calculate metrics for all different land cover classes (this would take a while). Instead, suppose you are interested in the metrics of "forested" patches in this landscape. First, reclassify into "forest" and "nonforest". You need to determine what the cover types are:

```
#unique(hood_nlcd)
## Look here at the "Class/Value" to figure out which cover type(s) you want: https://www.mrlc.gov/data
#hood_for <- hood_nlcd
#hood_for[hood_for < 41] <- 0 # make everything below 41 zero
#plot(hood_for)
#hood_for[hood_for > 43] <- 0 # make everything above 43 zero
#plot(hood_for)
#unique(hood_for)
## Assign 1 value (1) to forest.
#hood_for[hood_for == 41 | hood_for == 42 | hood_for == 43] <- 1
#show_landscape(hood_for)

## The code below follows part of this vignette: https://r-spatialecology.github.io/landscapemetrics/ar

## Check to see if the input data is suitable for computing patch metrics
#check_landscape(hood_for)

## Look at the distinct patches for this landscape
#show_patches(hood_for)

## Show patches of each class (we have 2 classes: 0 for non-forest and 1 for forest)
#show_patches(hood_for, class = "all", labels = FALSE)

## At the patch level, compute patch metrics
#hood_metrics <- calculate_lsm(hood_for, what = "patch")

## Look at the different types of metrics at the patch level. Note that you can compute these metrics a
#list_lsm(level='patch')
```

```
## Determine if any patch metrics are correlated
#show_correlation(hood_metrics, method = "pearson")
```

###QUESTION 1: How do the 12 patch metrics from `landscapemetrics` help you understand how an organism interacts with its environment? For 3 metrics of your choice, explain in 1-2 sentences what the metric tells us about an organism's interaction with its environment. Feel free to use examples. Refer to the *list_lsm()* results above. You may also want to refer to the FRAGSTATS documentation for more in-depth coverage (warning: this is a large document, I suggest searching for the patch metric term. Page 77 contains the start of the "metrics" discussion, but page 92 contains a condensed table): https://www.umass.edu/landeco/research/fragstats/documents/fragstats_documents.html

###QUESTION 2: Repeat the process above, but select a different management unit than Hood. Compare your other management unit patch metrics with the ones for Hood Management Area. Briefly, how do the patch metrics for Forested areas and non-Forested areas compare between the two management areas? Optional: make some plots showing the statistical distributions of patch metrics across the two areas (e.g., a histogram of patch area, by class, by management unit).

Part 2: Study Design

You decide to study the interactions between Spotted Owls and Barred Owls in Oregon forests (do a little internet searching to understand why studying these 2 species together is important, and what their habitat requirements are).

###QUESTION 3. Use GBIF data to generate a map showing "recent" point locations of each of these species (as different symbols and/or different colors), overlain on top of the 2 forest rasters you created above. Justify the timeframe you chose for the occurrences. Make 2 maps, one for each management unit with the points overlaid on top of the forest raster. Include a legend, scalebar (with units), and north arrow for your maps. You may want to reference Lab 2 for additional code.

###QUESTION 4. If you had unlimited resources, describe how you would sample the 2 management units to obtain new occurrence data next year for both species. Are there specific forest attributes that would be helpful to know in addition to just "forest vs. non-forest"? Use terminology we covered in Lecture and Discussion about sampling.

Some other potentially helpful code below:

`gbif`: Extract out just points from Oregon (or some extent): http://www.inside-r.org/packages/cran/dismo/docs/rgbif. There are 2 ways of extracting out the points:

The commands, `drawExtent` and `extent` are both ways to specify the extent of data in `gbif()`

`drawExtent` is in the raster package: http://www.inside-r.org/packages/cran/raster/docs/drawExtent. You have to have plotted a map first (and then click on it). This won't be possible if you're running a script via job submission on HPCC. You can also use `extent`, also in the raster package, to define boundaries: http://www.inside-r.org/packages/cran/raster/docs/extent

Homework in preparation for Lab 4:

In Lab 4 we will introduce GitHub, a versioning program helpful for collaborating and tracking changes in code. You can think of it as a Google Docs version for tracking code changes (you can also use it with text). If you are familiar with GitHub, no need to go over the following, but if you are not familiar, please work through the following tutorials [personally, I use the Desktop version but any version will work]:

At a minimum, please complete this first short tutorial ahead of Lab 4: 1. Command line version: https://guides.github.com/activities/hello-world/

Other versions: 2. Desktop version: https://help.github.com/desktop/guides/getting-started-with-github-desktop/

3. via RStudio: http://happygitwithr.com/rstudio-git-github.html and/or https://support.rstudio.com/hc/en-us/articles/200532077-Version-Control-with-Git-and-SVN