Manual-Applied-Spatial-Ecology

W. David Walter

Invalid Date

Table of contents

Chapter 1

Manual of Applied Spatial Ecology

New version of manual after conversion to terra and sf packages"

Chapter 2

Preface

The purpose of this manual is to assist researchers on methods for data management and analysis using the R environment or other software after data has been collected in the field. The impetus behind this manual was from many years of frustration in trying to analyze data in R using code and forgetting how it was done upon completion of a study. We wanted to find a way to avoid needing to search computers for folders to find old R code then try to remember what we did to the data to get the code to run properly. Over the years, advancements in data handling and manipulation, GIS capabilities, and methods of estimators for home range, movements, resource selection, and spatial epidemiology have occurred within the R environment. Program R is free and used by researchers world-wide but R also provides a platform to create and display spatial layers without the need for the variety of GUI software, free or otherwise. Furthermore, analyzing spatial data in R enables statistical analysis of data without the errors that may arise from bringing data from spreadsheet or GIS software to statistical programs. We would like to stress that this manual is not the authority on all topics presented herein. Our goal was to create an online manual that could be easily followed by researchers, biologists, or graduate students to analyze their data in R. Although the user would benefit from general introductory knowledge of using R and ArcMap, most of the manual is for mid-level users of R that need guidance beyond the basics of introductory R and GIS coureses. We also provide numerous citations throughout each section should the user choose to learn the theory or more details behind each topic. In addition, this manual provides a handy outline of course materials for an Applied Spatial Ecology course that will surely expand or change as the field evolves. As time permits and errors are brought to our attention, we plan to update and correct problems so be sure to send any corrections or comments our way.

Any use of trade, firm, or product names is for descriptive purposes only and does not imply endorsement by the U.S. Government.

Recommended citation:

Walter, W.D. Manual of Applied Spatial Ecology. Walter Applied Spatial Ecology Lab, Pennsylvania State University, University Park. Access Date. http://ecosystems.psu.edu/research/labs/walter-lab.

Acknowledgments

Numerous colleagues have provide assistance with R packages they created or with code they have provided in some other form. We would be remiss if we failed to thank these colleagues for their hard work:

Sharon Baruch-Mordo, The Nature Conservancy, Fort Collins, CO 80524 ; sbaruch-mordo@tnc.org Simon Benhamou, Centre d'Ecologie Fonctionnelle et Evolutive, France Clément Calenge, Data Analysis Support Unit, Directorate for Studies and Research, National Office of Hunting and Wild Fauna, Saint Benoist - 78610 Auffargis, France Mevin Hooten, Colorado Cooperative Fish and Wildlife Research Unit, 201 Wagar Bldg, Colorado State University, Fort Collins, CO 8052 ; Mevin.Hooten@colostate.edu Bill Kanapaux, Pennsylvania Cooperative Fish and Wildlife Research Unit, 406 Forest Resources Bldg., Pennyslvania State University, University Park, PA 16802 ; wjk15@psu.edu Bart Kranstauber, Max Planck Institute for Ornithology, Eberhard-Gwinner-Str., 82319 Seewiesen; bart.kranstauber@uni-konstanz.de Ryan Nielsen, West Inc.,415 W. 17th St. Suite 200, Cheyenne, WY 82001 ; rnielson@westinc. com Glen Sargeant, Northern Prairie Wildlife Research Center, Jamestown, North Dakota; glen_sargeant@usgs.gov Peter Singleton, USDA Forest Service, Pacific Northwest Research Station in Wenatchee, WA; singlep@u.washington.edu Marcó Smolla, Max Planck Institute for Ornithology, Dept. Migration and Immuno-ecology, Am Obstberg 1, 78315 Radolfzell, Germany; msmolla@orn.mpg.de Tyler Wagner, US Geological Survey, Pennsylvania Cooperative Fish and Wildlife Research Unit, 402 Forest Resources Bldg., Pennsylvania State University, University Park, PA 16802 ; twagner@psu.edu

Chapter 3

Introduction

The original Manual of Applied Spatial Ecology (hereafter referred to as *Manual*) went online in 2014 with a 173 page manual created in Latex and scripts/datasets available on GitHub a short time thereafter. Since the scripts were .R files, they were then converted into .Rmd files in attempts to create stand-alone pdfs for each chapter. Latex of the entire compilation was too time consuming to maintain so .Rmds for each exercise were an early solution. The resulting chapter pdfs were created with every update and version of each exercise pushed to GitHub to easily update code or package changes. Although the manual pdf was not edited or changed since 2016, the pdfs within each exercise on GitHub were updated each year.

Fast forward to 2023, I was challenged with whether it was even worth maintaining this *Manual* considering so many R packages and code were required to be published with most manuscripts. In addition, with the increase in spatial data available and R code to prepare spatial data (some within my own lab), several chapters in the *Manual* seem outdated with no reason to maintain these chapters (Chapter 2 for example). Combine that with deprecating of packages rgdal and raster for their equivalents, sf and terra, respectively, most exercises in the *Manual* would no longer be functional after December 2023.

The impetus for creating this manual in 2014, however, was a place to store code and revisit functions and code I created or found online in one place to use on new projects. GitHub provides a nice portal to store and organize code that might not be published and for my exercises to be available for graduate students in my course at The Pennsylvania State University - Applied Spatial Ecology in R. Even though I have countless scripts and functions that are not available in the *Manual*, maintaining R code (GitHub) and a pdf manual (bookdown or similar) has never been easier. With this in mind, the link on my lab webpage will stay as an archive as long as the university allows it, but the new manual and all updates will move to my GitHub page walterASEL under a new repository.

Part I

Data Manipulation and Management

Chapter 4

Transformations between coordinate systems

Transformations in ArcMap can be the most troublesome component of spatial analysis that is often overlooked as the reason for errors in data analysis. We will briefly go into the 2 most common problems requiring our assistance from collaborators and potential solutions.

1. What coordinate system were the data collected in?

It seems that every GPS collar, handheld GPS unit, GIS landcover layer, etc. has been created using a different coordinate system and it's not the one you have at your study site. Or perhaps NAD 1927 was used and you decided to use an updated reference system and want to use NAD 1983. Regardless, the coordinate systems must match even though ArcMap often overlays them with "on the fly projections". The "on the fly" component of ArcMap is great for visualization but not for spatial ecologists that need data analysis. We often can determine which coordinate system the data were created in using the metadata to define a coordinate system or project the data into a coordinate system for data analysis.

2. A Toolbox in ArcMap may not extract data or clip data properly

As mentioned previously, data collected with a GPS collar or handheld GPS may not be in the same geographic or projected coordinate system as the GIS layers you download or receive from collaborators (i.e., Digital Elevation Data, National Land Cover Data). As you attempt to use a Toolbox function, such as clipping National Land Cover Data within the extent of your GPS locations, an error may result.

We will now explore some transformations of data in R to help understand what Projections and Transformations are all about. The dataset that follows is for

a project in Colorado with mule deer equipped with GPS collars that collected locations every 3 hours. The purpose of the study was to determine mule deer use of agricultural crops, sunflowers in this case, in response to years of damage complaints from farmers. We will use this subset of dataset in later exercises as well.

1. Open the script "MDprojections.Rmd" and run code directly from the script

2. First we need to load the packages needed for the exercise

```
library(sf)
```

3. Now let's have a separate section of code to include projection information we will use throughout the exercise. In previous versions, these lines of code were within each block of code

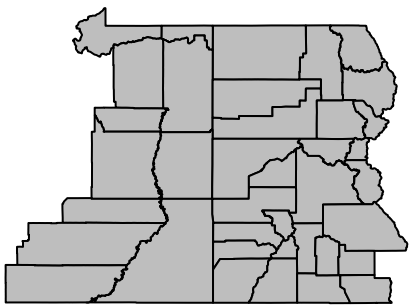
```
ll.crs <- st_crs(4269)
utm.crs <- st_crs(26912)
albers.crs <- st_crs(5070)
```

4. Read in some shapefiles of the study area

```
study.states<-st_read("data/MDcounties.shp")
```

```
Reading layer `MDcounties' from data source
  `/Users/davidwalter/Library/CloudStorage/OneDrive-ThePennsylvaniaStateUniversity/Walt
  using driver `ESRI Shapefile'
Simple feature collection with 38 features and 9 fields
Geometry type: POLYGON
Dimension:     XY
Bounding box:  xmin: -112.9059 ymin: 36.99228 xmax: -105.4277 ymax: 41.25376
Geodetic CRS:  WGS 84
```

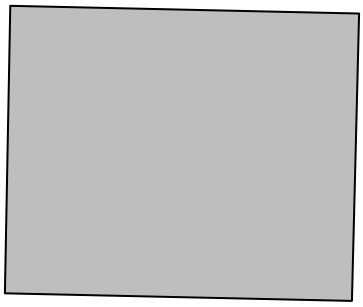
```
plot(st_geometry(study.states), col="grey")
```



```
#Let's zoom into the region we have locations instead of county level
study.zoom<- st_read("data/MDzoom.shp")
```

```
Reading layer `MDzoom' from data source
  `/Users/davidwalter/Library/CloudStorage/OneDrive-ThePennsylvaniaStateUniversity/WalterRproject
  using driver `ESRI Shapefile'
Simple feature collection with 1 feature and 1 field
Geometry type: POLYGON
Dimension:     XY
Bounding box:  xmin: -109.2433 ymin: 37.56257 xmax: -108.6488 ymax: 37.95425
Geodetic CRS:  WGS 84
```

```
plot(st_geometry(study.zoom), col="grey")
```

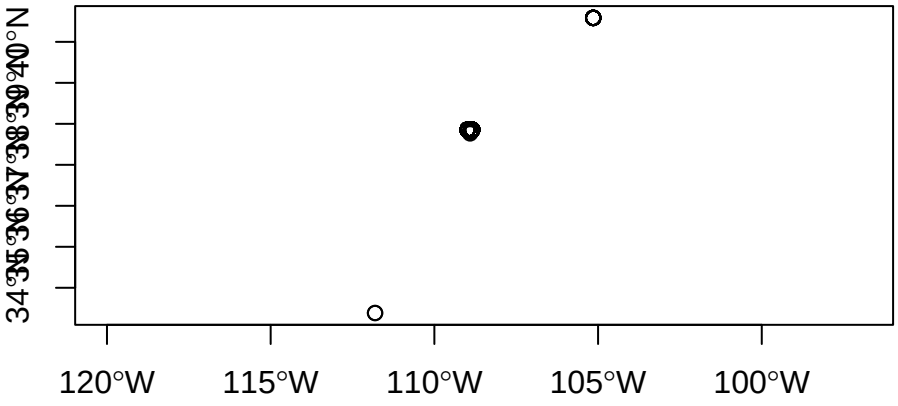


5. Import the csv file that contains all the mule deer locations by ID

```
muleys <-read.csv("data/muleysexample.csv",header=T)
```

6. Create an sf object of raw mule deer locations with projection defined similar to study site shapefile (i.e., WGS84) then remove obvious outliers using st_crop

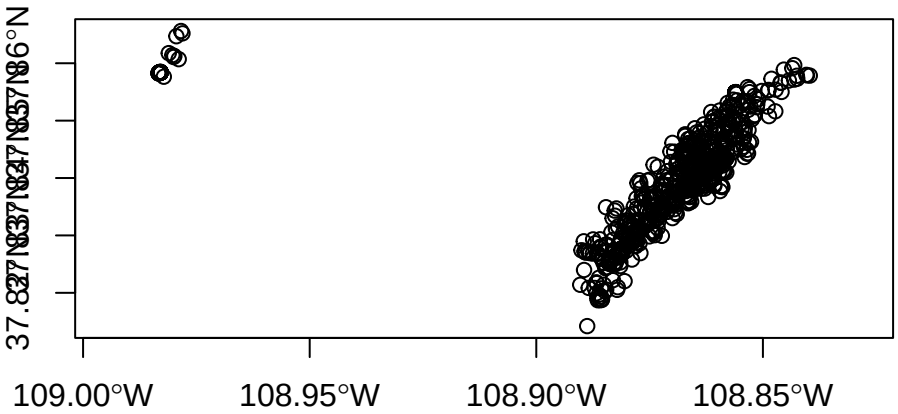
```
coords <- st_as_sf(muleys, coords = c("Long", "Lat"), crs = ll.crs)
plot(st_geometry(coords),axes=T)
```




```
deer.spdf <- st_crop(coords, xmin=-107.0,xmax=-110.5,ymin=37.8,ymax=39.0)
```

```
Warning: attribute variables are assumed to be spatially constant throughout
all geometries
```

```
plot(st_geometry(deer.spdf),axes=T)
```



7. Now let's project both the mule deer locations and study site shapefile to NAD83 UTM Zone 12 (Fig. 1.4, 1.5)

```
#projection for mule deer locations
deerUTM12 <-st_transform(deer.spdf, st_crs(utm.crs))
plot(st_geometry(deerUTM12), axes=T)
class(deerUTM12)
st_crs(deerUTM12)
```

```
#See new projected coordinates in UTM 12N for the first 5 locations compared to similar line o
st_coordinates(deerUTM12[1:5,])
```

Chapter 5

Import and Format Datasets

1. Open the script "TimeLagScript.Rmd" and run code directly from the script

2. First we need to load the packages needed for the exercise

```
library(chron)
library(sf)
#library(RAtmosphere)#This package has not been updated for newer versions of R
```

3. Now let's have a separate section of code to include projection information we will use throughout the exercise. In previous versions, these lines of code were within each block of code

```
ll.crs<-"+proj=longlat +datum=WGS84 +no_defs +ellps=WGS84 +towgs84=0,0,0"
utm.crs <-"+proj=utm +zone=12 +datum=WGS84"
ll.crs=st_crs(4269)
utm.crs <- st_crs(9001)
albers.crs <- st_crs(5070)
```

4. Determine the name of your file ("temp" in our case here).

```
temp <- read.csv("data/Y2005_UTM_date.csv", header=T)
```

5. It is often necessary to determine the time lag between successive locations within your dataset

```
# Modify time to include seconds
temp$time <- paste(as.character(temp$LMT_TIME),"00",sep=":")

# Convert to chron date
temp$date_time <- chron(as.character(temp$LMT_DATE),temp$time,format=c(dates="m/d/y'

# Calculate difference in time in minutes
timediff <- diff(temp$date_time)*24*60
summary(timediff)
```

```
   Min.    1st Qu.     Median       Mean    3rd Qu.       Max.
7.00000   30.00000   30.00000   72.01485   60.00000 7859.00000
```

```
# Remove first entry without any difference
temp <- temp[-1,]

# Assign timediff column to original "temp" dataset for use later
temp$timediff <- as.numeric(timediff)
```

6. We can then either export this file as an excel file for use in other programs or use it in R in subsequent analysis

```
#write.csv(temp, "TimeDiffdata.csv", row.names=TRUE,sep=" ", col.names=TRUE, quote=T
```

7. Next we can add code below to include night and day into datasets and to also to account for daylight savings. Package RAtmosphere will eliminate the need for the chunk of code below if working on an earlier version of R (i.e., code not available for R 3.2.1 plus). Thanks to Duane Diefenbach, PA Coop Unit Leader, for compiling all this from online sources.

We may first need to create a SPDF and transform to Lat Long then return to a Data Frame You only need this section of code if you need to acquire Lat Long coordinates for dataset

```
#ll.crs<-"+proj=longlat +datum=WGS84 +no_defs +ellps=WGS84 +towgs84=0,0,0"
utm.crs <-"+proj=utm +zone=12 +datum=WGS84"
dataspdf <- st_as_sf(temp, coords = c("UTMe", "UTMn"), crs=utm.crs)

datall <-st_transform(dataspdf, ll.crs)
temp <- data.frame(datall)
```

8. Separate times into categories “Day” and “Night” based on sunrise-sunset table by running function below or simply using the RAtmosphere package

9. First run line of code below with d being the day of year, Lat is latitude in decimal degrees, and Long is longitude in decimal degrees (negative ==

West) available at suncalc suncalc This method is copied from: Teets, D.A. 2003. Predicting sunrise and sunset times. The College Mathematics Journal 34(4):317-321.

At the default location the estimates of sunrise and sunset are within seven minutes of the correct times (http://aa.usno.navy.mil/data/docs/RS_OneYear.php) with a mean of 2.4 minutes error.

NOTE: Function is in package RAtmosphere but does not work in newer version of R along with other issues!

```
suncalc <- function(d,Lat=39.14133,Long=-106.7722){

  ## Function to convert degrees to radians
  rad <- function(x)pi*x/180

  ##Radius of the earth (km)
  R=6378

  ##Radians between the xy-plane and the ecliptic plane
  epsilon=rad(23.45)

  ##Convert observer's latitude to radians
  L=rad(Lat)

  ## Calculate offset of sunrise based on longitude (min)
  ## If Long is negative, then the mod represents degrees West of
  ## a standard time meridian, so timing of sunrise and sunset should
  ## be made later.
  ##NOTE: If working with UTC times use timezone = -4*(abs(Long)%%15)*sign(Long)
  timezone = -7*(abs(Long)%%15)*sign(Long)

  ## The earth's mean distance from the sun (km)
  r = 149598000

  theta = 2*pi/365.25*(d-80)

  z.s = r*sin(theta)*sin(epsilon)
  r.p = sqrt(r^2-z.s^2)

  t0 = 1440/(2*pi)*acos((R-z.s*sin(L))/(r.p*cos(L)))

  ##a kludge adjustment for the radius of the sun
  that = t0+5

  ## Adjust "noon" for the fact that the earth's orbit is not circular:
```

```
  n = 720-10*sin(4*pi*(d-80)/365.25)+8*sin(2*pi*d/365.25)

  ## now sunrise and after sunset are:
  sunrise = (n-that+timezone)/60
  sunset = (n+that+timezone)/60
  suntime <- cbind(sunrise,sunset)

  return(suntime)
}
```

11. Read in location data and retain lat, lon, and date

```
temp$Date <- paste((temp$Year),substr(temp$date_time, 2,3),substr(temp$date_time, 5,

#calculate calendar day and center of locations
calday <- as.numeric(as.Date(temp$Date)-as.Date("2005-01-01"), units="days")
dat1 <- cbind(temp,calday)
moda <- format(as.Date(temp$Date),"%d-%b")

dat1 <- cbind(dat1, suncalc(dat1$calday, Lat=dat1$LATITUDE, Long=dat1$LONGITUDE),mod
hrchar <- as.character(substr(dat1$time,1,2))
hr <- as.numeric(as.character(substr(dat1$time,1,2)))
minchar <- as.character(substr(dat1$time,4,5))
min <- as.numeric(minchar)
localhr <- hr+min/60
dat1 <- cbind(dat1,hr,hrchar,minchar,localhr)
Diel <- ifelse(localhr<dat1$sunrise | localhr>dat1$sunset, 'Night', 'Day')
dat1 <- cbind(dat1,Diel)
```

```
dat1[15:50,]
```

Chapter 6

Manipulate Polygon Layer

1. Open the script "SoilScript.Rmd" and run code directly from the script

2. First we need to load the packages needed for the exercise

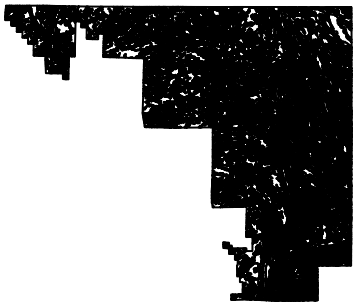
```
library(sf)
library(terra)
```

3. Import shapefile of soils dataset

```
soils<-st_read("data/Soil_Properties.shp")
```

```
Reading layer `Soil_Properties' from data source
  `/Users/davidwalter/Library/CloudStorage/OneDrive-ThePennsylvaniaStateUniversity/WalterRproject
  using driver `ESRI Shapefile'
Simple feature collection with 12812 features and 11 fields
Geometry type: POLYGON
Dimension:     XY
Bounding box:  xmin: 408283 ymin: 4456763 xmax: 504816.1 ymax: 4538989
Projected CRS: NAD83 / UTM zone 13N
```

```
plot(st_geometry(soils))
```



```
names(soils) #get attribute data
```

```
 [1] "Join_Count" "TARGET_FID" "Join_Cou_1" "TARGET_F_1" "AREASYMBOL"
 [6] "SPATIALVER" "MUSYM"      "MUKEY"      "SdvOutpu_1" "SdvOutpu_2"
[11] "SdvOutpu_3" "geometry"
```

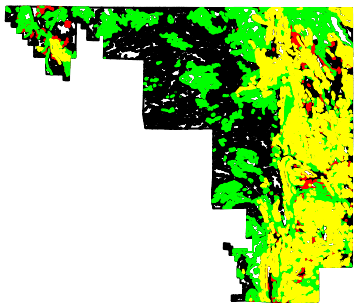
```
#Rename original category headings to something more familiar
soils$Clay <- soils$SdvOutpu_1
soils$pH <- soils$SdvOutpu_2
soils$CEC <- soils$SdvOutpu_3
```

4.Shapefiles contain several features associated with each polygons that collectively make up the entire layer so let's explore

```
soils #a data frame with geometry type "polygon" associated with 10 features
st_bbox(soils) #boundary box
st_crs(soils) #projection information
soils[1, ] #will bring up data associated with the first polygon
```

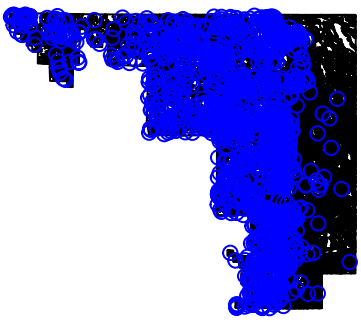
5. Select portions of the data that fit some set criteria

```
#Highlights the areas that Percent Clay polygons are over 30%
plot(st_geometry(soils))
high.clay <- soils[soils$Clay>30,]
plot(st_geometry(high.clay), border="red", add=TRUE)

##Highlights the areas that Cation Exchange Capacity is greater than 14
high.CEC<- soils[soils$CEC>14,]
plot(st_geometry(high.CEC), border="green", add=TRUE)

##Highlights the areas that soil pH is greater than 8
high.pH <- soils[soils$pH>8,]
plot(st_geometry(high.pH), border="yellow", add=TRUE)
```



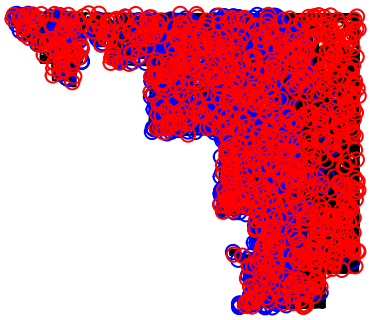
6. Bring in locations of harvested mule deer

```
#Import mule deer locations from harvested animals tested for CWD
#Note: Locations have been offset or altered so do not reflect actual locations of samples
mule <- read.csv("data/MDjitterclip.csv", header=T)
crs<-"+proj=utm +zone=13 +datum=WGS84 +no_defs +towgs84=0,0,0"
coords <- st_as_sf(mule, coords = c("x", "y"), crs = st_crs(soils))

plot(st_geometry(soils))
plot(st_geometry(coords), col="blue",add=T)
```



7. Let's generate random points with the extent of the soil layer

```
#Sampling points in a Spatial Object "type=regular" will give a regular grid
samples<-st_sample(soils, 1000, type="random")

plot(st_geometry(soils), col="wheat")
plot(st_geometry(coords), col="blue", add=T)
plot(st_geometry(samples), col="red", add=T)
```



8. Extract and tally Clay soil types for random samples and mule deer locations

```
#Matches points with polygons:
soils.idx<- st_intersects(samples,soils)
soil.samples <- soils[unlist(soils.idx), "Clay", drop = TRUE] # drop geometry
# locs <- SpatialPoints(coords)
# locs@proj4string <- soils@proj4string
soils.locations<- st_intersects(coords, soils)
soils.locs <- soils[unlist(soils.locations), "Clay", drop = TRUE]
#Tally clay soil types for random samples
obs.tbl <- table(soil.samples[soil.samples])
obs.tbl
```

```
  0 11.7 12.6 13.2 16.5 18.5 19.8 20.1 20.5   21 23.7 24.3 24.7 25.8 25.9 26.1
 35   27  238   68   83   89  196    3   27   21   51   24    8    9    5   33
26.7 37.1
  3   14
```

```
#Also tally soil types for each mule deer sampled
obs.tbl2 <- table(soils.locs[soils.locs])
obs.tbl2
```

```
  0  8.7  9.7 11.5 11.7 12.1 12.6   13 13.1 16.5 18.5   21   40
 71   31  670  237   96    1    8   89    2  236   12   30   31
```

10. Converts the counts to proportions

```
obs <- obs.tbl/sum(obs.tbl)
obs
```

```
          0        11.7        12.6        13.2        16.5        18.5
0.037473233 0.028907923 0.254817987 0.072805139 0.088865096 0.095289079
       19.8        20.1        20.5          21        23.7        24.3
0.209850107 0.003211991 0.028907923 0.022483940 0.054603854 0.025695931
       24.7        25.8        25.9        26.1        26.7        37.1
```

```
0.008565310 0.009635974 0.005353319 0.035331906 0.003211991 0.014989293
```

```
obs2 <- obs.tbl2/sum(obs.tbl2)
obs2
```

```
          0         8.7         9.7        11.5        11.7        12.1
0.046895641 0.020475561 0.442536328 0.156538970 0.063408190 0.000660502
       12.6          13        13.1        16.5        18.5          21
0.005284016 0.058784676 0.001321004 0.155878468 0.007926024 0.019815059
         40
0.020475561
```

Chapter 7

Manipulate Raster Data Layer

1. Open the script "RasterScript.R"
2. First we need to load the packages needed for the exercise

```
#install.packages(c("adehabitatHR","maptools","raster", "rgdal"))
library(terra)
library(sf)
library(dplyr)
library(tigris)
#options(tigris_use_cache = TRUE)
#tigris_cache_dir("/Users/davidwalter/Library/CloudStorage/OneDrive-ThePennsylvaniaStateUniver
#'
#install.packages("remotes")
# remotes::install_github(repo = "r-lib/devtools",
#                                dependencies = TRUE,
#                                upgrade = TRUE)
# devtools::install_github(repo = "r-lib/devtools",
#                            dependencies = TRUE,
#                            upgrade = TRUE)
#devtools::install_github("ropensci/FedData")
#remotes::install_github("ropensci/FedData")
library(FedData)
sessionInfo() #Search for and confirm FedData_4.0.0 or newer was loaded
```

```
R version 4.3.2 (2023-10-31)
Platform: x86_64-apple-darwin20 (64-bit)
Running under: macOS Sonoma 14.3.1
```

```
Matrix products: default
BLAS:   /Library/Frameworks/R.framework/Versions/4.3-x86_64/Resources/lib/libRblas.0.dy
LAPACK: /Library/Frameworks/R.framework/Versions/4.3-x86_64/Resources/lib/libRlapack.dy

locale:
[1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8

time zone: America/New_York
tzcode source: internal

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
[1] FedData_4.0.1 tigris_2.0.3  dplyr_1.1.3   sf_1.0-14     terra_1.7-46

loaded via a namespace (and not attached):
 [1] jsonlite_1.8.7     compiler_4.3.2     tidyselect_1.2.0   Rcpp_1.0.11
 [5] stringr_1.5.0      uuid_1.1-0         yaml_2.3.7         fastmap_1.1.1
 [9] readr_2.1.4        R6_2.5.1           generics_0.1.3     classInt_0.4-10
[13] knitr_1.42         tibble_3.2.1       units_0.8-4        DBI_1.1.3
[17] tzdb_0.4.0         pillar_1.9.0       rlang_1.1.1        utf8_1.2.3
[21] stringi_1.7.12     xfun_0.39          cli_3.6.1          magrittr_2.0.3
[25] class_7.3-22       digest_0.6.31      grid_4.3.2         rstudioapi_0.14
[29] hms_1.1.3          rappdirs_0.3.3     lifecycle_1.0.3    vctrs_0.6.3
[33] KernSmooth_2.23-22 proxy_0.4-27       evaluate_0.21      glue_1.6.2
[37] codetools_0.2-19   fansi_1.0.4        e1071_1.7-13       rmarkdown_2.21
[41] httr_1.4.7         tools_4.3.2        pkgconfig_2.0.3    htmltools_0.5.5
```

```
#library(adehabitatHR)
#library(maptools)
```

3. Now let's have a separate section of code to include projection information we will use throughout the exercise

```
ll.crs=st_crs(4269)
utm.crs <- st_crs(9001)
albers.crs <- st_crs(5070)
#CRS of shapefile layers
#crs <- CRS("+proj=aea +lat_1=29.5 +lat_2=45.5 +lat_0=23 +lon_0=-96 +x_0=0
#  +y_0=0 +datum=NAD83 +units=m +no_defs +ellps=GRS80 +towgs84=0,0,0")
#CRS of raster layers
#crs2 <- 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 +towgs84=0,0,0,0,0,0,0 +units=m +no_defs")
```

4. We will use the tigris package to downloaded statewide layers for state and county outlines

```
st <- tigris::states() %>%
  dplyr::filter(GEOID < "60") %>%
  tigris::shift_geometry()
#GEOID's above 60 are territories and islands, etc. So I'm removing them for scaling.
plot(st_geometry(st))

st <- st_transform(st, albers.crs)
SC.outline <- subset(st, st$NAME == "South Carolina")

SCcounties <- counties("South Carolina", cb = TRUE)
SCcounties <- st_transform(SCcounties, albers.crs)
```

5. Now let's add some shapefiles to our raster specific to the military base of interest

```
#Load county shapefile
county<- subset(SCcounties, SCcounties$NAME == "Beaufort")

#Load airport runway shapefile
run<-st_read("data/RunwayAlbers.shp")

#Load aircraft flight pattern shapefile
path<-st_read("data/FlightImage.shp")

#Load aircraft flight pattern shapefile
road <- st_read("data/CountyRoadAlbers.shp")
```

6. Using county outline, we can then download National Land Cover data for any year of interest using the FedData package

```
#new.outline <- sf::as_Spatial(county)#need to do this line due to error with dplyr package
SCnlcd <- get_nlcd(template=SC.outline, year = 2019, label = 'SC',force.redo = T)
#Write raster if needed
#writeRaster(SCnlcd, filename = "SCnlcd2019.tif", filetype = "GTiff", datatype = 'INT4U',overw

plot(SCnlcd)
plot(st_geometry(SCcounties), add=T, lwd=3)

#Import raster saved above just for demonstration
```

```
#SCnlcd <-rast("SCnlcd2019.tif")
plot(SCnlcd)
plot(st_geometry(SCcounties), add=T, lwd=3)
```

7. Plot out all the shapefiles overlayed on each other with and without the raster.

```
plot(st_geometry(county))
plot(st_geometry(road), add=T)
plot(st_geometry(run), col="red",add=T)
plot(st_geometry(path), col="blue",add=T)
```

8. Let's reclassify NLCD layer to get fewer land cover categories to make it easier to work with the raster.

```
# reclassify the values into 7 groups all values between 0 and 20 equal 1, etc. whil
m <- c(0, 19, NA, 20, 39, 1, 40, 50, 2, 51, 68, 3, 69,79, 4, 80, 88, 5, 89, 99, 6)
rclmat <- matrix(m, ncol=3, byrow=TRUE)
rc <- classify(SCnlcd, rclmat)
plot(rc)

#Clip the raster within the county polygon for a zoomed in view then plot
clip <- crop(rc, county)
plot(clip)
plot(st_geometry(county),add=T,lwd=2)
plot(st_geometry(run),add=T,lwd=2, col="red")
plot(st_geometry(path),add=T,lty="62", col="blue")
plot(st_geometry(road),add=T,lty="22", col="yellow")
```

9. We can load some vulture locations to extract landcover that each location occurs in that will be considered "used" habitat in resource selection analysis.

```
#Import bird 49 locations to R
bv49 <-read.csv("data/Bird49.csv", header=T)#How many bird locations?

utm.crs<-"+proj=utm +zone=17 +ellps=WGS84" #NOTE: If using a mac, we need to remove
#Include geometry
bvspdf <- st_as_sf(bv49, coords = c("x", "y"), crs = utm.crs)

plot(st_geometry(bvspdf), col="red")
#NOTE: Locations must be assigned to the UTM coordinate system prior to projection t
#so won't overlay on veg layer at this point because veg is in Albers
bv49Albers <- st_transform(bvspdf, crs = albers.crs)
plot(clip)
```

```
plot(st_geometry(bv49Albers), col="blue",add=T)
```

10. Determine which of those points lie within a cell that contains data by using the extract function. The extract function will extract covariate information from the raster for xy coordinates at for each location.

```
bvspdf.df <- data.frame(st_coordinates(bv49Albers))
bvspdf2 <- st_as_sf(bvspdf.df, coords= c("X", "Y"), crs = albers.crs)
bvspdf.sv <- vect(bvspdf2)#Need SpatVector for extract function in terra
veg.survey<-terra::extract(clip, bvspdf.sv)
veg.survey<-subset(veg.survey,!is.na(veg.survey$Class))
```

11. We can also create some random points within the extent of the area to be considered as "available" habitat.

```
##First we need to create a box across the study site to generate a random sample of locations
e.box <- as.polygons(clip,extent=TRUE)
e.box <- st_as_sf(e.box)
Sample.points <- st_sample(e.box, 1000, type="random")

plot(clip)
plot(st_geometry(Sample.points),add=T)

#Determine which of those points lie within a cell that contains data by using the extract fun
Sample.points.sv <- vect(Sample.points)#Need SpatVector class for extract function in terra
samp.survey<-terra::extract(clip, Sample.points.sv)
samp.survey<-subset(samp.survey,!is.na(samp.survey$Class))
```

12. New code addition below (Update 1/7/2021) to use mapview package to treat plotting data layers as done with GIS software. This package enables user to zoom in and out and scroll around layer after calling a plot function

```
library(mapview)
library(stars)
mapview(st_as_stars(clip)) + mapview(run) + mapview(path)
mapview(county) + mapview(bv49Albers, color="yellow")
mapview(run) + bv49Albers + road
```

Chapter 8

Creating a Hexagonal Polygon Grid Over a Study Area

1. Open the script "GridScripts.Rmd" and run code directly from the script

2. First we need to load the packages needed for the exercise

```
library(terra)
library(sf)
library(tigris)
library(FedData)
library(ggplot2)
```

3. Now let's have a separate section of code to include projection information we will use throughout the exercise. In previous versions, these lines of code were within each block of code

```
ll.crs=st_crs(4269)
utm.crs <- st_crs(9001)
albers.crs <- st_crs(5070)
```

4. We will use the tigris package to downloaded statewide layers for state and county outlines

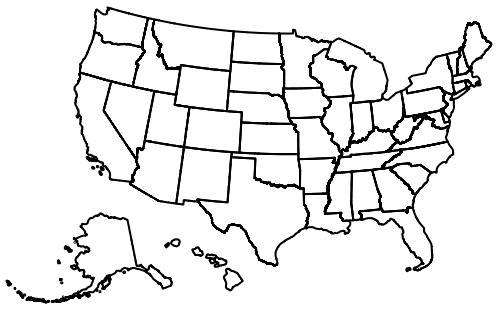
```
st <- tigris::states() %>%
  dplyr::filter(GEOID < "60") %>% #GEOID's above 60 are territories and islands, etc. So I'm r
```

```
  tigris::shift_geometry()
```

```
|
|                                                                      |   0%
|
|                                                                      |   1%
|
|=                                                                     |   1%
|
|=                                                                     |   2%
|
|==                                                                    |   2%
|
|==                                                                    |   3%
|
|===                                                                   |   4%
|
|===                                                                   |   5%
|
|====                                                                  |   5%
|
|====                                                                  |   6%
|
|=====                                                                 |   7%
|
|======                                                                |   8%
|
|======                                                                |   9%
|
|==========                                                            |  15%
|
|===========                                                           |  15%
|
|===========                                                           |  16%
|
|============                                                          |  17%
|
|============                                                          |  18%
|
|=============                                                         |  18%
|
|=============                                                         |  19%
|
|==============                                                        |  20%
|
```

```
|===============                                                       |  21%
|
|================                                                      |  23%
|
|============================                                          |  39%
|
|=======================================                               |  56%
|
|==================================================                    |  73%
|
|============================================================          |  89%
|
|======================================================================| 100%
```

```
plot(st_geometry(st))
```



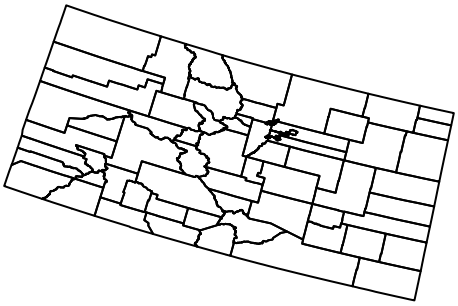
```
st <- st_transform(st, utm.crs)
CO.outline <- subset(st, st$NAME == "Colorado")

COcounties <- counties("Colorado", cb = TRUE)
```

```
|
|                                                                      |   0%
|
|=                                                                     |   1%
|
|=                                                                     |   2%
|
|==                                                                    |   2%
|
|==                                                                    |   3%
|
|========                                                              |  11%
|
```

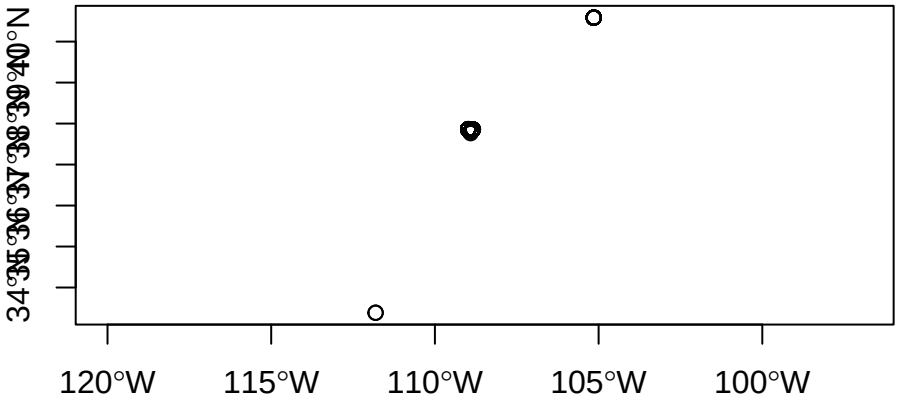
```
|=================                                                     |  25%
|
|=========================                                             |  39%
|
|================================                                      |  50%
|
|================================                                      |  51%
|
|====================================                                  |  55%
|
|========================================                              |  60%
|
|============================================                          |  68%
|
|=====================================================                 |  82%
|
|======================================================                |  83%
|
|======================================================                |  84%
|
|=======================================================               |  84%
|
|========================================================              |  86%
|
|=========================================================             |  87%
|
|===========================================================           |  88%
|
|============================================================          |  89%
|
|============================================================          |  89%
|
|============================================================          |  90%
|
|============================================================          |  91%
|
|=============================================================         |  91%
|
|=============================================================         |  92%
|
|==============================================================        |  92%
|
|==============================================================        |  93%
|
|================================================================      |  97%
|
```

```
|======================================================================  |  98%
|
|======================================================================= |  98%
|
|=======================================================================| 100%
```

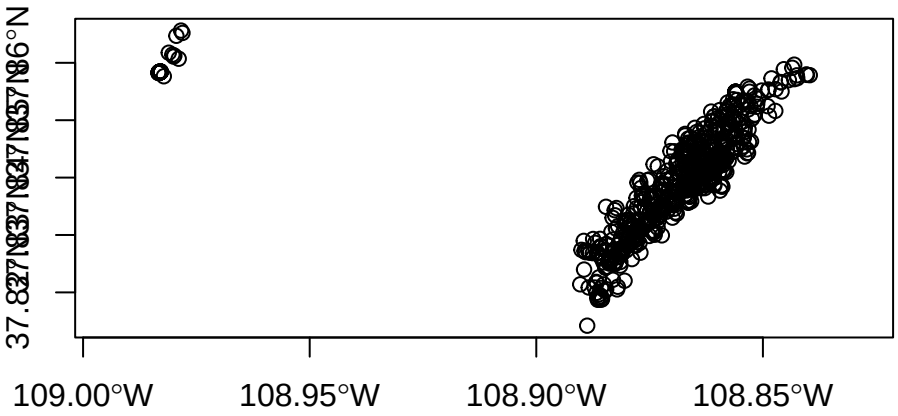
```
COcounties <- st_transform(COcounties, utm.crs)
plot(st_geometry(COcounties))
```



```
#Import the location from earlier exercise
muleys <-read.csv("data/muleysexample.csv", header=T)

#Remove outlier locations
coords <- st_as_sf(muleys, coords = c("Long", "Lat"), crs = ll.crs)
plot(st_geometry(coords),axes=T)
```



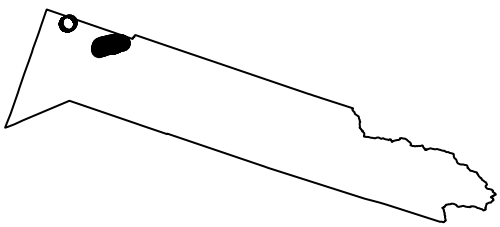

```
deer.spdf <- st_crop(coords, xmin=-107.0,xmax=-110.5,ymin=37.8,ymax=39.0)#Visually identified
plot(st_geometry(deer.spdf),axes=T)
```




```
deer.spdf <-st_transform(deer.spdf, crs=utm.crs)
```

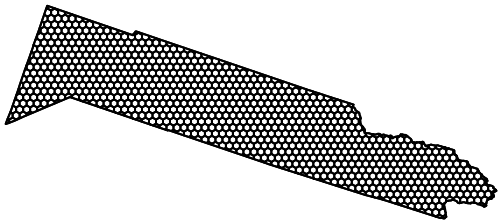
5. Now lets extract counties within the extent of the mule deer locations

```
int <- st_intersection(COcounties,deer.spdf)
clipped <- COcounties[int,]
plot(st_geometry(clipped))
plot(st_geometry(deer.spdf), add=T)
```



6. We also can create a hexagonal grid across the study site

```
grid_spacing <- 1500
HexPols <- st_make_grid(clipped, square = F, cellsize = c(grid_spacing, grid_spacing
  st_intersection(clipped)

plot(st_geometry(clipped))
plot(st_geometry(HexPols),add=T)
```

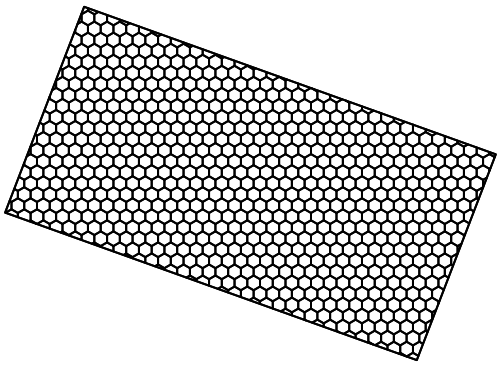


7. Create this hexagonal grid across our study site by zooming into deer locations

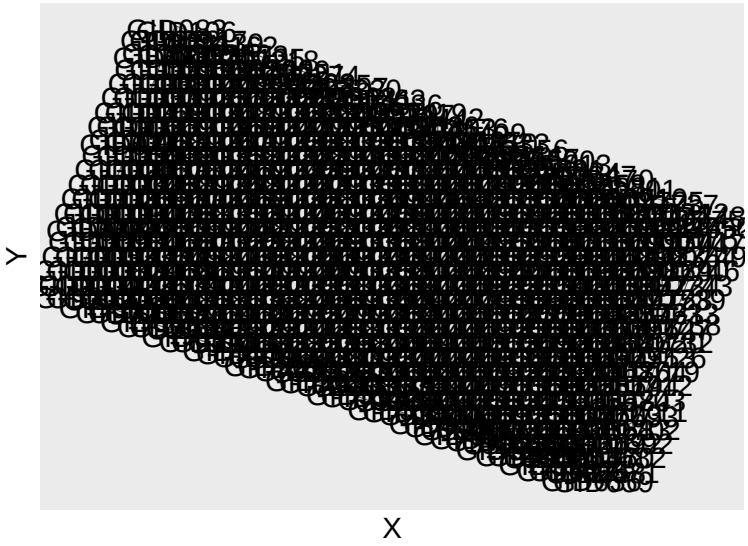
```
#Import the study site zoomed in shapefile
study.zoom<-st_read("data/MDzoom.shp")
```

```
Reading layer `MDzoom' from data source
  `/Users/davidwalter/Library/CloudStorage/OneDrive-ThePennsylvaniaStateUniversity/WalterRproject
  using driver `ESRI Shapefile'
Simple feature collection with 1 feature and 1 field
Geometry type: POLYGON
Dimension:     XY
Bounding box:  xmin: -109.2433 ymin: 37.56257 xmax: -108.6488 ymax: 37.95425
Geodetic CRS:  WGS 84
```

```
study.zoom <- st_transform(study.zoom, crs=utm.crs)

#Create new hexagonal grid
grid_spacing <- 1500
HexPols <- st_make_grid(study.zoom, square = F, cellsize = c(grid_spacing, grid_spacing)) %>%
  st_intersection(study.zoom) %>%
    cbind(data.frame(ID = sprintf(paste("GID%0",nchar(length(.)),"d",sep=""), 1:length(.)))) %
    st_sf()
plot(st_geometry(study.zoom))
plot(st_geometry(HexPols),add=T)
```



```
#Now add labels to each hexagon for unique ID
hex.centroids <- sf::st_point_on_surface(HexPols)
hex_coords <- as.data.frame(sf::st_coordinates(hex.centroids))
hex_coords$NAME <- HexPols$ID
ggplot() +
  geom_sf(data = HexPols) +
  geom_text(data = hex_coords, aes(X, Y, label = NAME), colour ="black")
```



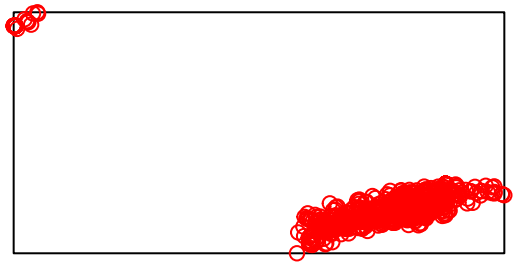
8. We can intersect the mule deer locations with the polygon shapefile (i.e., county) they occurred in Hexagon ID if needed

```
o = st_intersection(deer.spdf,COcounties)
```

```
Warning: attribute variables are assumed to be spatially constant throughout
all geometries
```

9. As an alternative to importing a polygon that we created in ArcMap above, we can create a polygon in R using the coordinates of the boundary box of the area of interest. In our case here, the bounding box will be the mule deer locations.

```
bb <- st_bbox(deer.spdf) %>% st_as_sfc()
#bb <- cbind(x=c(-108.83966,-108.83966,-108.9834,-108.9834, -108.83966),
#  y=c(37.8142, 37.86562,37.86562,37.8142,37.8142))
plot(st_geometry(bb))
plot(st_geometry(deer.spdf),col="red",add=T)
```



10. Now make practical use of the new bounding box we created by clipping a larger raster dataset. A smaller raster dataset runs analyses faster, provides a zoomed in view of mule deer locations and vegetation, and is just easier to work with.

```
#Load vegetation raster layer textfile clipped in ArcMap
veg <-raster("extentnlcd2.txt")
plot(veg)
class(veg)
```

```
#Clip using the raster imported with "raster" package
bbclip <- st_crop(veg, bb)
veg
```

```
#WON'T WORK because projections are not the same, WHY?
#Let's check projections of layers we are working with now.
st_crs(MDclip)
st_crs(deer.spdf)
st_crs(SP)
```

```
st_crs(veg)
```

11. We need to have all layers in same projection so project the deer.spdf to Albers and then clip vegetation layer with new polygon we created in the Albers projection.

```
deer.albers <-st_transform(deer.spdf, crs=albers.crs)
bb.albers <- st_bbox(deer.albers) %>% st_as_sfc()
#Check to see all our layers are now in Albers projection
st_crs(veg)
st_crs(deer.albers)
st_crs(bb.albers)

#Clip using the raster imported with "raster" package
bbclip <- st_crop(veg, AlbersSP)
plot(bbclip)
plot(st_geometry(deer.albers), col="red", add=T)
plot(st_geometry(bb.albers), lwd=5, add=T)
```

Chapter 9

Creating a Square Polygon Grid Over a Study Area

1. Open the script "GridSystem2Script.Rmd" and run code directly from the script

2. First we need to load the packages needed for the exercise

```
library(sf)
library(terra)
library(adehabitatMA)
library(FedData)
```

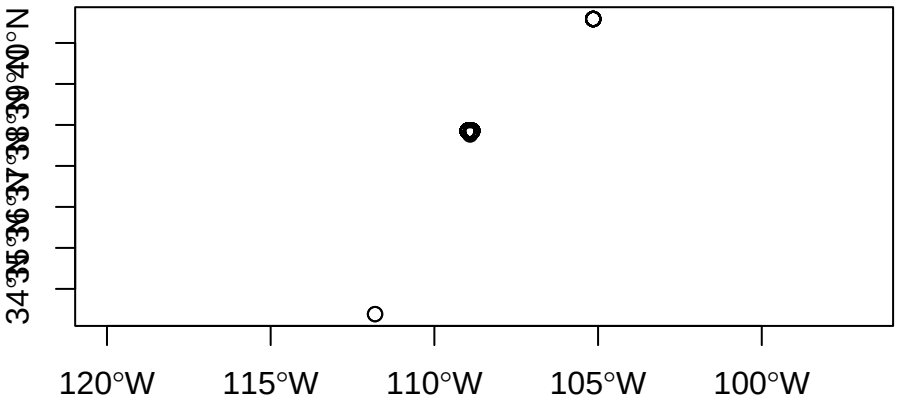
3. Now let's have a separate section of code to include projection information we will use throughout the exercise. In previous versions, these lines of code were within each block of code

```
ll.crs <- st_crs(4269)
utm.crs <- st_crs(9001)
albers.crs <- st_crs(5070)
```

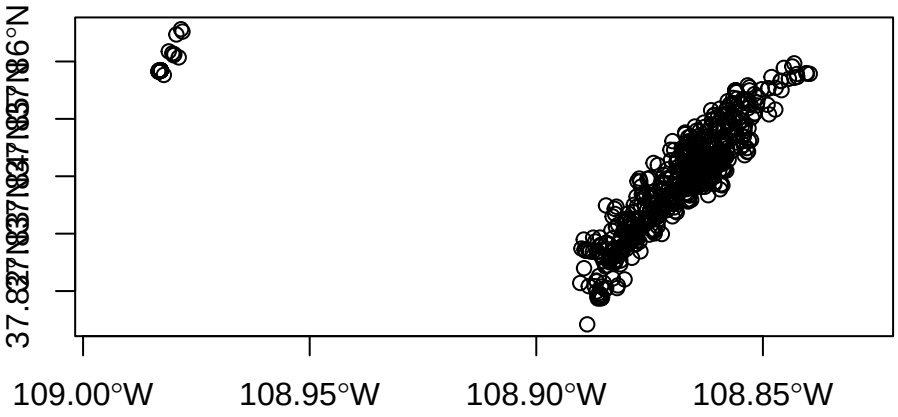
4. We need to have all layers in same projection so import, create, and remove outliers for mule deer locations then project all to the Albers projection as we did previously.

```
#Import the location from earlier exercise
muleys <-read.csv("data/muleysexample.csv", header=T)

#Remove outlier locations
```

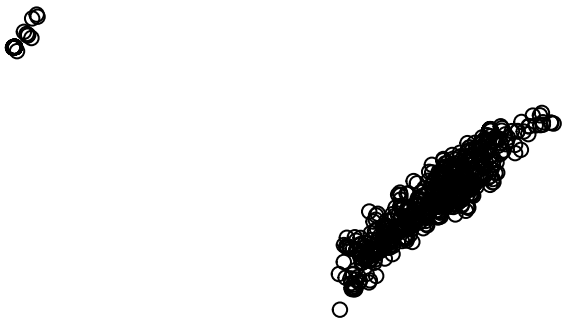
```
coords <- st_as_sf(muleys, coords = c("Long", "Lat"), crs = ll.crs)
plot(st_geometry(coords),axes=T)
```




```
deer.spdf <- st_crop(coords, xmin=-107.0,xmax=-110.5,ymin=37.8,ymax=39.0)#Visually i
plot(st_geometry(deer.spdf),axes=T)
```

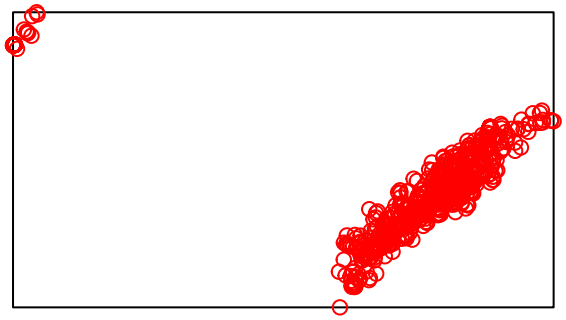



```
deer.spdf <-st_transform(deer.spdf, crs=utm.crs)

#Project deer.spdf to Albers as in previous exercise
deer.albers <-st_transform(deer.spdf, crs=albers.crs)
plot(st_geometry(deer.albers,axes=T))
```



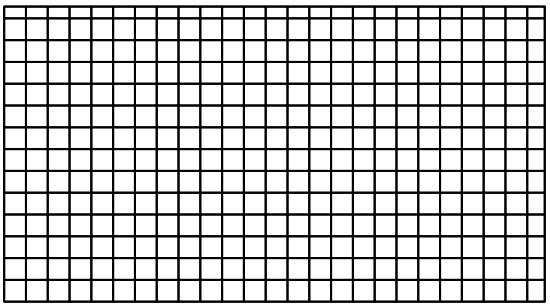
5. Create points for x and y from the bounding box of all mule deer locations with 1500 m spacing between each point.

```
bb <- st_bbox(deer.albers) %>% st_as_sfc()
#bb <- cbind(x=c(-108.83966,-108.83966,-108.9834,-108.9834, -108.83966),
#  y=c(37.8142, 37.86562,37.86562,37.8142,37.8142))
plot(st_geometry(bb))
plot(st_geometry(deer.albers),col="red",add=T)
```



6. Create a grid of all pairs of coordinates (as a data.frame) using the "expand grid" function and then make it a gridded object.

```
grid_spacing <- 500
GridPols <- st_make_grid(bb, square = T, cellsize = c(grid_spacing, grid_spacing)) %>% # the g
  st_intersection(bb) %>%
    cbind(data.frame(ID = sprintf(paste("GID%0",nchar(length(.)),"d",sep=""), 1:length(.)))) %
    st_sf()
plot(st_geometry(bb))
plot(st_geometry(GridPols),add=T)
```



7. Similar to the hexagonal grid, identify the cell ID that contains each mule deer location.

```
o = st_intersection(deer.albers,GridPols)
```

```
Warning: attribute variables are assumed to be spatially constant throughout
all geometries
```

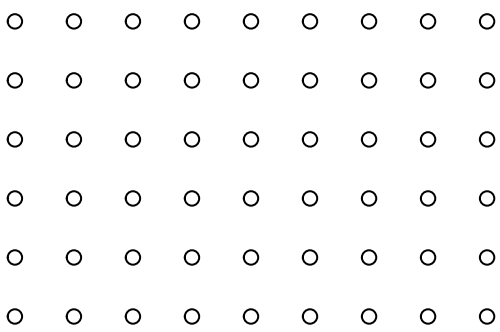
```
head(o)
```

```
Simple feature collection with 6 features and 21 fields
Geometry type: POINT
Dimension:     XY
Bounding box:  xmin: -1120488 ymin: 1718097 xmax: -1120110 ymax: 1718916
Projected CRS: NAD83 / Conus Albers
         DateTimeAnimalID id  Serial            Acq_Time              Acq_ST
197 D82011.10.20 12:00:00 D8 647578A 2011.10.20 12:00:40 2011.10.20 12:00:00
230 D82011.10.24 15:00:00 D8 647578A 2011.10.24 15:00:48 2011.10.24 15:00:00
231 D82011.10.24 18:00:00 D8 647578A 2011.10.24 18:01:58 2011.10.24 18:00:00
232 D82011.10.24 21:00:00 D8 647578A 2011.10.24 21:00:49 2011.10.24 21:00:00
333 D82011.11.06 18:00:00 D8 647578A 2011.11.06 18:01:04 2011.11.06 18:00:00
328 D82011.11.06 03:00:00 D8 647578A 2011.11.06 03:00:52 2011.11.06 03:00:00
             GPSFixTime      GPSFixAtt UTM_Zone       Y      X GPS_Alt
197 2011.10.20 12:00:40 Succeeded (3D)      12S 4187300 685838    2176
230 2011.10.24 15:00:48 Succeeded (3D)      12S 4187827 686114    2237
231 2011.10.24 18:01:58 Succeeded (3D)      12S 4187807 686078    2217
232 2011.10.24 21:00:49 Succeeded (3D)      12S 4187812 686107    2233
333 2011.11.06 18:01:04 Succeeded (3D)      12S 4187805 686032    2222
328 2011.11.06 03:00:52 Succeeded (3D)      12S 4188096 685684    2173
    GPS_Speed GPSHeading GPSHorizEr GPS_PD    GPS_SB GPS_SC GPS_NT Pre_Data
197      0.05        0.7      18.46    3.4   4227808      6     40       No
230      0.07        0.7      24.00    2.8 134299840      5     48       No
231      0.04        0.7     108.00    4.7      2314      4    118       No
232      0.06      348.0      24.00    3.8  16779290      5     49       No
```

```
333      0.03        0.7      45.00    3.3     66314      5     64       No
328      0.02        0.7      48.00    3.8   1318948      5     52       No
    Error     ID                  geometry
197    NA GID015 POINT (-1120464 1718097)
230    NA GID016 POINT (-1120110 1718579)
231    NA GID016 POINT (-1120149 1718566)
232    NA GID016 POINT (-1120119 1718566)
333    NA GID016 POINT (-1120193 1718571)
328    NA GID040 POINT (-1120488 1718916)
```

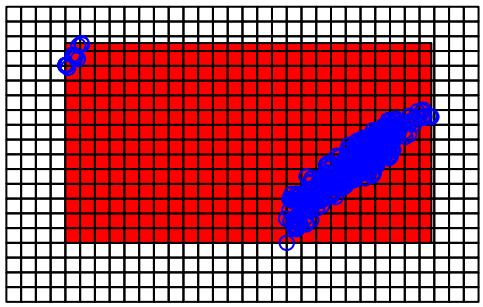
8. If we get some NA errors because our grid does not encompass all mule deer locations then we can expand the grid size extending beyond our locations.

```
# Create vectors of the x and y points using boundary box created around deer locations
bb1 <- st_bbox(GridPols)

increment = 2000
minx=(min(bb1$xmin)-(increment))
maxx=(max(bb1$xmax)+(increment))
miny=(min(bb1$ymin)-(increment))
maxy=(max(bb1$ymax)+(increment))
  # Create vectors of the x and y points using mean size of deer home range of X square kilome
x=seq(from=minx,to=maxx,by=increment)
y=seq(from=miny,to=maxy,by=increment)
# Create a grid of all pairs of coordinates (as a data.frame)
xy=expand.grid(x=x,y=y)
grid.pts<-st_as_sf(xy, coords = c("x","y"),crs = albers.crs)
plot(st_geometry(grid.pts))
```



```
GridPols2 <- st_make_grid(grid.pts, square=T,cellsize = grid_spacing) %>%
   cbind(data.frame(ID = sprintf(paste("GID%0",nchar(length(.)),"d",sep=""), 1:length(.)))) %>
   st_sf()
```

```
plot(st_geometry(GridPols2))
plot(st_geometry(GridPols),add=T, col="red")
plot(st_geometry(deer.albers),add=T, col="blue")
```



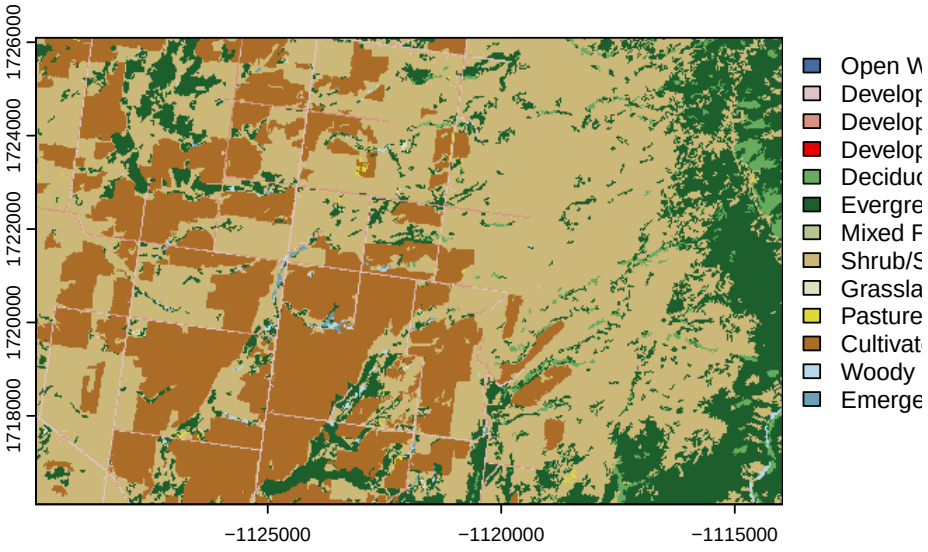
9. Then identify the cell ID that contains each mule deer location from the new expanded grid

```
##BE SURE TO RUN CODE FROM XY CREATION THROUGH NEW2 AGAIN THEN LOOK AT DATA!!
o2 = st_intersection(deer.albers,GridPols2)
```

```
Warning: attribute variables are assumed to be spatially constant throughout
all geometries
```

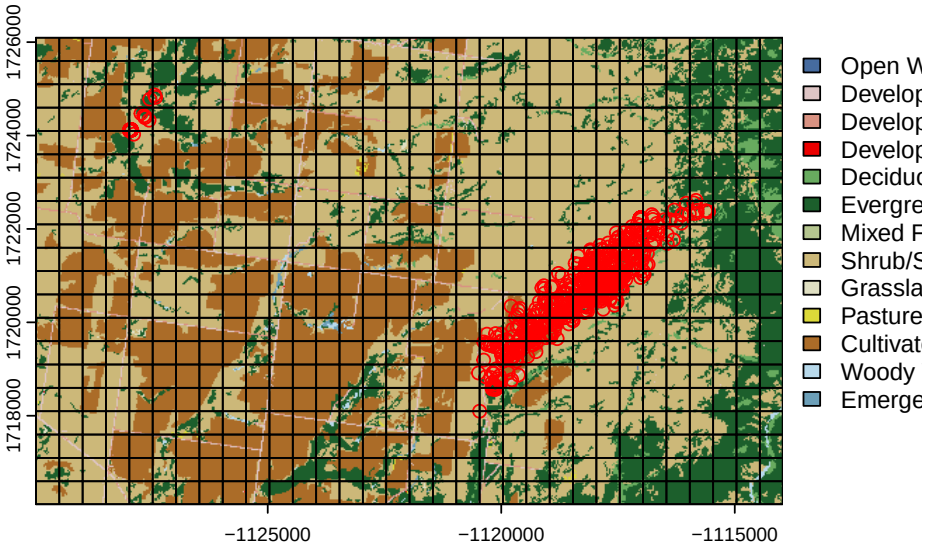
10. Now we can load a vegetation raster layer using the FedData package to summarize vegetation categories within each polygon grid cell.

```
nlcd <- get_nlcd(template=GridPols2, year = 2019, label = 'nlcd',force.redo = T)
plot(nlcd)
```



11. Clip the raster within the extent of the newly created grid

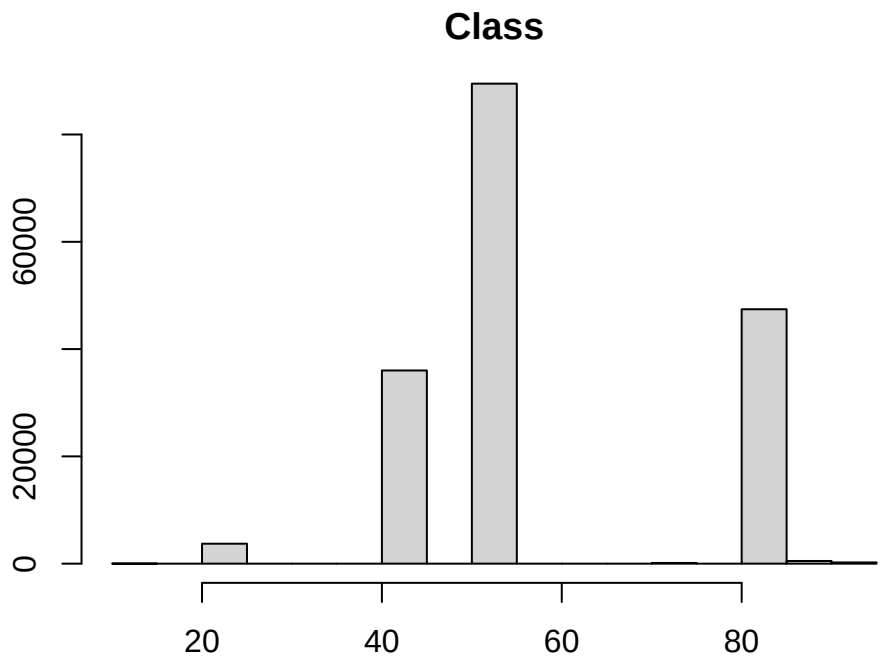
```
bbclip <- crop(nlcd, GridPols2)
plot(bbclip)
plot(st_geometry(deer.albers),add=T,col="red")
plot(st_geometry(GridPols2), add=T)
```



```
#Cell size of raster layer
xres(bbclip)
```

```
[1] 30
```

```
#Create histogram of vegetation categories in bbclip
hist(bbclip)
```




```
#Calculate cell size in square meters
ii <- st_area(GridPols2)#requires adehabitatMA package
ii[1]
```

```
250000 [m^2]
```

12. We can extract the vegetation characteristics within each polygon of the grid and then tabulate area of each vegetation category within each polygon by extracting vegetation within each polygon by ID then summarizing the vegetation characteristics in each cell to be used in future resource selection analysis or disease epidemiology.

```
library(exactextractr)
area = exact_extract(bbclip,GridPols2)
```

```
|
|                                                                      |   0%
|
|                                                                      |   1%
|
|=                                                                     |   1%
|
|=                                                                     |   2%
|
|==                                                                    |   2%
|
|==                                                                    |   3%
|
|===                                                                   |   4%
|
|===                                                                   |   5%
|
|====                                                                  |   5%
|
|====                                                                  |   6%
|
|=====                                                                 |   7%
|
|=====                                                                 |   8%
|
|======                                                                |   8%
|
|======                                                                |   9%
|
|=======                                                               |   9%
|
|=======                                                               |  10%
|
|=======                                                               |  11%
|
|========                                                              |  11%
|
|========                                                              |  12%
|
|=========                                                             |  12%
|
|=========                                                             |  13%
|
|==========                                                            |  14%
|
|==========                                                            |  15%
```

```
|
|==========                                                      |  15%
|
|==========                                                      |  16%
|
|===========                                                     |  17%
|
|===========                                                     |  18%
|
|============                                                    |  18%
|
|===========                                                     |  19%
|
|============                                                    |  19%
|
|============                                                    |  20%
|
|============                                                    |  21%
|
|==============                                                  |  21%
|
|==============                                                  |  22%
|
|===============                                                 |  22%
|
|===============                                                 |  23%
|
|================                                                |  24%
|
|================                                                |  25%
|
|=================                                               |  25%
|
|=================                                               |  26%
|
|==================                                              |  27%
|
|==================                                              |  28%
|
|====================                                            |  28%
|
|====================                                            |  29%
|
|=====================                                           |  29%
|
|=====================                                           |  30%
```

```
|
|=====================                                 |  31%
|
|======================                                |  31%
|
|======================                                |  32%
|
|======================                                |  32%
|
|=======================                               |  33%
|
|=======================                               |  34%
|
|========================                              |  35%
|
|========================                              |  35%
|
|=========================                             |  36%
|
|=========================                             |  37%
|
|==========================                            |  38%
|
|==========================                            |  38%
|
|===========================                           |  39%
|
|===========================                           |  39%
|
|============================                          |  40%
|
|============================                          |  41%
|
|=============================                         |  41%
|
|=============================                         |  42%
|
|==============================                        |  42%
|
|==============================                        |  43%
|
|===============================                       |  44%
|
|===============================                       |  45%
|
|================================                      |  45%
```

```
|
|================================                                 |  46%
|
|================================                                 |  47%
|
|================================                                 |  48%
|
|================================                                 |  48%
|
|================================                                 |  49%
|
|================================                                 |  49%
|
|================================                                 |  50%
|
|================================                                 |  51%
|
|================================                                 |  51%
|
|================================                                 |  52%
|
|================================                                 |  52%
|
|================================                                 |  53%
|
|================================                                 |  54%
|
|================================                                 |  55%
|
|================================                                 |  55%
|
|================================                                 |  56%
|
|================================                                 |  57%
|
|================================                                 |  58%
|
|================================                                 |  58%
|
|================================                                 |  59%
|
|================================                                 |  59%
|
|================================                                 |  60%
|
|================================                                 |  61%
```

```
|
|=========================================                              |  61%
|
|=========================================                              |  62%
|
|==========================================                             |  62%
|
|==========================================                             |  63%
|
|==========================================                             |  64%
|
|==========================================                             |  65%
|
|==========================================                             |  65%
|
|==========================================                             |  66%
|
|==========================================                             |  67%
|
|==========================================                             |  68%
|
|==========================================                             |  68%
|
|==========================================                             |  69%
|
|==========================================                             |  69%
|
|==========================================                             |  70%
|
|==========================================                             |  71%
|
|==========================================                             |  71%
|
|==========================================                             |  72%
|
|==========================================                             |  72%
|
|==========================================                             |  73%
|
|==========================================                             |  74%
|
|==========================================                             |  75%
|
|==========================================                             |  75%
|
|==========================================                             |  76%
```

```
|
|=================================================                  |  77%
|
|=================================================                  |  78%
|
|==================================================                 |  78%
|
|==================================================                 |  79%
|
|===================================================                |  79%
|
|===================================================                |  80%
|
|===================================================                |  81%
|
|====================================================               |  81%
|
|====================================================               |  82%
|
|=====================================================              |  82%
|
|=====================================================              |  83%
|
|======================================================             |  84%
|
|======================================================             |  85%
|
|=======================================================            |  85%
|
|=======================================================            |  86%
|
|========================================================           |  87%
|
|========================================================           |  88%
|
|=========================================================          |  88%
|
|=========================================================          |  89%
|
|==========================================================         |  89%
|
|==========================================================         |  90%
|
|===========================================================        |  91%
|
|============================================================       |  91%
```

```
  |
  |=================================================================      |  92%
  |
  |=================================================================      |  92%
  |
  |=================================================================      |  93%
  |
  |=================================================================      |  94%
  |
  |=================================================================      |  95%
  |
  |=================================================================      |  95%
  |
  |=================================================================      |  96%
  |
  |=================================================================      |  97%
  |
  |=================================================================      |  98%
  |
  |=================================================================      |  98%
  |
  |=================================================================      |  99%
  |
  |=================================================================      |  99%
  |
  |=================================================================      | 100%
```

```
classes <- sort(unique(bbclip[]))
combine <- lapply(area, FUN=function(x) { as.data.frame(prop.table(table(factor(x[,1],
                          levels = classes))))} )
head(combine)
```

```
[[1]]
   Var1       Freq
1    11 0.00000000
2    21 0.00000000
3    22 0.00000000
4    23 0.00000000
5    41 0.00000000
6    42 0.00000000
7    43 0.00000000
8    52 0.92733564
9    71 0.00000000
10   81 0.00000000
11   82 0.07266436
```

```
12   90 0.00000000
13   95 0.00000000

[[2]]
   Var1       Freq
1    11 0.00000000
2    21 0.00000000
3    22 0.00000000
4    23 0.00000000
5    41 0.00000000
6    42 0.05228758
7    43 0.00000000
8    52 0.83986928
9    71 0.00000000
10   81 0.00000000
11   82 0.10784314
12   90 0.00000000
13   95 0.00000000

[[3]]
   Var1       Freq
1    11 0.00000000
2    21 0.06535948
3    22 0.00000000
4    23 0.00000000
5    41 0.00000000
6    42 0.00000000
7    43 0.00000000
8    52 0.93464052
9    71 0.00000000
10   81 0.00000000
11   82 0.00000000
12   90 0.00000000
13   95 0.00000000

[[4]]
   Var1        Freq
1    11 0.000000000
2    21 0.076124567
3    22 0.024221453
4    23 0.000000000
5    41 0.000000000
6    42 0.013840830
7    43 0.000000000
8    52 0.882352941
9    71 0.000000000
```

```
10   81 0.000000000
11   82 0.003460208
12   90 0.000000000
13   95 0.000000000

[[5]]
   Var1        Freq
1    11 0.000000000
2    21 0.026143791
3    22 0.000000000
4    23 0.000000000
5    41 0.000000000
6    42 0.196078431
7    43 0.000000000
8    52 0.549019608
9    71 0.000000000
10   81 0.003267974
11   82 0.225490196
12   90 0.000000000
13   95 0.000000000

[[6]]
   Var1      Freq
1    11 0.0000000
2    21 0.0000000
3    22 0.0000000
4    23 0.0000000
5    41 0.0000000
6    42 0.3986928
7    43 0.0000000
8    52 0.4052288
9    71 0.0000000
10   81 0.0000000
11   82 0.1960784
12   90 0.0000000
13   95 0.0000000
```

Chapter 10

Creating Buffers

For this exercise, we will again be working with the Colorado mule deer locations and rasters from earlier sections (1.3, 1.7). Creating buffers around locations of animals, plots, or some other variable may be necessary to determine what occurs around the locations. Often times,in resource selection studies, we may want to generate buffers that can be considered used habitat within the buffer as opposed to simply counting only the habitat that the location is found. Lets begin with loading the proper packages and mule deer locations from previous exercise. Because we are dealing with the raster layer projected in Albers, we will need to project our mule deer locations as we did above.

1. Open the script "BufferScript.Rmd" and run code directly from the script

2. First we need to load the packages needed for the exercise

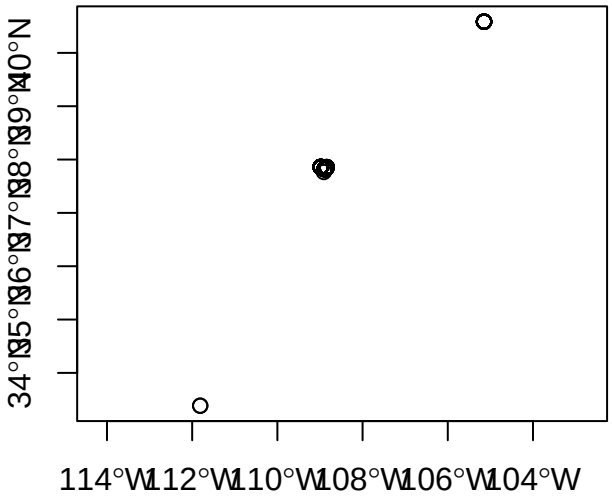
```
library(sf)
library(terra)
library(exactextractr)
library(FedData)
library(zoo)
```

3. Now let's have a separate section of code to include projection information we will use throughout the exercise. In previous versions, these lines of code were within each block of code

```
ll.crs <- st_crs(4269)
utm.crs <- st_crs(9001)
albers.crs <- st_crs(5070)
```

4. We will import a dataset then subset by a single deer to make processing time faster

```
#Import the location from earlier exercise
muleys <-read.csv("data/muleysexample.csv", header=T)

#Remove outlier locations
coords <- st_as_sf(muleys, coords = c("Long", "Lat"), crs = ll.crs)
plot(st_geometry(coords),axes=T)
```




```
deer.spdf <- st_crop(coords, xmin=-107.0,xmax=-110.5,ymin=37.8,ymax=39.0)#Visually i
plot(st_geometry(deer.spdf),axes=T)
```