1.6 Manipulate Polygon Layer

Manual of Applied Spatial Ecology

3/11/2022

1. Exercise 1.6 - Download and extract zip folder into your preferred location
2. Set working directory to the extracted folder in R under Session - Set Working Directory...
3. Now open the script "SoilScript.Rmd" and run code directly from the script
4. First we need to load the packages needed for the exercise

```
library(rgdal)
library(maptools)
library(foreign)
```

```
soils<-readOGR(dsn=".",layer="Soil_Properties",verbose=FALSE)
plot(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"
```

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

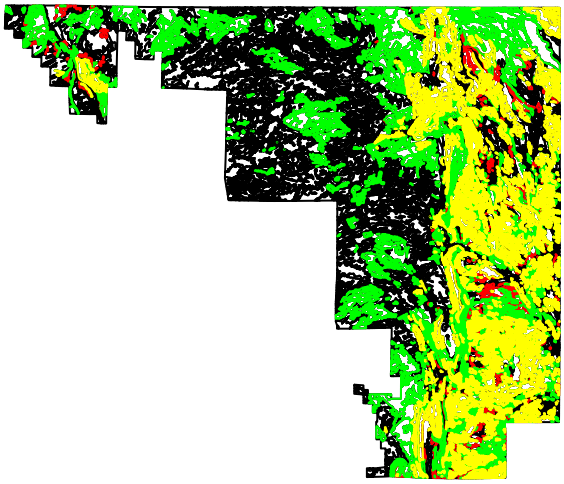
```
#Shapefiles contain several slots which can be called with the "@" symbol or slot(object, "data")
soils@data[1:10,] #a data frame with n observations associated with X covariates,
#soils@polygons #the number of polygons that the shapefile consists of
soils@plotOrder #the order of the polygons
soils@bbox #boundary box
soils@proj4string #projection
#Within the slot
soils@polygons[[1]] #will bring up the first polygon
soils@polygons[[1]]@area #will bring up the area for the first polygon
soils@polygons[[1]]@ID #will retrieve the ID of the first polygon
soils@polygons[[1]]@plotOrder #will retrieve the order of 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(soils)
high.clay<- soils[soils$Clay>30,]
plot(high.clay, border="red", add=TRUE)

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

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

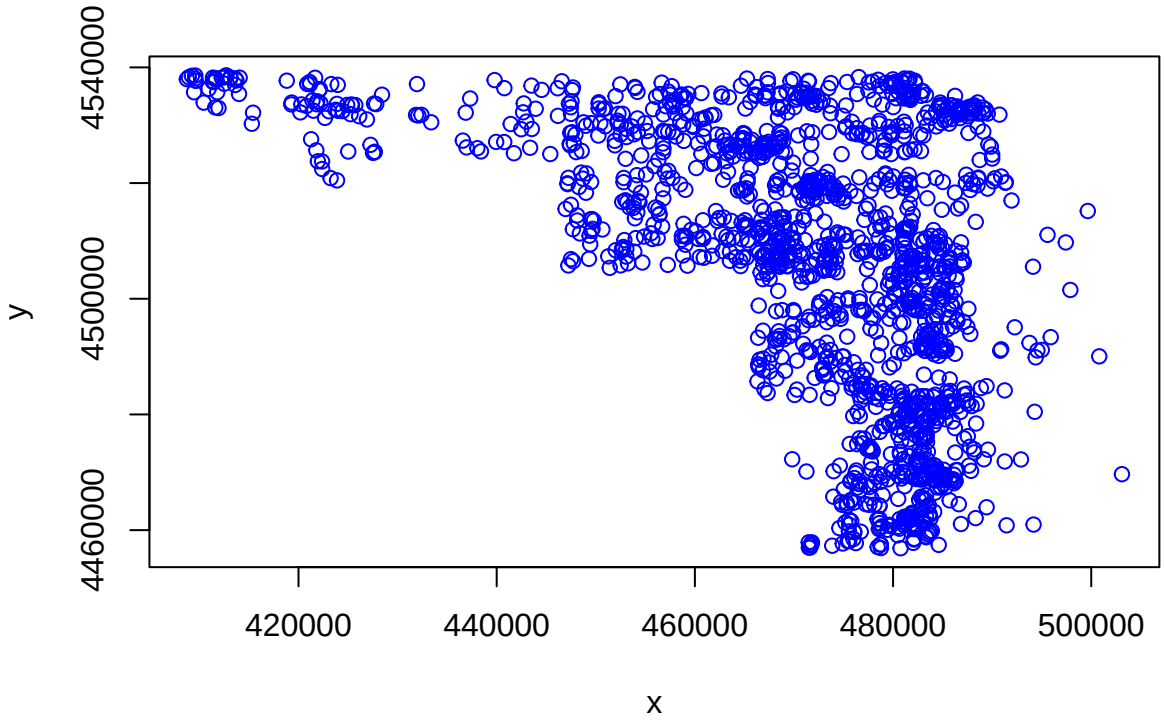


6. Bring in locations of harvested mule deer

```
#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("MDjitterclip.csv", header=T)

coords<-data.frame(x = mule$x, y = mule$y)
crs<-"+proj=utm +zone=13 +datum=WGS84 +no_defs +towgs84=0,0,0"

plot(coords, col="blue")
```



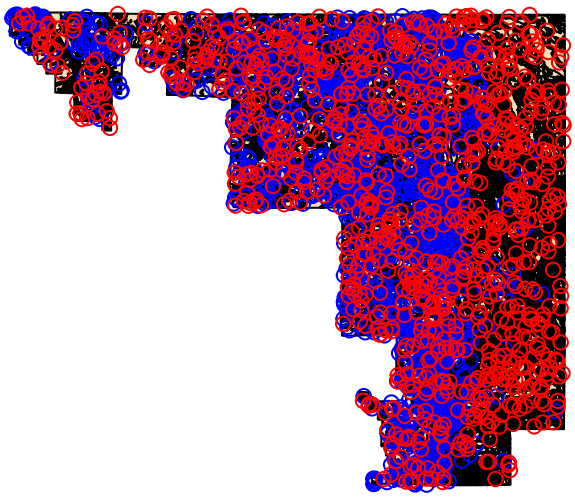

```
#par(new=TRUE)
```

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<-spsample(soils, n=1000, type="random")
```

```
## Warning in proj4string(obj): CRS object has comment, which is lost in output; in tests, see
## https://cran.r-project.org/web/packages/sp/vignettes/CRS_warnings.html
```

```
plot(soils, col="wheat")
points(coords, col="blue")
points(samples, col="red")
```



8. Creates a SpatialPoints object from the location coordinates

```
samples@bbox <- soils@bbox
samples@proj4string <- soils@proj4string
```

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

```
#Matches points with polygons:
soils.idx<- over(samples,soils)
locs <- SpatialPoints(coords)
locs@proj4string <- soils@proj4string
soils.locs<- over(locs, soils)

#Tally clay soil types for random samples
obs.tbl <- table(soils.idx$Clay[soils.idx$Clay])
obs.tbl
```

```
##
##    0  8.7  9.7 11.5 11.7 12.6 13.1 16.5 17.4 18.5 19.8   20 20.1 20.7 23.5 25.9
##   17    7    1   44   74  125   25    9  172   32   28   39  136   15   24    6
## 26.1 26.2 26.5 26.7 27.1   40
##   56   50    1    2    9   75
```

```
#Also tally soil types for each mule deer sampled
obs.tbl2 <- table(soils.locs$Clay[soils.locs$Clay])
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         8.7         9.7        11.5        11.7        12.6
## 0.017951426 0.007391763 0.001055966 0.046462513 0.078141499 0.131995776
##        13.1        16.5        17.4        18.5        19.8          20
## 0.026399155 0.009503696 0.181626188 0.033790919 0.029567054 0.041182682
##        20.1        20.7        23.5        25.9        26.1        26.2
## 0.143611404 0.015839493 0.025343189 0.006335797 0.059134108 0.052798310
##        26.5        26.7        27.1          40
## 0.001055966 0.002111932 0.009503696 0.079197466
```

```
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
```