**PART 1: GIS Tutorial**

Objectives: Find data and make a map. Map lon/lats of a point file, do a table join, spatial join, buffer, dissolve.

Input data

1. STATE TO REGION TABLE (**PROVIDED**)

[state\_census\_regions\_tablejoin\_dissolve.csv]

1. XY TABLE of STATE CAPITAL COORDINATES (**PROVIDED**)

[us\_state\_capitals\_xy\_buffer\_spatialjoin.csv]

1. US STATE SHAPEFILE (**NOT PROVIDED**)

To turn in:

A map of US dissolved into regions. Each region is a different color and has a thick border. Put state outlines on top (no fill) with a thin border. Map the state capitals, put a 50 km buffers around them, and label them as capital, state (absorb the state underneath them per spatial join). Maps need a scale bar, and legend.   
  
Extra credit: clip the boundaries of the buffers that go into the ocean.

**To turn in: Put your name on your map and upload to Canvas. If you had a ‘tutor,’ put their name on it too. The tutor will get credit as if it was their own submission.**

**PART 2: Basic Data Manipulation**

Objectives: practice some basic functions for data cleaning. You can do all of these things in Excel if you need to (PDFs aren’t recommended in Excel), but the code below is for R users. Python, Java, MATLAB is fine.  
(Watch your curly quotes.)

**R Basics for Data Manipulation**

Environment**: setwd("C:/R")**

**Reading in data**

EnergyNY = read.csv("EnergyNY.csv", header = TRUE, stringsAsFactors = FALSE)  
  
View(EnergyNY)

**Drop columns, drop rows**

NewData <- EnergyNY [,1:3]  
NewData <- EnergyNY [1:25,] #I really never use this.

# Rm(Olddata)if you want to get rid of something.

**Rename columns**

names(EnergyNY)[names(EnergyNY) == "Tract"] <- "MyTract"  
  
ALSO, PLEASE SEE **PLYR and TIDYVERSE**. **Find and replace**   
EnergyNY$Place <- gsub("Brooklyn|Queens|Staten Island" ,"DiffIsland", EnergyNY$Borough)

(Google regular expressions as well—if you wanted to delete everything after a number, for example)

**Pick out part of a code**  
EnergyNY$shortLat <- substr(EnergyNY$Latitude, 0, 4)  
  
**Concatenate**  
EnergyNY$coordinates <- paste(EnergyNY$Latitude, EnergyNY$Longitude, sep = ", ")

**Table joins (hard in excel—use arcmap)**

sub1 <- EnergyNY[,1:4]

sub2 <- EnergyNY[,c(1,5:ncol(EnergyNY))]

fulldata <- merge(sub1,sub2)

INNER JOIN is default  
LEFT JOIN set all.x = TRUE;  
RIGHT JOIN set all.y = TRUE  
OUTER JOIN set all = TRUE  
  
If the column names to join differ, and they will, try this:

*Set up a fake case*

sub1 <- EnergyNY[,1:4]

sub2 <- EnergyNY[,c(1,5:ncol(EnergyNY))]

names(sub2)[names(sub2) == "BBL"] <- "BBL2"

fulldata <- merge(sub1, sub2, by.x = "BBL", by.y = "BBL2")

**Aggregate & filter data**

***A Create a frequency table***

BoroughFreq <- table(EnergyNY$Borough)

***B Filter data***

D <- filter(EnergyNY, CouncilDistrict < 10)

***C Produce summary statistics***

buildingMeanCenters = aggregate(EnergyNY[,c("Latitude","Longitude")], by = list(EnergyNY$Postcode),FUN = mean)

IF YOU LIKE SQL INSTEAD, I RECOMMEND **SQLDF**. To install a package, use install.packages("sqldf") and then library("sqldf")

section <- sqldf("select \* from EnergyNY where Borough = 'Manhattan' or Borough = 'Queens'")

courts <- sqldf("select \* from EnergyNY where BuildingName LIKE '%Court%'")  
  
sqldf("select Borough, sum(EnergyUse\_MMBTU) as TotalEnergy from EnergyNY group by Borough")

**Plotting**

***Scatterplot (General)***

plot(x,y, main = "my plot", xlab = "x axis", ylab = "y axis")

PDFs   
  
plot(density(EnergyNY$EnergyUse\_MMBTU))  
  
lines(density(EnergyNY[EnergyNY$Borough=="Manhattan",]$EnergyUse\_MMBTU), col="red")  
lines(density(EnergyNY[EnergyNY$Borough=="Bronx",]$EnergyUse\_MMBTU), col="blue")

CDFs

plot(ecdf(EnergyNY$EnergyUse\_MMBTU), main = "Cumulative Density Function of Energy Usage Per Building", xlab = "Energy Usage in MMBTUs")

I RECOMMEND **GGPLOT2.**

**Writing data**

write.csv(EnergyNY, file = "Energy.csv", row.names = FALSE)

**Merging files**

years <- c(12,15,13,19,18,11,16)  
  
for (i in years) {

stringfilename <- paste("Energy20",i,sep="")

energy = read.csv(stringfilename, header = TRUE, stringsAsFactors = FALSE)   
  
write.csv(energy, file = "EnergyAllYears.csv", row.names = FALSE, append = TRUE)  
 }

**To turn in: nothing, but be aware that these are important things you have to do.**