Alvin Luong

02/27/2016

P. Thomas

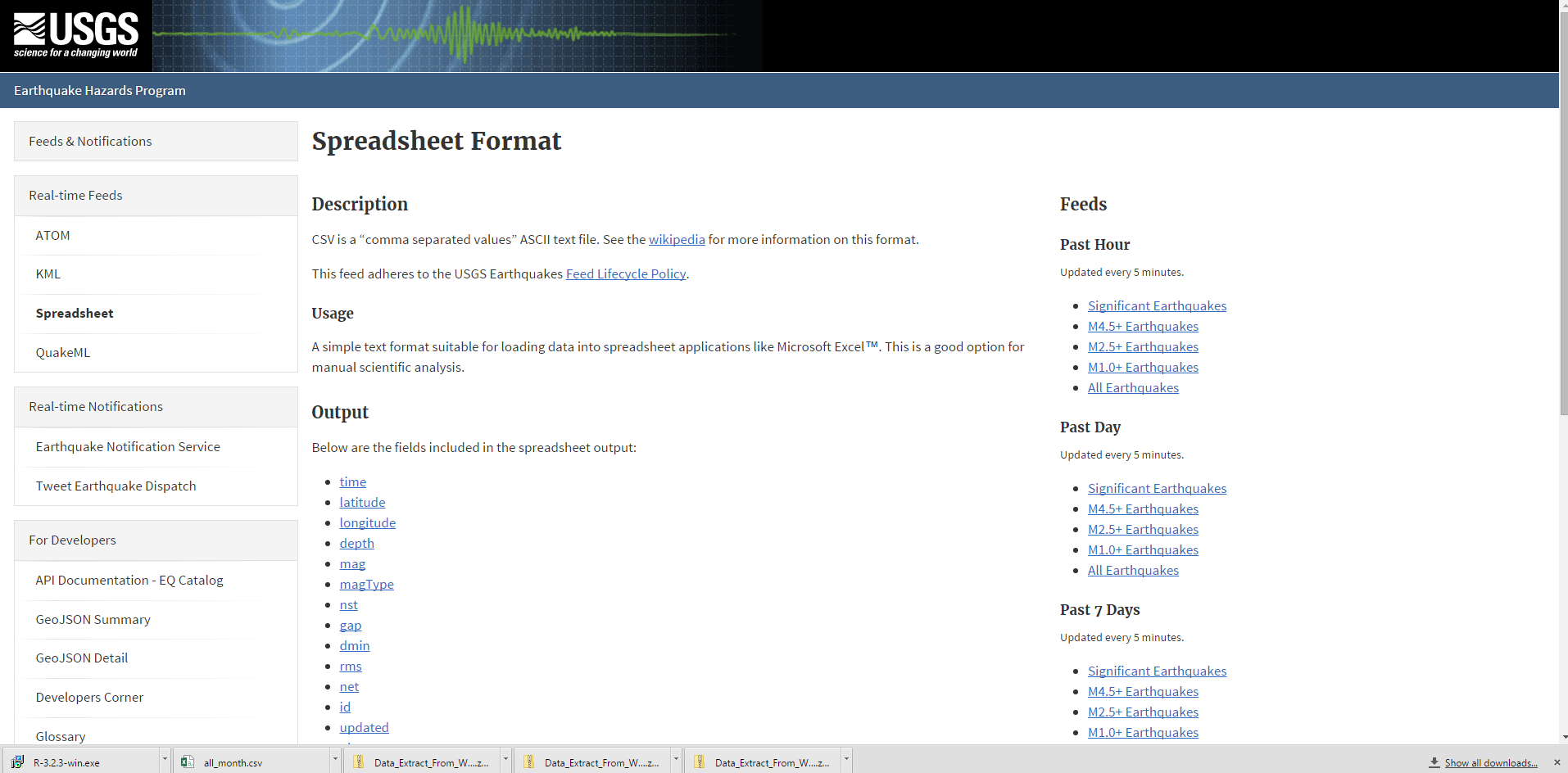
**CIS320 R Project**

I plan on gathering live data from around the world at any given time to see what is going on geologically. Where the most earth quakes are occurring as well as where the biggest ones are occurring. This was brought to my attention because of the earthquake that just happened in Taiwan.

**World Earthquake Data Set @ USGS**

<http://earthquake.usgs.gov/earthquakes/feed/v1.0/summary/all_month.csv>

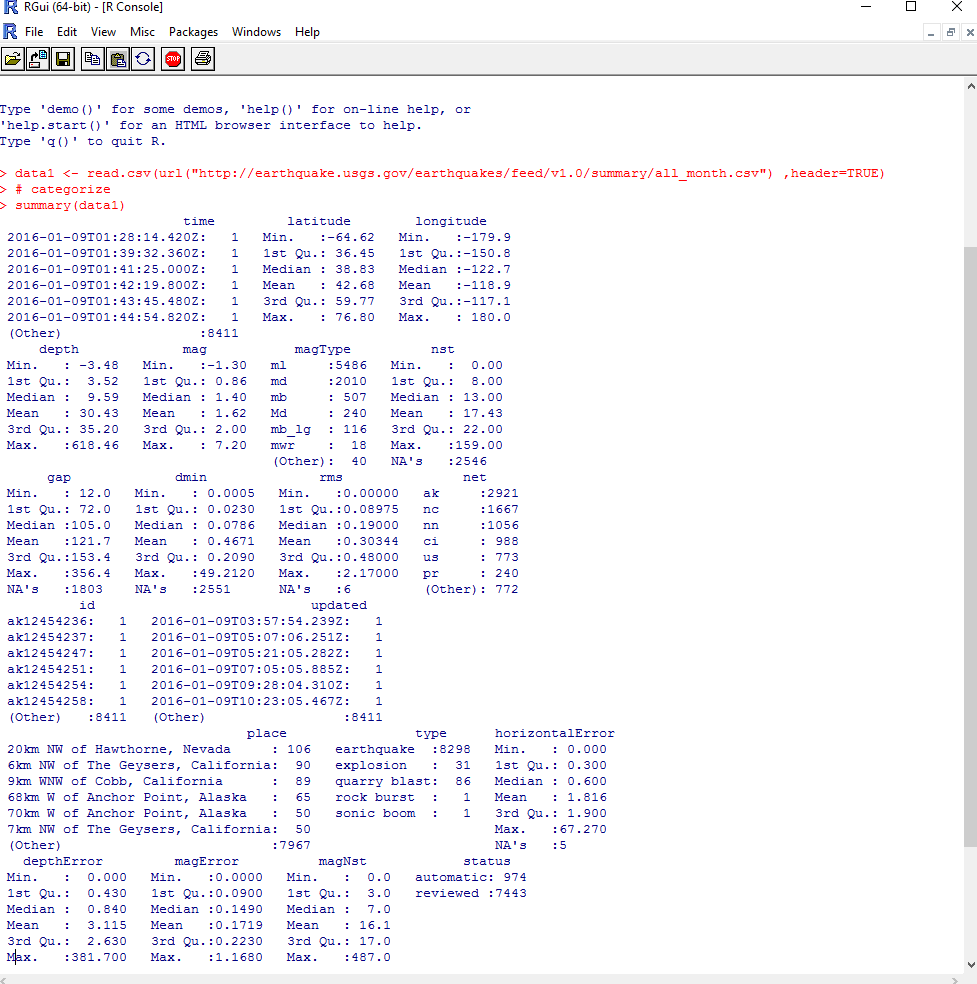
Real time live data set of over 9000 earthquake in the past 30 days, world wide.

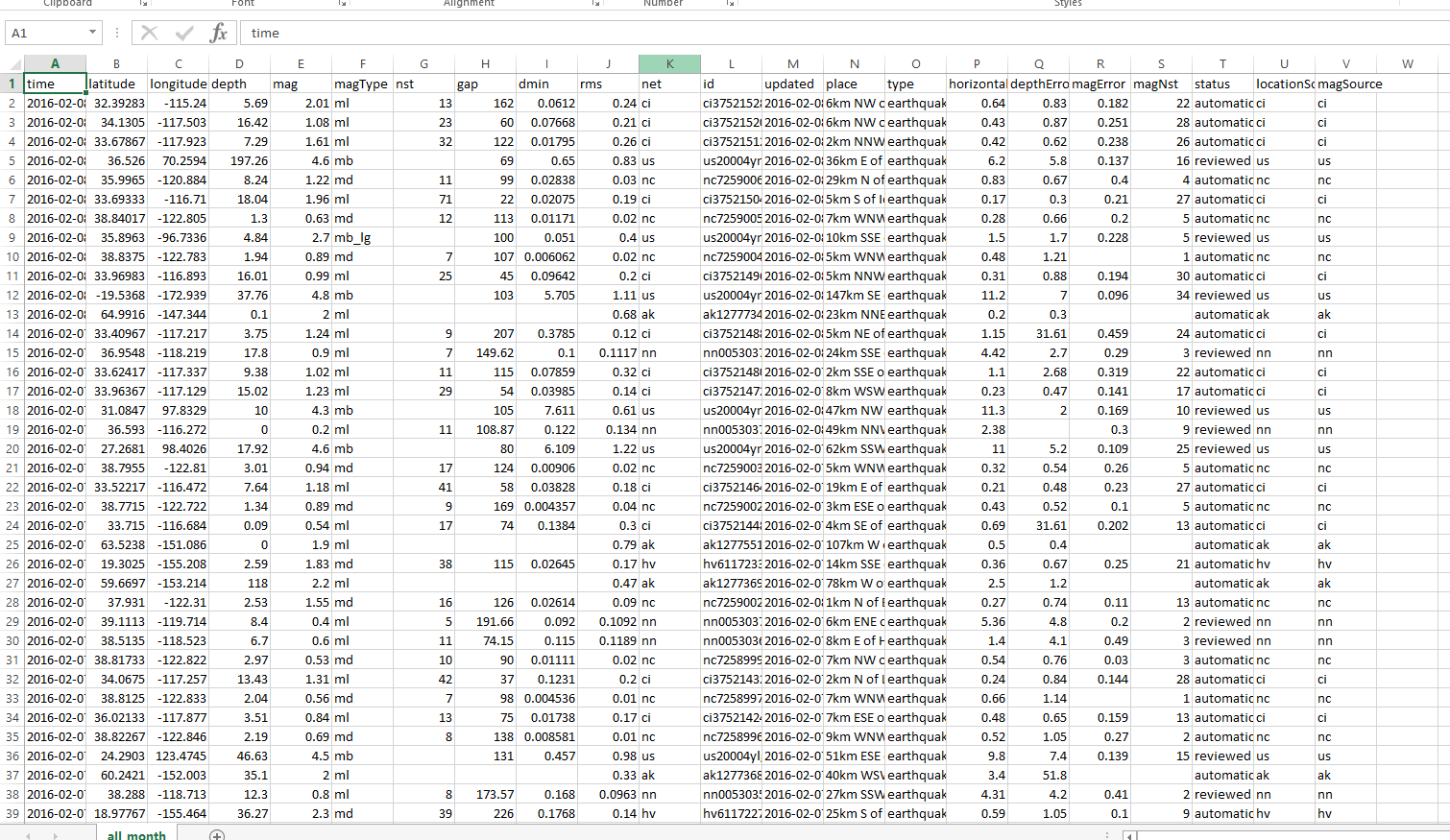


data1 <- read.csv(url("http://earthquake.usgs.gov/earthquakes/feed/v1.0/summary/all\_month.csv") ,header=TRUE)

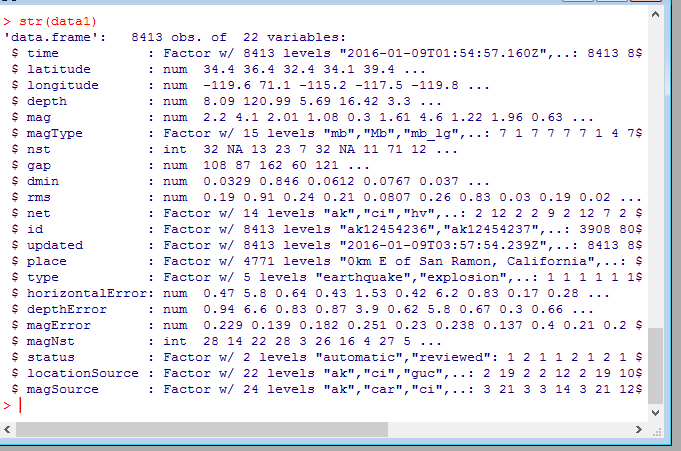
# categorize

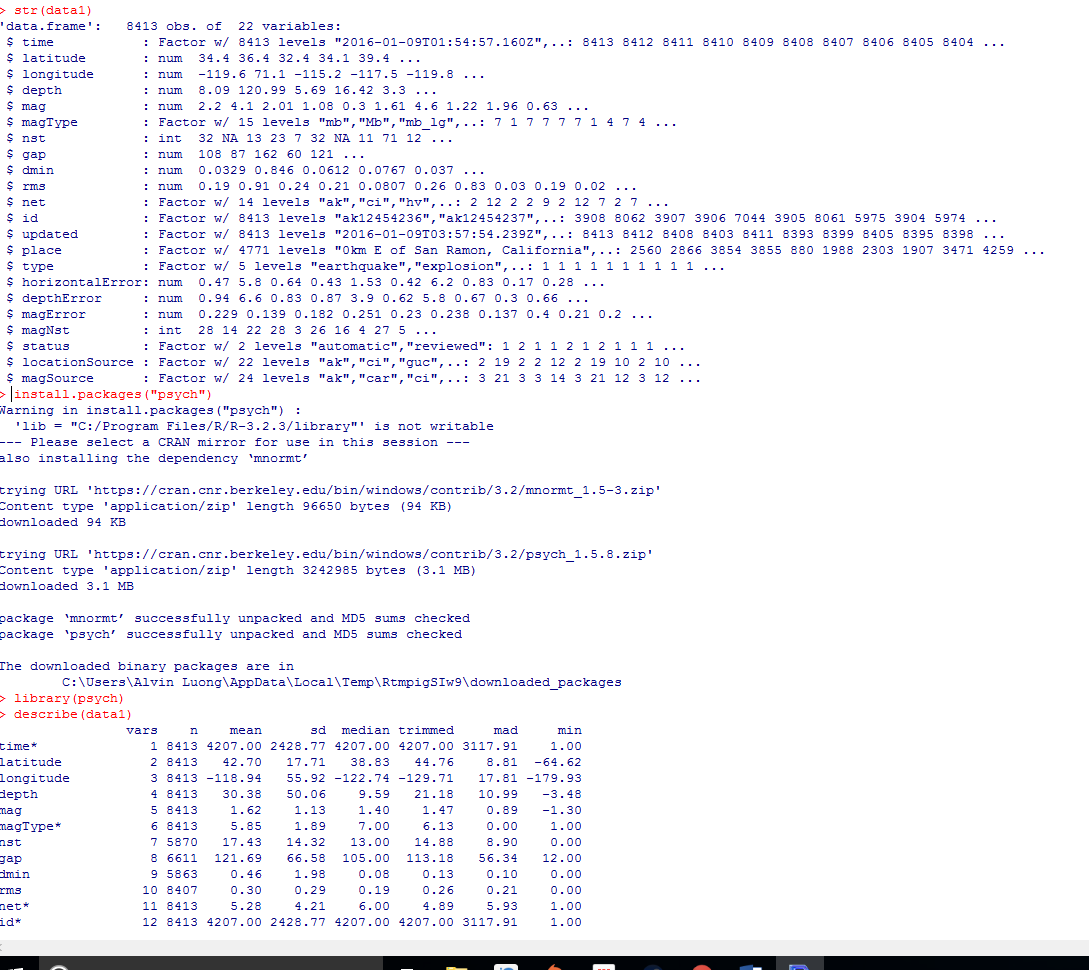
summary(data1)





Data set continues down to 9000+ rows of data to sort through. With this information I can kind of draw the attributes or relationship between the size of the quakes and how deep in the earth they are occurring.

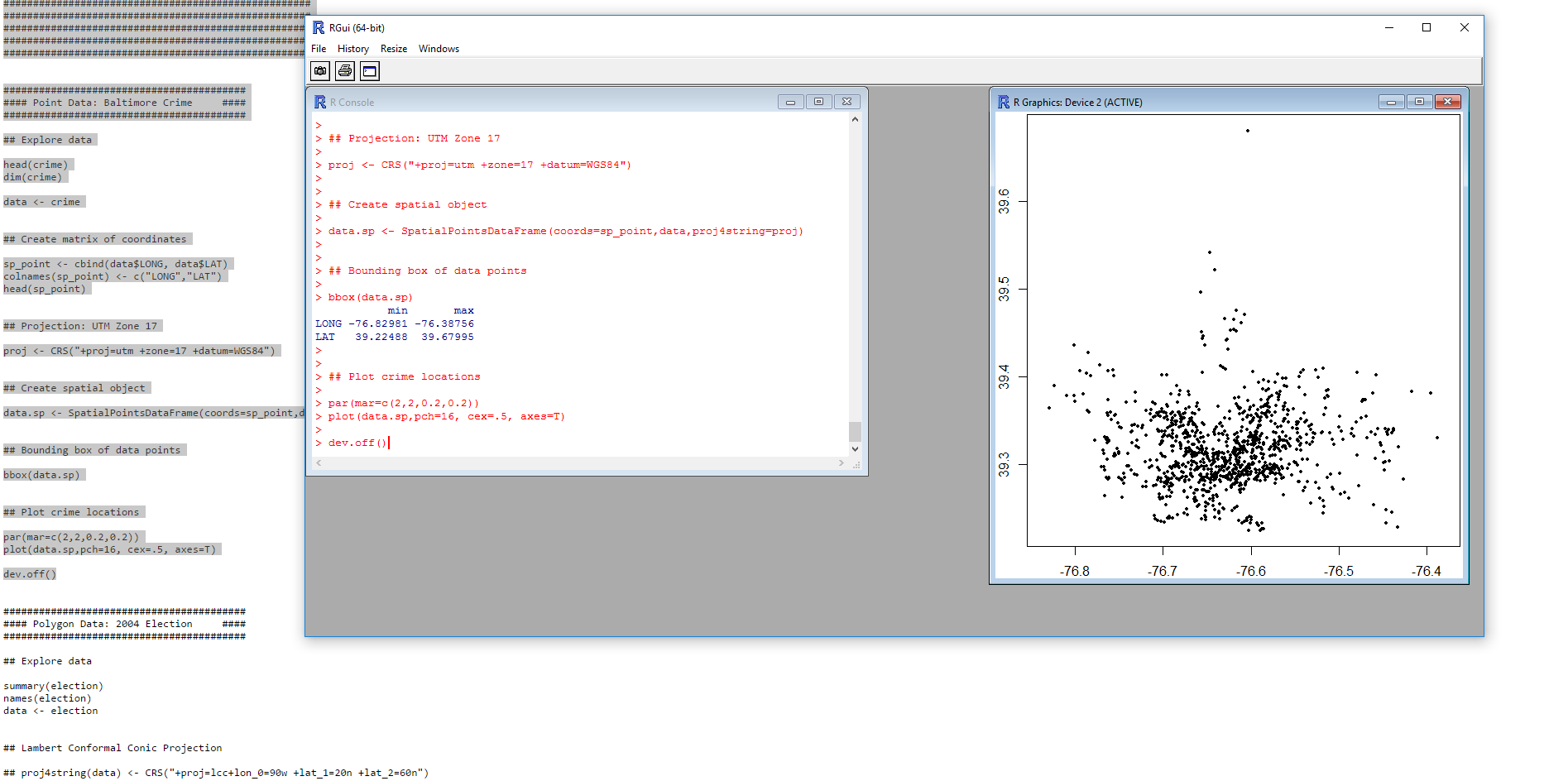


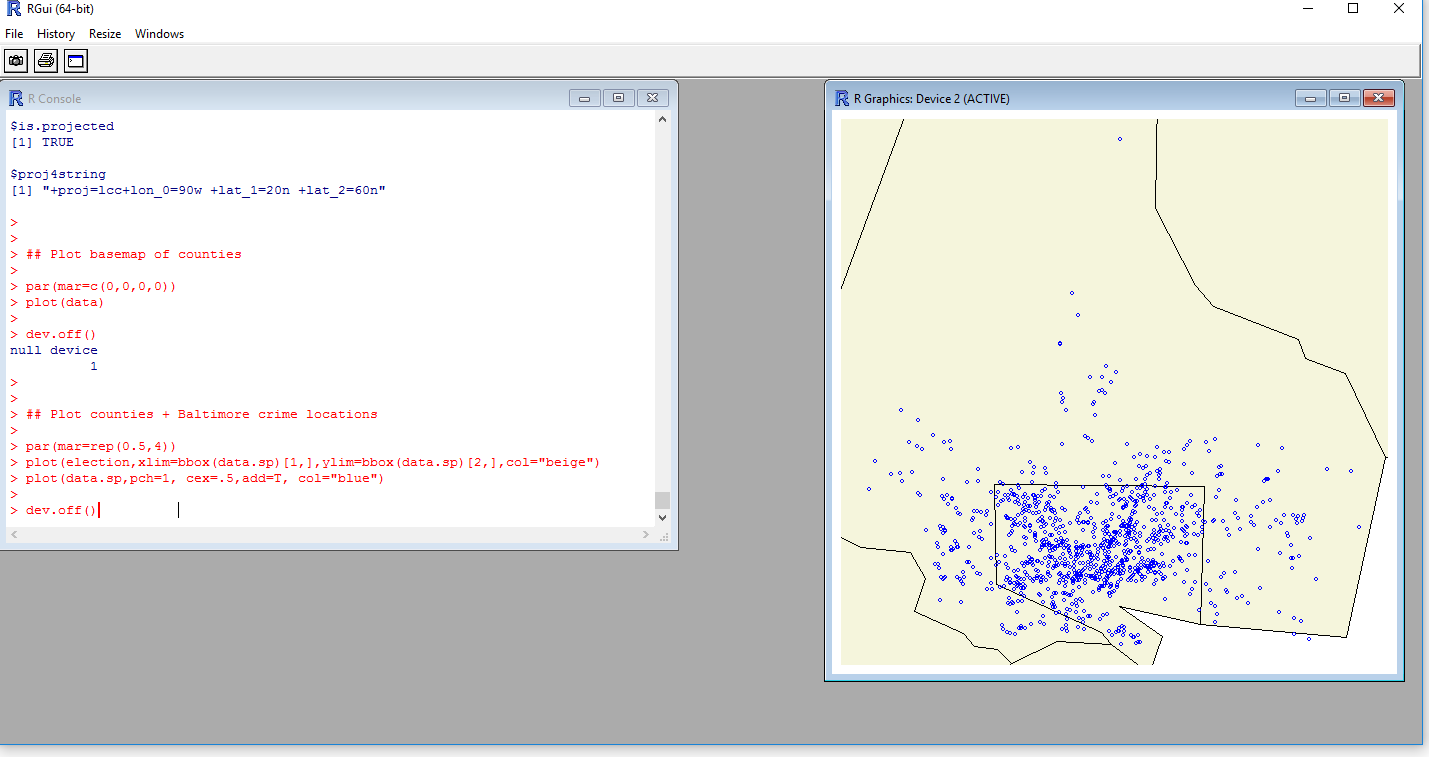


I was seeing what people did with seismic data and what I found was that spatial statistics was one of the primary tools in analyzing the data. All the points can be plotted and overlaid over a template of the earth.

There are ways to color code all the individual points by time, magnitude or even how many times of occurrence by setting transparency and they can plot over each other.

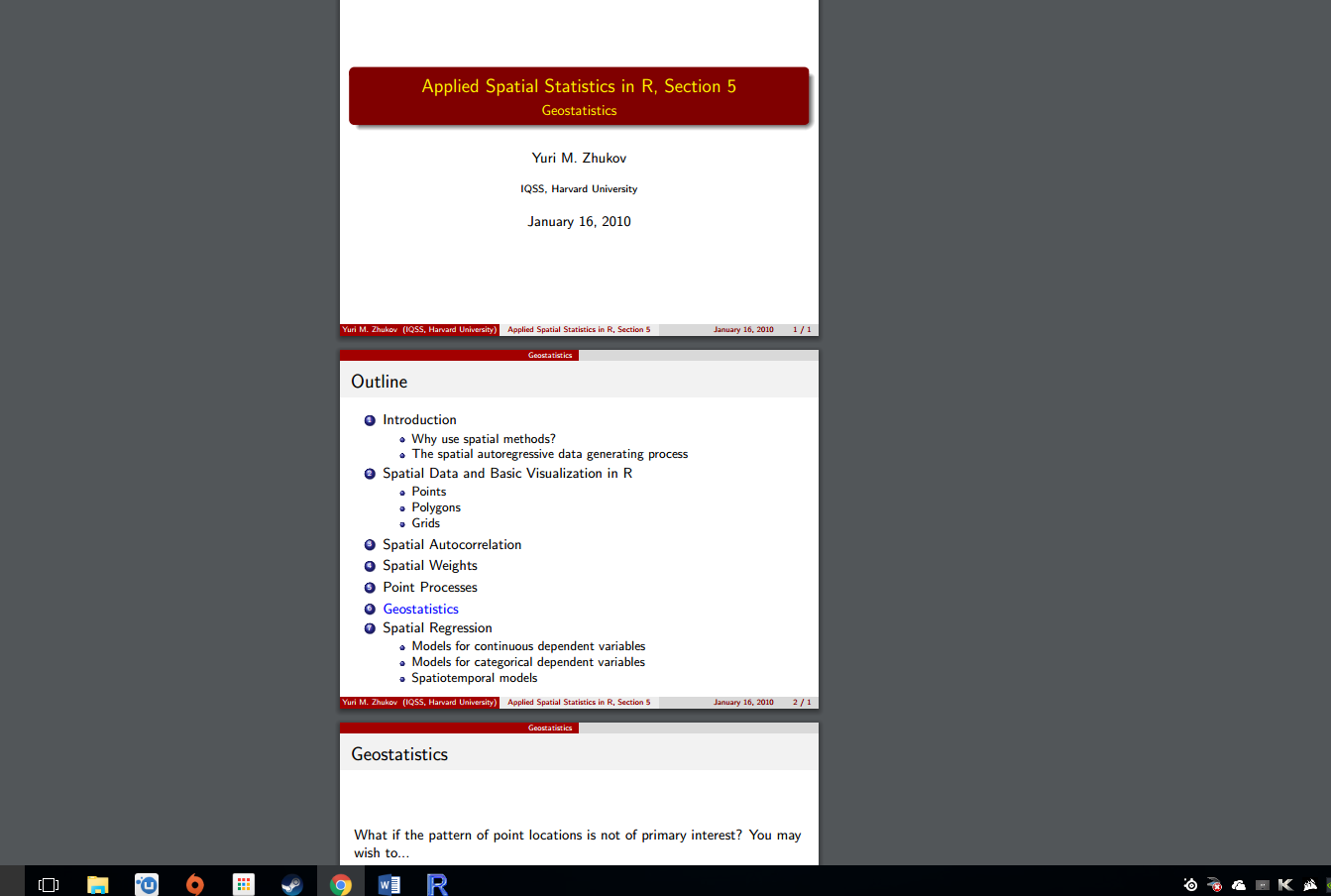
I am learning how to use R, but the programming becomes very difficult as the logic and syntax involved is new to me. I’ve found guides online and this particular one is from Harvard regarding spatial point data on crime analysis. I ran the code to see what was going on and get more understanding of what R can do.





<http://www.people.fas.harvard.edu/~zhukov/Spatial5.pdf>

I’m going to study Applied Spatial Statistics in R from a Harvard Course



Hope I am not in over my head.

I have found many resources explaining how people are using R in analyzing USGS data. Cluster analysis seems to be the standard way to model the data that is available to us.

library(sp)

library(plotrix)

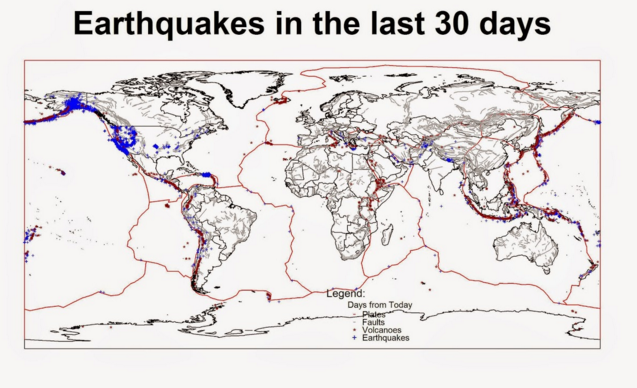
library(raster)

library(rgeos)

library(rgdal)

library(scatterplot3d)

These packages are loaded into R to analyze the data. A map of the earth is loaded and all the latitude and longitude points for the earthquakes are overlaid onto the world map. Then they are color coded to show occurrences of earthquakes by either fault lines or volcanoes. The data points organized this way are very easy to understand and give a nice visual representation on what is going on.



I spent a lot of time trying to understand all this R code but there are too many errors to get it to run correctly in R. I understand what should be happening but the program is very fussy about missing files.

library(sp)

library(plotrix)

library(raster)

library(rgeos)

library(rgdal)

setwd("C:/Visualize Earthquake")

#http://earthquake.usgs.gov/earthquakes/feed/v1.0/csv.php

#From this address we can download csv files with locations of Earthquakes

#For this experiment we will download the summary of the last 30 days

#This dataset is updated every 15 minutes so your output may differ from the following

URL <- "http://earthquake.usgs.gov/earthquakes/feed/v1.0/summary/all\_month.csv"

Earthquake\_30Days <- read.table(URL, sep = ",", header = T)

#Check the data.frame dimensions

nrow(Earthquake\_30Days)

ncol(Earthquake\_30Days)

#We can check the contents of the table with the following function

str(Earthquake\_30Days)

#The function names() is used to generate a list of the column names of a data.frame

names(Earthquake\_30Days)

#Now we can transform the data.frame into a Spatial object

coordinates(Earthquake\_30Days)=~longitude+latitude

str(Earthquake\_30Days)

#Slot data

str(Earthquake\_30Days@data)

#Check and set the projection

Earthquake\_30Days@proj4string

#We can set the projection to WGS84 using the following line

projection(Earthquake\_30Days)=CRS("+init=epsg:4326")

#http://spatialreference.org/ref/epsg/wgs-84/

#Downloading polygons with the border of each country

#Download, unzip and load the polygon shapefile with the countries' borders

download.file("http://thematicmapping.org/downloads/TM\_WORLD\_BORDERS\_SIMPL-0.3.zip",destfile="TM\_WORLD\_BORDERS\_SIMPL-0.3.zip")

unzip("TM\_WORLD\_BORDERS\_SIMPL-0.3.zip",exdir=getwd())

polygons <- shapefile("TM\_WORLD\_BORDERS\_SIMPL-0.3.shp")

#Let us check the time variable

Earthquake\_30Days$time[1]

#The time variable has the following format: year-month-dayThour:minutes:second.milliseconds

#Now that we understood the format we can change this variable from a factor to a time stamp

conv.time <- function(vector){

split1 <- strsplit(paste(vector),"T")

split2 <- strsplit(split1[[1]][2],"Z")

fin <- paste0(split1[[1]][1],split2[[1]][1])

paste(as.POSIXlt(fin,formate="%Y-%m-%d%H:%M:%OS3"))

}

conv.time(Earthquake\_30Days$time[1])

DT <- sapply(Earthquake\_30Days$time,FUN=conv.time)

Earthquake\_30Days$DateTime <- as.POSIXlt(DT)

#####################################

#Plot Color:Days - Size:Magnitude

#Color Scale

days.from.today <- round(c(Sys.time()-Earthquake\_30Days$DateTime)/60,0)

colors.DF <- data.frame(days.from.today,color.scale(days.from.today,color.spec="rgb",extremes=c("red","blue")))

colors.DF <- colors.DF[with(colors.DF, order(colors.DF[,1])), ]

colors.DF$ID <- 1:nrow(colors.DF)

breaks <- seq(1,nrow(colors.DF),length.out=10)

#Size scale

size.DF <- data.frame(Earthquake\_30Days$mag,Earthquake\_30Days$mag/5)

size.DF <- size.DF[with(size.DF, order(size.DF[,1])), ]

size.DF$ID <- 1:nrow(size.DF)

breaks.size <- seq(0,max(Earthquake\_30Days$mag/5),length.out=5)

#Save plot in JPEG

tiff(filename="Earthquake\_Map.tif",width=7000,height=4000, res=300)

#Plot

plot(polygons)

plot(Earthquake\_30Days, col= colour.scale, cex=Earthquake\_30Days$mag/5, pch=16, add=T)

#Title and Legend

title("Earthquakes in the last 30 days",cex.main=3)

legend.pos <- list(x=-28.52392,y=-20.59119)

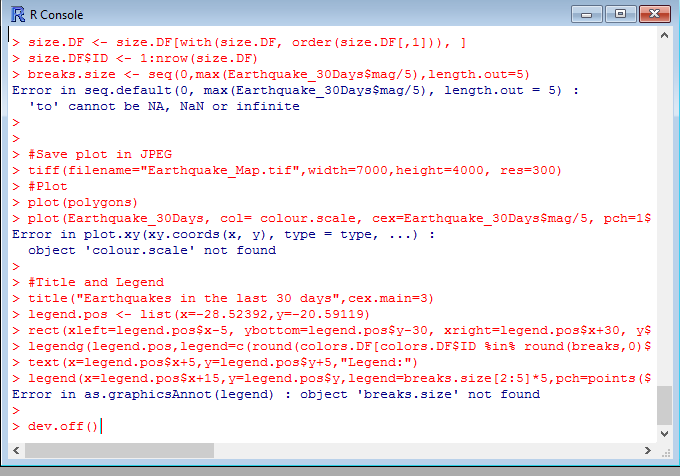
rect(xleft=legend.pos$x-5, ybottom=legend.pos$y-30, xright=legend.pos$x+30, ytop=legend.pos$y+10, col="white", border=NA)

legendg(legend.pos,legend=c(round(colors.DF[colors.DF$ID %in% round(breaks,0),1],2)),fill=paste(colors.DF[colors.DF$ID %in% round(breaks,0),2]),bty="n",bg=c("white"),y.intersp=0.75,title="Depth",cex=0.8)

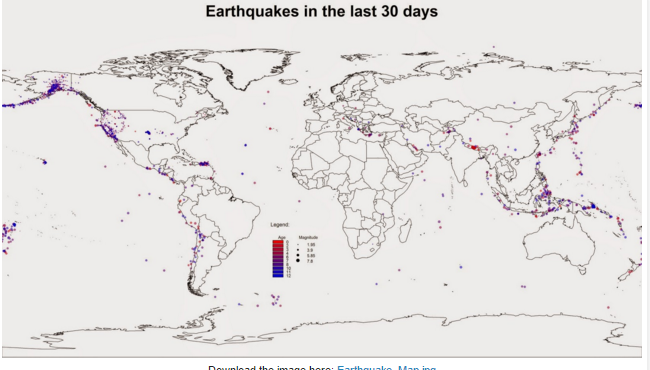
text(x=legend.pos$x+5,y=legend.pos$y+5,"Legend:")

legend(x=legend.pos$x+15,y=legend.pos$y,legend=breaks.size[2:5]\*5,pch=points(rep(legend.pos$x+15,4),c(legend.pos$y-6,legend.pos$y-9,legend.pos$y-12,legend.pos$y-15),pch=16,cex=breaks.size[2:5]),cex=0.8,bty="n",bg=c("white"),y.intersp=1.1,title="Magnitude")

dev.off()



Attempting to change and run the code did not get me anywhere and I was not able to replicate what was supposed to happen



So I got to another article and tried modifying/simplifying some code to get similar results.

library(sp)

library(plotrix)

library(raster)

library(rgeos)

library(rgdal)

library(scatterplot3d)

setwd("C:/Visualize Earthquake")

URL <- "http://earthquake.usgs.gov/earthquakes/feed/v1.0/summary/all\_month.csv"

Earthquake\_30Days <- read.table(URL, sep = ",", header = T)

#Download, unzip and load the polygon shapefile with the countries' borders

download.file("http://thematicmapping.org/downloads/TM\_WORLD\_BORDERS\_SIMPL-0.3.zip",destfile="TM\_WORLD\_BORDERS\_SIMPL-0.3.zip")

unzip("TM\_WORLD\_BORDERS\_SIMPL-0.3.zip",exdir=getwd())

polygons <- shapefile("TM\_WORLD\_BORDERS\_SIMPL-0.3.shp")

dir.create(paste(getwd(),"/GeologicalData",sep=""))

#Faults

download.file("http://legacy.jefferson.kctcs.edu/techcenter/gis%20data/World/Zip/FAULTS.zip",destfile="GeologicalData/FAULTS.zip")

unzip("GeologicalData/FAULTS.zip",exdir="GeologicalData")

faults <- shapefile("GeologicalData/FAULTS.SHP")

#Plates

download.file("http://legacy.jefferson.kctcs.edu/techcenter/gis%20data/World/Zip/PLAT\_LIN.zip",destfile="GeologicalData/plates.zip")

unzip("GeologicalData/plates.zip",exdir="GeologicalData")

plates <- shapefile("GeologicalData/PLAT\_LIN.SHP")

#Volcano

download.file("http://legacy.jefferson.kctcs.edu/techcenter/gis%20data/World/Zip/VOLCANO.zip",destfile="GeologicalData/VOLCANO.zip")

unzip("GeologicalData/VOLCANO.zip",exdir="GeologicalData")

volcano <- shapefile("GeologicalData/VOLCANO.SHP")

Earthquakes <- Earthquake\_30Days[paste(Earthquake\_30Days$type)=="earthquake",]

coordinates(Earthquakes)=~longitude+latitude

jpeg("Earthquake\_Origin.jpg",4000,2000,res=300)

plot(plates,col="red")

plot(polygons,add=T)

title("Earthquakes in the last 30 days",cex.main=3)

lines(faults,col="dark grey")

points(Earthquakes,col="blue",cex=0.5,pch="+")

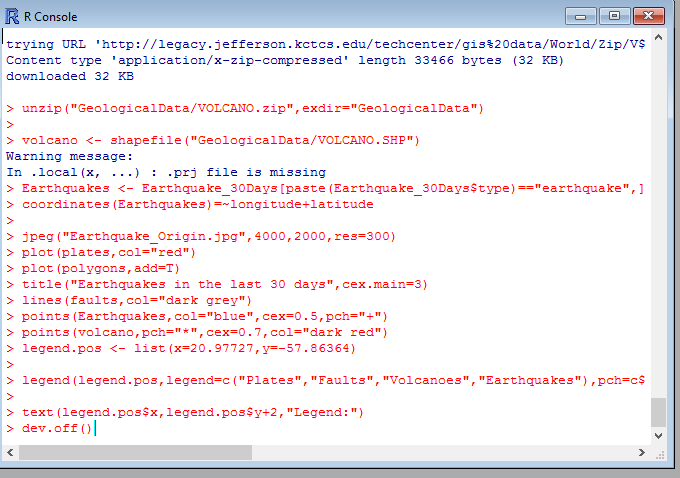
points(volcano,pch="\*",cex=0.7,col="dark red")

legend.pos <- list(x=20.97727,y=-57.86364)

legend(legend.pos,legend=c("Plates","Faults","Volcanoes","Earthquakes"),pch=c("-","-","\*","+"),col=c("red","dark grey","dark red","blue"),bty="n",bg=c("white"),y.intersp=0.75,title="Days from Today",cex=0.8)

text(legend.pos$x,legend.pos$y+2,"Legend:")

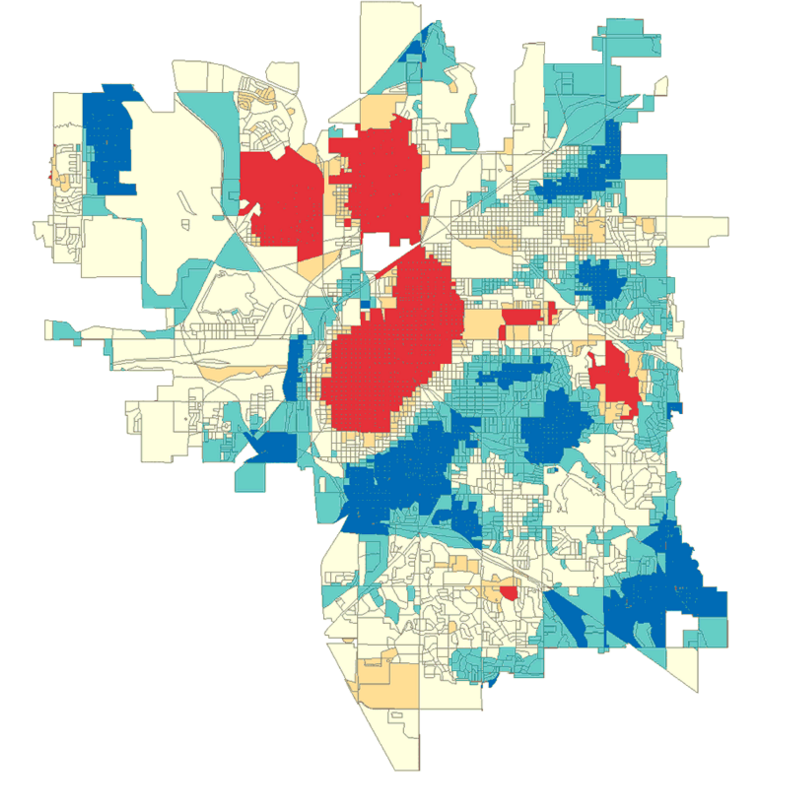
dev.off()



…And I get a missing .prj file which I cannot locate anywhere.

I am still going to keep trying to get this to work because I feel this technique of analyzing the data will be a useful way to apply to other types of data.

This link (<https://data.lacity.org/api/views/ttiz-7an8/rows.csv?accessType=DOWNLOAD>) has live access to crime in the Los Angeles area. I should be able to load a LA County map and overlay crime occurrences and color code by types of crime.



This is something that ESRI has up for crime using GIS data. We should be able to get close to this in R.