Code Violations GIS Visualization

Rory Tikalsky & Skyler Halbritter

March 9, 2016

```
setwd(C:/Users/Rory/Google Drive/Class/Data-Driven Management II/GIS project/01-05-2015)
```

Visualizing Using Point Mapping

Map Basics

```
library(ggmap)
```

```
## Warning: package 'ggmap' was built under R version 3.2.3
```

```
## Loading required package: ggplot2
```

```
## Warning: package 'ggplot2' was built under R version 3.2.3
```

```
syracuse <- get_map(location="Syracuse NY", zoom = 13, color="bw")
```

```
## Map from URL : http://maps.googleapis.com/maps/api/staticmap?center=Syracuse+NY&zoom=13&size=640x640&scale=2&maptype=terrain&language=en-EN&sensor=false
## Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Syracuse%20NY&sensor=false
```

```
syr.map <- ggmap(syracuse, extent = "device")
```

This code was originally used to create the dataframes from the original file. Because of the length of geocoding, this was not run for the R Markdown. Instead, outputs were saved in a dataframe and converted to new CSV files.

---

dat2 <- read.csv(“new data.csv”)

Open Violations open.dat <- dat2[dat2$Complaint.Status == “Open”,]

Open Bed Bug Violations open.bedbugs <- open.dat[open.dat$Complaint.Type == “Bed Bugs”,] open.bedbugs[“fulladdress”] <- NA open.bedbugs$fulladdress < -paste(open.bedbugsAddress, “Syracuse, NY”, sep=" “) long.lat.bugs <- geocode(open.bedbugs$fulladdress) open.bedbugs <- cbind(open.bedbugs, long.lat.bugs)

All Bed Bug Violations bed.bugs.all <- dat2[dat2$Complaint.Type == “Bed Bugs”,] bed.bugs.all[“fulladdress”] <- NA bed.bugs.all$fulladdress < -paste(bed.bugs.allAddress, “Syracuse, NY”, sep=" “) long.lat.all.bugs <- geocode(bed.bugs.all$fulladdress) bed.bugs.all <- cbind(bed.bugs.all, long.lat.all.bugs)

Unsafe conditions unsafe.dat <- dat2[dat2$Complaint.Type == “Unsafe Conditions”,] unsafe.dat[“fulladdress”] <- NA unsafe.dat$fulladdress < -paste(unsafe.datAddress, “Syracuse, NY”, sep=" “) long.lat.unsafe <- geocode(unsafe.dat$fulladdress) unsafe.dat <- cbind(unsafe.dat, long.lat.unsafe)

Add color vector to unsafe conditions col.vec <- rep(NA, nrow(unsafe.dat)) col.vec <- ifelse(unsafe.dat$Violation.Status == “Open”, “orange”, “blue”)

Load new dataset (and Subset and Geocode – see above code)

```
open.bedbugs <- read.csv("openbedbugs.csv")
bed.bugs.all <- read.csv("bedbugsall.csv")
unsafe.dat <- read.csv("unsafe.csv")

#add color vector to unsafe conditions
col.vec <- rep( NA, nrow(unsafe.dat) )
col.vec <- ifelse( unsafe.dat$Violation.Status == "Open", "orange", "blue")
```

Load Multiplot Function

This function is not in the ggmap package but is widely used for it.

```
multiplot <- function(..., plotlist=NULL, file, cols=1, layout=NULL) {
  library(grid)

  # Make a list from the ... arguments and plotlist
  plots <- c(list(...), plotlist)

  numPlots = length(plots)

  # If layout is NULL, then use 'cols' to determine layout
  if (is.null(layout)) {
    # Make the panel
    # ncol: Number of columns of plots
    # nrow: Number of rows needed, calculated from # of cols
    layout <- matrix(seq(1, cols * ceiling(numPlots/cols)),
                     ncol = cols, nrow = ceiling(numPlots/cols))
  }

  if (numPlots==1) {
    print(plots[[1]])

  } else {
    # Set up the page
    grid.newpage()
    pushViewport(viewport(layout = grid.layout(nrow(layout), ncol(layout))))

    # Make each plot, in the correct location
    for (i in 1:numPlots) {
      # Get the i,j matrix positions of the regions that contain this subplot
      matchidx <- as.data.frame(which(layout == i, arr.ind = TRUE))

      print(plots[[i]], vp = viewport(layout.pos.row = matchidx$row,
                                      layout.pos.col = matchidx$col))
    }
  }
}
```

Plot All 3 Maps

```
multiplot(
syr.map + geom_point(data=open.bedbugs, aes(x=lon, y=lat), size=3, col="orange", alpha=.
8) +
  ggtitle("Open Bed Bug Violations") +
  theme(plot.title=element_text(family="Times", face="bold", size=12)),
syr.map + geom_point(data=bed.bugs.all, aes(x=lon, y=lat), size=3, col="orange", alpha=.
4) +
  ggtitle("All Bed Bug Violations 2012-Present") +
  theme(plot.title=element_text(family="Times", face="bold", size=12)),
syr.map + geom_point(data=unsafe.dat, aes(x=lon, y=lat), size=3, col=col.vec, alpha=.4) +
  ggtitle("All Unsafe Condition Violations 2012-Present") +
  theme(plot.title=element_text(family="Times", face="bold", size=12)),
layout=matrix(c(1,2,3,3), nrow=2, byrow=TRUE)
)
```

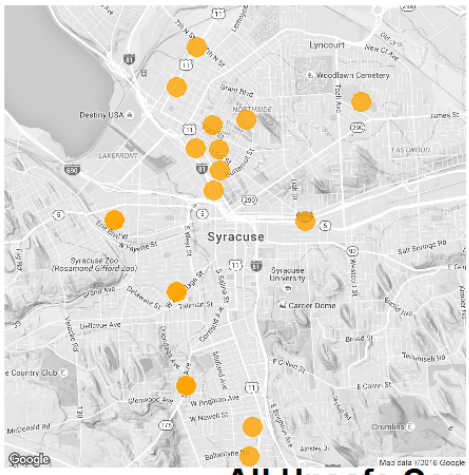
```
## Warning: Removed 1 rows containing missing values (geom_point).
```

```
## Warning in grid.Call(L_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database
```

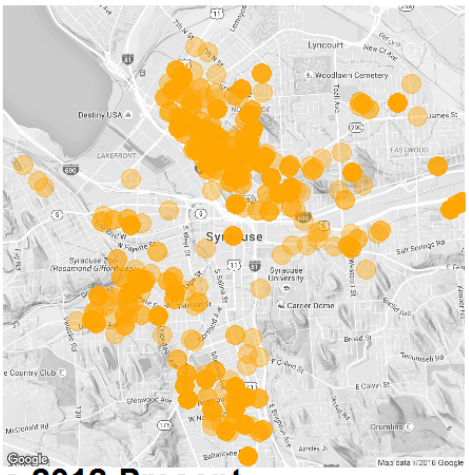
```
## Warning: Removed 103 rows containing missing values (geom_point).
```

```
## Warning: Removed 104 rows containing missing values (geom_point).
```

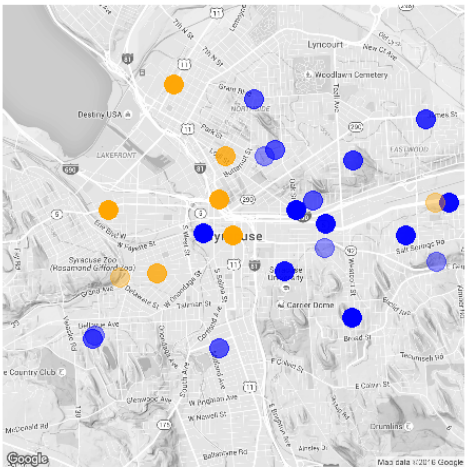
Open Bed Bug Violations



All Bed Bug Violations 2012-Present



All Unsafe Condition Violations 2012-Present



Visualizing Using Choropleth Mapping

```
require(maptools)
```

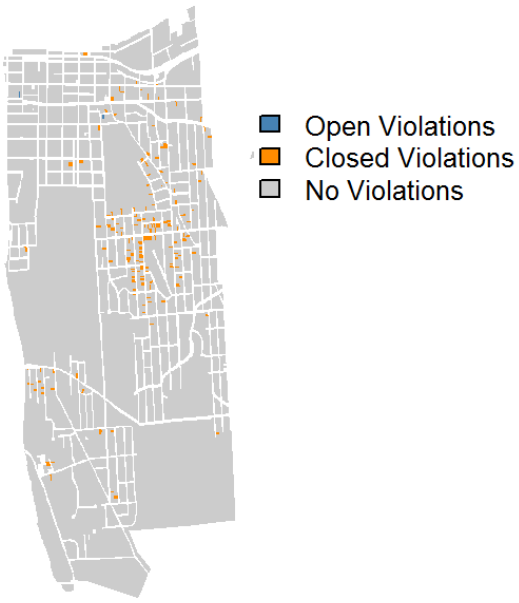
```
## Loading required package: maptools
```

```
## Warning: package 'maptools' was built under R version 3.2.3
```

```
## Loading required package: sp
## Checking rgeos availability: FALSE
##      Note: when rgeos is not available, polygon geometry     computations in maptools
depend on gpclib,
##      which has a restricted licence. It is disabled by default;
##      to enable gpclib, type gpclibPermit()
```

```
require(sp)

# get your data sets
syr <- readShapePoly( fn="01-05-2015", proj4string=CRS("+proj=longlat +datum=WGS84") )

violations <- read.csv("Violation Report with lot ID.csv")

#cast as numeric
syr$SBL <- as.character(syr$SBL)
violations$Identifier <- as.character(violations$Identifier)

#merge them
syr@data <- data.frame(syr@data, violations[match(syr@data[, "SBL"], violations[, "Identi
fier"]), ])

#########################################################################
############### Trash Violations in University Area ###############
#########################################################################

#subset for university area
univ <- syr[which(syr@data$Nhood=="Westcott"| syr@data$Nhood=="University Neighborhood" |
syr@data$Nhood=="University Hill" | syr@data$Nhood=="Near Eastside" | syr@data$Nhood=="So
uth Campus" | syr@data$Nhood=="Outer Comstock"), ]

#create color vector of open violations, closed violations and no violations
col.vec <- ifelse( univ@data$Complaint.Type=="Trash/Debris-Private, Occ" & univ@data$Viol
ation.Status=="Open", "steel blue", "gray80")
col.vec <- ifelse( univ@data$Complaint.Type=="Trash/Debris-Private, Occ" & univ@data$Viol
ation.Status=="Closed", "dark orange", col.vec)
col.vec[ is.na(col.vec) ] <- "gray80"

#plot and create legend
plot( univ, col=col.vec, border=FALSE, main="Trash Violations near Syracuse University (2
012-2015)", cex.main=1.4)
legend(x=626459, y=1110819, legend= c("Open Violations", "Closed Violations", "No Violati
ons"), fill=c("steel blue","dark orange", "gray80"), bty="n", cex=.8)
```

Trash Violations near Syracuse University (2012-2015)




```
################################################################################
############## Overgrowth Violations in University Area #################
################################################################################

#create color vector of open violations, closed violations and no violations
col.vec <- ifelse( univ@data$Complaint.Type=="Overgrowth: Private, Occ" & univ@data$Viola
tion.Status=="Open", "steel blue", "gray80")
col.vec <- ifelse( univ@data$Complaint.Type=="Overgrowth: Private, Occ" & univ@data$Viola
tion.Status=="Closed", "dark orange", col.vec)
col.vec[ is.na(col.vec) ] <- "gray80"

#plot and create legend
plot( univ, col=col.vec, border=FALSE, main="Overgrowth Violations Near Syracuse Universi
ty (2012-2015)", cex.main=1.4)
legend(x=626459, y=1110819, legend= c("Open Violations", "Closed Violations", "No Violati
ons"), fill=c("steel blue","dark orange", "gray80"), bty="n", cex=.8)
```

Overgrowth Violations Near Syracuse University (2012-2015)