Nicholas Lytal
ID: 991834259
STA 250
3/18/14

ASSIGNMENT #4: Data Visualization

Introduction

When explaining a project or bridging a knowledge gap, visualization is a critical part of conveying your message and research. Properly constructed graphics, even static ones, can reveal much more information than a well-crafted page or two of technical details. However, sometimes even static plots cannot adequately display all facets of the data in question. We must consider interactive plots when managing data that involves several important variables at once, all of which would not fit on a single static plot. In addition to displaying more variables, interactivity can let us analyze a single point in particular more effectively, providing extra information upon clicking or rolling our cursor over it.

As an example of data visualization, we consider a geolocation data set that measures wireless signal reception in a building depending on a person's position, their orientation, the router receiving the signal, and so forth. The data gives us an effective way to predict someone's location in the building depending on the signal strength of their device, which could be expanded beyond just a single building in future experiments. The variables involved are too many for a single plot to describe, so we approach the data through interactive plots to display data more effectively. In this report, we consider the main six access points in the building when measuring signal strength.

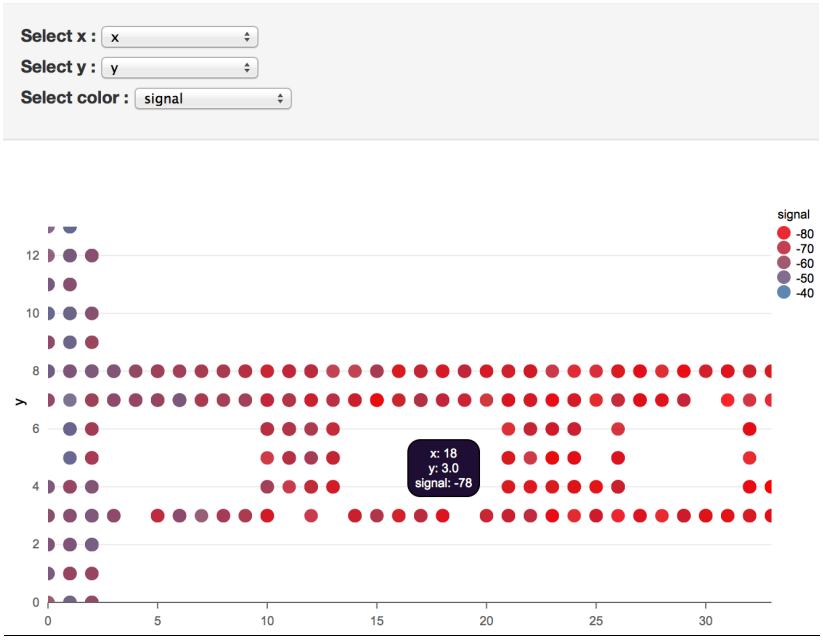
Methods

First, we divide the data into different groups depending on the router in question. With a single static plot, we could display the position and signal strength for a single router's data, but with an interactive plot we can allow users to easily switch among different routers' observations using the same plotting space. Here, signal strength, in dB, is stronger the larger the number is (so -40 is stronger than -60).

By using the rCharts package in R, we can get a single plot and alter the HTML code as necessary to add other touches to the existing plot. From what we can see, even without knowing the locations of the routers, we can reasonably determine them based on the existing signal data.

As an alternative to simply using the pre-made plots in rCharts, we can use the "SVGAnnotation" and "gridSVG" packages to craft necessary code from scratch to run the data. The resulting Javascript object can be described in R and converted to JSON to be reconstructed.

The resulting plots can be constructed using the code provided, but an example of a static snapshot is seen on the next page:



Complications

One of the most difficult tasks when experimenting with data visualization methods was properly separating the data to obtain a plottable form. With the presence of many overlapping data points, even for the same router, not to mention the eight different orientation groups for each x-y coordinate, there is a lot of separation necessary to make a plot of such data remotely readable and interpretable. Since each location/orientation/router combination has several dozen measurements, with no consistent number of samples among the measurements as well, it is more difficult to take an appropriate average of signal strength for the measurements in each such combination. This data processing is necessary before we can even construct a plot, as the grobs (graphical objects) involved must be clearly defined, and not merely dozens of overlapping points that cannot be discerned by a mouse-over.

Even with the grobs properly averaged and combined, separating the data into individual plots that can be accessed with a slider posed another challenge. There didn't appear to exist a good way to recalculate the data properly for each individual router in such a way that would include all possibilities. Given more time, I would explore SVG notation much more thoroughly to understand how to plot and access grobs without relying on rCharts to compose the majority of the code automatically. Using shiny to create webpages for the charts in question would be worth considering as well. Although rCharts is a useful package, the plots it produces still only consider 3 variables at a time by default, thus not allowing a simultaneous plot of signal and orientation. It doesn't appear possible to plot an rChart object on top of an existing image either, meaning we can't have the actual floor plan for a reference as far as locations are concerned.

R Code Appendix

```
# STA 250 Project 4 Notes

setwd("~/Dropbox/Work/STA 250 Files & Project/STA250_HW4_Files")

# Load in offline and AccessPointLocations files

off.samp = offline[sample(nrow(offline),size=5000,replace=FALSE),]

test.plot = rPlot(y~x, color = "channelFrequency",
                  data = offline.samp, type = "point")

# NOTE: The picture in question's boundaries are
# roughly x:(0.5,33), y:(-3,14)

# These are the six access points we care about
# and their physical coordinates
rownames(AccessPointLocations)
apl.x = AccessPointLocations[,1]
apl.y = AccessPointLocations[,2]

# Rough idea of boundaries for the data
summary(offline$x)
summary(offline$y)

# Let's take our building image as a background.
# This confirms the coordinates we're working with, more or less
library('png')
bg <- readPNG("building.png") # If picture is in current dir
plot(1:2, type='n', main="Building Background",
     xlab="x", ylab="y", xlim=c(0.5,33), ylim=c(-3,14))
lim <- par()
rasterImage(bg, lim$usr[1], lim$usr[3], lim$usr[2], lim$usr[4])
grid()

points(apl.x,apl.y, col = "red", pch = 15)


# GOAL: Need a separate plot for each router
# so want to split the data accordingly

# Also want to consider the presence of multiple entries at the same point
# and orientation

# Separate plots for signal and for orientation?

# Names of the six routers we want to consider
routers = rownames(AccessPointLocations)


# Data for each router, separated
router1 = offline[which(offline$mac==routers[1]),]
router2 = offline[which(offline$mac==routers[2]),]
router3 = offline[which(offline$mac==routers[3]),]
router4 = offline[which(offline$mac==routers[4]),]
router5 = offline[which(offline$mac==routers[5]),]
router6 = offline[which(offline$mac==routers[6]),]

# offline data for only the six desired routers
router.all = rbind(router1,router2,router3,router4,router5,router6)
```

```
# Interactive plots with the assignment data

# Example plot
plot1 = rPlot(y ~ x, data = router1, color="signal", type = "point",
              main = "Signal Data")

# Alternatively, enter:
# router: router (router1, router2, etc.)
# router.all: As defined above. Only used as universal source for variable names
# This will produce six interactive plots, one for each router.
all.plots = function(router, router.all)
{
    plot1 = rPlot(y ~ x, data = router, color="signal", type = "point",
                  main = "Signal Data")

    # Adds simple menus to choose which variables to use for x, y, and color
    # (signal, orientation, x) and (signal, orientation, y) can tell us a bit
    # more about the data in question.
    plot1$addControls("x", value = "x", values = names(router.all))
    plot1$addControls("y", value = "y", values = names(router.all))
    plot1$addControls("color", value = "signal", values = names(router.all))

    print(plot1)

}

# Additional notes
plot1$show('inline') # Displays javascript
print(plot1) # Displays graph
plot1$print("chart1")

# Publishes the plot for public viewing on Rpubs
plot1$publish('Scatterplot', host = 'rpubs')
```