HW3

Jordan Hilton

April 16, 2019

Let's load our data. I'm going to assign the mtcars dataset to "d" so that the lessR defaults will work.

```
d <- mtcars
```

1 Categorical with continuous variables.

a.

Here's the head of our data:

```
head(d)
```

```
##                    mpg cyl disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
## Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

b.

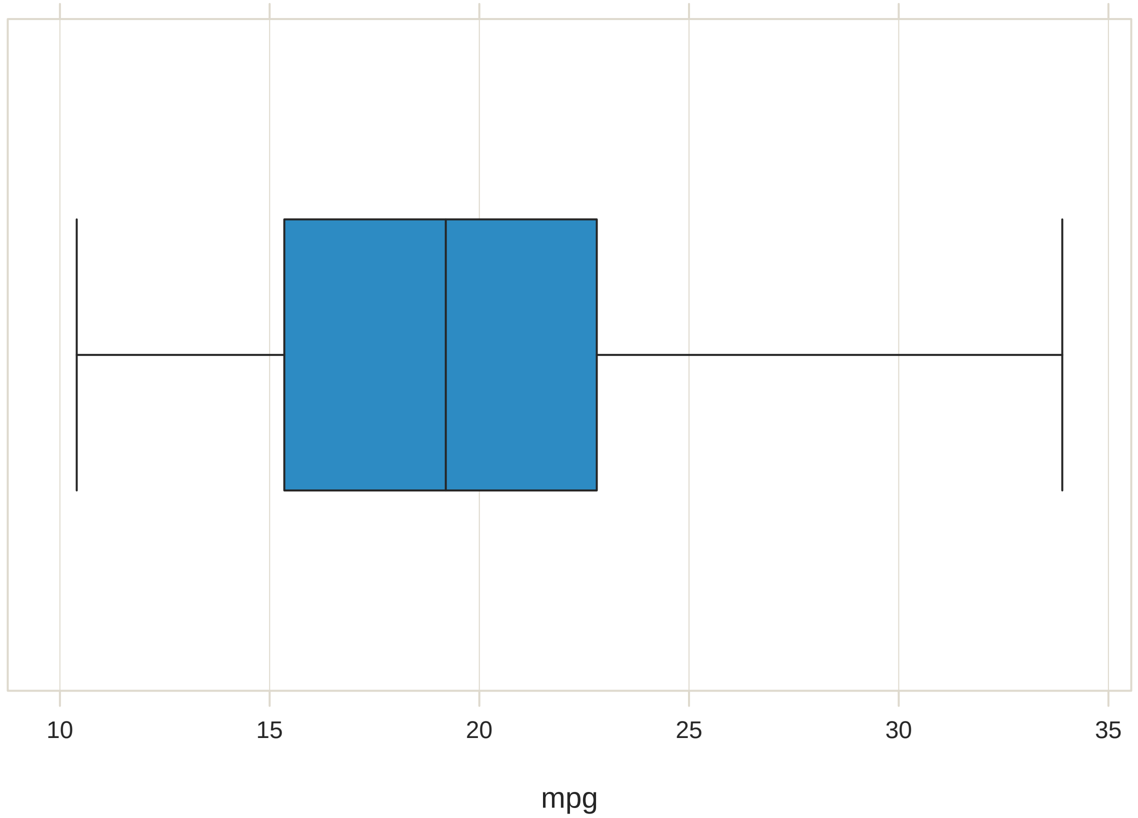
It looks like MPG, displacement, horsepower, rear axle ratio, weight, and "qsec" are continuous, while the number of cylinders, the engine shape "vs", the transmission "am", the number of forward gears, and the number of carburetors are categorical. All of the data are currently of type double, so let's go ahead and transform the categorical variables to factors now.

```
d<-Transform(cyl=factor(cyl), quiet=TRUE)
d<-Transform(vs=factor(vs), quiet=TRUE)
d<-Transform(am=factor(am), quiet=TRUE)
d<-Transform(carb=factor(carb), quiet=TRUE)
```

c.

Here’s a box plot for mpg:

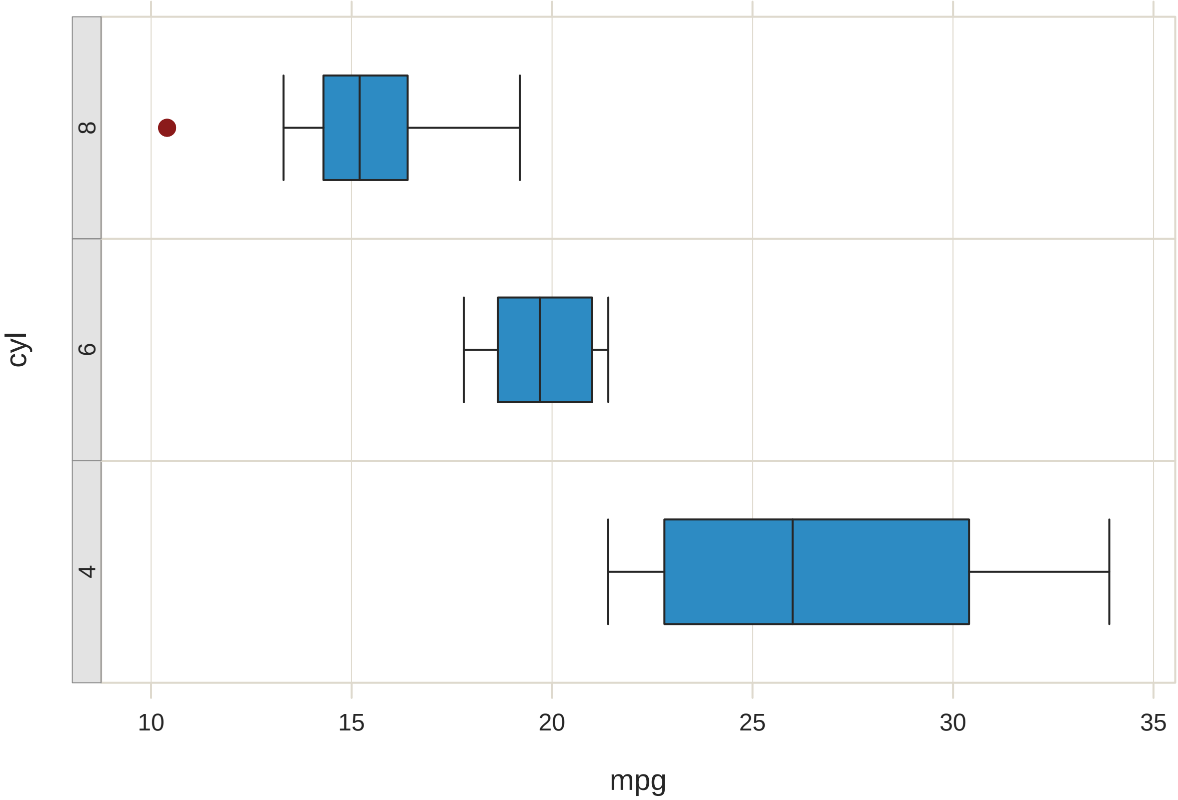
```
BoxPlot(mpg, quiet=TRUE)
```



d.

Here's a box plot for mpg at each level of cyl:

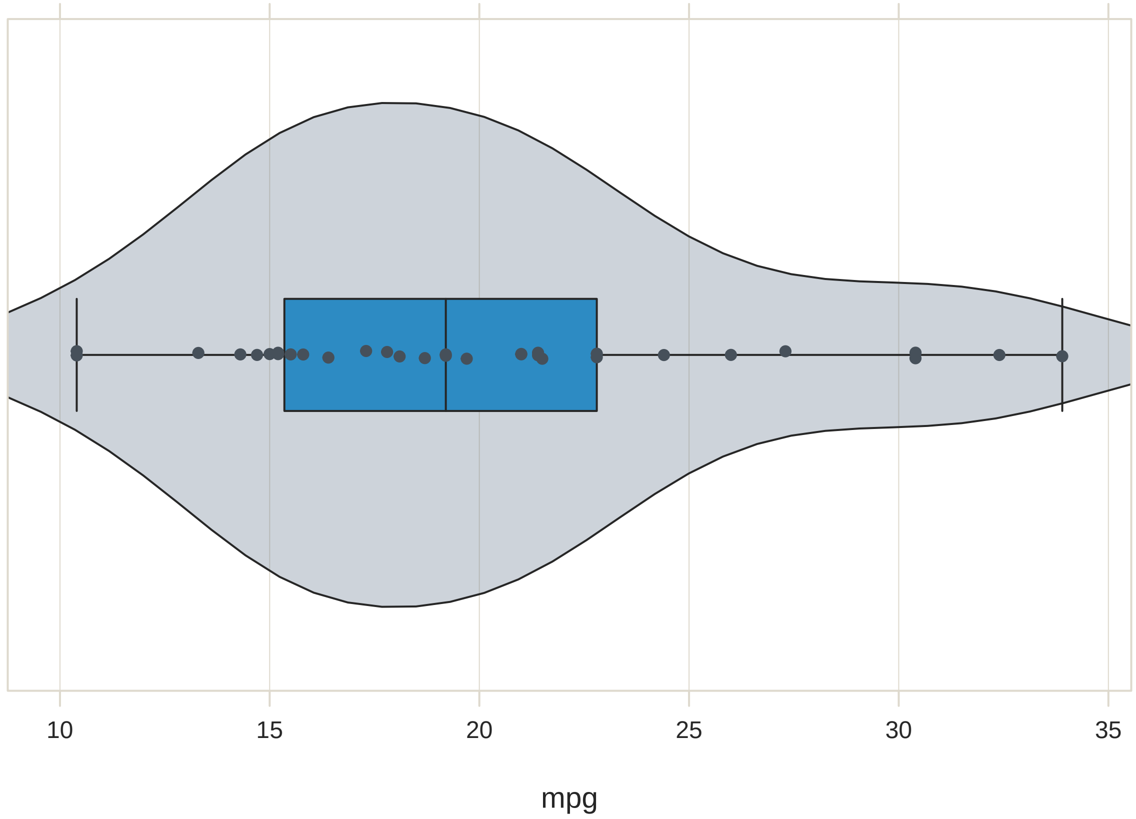
```
BoxPlot(mpg, by1=cyl, quiet=TRUE)
```



e.

Here's a VBS plot of mpg:

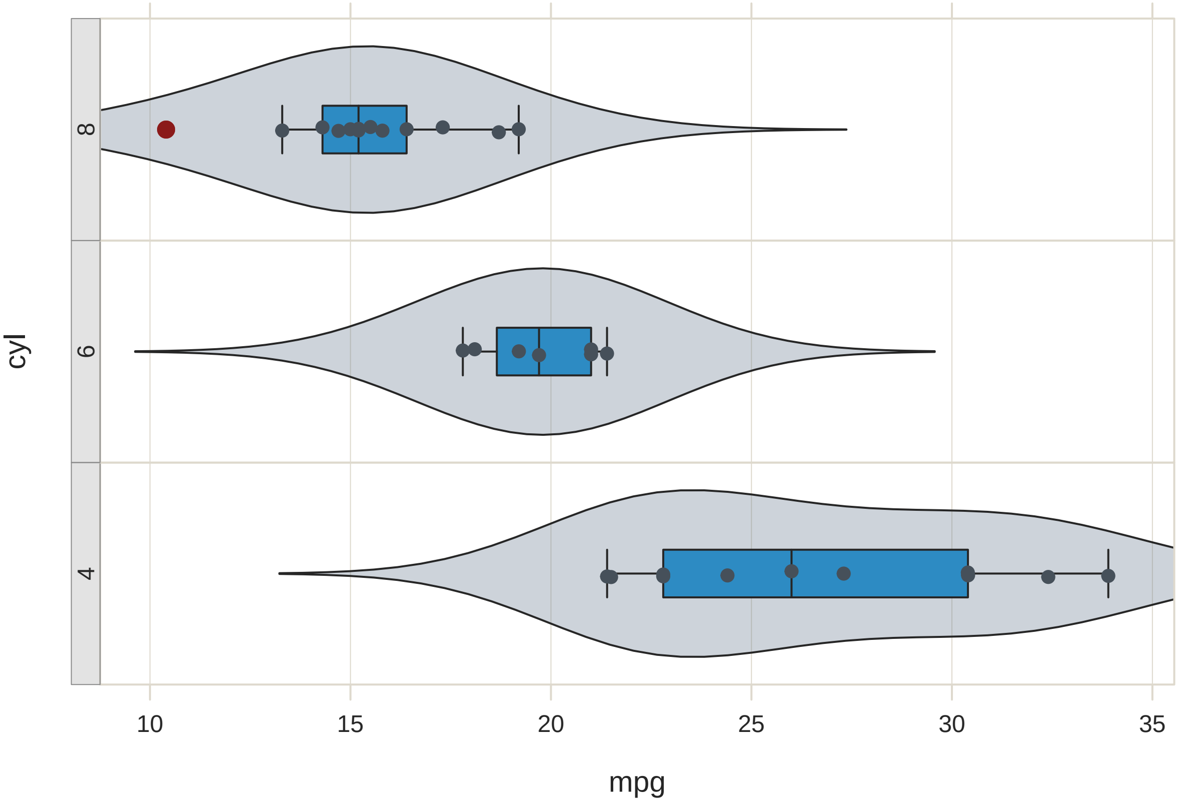
```
Plot(mpg,  quiet=TRUE)
```



f.

Here's a VBS plot of mpg at each level of cyl:

```
Plot(mpg, by1=cyl, quiet=TRUE)
```



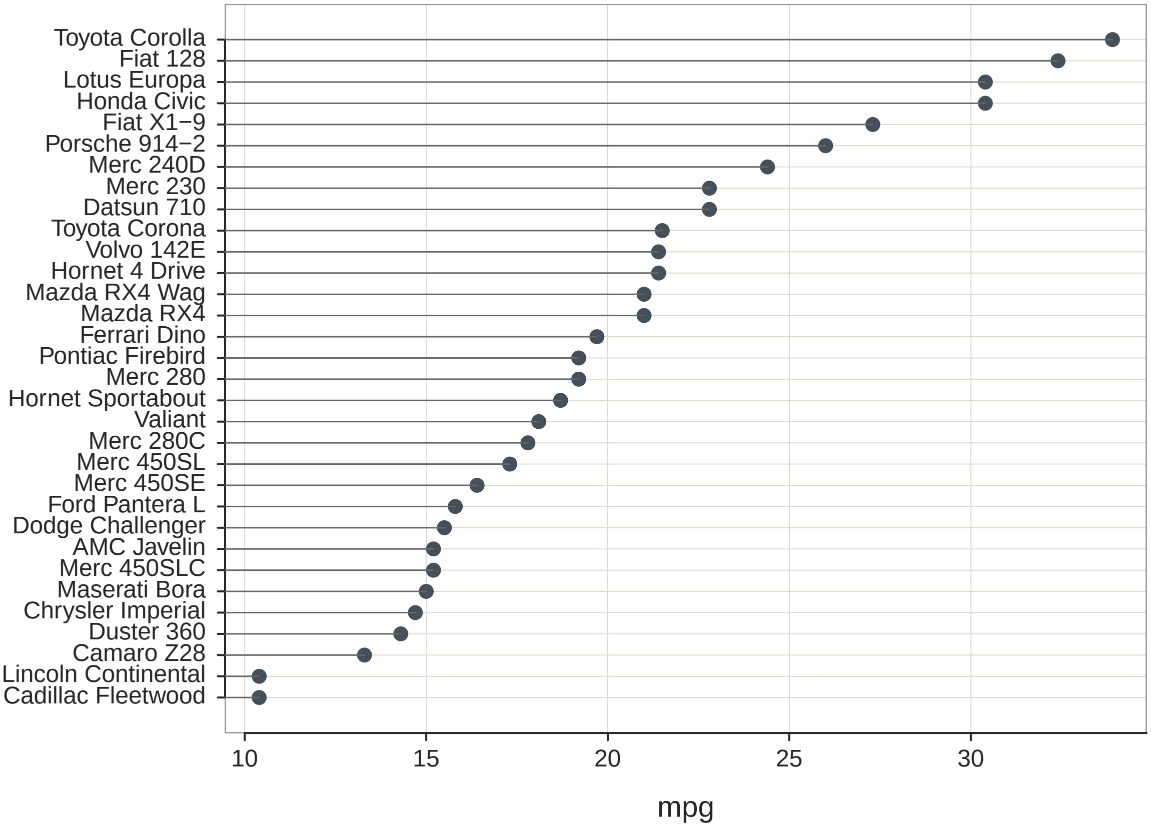
g.

The VBS plot includes the location of specific points with color coding so that we can more easily hunt outliers (in this example you're immediately curious what that 10.4 MPG monstrosity is. . . .I checked and it's a Cadillac), and the violin plot helps interpret the width of the distribution also shown by the box plot. The box plot however is cleaner and simpler.

h.

Here's a Cleveland dot plot for mpg:

```
Plot(mpg, row.names, quiet=TRUE)
```



It looks like a Corolla is around 3x more fuel efficient than a Cadillac Fleetwood.

2. Asymmetric Distribution Box Plot

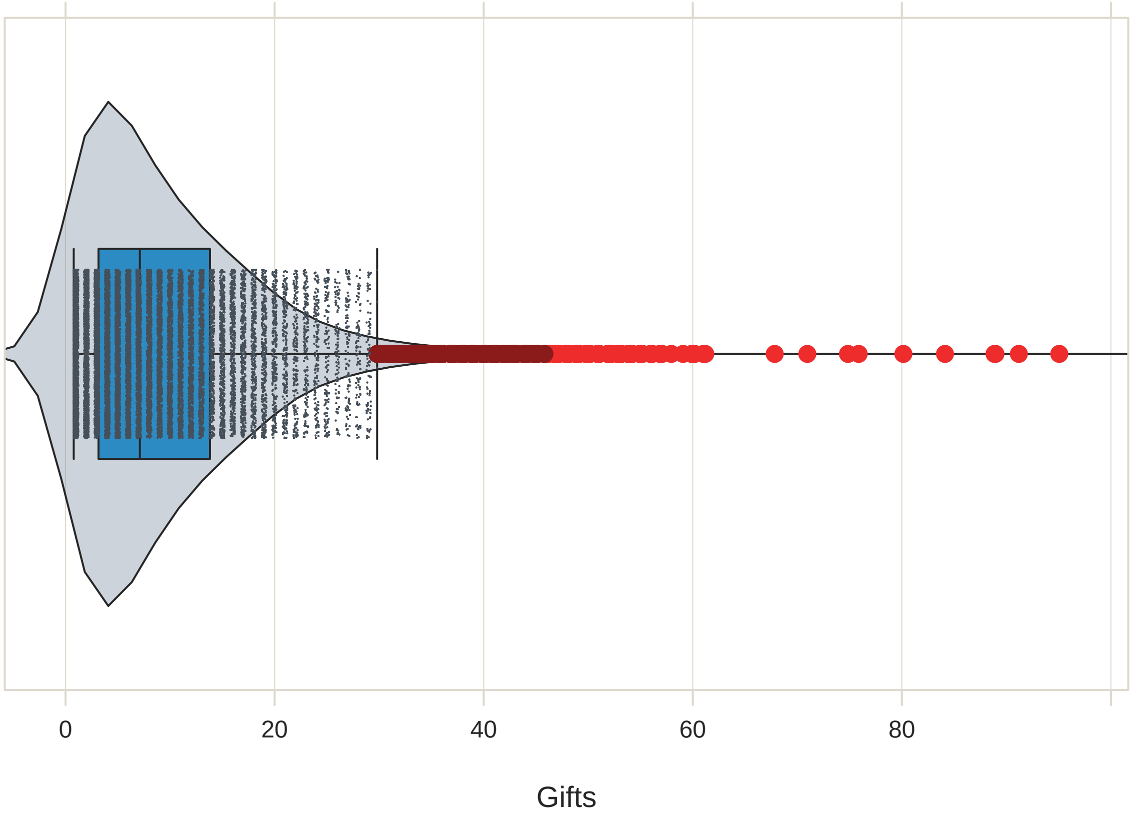
Let's load the donations data:

```
d<-Read("Donations.csv", quiet=TRUE)
```

a.

Here's a VBS of gifts:

```
Plot(Gifts, quiet=TRUE)
```



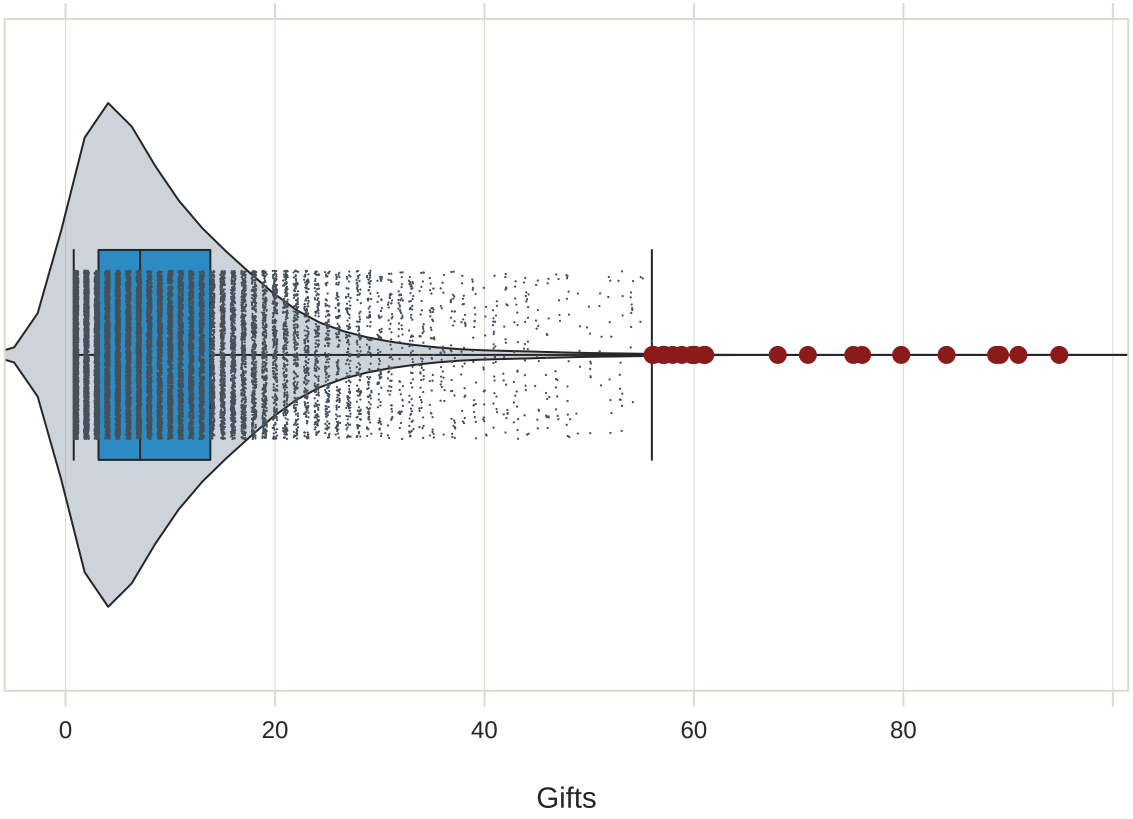
b.

There are a lot of outliers because this is a skewed distribution- most rows are donors who give between 0 and 25 gifts, but there are quite a few rows of donors who give a large number of gifts well separated from the mean.

c.

Here's the adjusted plot:

```r
Plot(Gifts, box.adj=TRUE, quiet=TRUE)
```



d.

To adjust the plot, I set `box.adj=TRUE`, which if I'm understanding the VBS paper correctly changes the central measure to the median and uses an adjusted version of the standard deviation. For skewed distributions like this one, this is the better plot.

# 3. Maps

I'm using Azerbaijan with country code AZ, reading in the .txt file from geonames.org here. The "complete.cases" row is to remove some pesky missing value rows at the end of my data; I suspect there was some data loading problem due to encoding of city names.

```
d<-Read("cities1000.txt", col.names = c("id","name","ascii_name", "alt_names","latitude","longitude","fe
```

```
## >>> A tab character detected in the first row of the
##      data file. Presume tab delimited data.

## Warning in scan(file = file, what = what, sep = sep, quote = quote, dec =
## dec, : EOF within quoted string
```

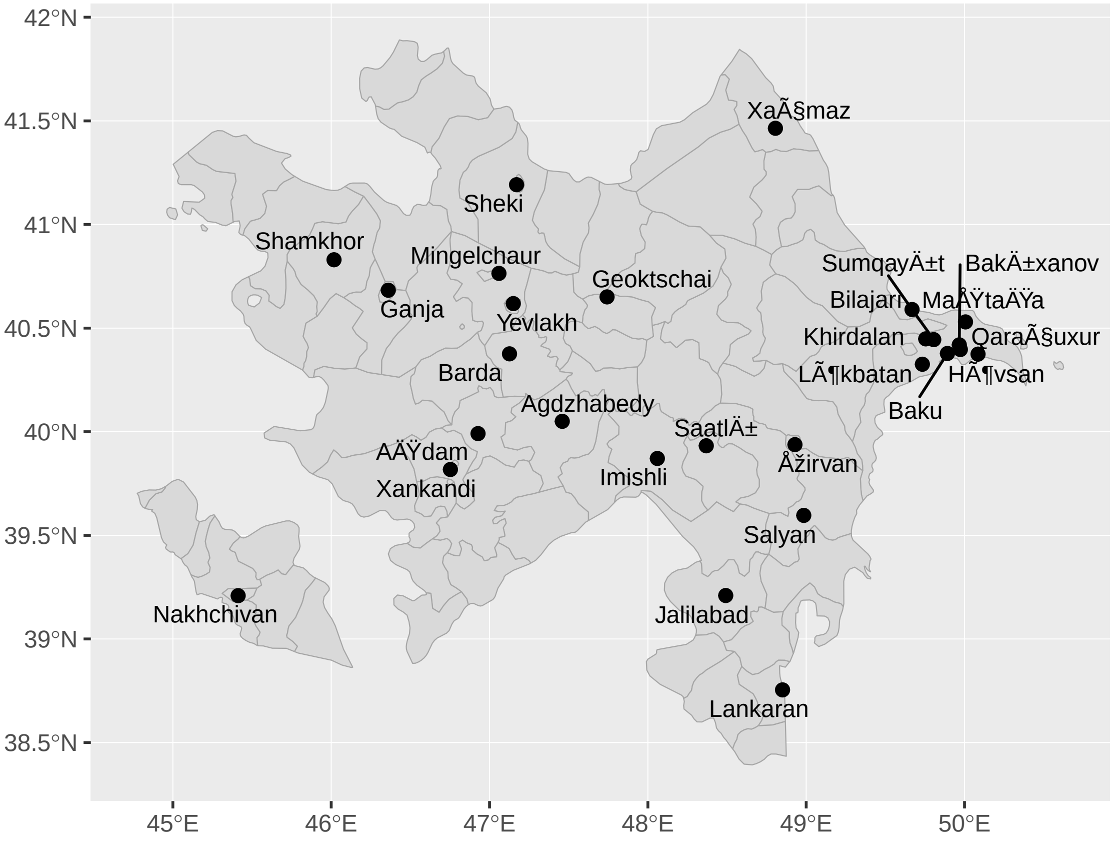
```
cols <- c("name", "longitude", "latitude", "population")
rows <- d$country.code=="AZ" & d$population > 30000
d <- d[rows, cols]
d<-d[complete.cases(d),]
```

Here I've adapted the code from the book directly to produce the map:

```
azerbaijan <- ne_states(country="azerbaijan", returnclass="sf")
cities <- st_as_sf(d, coords = c("longitude", "latitude"), crs=st_crs(azerbaijan), remove=FALSE)
```

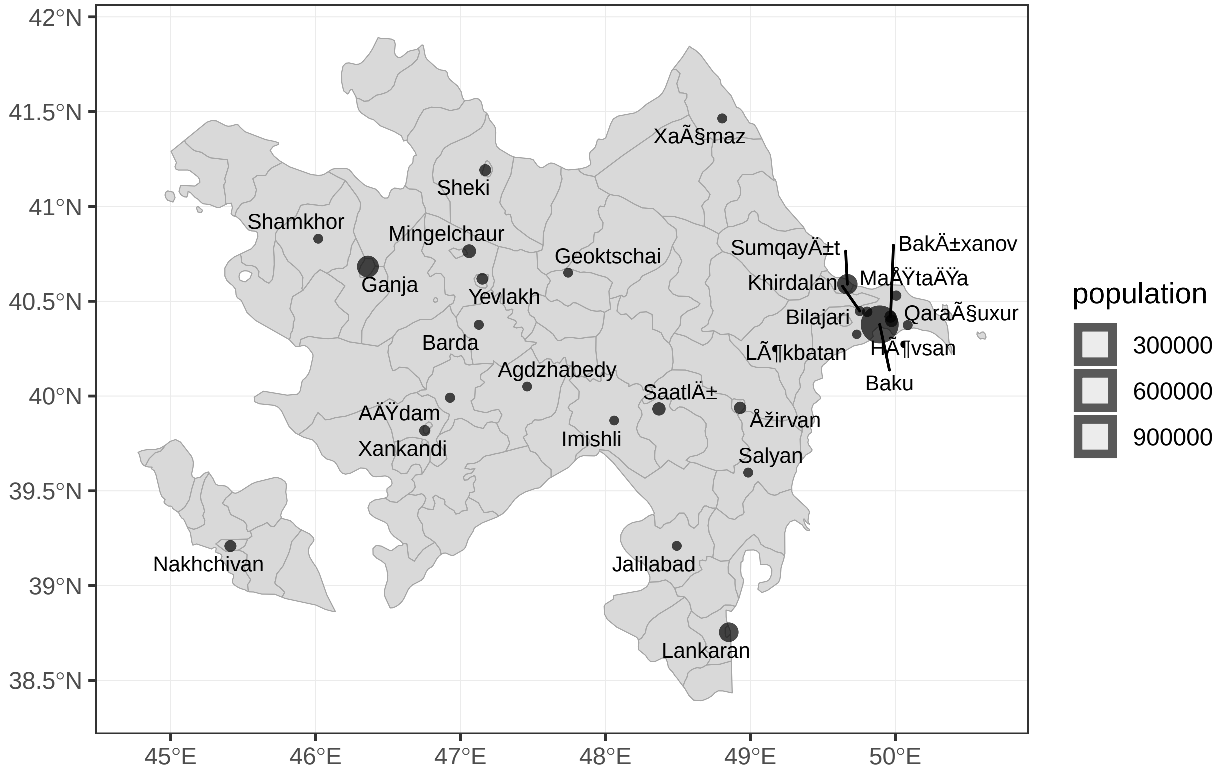
Here's the map with just points for each city:

```
ggplot() + geom_sf(data=azerbaijan, fill="gray85", color="gray65", size=0.2) + geom_sf(data=cities, size
```



And here's the map with each city's size indicated by the bubble on the map:

```
ggplot() + geom_sf(data=azerbaijan, fill="gray85", color="gray65", size=.2) + geom_sf(aes(size=populati
```



# 4. Time Series

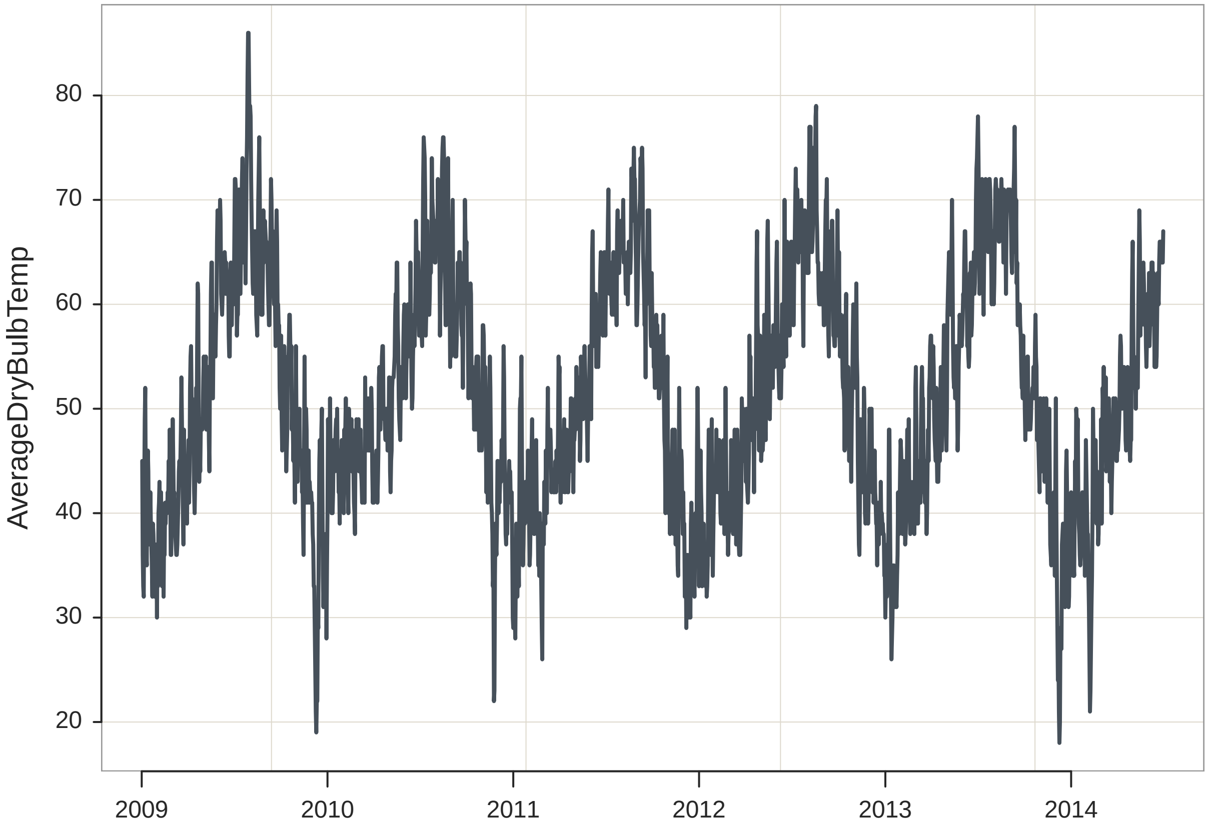
I have some Hillsboro airport climate data lying around, so let's use that. It's slightly long so I've subset it to the first 2,000 rows and only 3 of the relevant data columns Incidentally the "as.Date" helpfile is the funniest R helpfile I have ever read.

```
d<-rd("dailyclimatedata.xlsx", quiet=TRUE)
```

```
d<-Transform(DATE=as.Date(d$DATE,origin = "1899-12-30"), quiet=TRUE)
d<-d[1:2000,]
d<-d[,c(1,3,4,5)]
```

a. Here's a plot of average daily temperature:

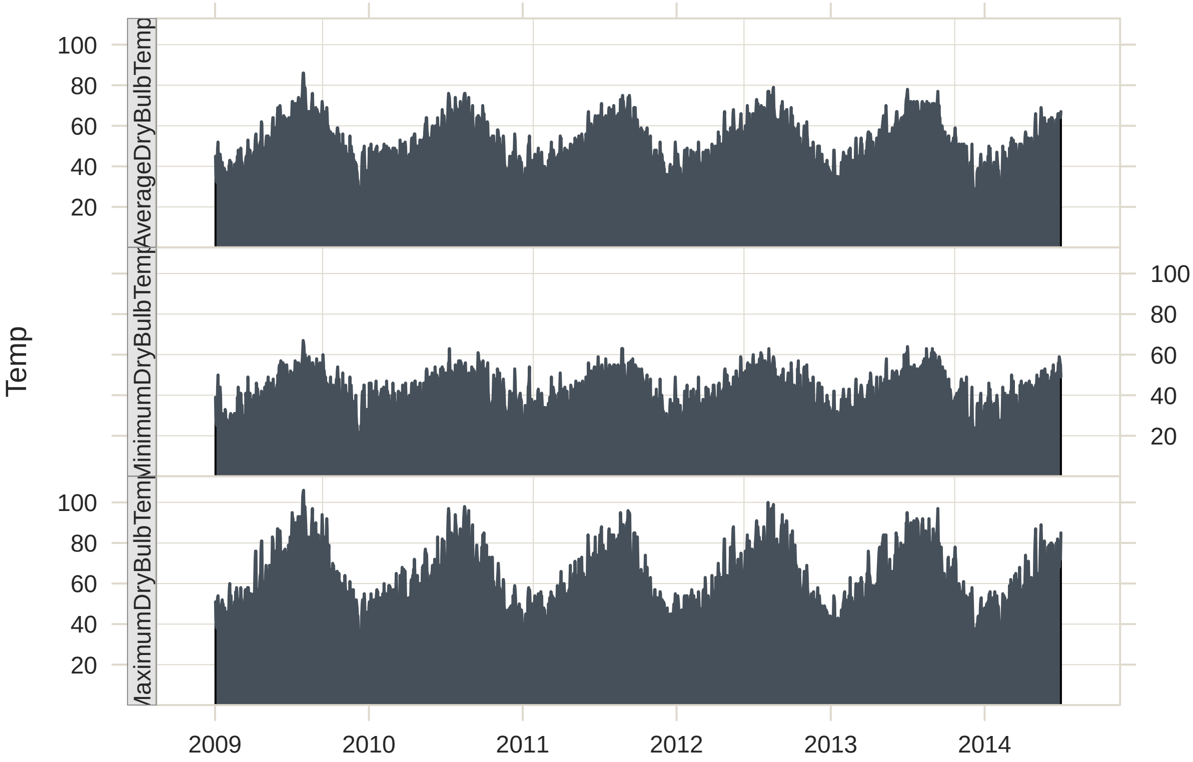
```
Plot(DATE, AverageDryBulbTemp, quiet=TRUE)
```



b.

These are average temperature, minimum temperature, and maximum temperature by day:

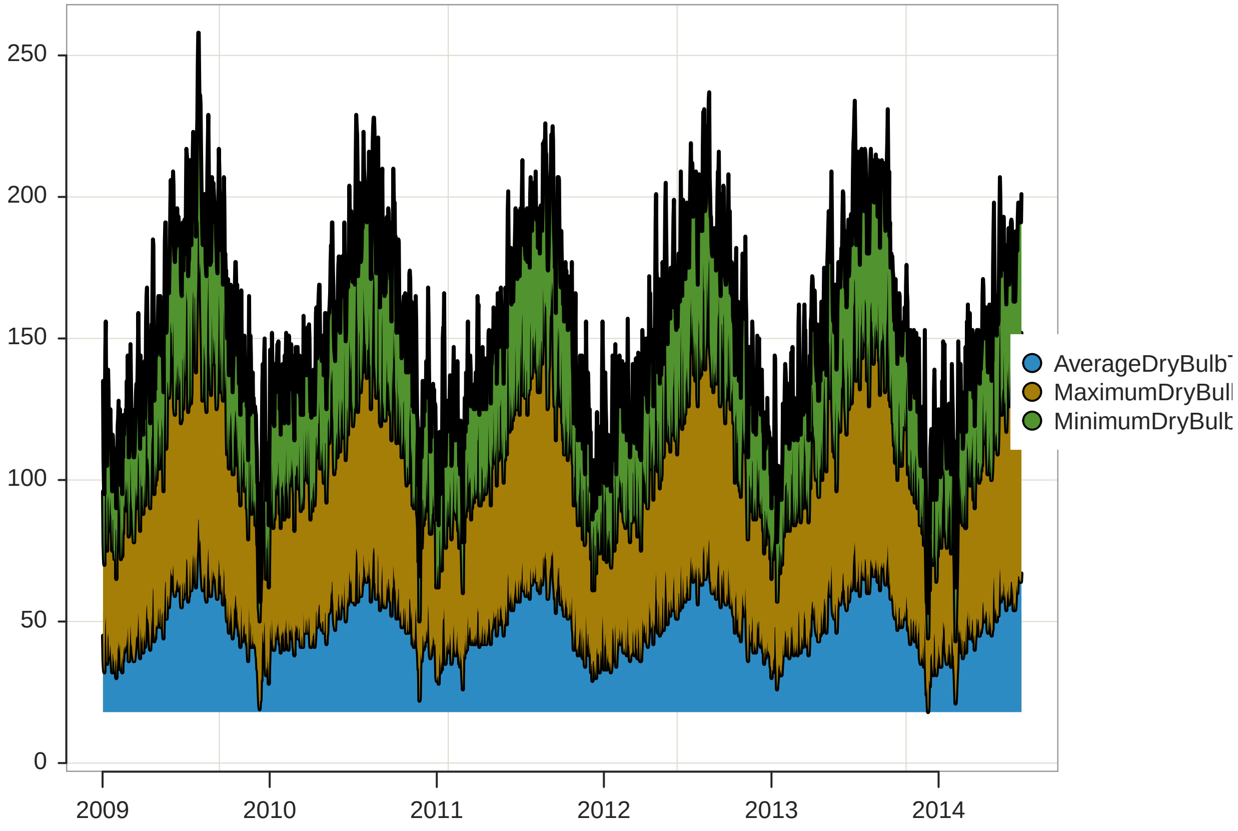
```
myd <- melt(d, id.vars="DATE", variable.name="TempType", value.name="Temp")
Plot(DATE, Temp, by1=TempType, data=myd, stack=TRUE, quiet=TRUE)
```



c.

My data's not greatly suited to this so it looks a little hideous, but here's the stacked chart:

```
Plot(DATE, c(AverageDryBulbTemp, MaximumDryBulbTemp, MinimumDryBulbTemp), stack=TRUE, quiet=TRUE)
```

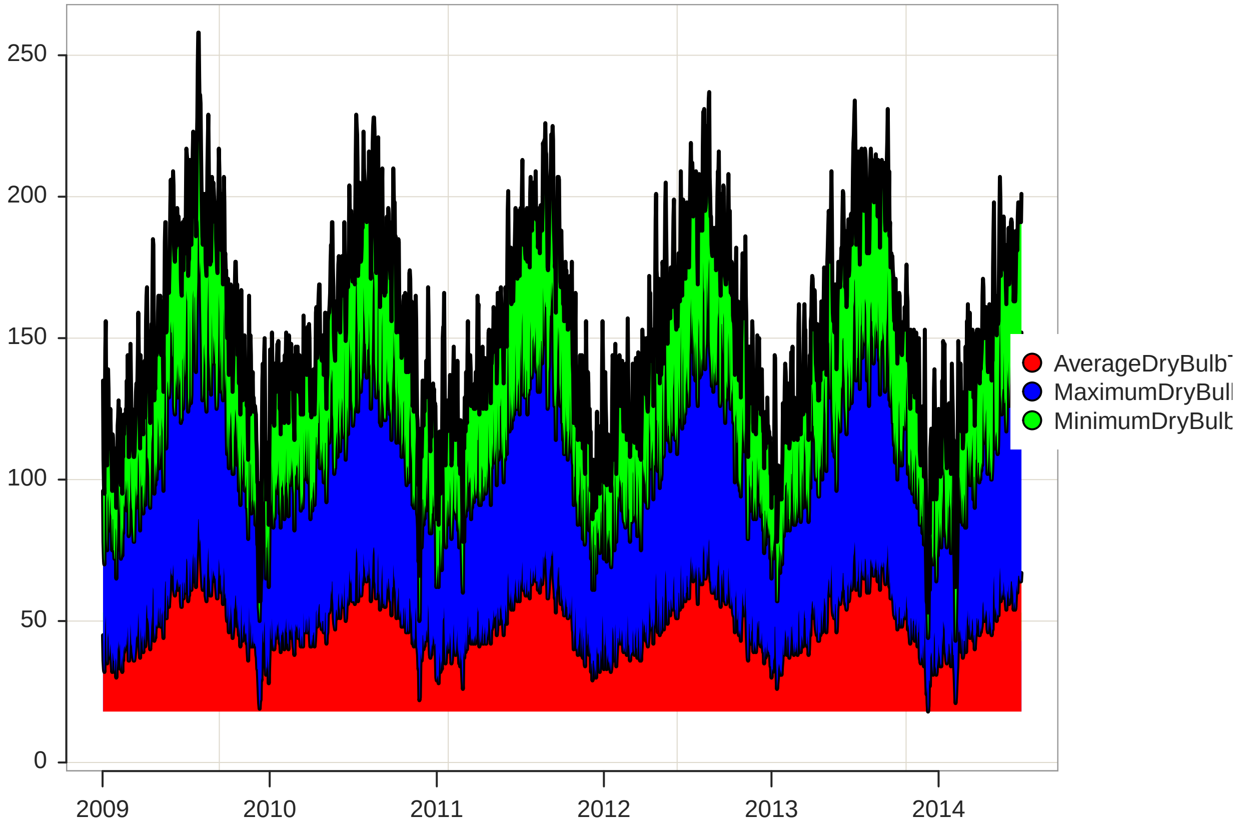


d.

And again with custom colors. I shouldn’t be allowed to choose custom colors.

```
Plot(DATE, c(AverageDryBulbTemp, MaximumDryBulbTemp, MinimumDryBulbTemp), stack=TRUE, fill=getColors(c(
```

```
## Warning in if (fill == "on") fill <- getOption("violin.fill"): the
## condition has length > 1 and only the first element will be used
```


AverageDryBulbT
MaximumDryBull
MinimumDryBulb