A brief introduction to plotting geographical data.

As published in Benchmarks RSS Matters, April 2016

http://web3.unt.edu/benchmarks/issues/2016/04/rss-matters

Jon Starkweather, PhD

Jon Starkweather, PhD
`jonathan.starkweather@unt.edu`
Consultant
Research and **S**tatistical **S**upport

UNT UNIVERSITY OF NORTH TEXAS A green light to greatness.™

http://www.unt.edu

RSS
Research and Statistical Support

http://www.unt.edu/rss

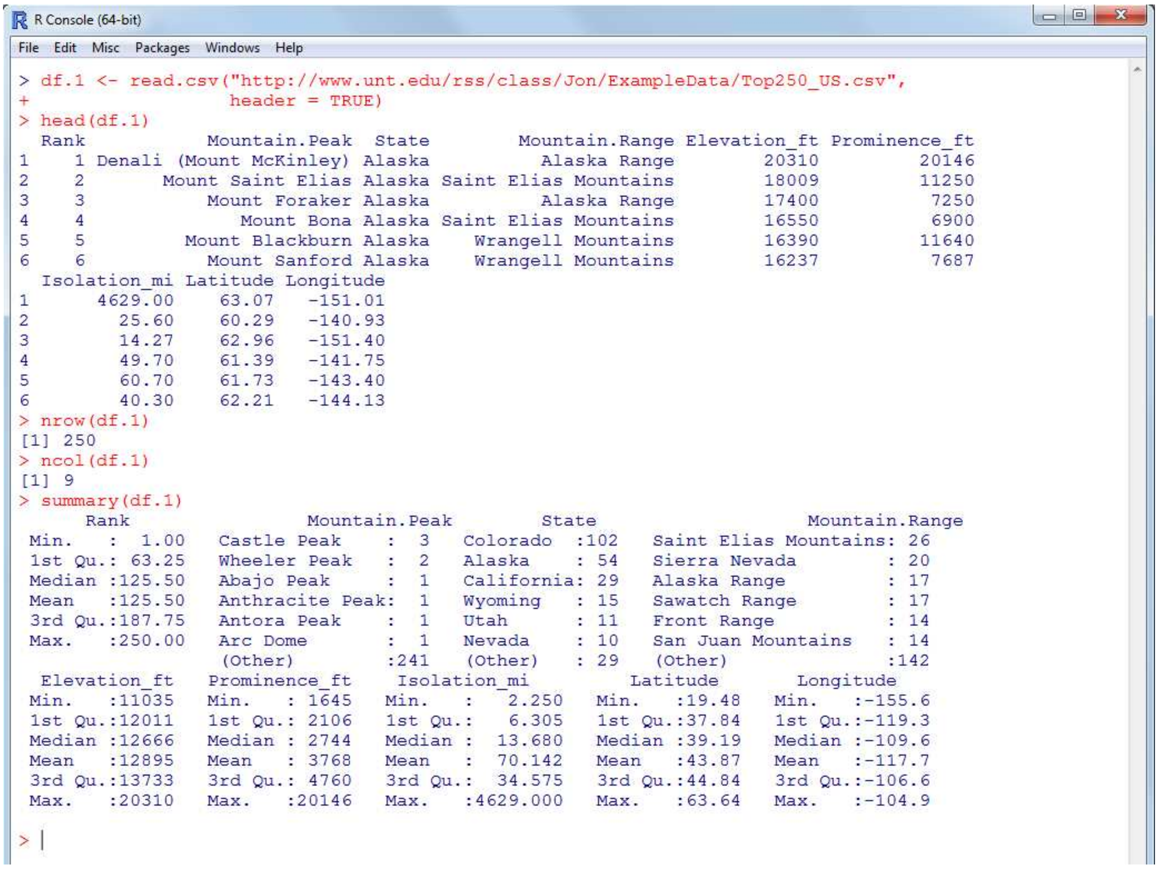
RSS hosts a number of “Short Courses”.
A list of them is available at:
http://www.unt.edu/rss/Instructional.htm

Those interested in learning more about R, or how to use it, can find information here:
`http://www.unt.edu/rss/class/Jon/R_SC`

A brief introduction to plotting geographical data.

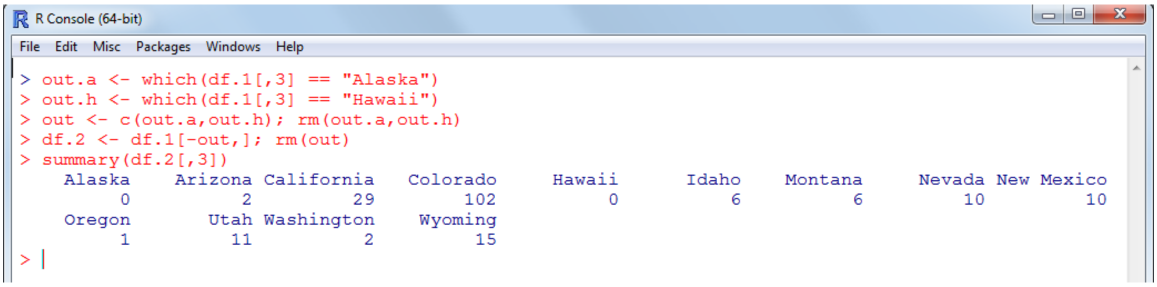
This month's article reviews some of the ways which a data analyst can plot geographical data in R using a two very handy packages. The two packages used here are 'ggmap' (Kahle & Wickham, 2013) and 'ggplot2' (Wickham, 2009). The package 'ggmap' requires the 'ggplot2' package. There are a variety of functions for using these two packages to plot geographical data using several types of maps. The examples below use topographical (i.e. terrain) maps produced by Google(TM). The examples below also utilize data from Wikipedia(TM). The data used in the examples below contains the highest 250 mountain peaks in the United States (Wikipedia, 2016).

First, import the data, which is available as a comma separated values (.csv) file on the R&SS server, and take a look at what is included.



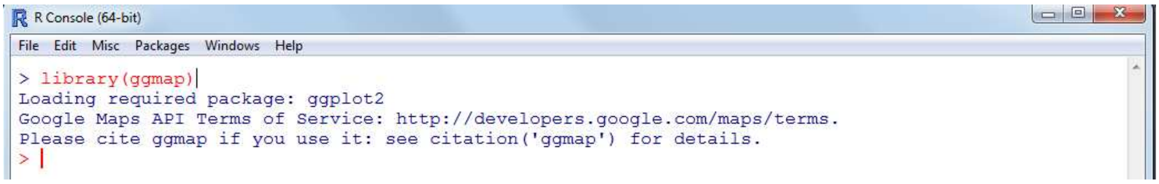
```
> df.1 <- read.csv("http://www.unt.edu/rss/class/Jon/ExampleData/Top250_US.csv",
+                  header = TRUE)
> head(df.1)
  Rank              Mountain.Peak  State        Mountain.Range Elevation_ft Prominence_ft
1    1 Denali (Mount McKinley) Alaska          Alaska Range        20310         20146
2    2       Mount Saint Elias Alaska Saint Elias Mountains        18009         11250
3    3           Mount Foraker Alaska          Alaska Range        17400          7250
4    4              Mount Bona Alaska Saint Elias Mountains        16550          6900
5    5         Mount Blackburn Alaska    Wrangell Mountains        16390         11640
6    6           Mount Sanford Alaska    Wrangell Mountains        16237          7687
  Isolation_mi Latitude Longitude
1      4629.00    63.07   -151.01
2        25.60    60.29   -140.93
3        14.27    62.96   -151.40
4        49.70    61.39   -141.75
5        60.70    61.73   -143.40
6        40.30    62.21   -144.13
> nrow(df.1)
[1] 250
> ncol(df.1)
[1] 9
> summary(df.1)
      Rank                Mountain.Peak        State                  Mountain.Range
 Min.   :  1.00   Castle Peak     :  3   Colorado  :102   Saint Elias Mountains: 26
 1st Qu.: 63.25   Wheeler Peak    :  2   Alaska    : 54   Sierra Nevada        : 20
 Median :125.50   Abajo Peak      :  1   California: 29   Alaska Range         : 17
 Mean   :125.50   Anthracite Peak:  1   Wyoming   : 15   Sawatch Range        : 17
 3rd Qu.:187.75   Antora Peak     :  1   Utah      : 11   Front Range          : 14
 Max.   :250.00   Arc Dome        :  1   Nevada    : 10   San Juan Mountains   : 14
                  (Other)         :241   (Other)   : 29   (Other)              :142
  Elevation_ft    Prominence_ft    Isolation_mi          Latitude        Longitude
 Min.   :11035   Min.   : 1645   Min.   :   2.250   Min.   :19.48   Min.   :-155.6
 1st Qu.:12011   1st Qu.: 2106   1st Qu.:   6.305   1st Qu.:37.84   1st Qu.:-119.3
 Median :12666   Median : 2744   Median :  13.680   Median :39.19   Median :-109.6
 Mean   :12895   Mean   : 3768   Mean   :  70.142   Mean   :43.87   Mean   :-117.7
 3rd Qu.:13733   3rd Qu.: 4760   3rd Qu.:  34.575   3rd Qu.:44.84   3rd Qu.:-106.6
 Max.   :20310   Max.   :20146   Max.   :4629.000   Max.   :63.64   Max.   :-104.9

> |
```

Next, select only the mountain peaks contained in the continental United States (i.e. exclude Alaska & Hawaii).



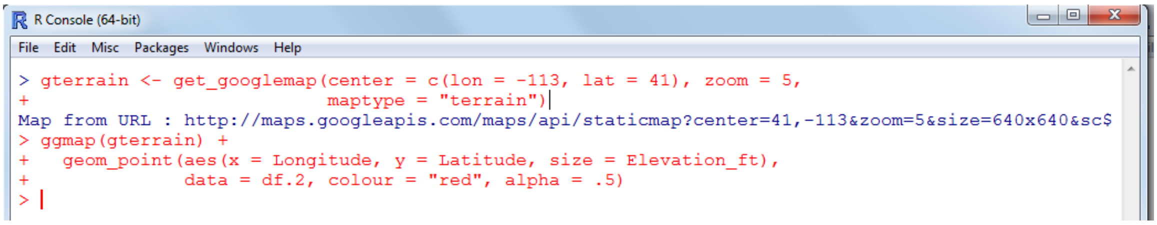
```
> out.a <- which(df.1[,3] == "Alaska")
> out.h <- which(df.1[,3] == "Hawaii")
> out <- c(out.a,out.h); rm(out.a,out.h)
> df.2 <- df.1[-out,]; rm(out)
> summary(df.2[,3])
    Alaska    Arizona California   Colorado     Hawaii      Idaho    Montana     Nevada New Mexico
         0          2         29        102          0          6          6         10         10
    Oregon       Utah Washington    Wyoming
         1         11          2         15
> |
```

Next, load the libraries ‘ggmap’, which requires ‘ggplot2’.

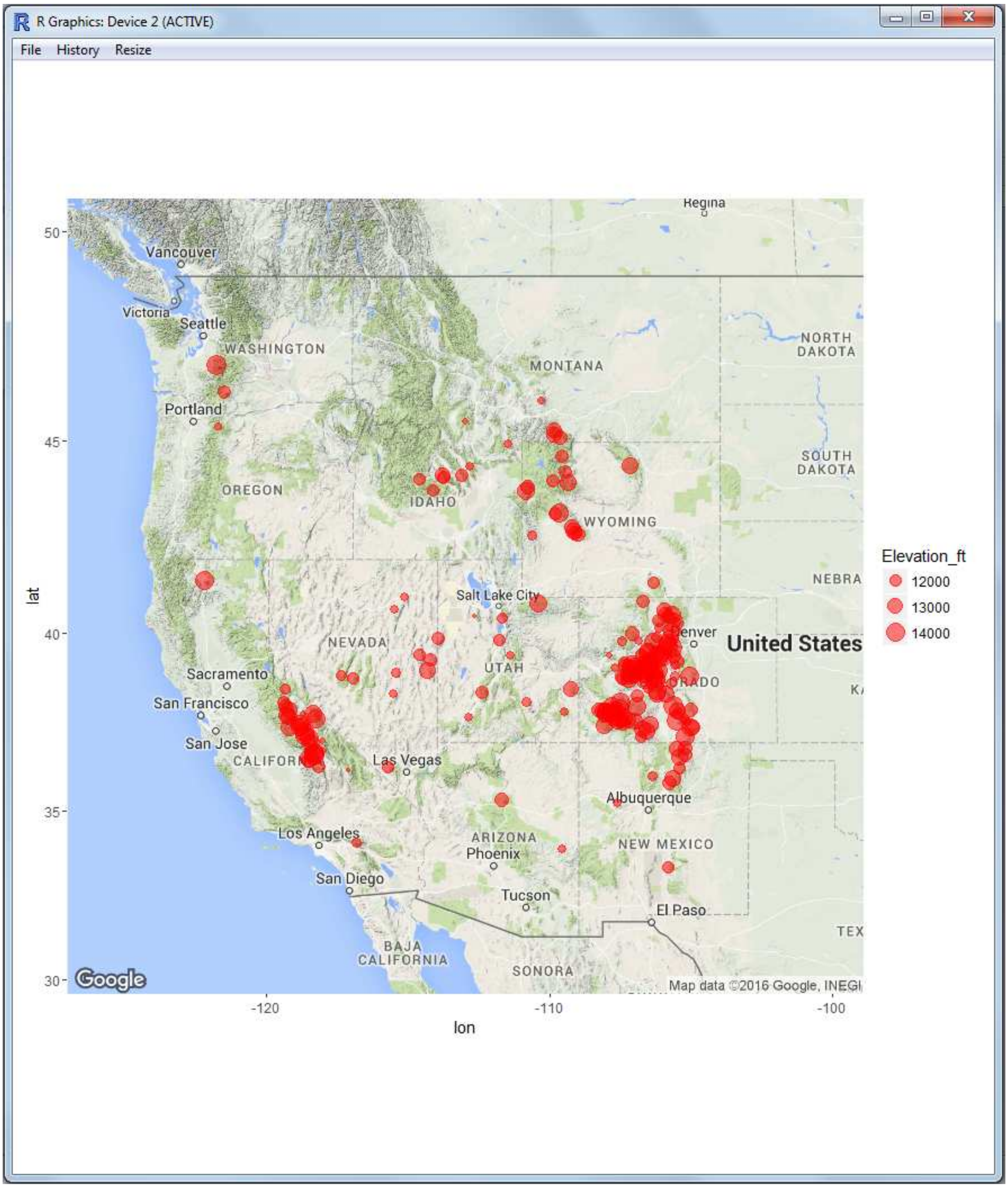


```
> library(ggmap)
Loading required package: ggplot2
Google Maps API Terms of Service: http://developers.google.com/maps/terms.
Please cite ggmap if you use it: see citation('ggmap') for details.
>
```

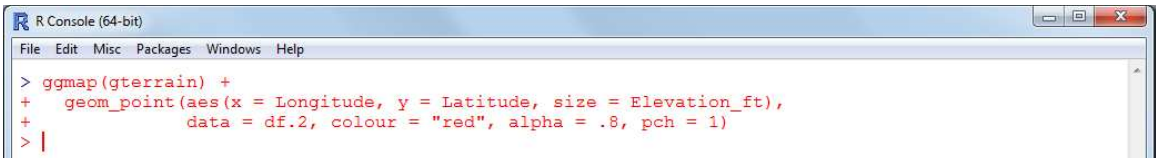
Next, get and plot the initial map. It is centered near Salt Lake City, UT. Keep in mind, the ‘zoom’ argument refers to: “...an integer from 3 (continent) to 21 (building), default value 10 (city)” (Kahle & Wickham, 2013). We use the longitude (x-axis) and latitude (y-axis) to locate the mountain peaks. Notice we are also using the size of the points to represent the elevation of the mountain peaks.



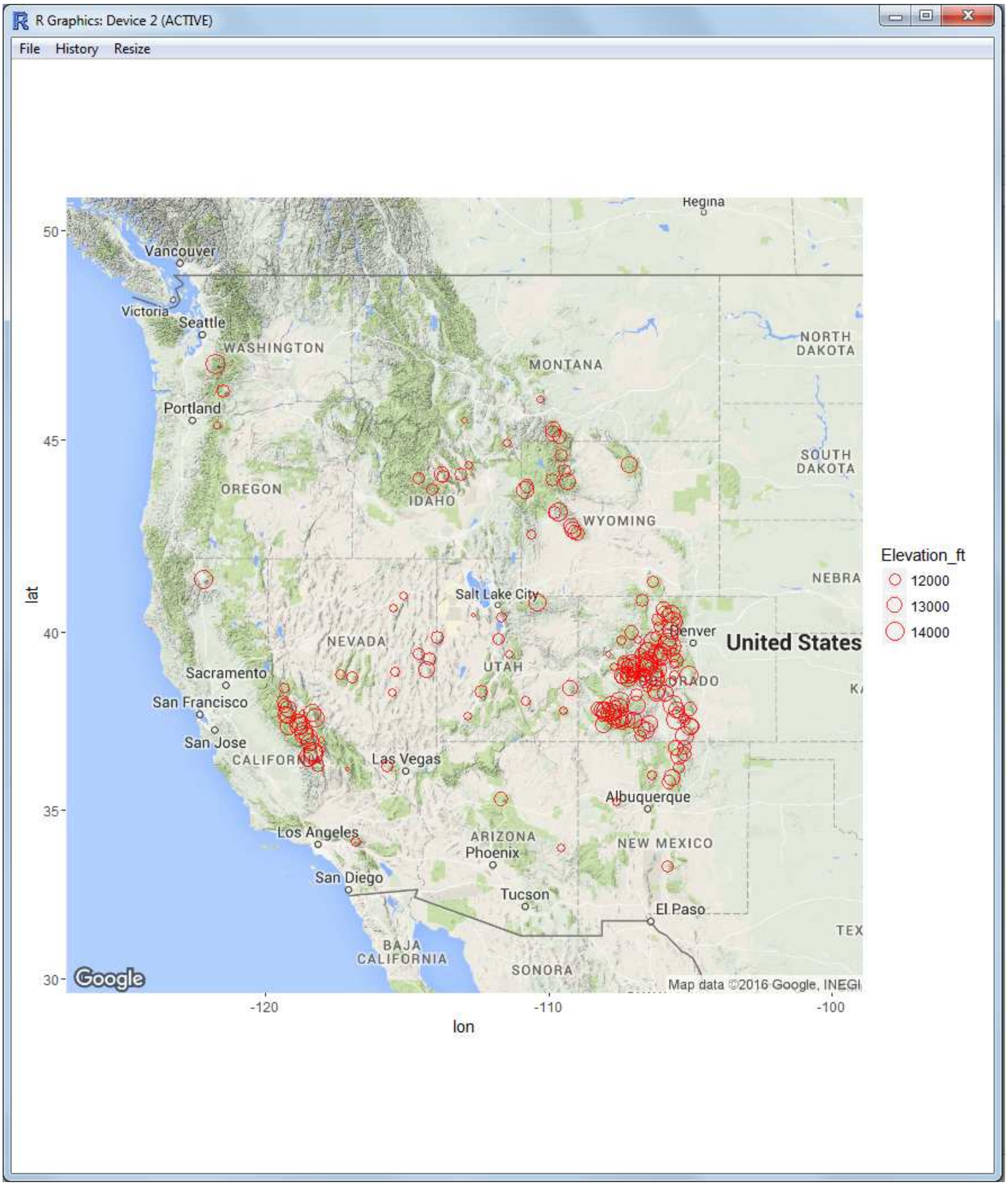
```
> gterrain <- get_googlemap(center = c(lon = -113, lat = 41), zoom = 5,
+                           maptype = "terrain")
Map from URL : http://maps.googleapis.com/maps/api/staticmap?center=41,-113&zoom=5&size=640x640&sc$
> ggmap(gterrain) +
+   geom_point(aes(x = Longitude, y = Latitude, size = Elevation_ft),
+              data = df.2, colour = "red", alpha = .5)
>
```



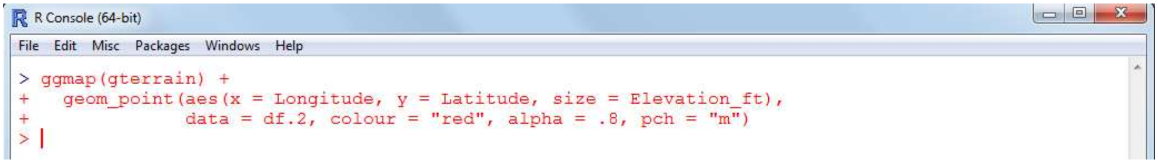
Unfortunately, the larger points are simply obscuring the smaller ones. So, we need to make the points hollow (rather than solid). This is due to two things, first, the points are solid and second, the 'alpha' sets the transparency. If we lower the transparency further, the points would disappear into the map. So, we increase the transparency, but, use hollow points rather than solid (pch = 1).



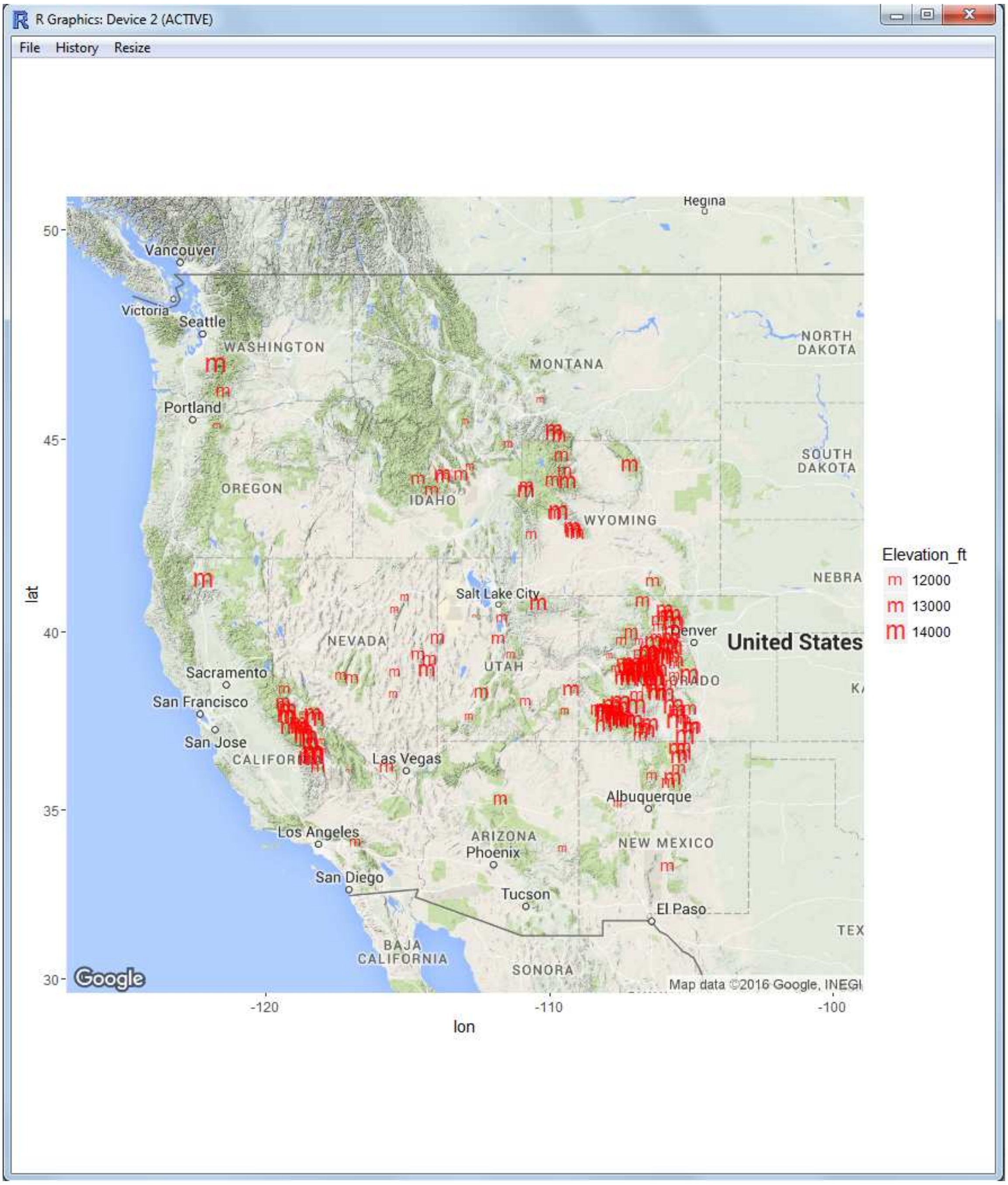
```
> ggmap(gterrain) +
+   geom_point(aes(x = Longitude, y = Latitude, size = Elevation_ft),
+              data = df.2, colour = "red", alpha = .8, pch = 1)
> 
```



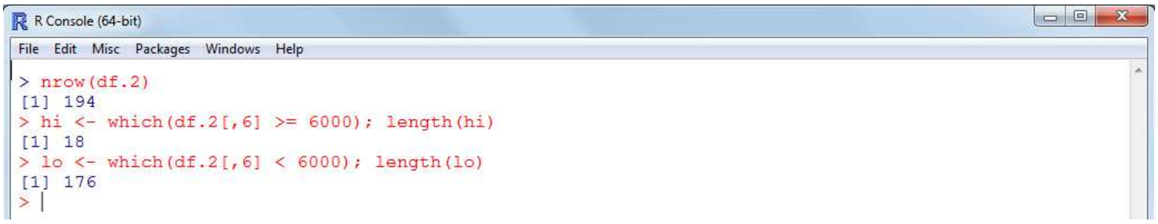
We can also change the points to any of the 25 available or simply use a particular character by simply inserting the character we want inside quotation marks for the ‘pch’ argument.



```
> ggmap(gterrain) +
+   geom_point(aes(x = Longitude, y = Latitude, size = Elevation_ft),
+              data = df.2, colour = "red", alpha = .8, pch = "m")
> |
```

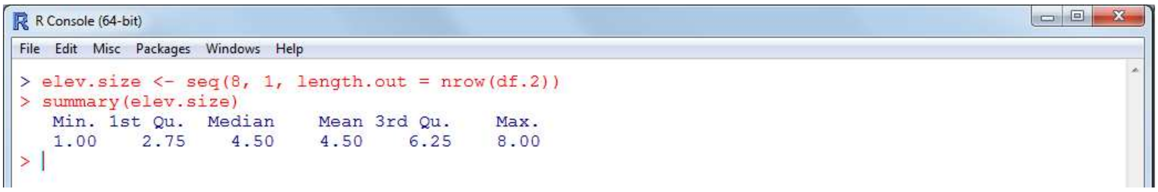


What if we had a grouping variable we wanted to include in the plot? For example, we can create 2 (arbitrary) groups based on prominence by dividing the peaks with prominence greater than or equal to 6000 feet or less than 6000 feet.



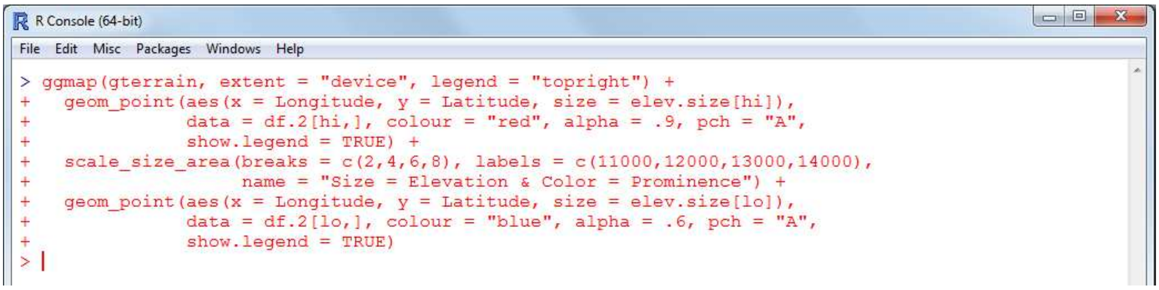
```
> nrow(df.2)
[1] 194
> hi <- which(df.2[,6] >= 6000); length(hi)
[1] 18
> lo <- which(df.2[,6] < 6000); length(lo)
[1] 176
> |
```

We are also going to need a better indicator of elevation — in order to better differentiate between the mountains. So, we create a sequential vector which runs between 8 and 1 with an equal number of sequential values as the number of mountains.

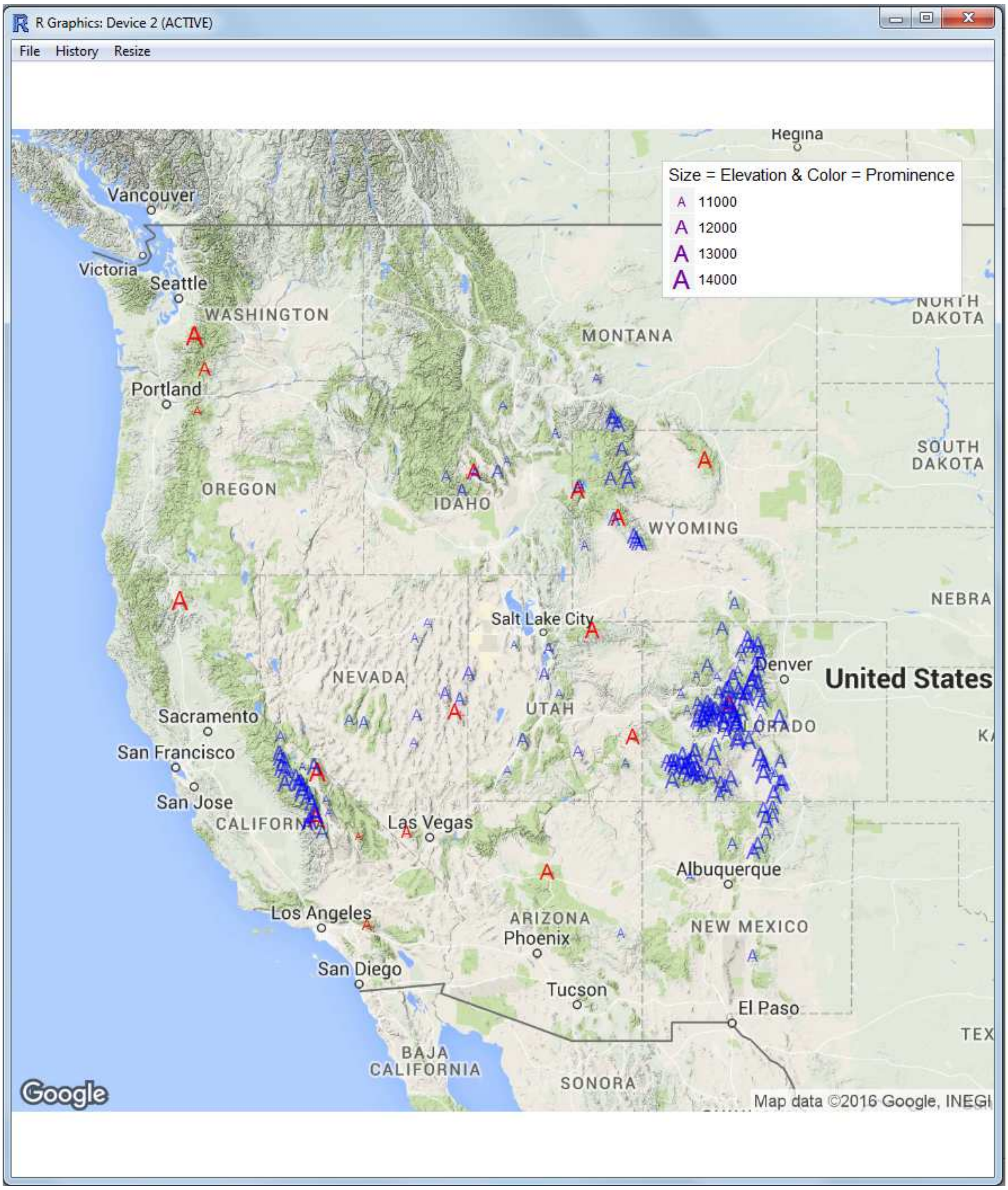


```
> elev.size <- seq(8, 1, length.out = nrow(df.2))
> summary(elev.size)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
   1.00    2.75    4.50    4.50    6.25    8.00 
> 
```

Next, we can combine the two groups into one map with red (reddish) points represent the peaks greater than or equal to 13000 feet and blue (blueish) points represent peaks less than 13000 feet. We need to tune the legend using the ‘scale_size_area’ function since we are using a different vector to represent the elevations. Notice in the plot below, we also changed the points to the character “A” by passing “A” to the ‘pch’ argument.



```
> ggmap(gterrain, extent = "device", legend = "topright") +
+   geom_point(aes(x = Longitude, y = Latitude, size = elev.size[hi]),
+              data = df.2[hi,], colour = "red", alpha = .9, pch = "A",
+              show.legend = TRUE) +
+   scale_size_area(breaks = c(2,4,6,8), labels = c(11000,12000,13000,14000),
+                   name = "Size = Elevation & Color = Prominence") +
+   geom_point(aes(x = Longitude, y = Latitude, size = elev.size[lo]),
+              data = df.2[lo,], colour = "blue", alpha = .6, pch = "A",
+              show.legend = TRUE)
> 
```



So, there we have a gentle introduction to the production of plots for representing geographical or geospatial data. There are other packages which can produce similar plots, the ‘ggmap’ and ‘ggplot2’ packages were used here simply because the author has an interest in hiking mountains and Google(TM) allows access to topographical (i.e. terrain) maps. As previous articles of Research Matters have stated, graphing data is as important as computation. A version of the R script used in this article can be found on the R&SS Do-It-Yourself Introduction to R[1] in the Module 12 section.

Until next time; be wary of the *mis–measure* of human attributes...[2]

[1] `http://www.unt.edu/rss/class/Jon/R_SC/`

[2] A perhaps too subtle nod to a book I recently read and recommend: *The Mismeasure of Man*.

References and Resources

Gould, S. J. (1996). *The Mismeasure of Man (Revised & Expanded)*. New York: W. W. Norton & Company.

Kahle, D. & Wickham, H. (2013). ggmap: Spatial Visualization with ggplot2. *The R Journal*, *5*(1), 144-161. URL: `http://journal.r-project.org/archive/2013-1/kahle-wickham.pdf`

Wickham, H. (2009). *ggplot2: Elegant Graphics for Data Analysis*. New York: Springer-Verlag.

Wikipedia. (2016). “List of the highest major summits of the United States.” Accessed April 7, 2016 at: `https://en.wikipedia.org/wiki/List_of_the_highest_major_summits_of_the_Unit`

This article was last updated on April 7, 2016.

This document was created using LaTeX