Introduction to R - Making Maps and Graphs using your Data

Key Points to Remember
- Low barrier to entry
  - Open Source
  - Huge online library of tutorials and a active support community
- You learn by doing and failing.
  - Experimentation is key.
  - Expect to invest more time and energy using open source tools.
  - Expect to have a more complete understanding of the technology you are using.
- Visualizing data is just as important when analyzing as it is when publishing.
- Don't use Windows or MS Office if you can help it... Linux > Mac > MS

So let's get started...

Intro to R and R Studio - The parts and pieces.

Getting your data into R and looking at it.

Download plants.csv from your email

```r
> ?read.csv

> plants <- read.csv("plantdata.csv", encoding = "utf-8", stringsAsFactors = FALSE, header = TRUE)
> view(plants)
Error: could not find function "view"

> View(plants)

> class(plants$lon)
```
A Basic Mapping Example

Usually made up of two pieces, background map and data points.

Building the Background Map

```r
> plantmap <- get_map(location = c(lon = mean(plants$lon), lat = mean(plants$lat)), zoom = 15)
Error: could not find function "get_map"

> library("ggmap", lib.loc="~/R/win-library/3.3")
Loading required package: ggplot2
Please cite ggmap if you use it: see citation("ggmap") for details.

> plantmap <- get_map(location = c(lon = mean(plants$lon), lat = mean(plants$lat)), zoom = 18)
> ggmap(plantmap) + geom_point(data = plants, aes(x = lon, y = lat), fill = "green", size = 2, shape = 21)

Adding Data Points

```r
> ?geom_point

```r
> ggmap(plantmap) + geom_point(data = plants, aes(x = lon, y = lat), fill = "green", size = 2, shape = 21)

Zoom in?

```r
> plantmap <- get_map(location = c(lon = mean(plants$lon), lat = mean(plants$lat)), zoom = 18)
> ggmap(plantmap) + geom_point(data = plants, aes(x = lon, y = lat), fill = "green", size = 2, shape = 21)

Changing shape?

```r
size = 6, shape = 77

To summarize...

Three lines of code, easily read by a human, provide a simple map of your data.
Low barrier to entry!!

```r
> plants <- read.csv("planedata.csv", encoding = "utf-8", stringsAsFactors = FALSE, header = TRUE)
> plantmap <- get_map(location = c(lon = mean(plants$lon), lat = mean(plants$lat)), zoom = 15)
> ggmap(plantmap) + geom_point(data = plants, aes(x = lon, y = lat), fill = "green", size = 2, shape = 21)

Of course, you need to add a title, legend, edit labels, etc. etc.

If we have time at the end of the class we will try using the help documentation to figure out how to include those elements in our map.
Cleaning up and Graphing Data

> count(plants, "scientificName")

<table>
<thead>
<tr>
<th>scientificName</th>
<th>freq</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abutilon grandifolium</td>
<td>1</td>
</tr>
<tr>
<td>Abutilon menziesii</td>
<td>2</td>
</tr>
<tr>
<td>Acacia confusa</td>
<td>55</td>
</tr>
<tr>
<td>Acacia koa</td>
<td>5</td>
</tr>
<tr>
<td>Acacia koaia</td>
<td>3</td>
</tr>
<tr>
<td>Acacia nilotica</td>
<td>1</td>
</tr>
<tr>
<td>Acacia saligna</td>
<td>5</td>
</tr>
<tr>
<td>Acacia stenophylla</td>
<td>1</td>
</tr>
<tr>
<td>Acalypha hispida</td>
<td>1</td>
</tr>
<tr>
<td>Acalypha wilkesiana</td>
<td>17</td>
</tr>
<tr>
<td>Acalypha wilkesiana (A. godseffiana)</td>
<td>5</td>
</tr>
<tr>
<td>Acer floribundus</td>
<td>1</td>
</tr>
<tr>
<td>Acoelorrhaphe wrightii</td>
<td>11</td>
</tr>
<tr>
<td>Acrocomia aculeata</td>
<td>3</td>
</tr>
<tr>
<td>Adansonia digitata</td>
<td>1</td>
</tr>
<tr>
<td>Adenanthera pavonina</td>
<td>2</td>
</tr>
<tr>
<td>Adonidia merrillii</td>
<td>70</td>
</tr>
<tr>
<td>Agapanthus praecox</td>
<td>1</td>
</tr>
<tr>
<td>Agathis australis</td>
<td>3</td>
</tr>
<tr>
<td>Agathis robusta</td>
<td>1</td>
</tr>
</tbody>
</table>

> counts <- ggplot(plants, aes(factor(scientificName)))
> counts + geom_bar()

A somewhat useless graph is the result....

Let's clean up the data a little bit to get something useful

Put the data in a data frame.
> namecountdf<- data.frame(table(plants$scientificName))

The table function default is counting the frequency of each element in the field specified. We specify scientificName so we get two columns in the dataframe. One with each unique value in scientificName and one with counts of each unique value.

If you use view(namecountdf) you can see that so far we are in the same boat as we were with the original > counts + geom_bar().

However, the point is that we are now working with a dataframe instead of a set of values, which gives us some options for manipulation the data.
Since our first graph was difficult to read because there were so many unique scientific names included, we can try to eliminate all the low count names and just look a the most frequent occurrences.

Because we are now working on a dataframe, we can use the function `subset()`.

```r
> highcountsdf <- data.frame(subset(namecountsdf, Freq >= 40, select=Var1:Freq))
```

Now we are dealing with a dataframe that only contains the unique scientific names that occur 40 or more times in the data.

This command stores a `ggplot` function in a value, and we can pair that with a `geom_bar` function to give us a bar graph.

```r
> highcountsplot <- ggplot(data=highcountdf, aes(Var1))
```

Using the `weight` element of the `aes` argument lets us set the y axis range using the values in the `Freq` field.

```r
> highcountsplot + geom_bar(aes(weight = Freq))
```

Now we can adjust the display of the labels on the x axis so that they are legible.

```r
> highcountsplot + geom_bar(aes(weight = Freq)) + theme(axis.text.x = element_text(angle = 75, vjust = 1, hjust = 1))
```

There are a ton of other aesthetic options and label and format options available for `ggplot` and its various pieces. We are just scratching the surface of possibilities.

Please feel free to contact me with specific or general questions about R.

David Beales
956-2529
beales@hawaii.edu