Spencer Phillips

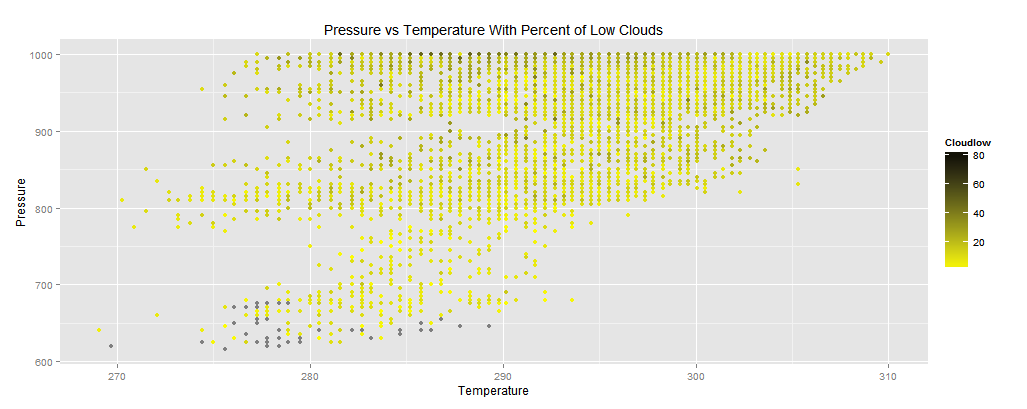
998887568

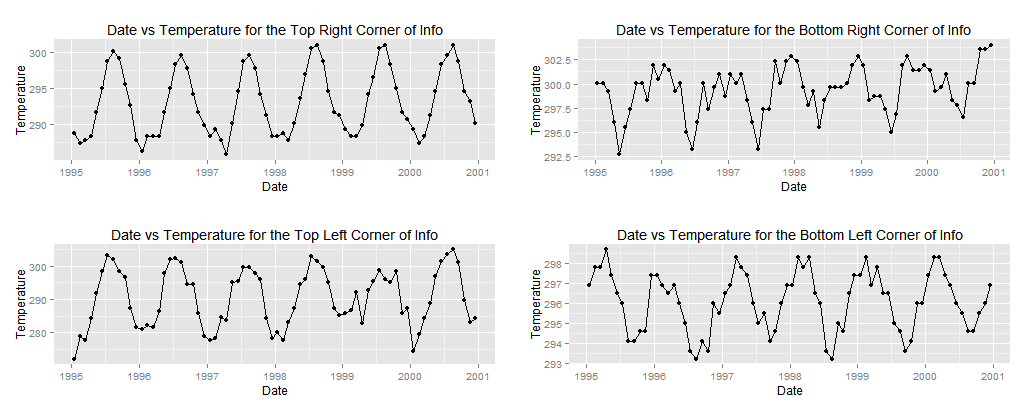
STA141 Assignment 2

Step 1. For this part, I started to try just to make one file into a data frame so I could then loop each set. I did this by extracting each of the variables I was interested in and then putting it all into one data frame, which I combined into a single function that I then looped across all 72 files for each set and used rbind to make it one large set.

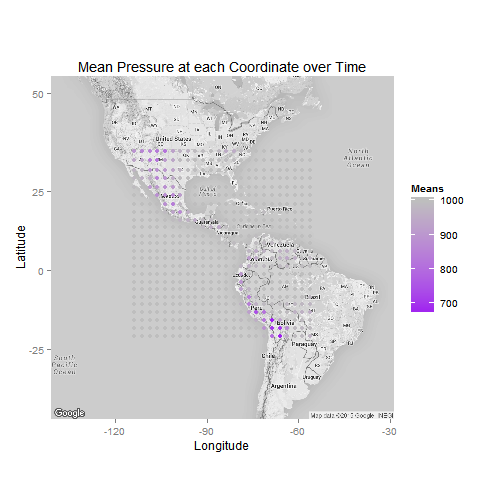
Step 2. For this section, I checked to see that each set was correct. I did this by using the function identical() and then pasted together the longitude, latitude, and date parts together into one long vector which I would then compare against each data set. If each were true, like so: identical(paste(cloudhigh$Longitude, cloudhigh$Latitude, cloudhigh$Date), paste(cloudlow$Longitude, cloudlow$Latitude, cloudlow$Date)). If each comparison returned TRUE (the next compared cloudlow to cloudmid since that would be equal to cloudhigh due to the law of transitivity) then we would know that they all are equal and can be combined together without any issues. All returned TRUE, so I combined the percentage values of each into one new complete data frame using Complete.df = cbind(cloudlow$Cloudlow, cloudmid$Cloudmid, cloudhigh$Cloudhigh, ozone$Ozone, pressure$Pressure, surftemp$Surftemp, temperature).

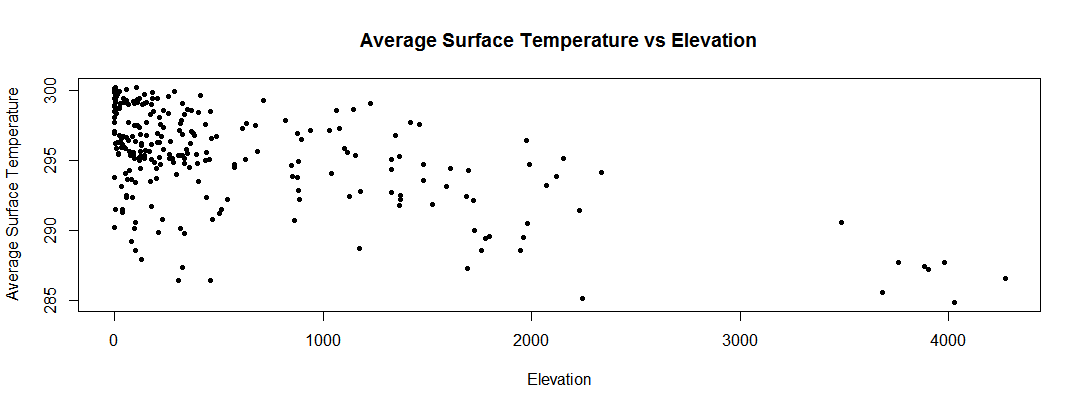
Step 3. For this section, I needed to read in the file intlvtn.dat which I did with read.csv() which put it into a data frame. I then noticed that the coordinates in intlvtn.dat was slightly off, perhaps a rounding error. The data was also, luckily, in the same layout as the data was in the rest of the data, so I created a function that took both values of the data frame going down the columns and in an apply function put it into a list that I then added to the Complete.df data frame using Complete.df$Elevation = ElevationList. I then checked to make sure all data was in the correct place, which it was.  
Code: GetElevation = function(x, y) return(elevation[x,y])  
ElevationList=apply(expand.grid(c(1:nrow(elevation)), c(1:ncol(elevation))), 1, function(xval, yval) GetFromDat(xval[1], xval[2]))

Step 4. I started this out by running unlist over all of the 7 numeric values to make them into numeric values instead of the factors they currently are. Next, I put the data into qplot to make the graph to show the data.   
Some of the dots are grey, which is due to a few NAs that were in the data for the Cloudlow variable. As we see, there are more low clouds in high pressure environments. There also tends to be more clouds being high pressure at high temperature with there being a a positive correlation between pressure and temperature.

For the second question we ask for the points at the four corners of the spatial grid (so the four corners of the goegraphical coordinates) to see their change in temperature over time for those places. For this, I had to change the Date variable from a function to a Date data type and make the coordinates numeric. This led to these four graphs for the four corners.  As shown, there is a similar cycle with the top corners but for the bottom corners we notice that they are off, this is due to the southern hemisphere having a different season cycle than the northern hemisphere.

The third question asks for the average and standard deviation of the 7 variables for each area over time. For this I created a function that would create a matrix of the means and standard deviations (seperately) and then give the names for each of these in the matrix. This was done using nested loops to get the means/sds of each variable more efficiently. Then the names were renamed in order for them to make logical sense when looking at them later.

For the fourth question, I needed to move the means of the pressure to a new dataframe which I named LongLatPressure. I then needed to add the Longitude and Latitude of each value to it. Since I could not find a way to use substring to make it work, I made a function using lapply(), substring, and strsplit to get what I needed to get done. It worked and after using a crazy ggplot2 function (mymap + geom\_point(data = LongLatPressure, aes(x = Longitude, y = Latitude, color = Means)) + scale\_color\_gradient(low = "purple", high = "grey") + ggtitle('Mean Pressure at each Coordinate over Time') + labs(x= 'Longitude', y = 'Latitude')) I got this.  This showed me that Pressure was very high when over the oceans and lower over the land, especially dry areas such as the US Southwest, Northern Mexico, and parts of Peru and Ecuador.

Part 4.5: In this part, I needed to get the average surface temperature and put it against Elevation, this would make me split my final dataframe by Elevation so then I could then manage the list of data frames by looping over them to get the means of each one’s surfacetemps. I would then put this into a list as part of lapply and get a list of the names of the many dataframes (in order to get the elevation levels). This would then go into the following plot.This plot shows that temperature drops as elevation increases but there are still some areas which are cold and at a low elevation though.