STA141 Homework 02

Zhen Zhang

October 20, 2015

Problem 1

The first step is to define a function to return a dataframe, based on the file path:

```
readData <- function(path) {
    # use textConnection to read into data
    data_scan <- scan(path, what = "")
    data_txtCon <- textConnection(data_scan)
    data_vec <- readLines(data_txtCon)
    # locate 'TIME'
    time_index <- which(data_vec == "TIME")
    # extract row names
    colName <- data_vec[(time_index + 4):(time_index + 27)]
    # extract column names
    rowName <- data_vec[seq(time_index + 52, by = 27, length.out = 24)]
    # create long and lat
    lat <- rep(rowName, each = 24)
    lon <- rep(colName, times = 24)
    # extract date
    data_date <- as.Date(data_vec[time_index + 2], "%d-%b-%Y")
    # extract raw data
    slash_index <- grep("/", data_vec)
    rawData_index <- unlist(lapply(slash_index[-1], function(x) (x + 2):(x + 25)))
    rawData <- as.numeric(data_vec[rawData_index])
    # combine them into a data frame
    data_frame <- data.frame(value = rawData, lat = lat, lon = lon, date = data_date)
}
```

What I need to do now is to render the names of all the txt files. There is multiple steps to do so: First define the function that accepts the variable name, then return the 72 pasted file names, and then define another function that accepts the 72 files as input and then apply them to the `readData()` function defined in the previous step.

```
# create names
createName <- function(rawName) {
    rawName_total <- paste(rawName, 1:72, sep = "")
    rawName_path <- paste("NASA/", rawName_total, ".txt", sep = "")
}

# define a function to return the final dataframe that accepts list of data
# frames as input and return the combined dataframes
createDataFrame <- function(rawName) {
    list_df <- lapply(createName(rawName), readData)
    df <- do.call("rbind", list_df)
}
```

Now define a global name vector corresponding to the name of the files

```
nameNASA <- c("cloudhigh", "cloudlow", "cloudmid", "ozone", "pressure", "surftemp",
    "temperature")
```

The final step is to call the `createDataFrame` function which will call the `createName` function and the `readData` function, then combine them together using rbind for each variable

```
dataListNASA <- lapply(nameNASA, createDataFrame)
names(dataListNASA) <- nameNASA
lapply(dataListNASA, head)
```

```
## $cloudhigh
##   value   lat    lon       date
## 1  26.0 36.2N 113.8W 1995-01-16
## 2  23.0 36.2N 111.2W 1995-01-16
## 3  23.0 36.2N 108.8W 1995-01-16
## 4  17.0 36.2N 106.2W 1995-01-16
## 5  19.5 36.2N 103.8W 1995-01-16
## 6  17.0 36.2N 101.2W 1995-01-16
##
## $cloudlow
##   value   lat    lon       date
## 1   7.5 36.2N 113.8W 1995-01-16
## 2   7.0 36.2N 111.2W 1995-01-16
## 3   7.0 36.2N 108.8W 1995-01-16
## 4   7.0 36.2N 106.2W 1995-01-16
## 5  11.0 36.2N 103.8W 1995-01-16
## 6  14.5 36.2N 101.2W 1995-01-16
##
## $cloudmid
##   value   lat    lon       date
## 1  34.5 36.2N 113.8W 1995-01-16
## 2  32.0 36.2N 111.2W 1995-01-16
## 3  32.0 36.2N 108.8W 1995-01-16
## 4  29.5 36.2N 106.2W 1995-01-16
## 5  33.0 36.2N 103.8W 1995-01-16
## 6  34.0 36.2N 101.2W 1995-01-16
##
## $ozone
##   value   lat    lon       date
## 1   304 36.2N 113.8W 1995-01-16
## 2   306 36.2N 111.2W 1995-01-16
## 3   306 36.2N 108.8W 1995-01-16
## 4   294 36.2N 106.2W 1995-01-16
## 5   308 36.2N 103.8W 1995-01-16
## 6   310 36.2N 101.2W 1995-01-16
##
## $pressure
##   value   lat    lon       date
## 1   835 36.2N 113.8W 1995-01-16
## 2   810 36.2N 111.2W 1995-01-16
## 3   810 36.2N 108.8W 1995-01-16
## 4   775 36.2N 106.2W 1995-01-16
## 5   795 36.2N 103.8W 1995-01-16
```

```
## 6   915 36.2N 101.2W 1995-01-16
##
## $surftemp
##   value   lat    lon       date
## 1 272.7 36.2N 113.8W 1995-01-16
## 2 270.9 36.2N 111.2W 1995-01-16
## 3 270.9 36.2N 108.8W 1995-01-16
## 4 269.7 36.2N 106.2W 1995-01-16
## 5 273.2 36.2N 103.8W 1995-01-16
## 6 275.6 36.2N 101.2W 1995-01-16
##
## $temperature
##   value   lat    lon       date
## 1 272.1 36.2N 113.8W 1995-01-16
## 2 270.3 36.2N 111.2W 1995-01-16
## 3 270.3 36.2N 108.8W 1995-01-16
## 4 270.9 36.2N 106.2W 1995-01-16
## 5 271.5 36.2N 103.8W 1995-01-16
## 6 275.6 36.2N 101.2W 1995-01-16
```

Problem 2

Part 1

I first define a function that compares the grid coordinates. It accepts two inputs: one dataframe and another dataframe. For each one, extract the information we need and then compare them. The information can depend on data or variable, based on the question. It all is equal, the result will be true, otherwise, false.

```
compareCoordinate <- function(x, y) {
    lonCompare <- sum(x$lon != y$lon)
    latCompare <- sum(x$lat != y$lat)
    return(lonCompare + latCompare == 0)
}
```

Now split the data by date

```
dataListNASAByDate <- lapply(dataListNASA, function(x) split(x[, 1:3], x[, 4]))
```

Compare between date by calling compareCoordinate and Map function

```
compareBetweenDate <- sapply(dataListNASAByDate, function(x, n) {
    Map(compareCoordinate, x[1:n - 1], x[2:n])
}, 72)
colnames(compareBetweenDate) <- nameNASA
compareBetweenDate
```

```
##            cloudhigh cloudlow cloudmid ozone pressure surftemp temperature
## 1995-01-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1995-02-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1995-03-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1995-04-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1995-05-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1995-06-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
```

```
## 1995-07-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1995-08-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1995-09-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1995-10-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1995-11-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1995-12-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1996-01-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1996-02-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1996-03-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1996-04-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1996-05-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1996-06-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1996-07-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1996-08-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1996-09-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1996-10-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1996-11-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1996-12-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1997-01-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1997-02-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1997-03-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1997-04-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1997-05-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1997-06-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1997-07-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1997-08-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1997-09-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1997-10-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1997-11-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1997-12-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1998-01-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1998-02-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1998-03-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1998-04-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1998-05-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1998-06-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1998-07-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1998-08-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1998-09-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1998-10-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1998-11-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1998-12-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1999-01-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1999-02-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1999-03-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1999-04-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1999-05-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1999-06-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1999-07-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1999-08-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1999-09-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1999-10-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1999-11-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 1999-12-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
```

```
## 2000-01-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 2000-02-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 2000-03-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 2000-04-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 2000-05-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 2000-06-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 2000-07-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 2000-08-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 2000-09-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 2000-10-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
## 2000-11-16 TRUE      TRUE     TRUE     TRUE  TRUE     TRUE     TRUE
```

Compare between variables by calling compareCoordinate and Map function

```
compareBetweenVar <- unlist(Map(compareCoordinate, dataListNASA[1:6], dataListNASA[2:7]))
compareBetweenVar
```

```
## cloudhigh  cloudlow  cloudmid     ozone  pressure  surftemp
##      TRUE      TRUE      TRUE      TRUE      TRUE      TRUE
```

They are the same.

Part 2

First I append a row named by its data source to each data frame in the big list: dataListNASA

```
addTag <- function(i) {
    data_i <- dataListNASA[[i]]
    data_i <- data.frame(data_i, type = nameNASA[i])
}
```

Then I use lapply to perform addTag function, combine them into a data frame with type variable

```
dataListNASANew <- lapply(1:7, addTag)
dataNASANew <- do.call("rbind", dataListNASANew)
```

Next I will use the package tidyr to convert from long format to wide format

```
library(tidyr)
dataNASA <- spread(dataNASANew, type, value)
head(dataNASA)
```

```
##    lat    lon       date cloudhigh cloudlow cloudmid ozone pressure surftemp
## 1 1.2N 101.2W 1995-01-16       3.0     47.0     21.5   246     1000    298.3
## 2 1.2N 101.2W 1995-02-16       2.0     28.0      9.0   246     1000    298.3
## 3 1.2N 101.2W 1995-03-16       2.5     26.5      6.5   252     1000    299.6
## 4 1.2N 101.2W 1995-04-16       1.5     16.0      3.5   252     1000    298.3
## 5 1.2N 101.2W 1995-05-16       1.5     22.0      4.5   256     1000    297.4
## 6 1.2N 101.2W 1995-06-16       0.5     37.5      4.5   262     1000    296.5
##   temperature
## 1       298.3
## 2       299.6
## 3       299.6
```

```
## 4       299.2
## 5       297.8
## 6       297.8
```

Problem 3

First read elevation data:

```
intlvtn_scan <- scan("NASA/intlvtn.dat", what = "")
intlvtn_txtCon <- textConnection(intlvtn_scan)
intlvtn_vec <- readLines(intlvtn_txtCon)
```

Extract row and col

```
intlvtn_col <- intlvtn_vec[1:24]
intlvtn_row <- intlvtn_vec[seq(25, by = 25, length.out = 24)]
```

Define function to return string coordinate row format in compliance with the previous data

```
coordinateStr <- function(i, lon = F) {
    i <- as.numeric(i)
    if (lon == T) {
        i <- abs(i)
        i <- ifelse(round(i) > i, round(i) - 0.2, round(i) + 0.2)
        paste(as.character(i), "W", sep = "")
    } else {
        if (i > 0) {
            i <- ifelse(round(i) > i, round(i) - 0.2, round(i) + 0.2)
            paste(as.character(i), "N", sep = "")
        } else {
            i <- abs(i)
            i <- ifelse(round(i) > i, round(i) - 0.2, round(i) + 0.2)
            paste(as.character(i), "S", sep = "")
        }
    }
}
```

Apply the coordinateStr function to the row and col:

```
intlvtn_lat <- unlist(lapply(intlvtn_row, coordinateStr))
intlvtn_lon <- unlist(lapply(intlvtn_col, coordinateStr, T))
```

Extract data:

```
intlvtn_rawData <- as.numeric(intlvtn_vec[(26:624)[-seq(25, by = 25, length.out = 23)]])
```

Combine them into a dataframe:

```
intlvtn_data <- data.frame(elevation = intlvtn_rawData, lat = rep(intlvtn_lat, each = 24),
    lon = rep(intlvtn_lon, times = 24))
```

Merge with dataNASA:

```r
dataNASAInt <- merge(dataNASA, intlvtn_data, by = c("lat", "lon"))
head(dataNASAInt)
```

```
##    lat    lon       date cloudhigh cloudlow cloudmid ozone pressure surftemp
## 1 1.2N 101.2W 1995-01-16       3.0     47.0     21.5   246     1000    298.3
## 2 1.2N 101.2W 1995-02-16       2.0     28.0      9.0   246     1000    298.3
## 3 1.2N 101.2W 1995-03-16       2.5     26.5      6.5   252     1000    299.6
## 4 1.2N 101.2W 1995-04-16       1.5     16.0      3.5   252     1000    298.3
## 5 1.2N 101.2W 1995-05-16       1.5     22.0      4.5   256     1000    297.4
## 6 1.2N 101.2W 1995-06-16       0.5     37.5      4.5   262     1000    296.5
##   temperature elevation
## 1       298.3         0
## 2       299.6         0
## 3       299.6         0
## 4       299.2         0
## 5       297.8         0
## 6       297.8         0
```

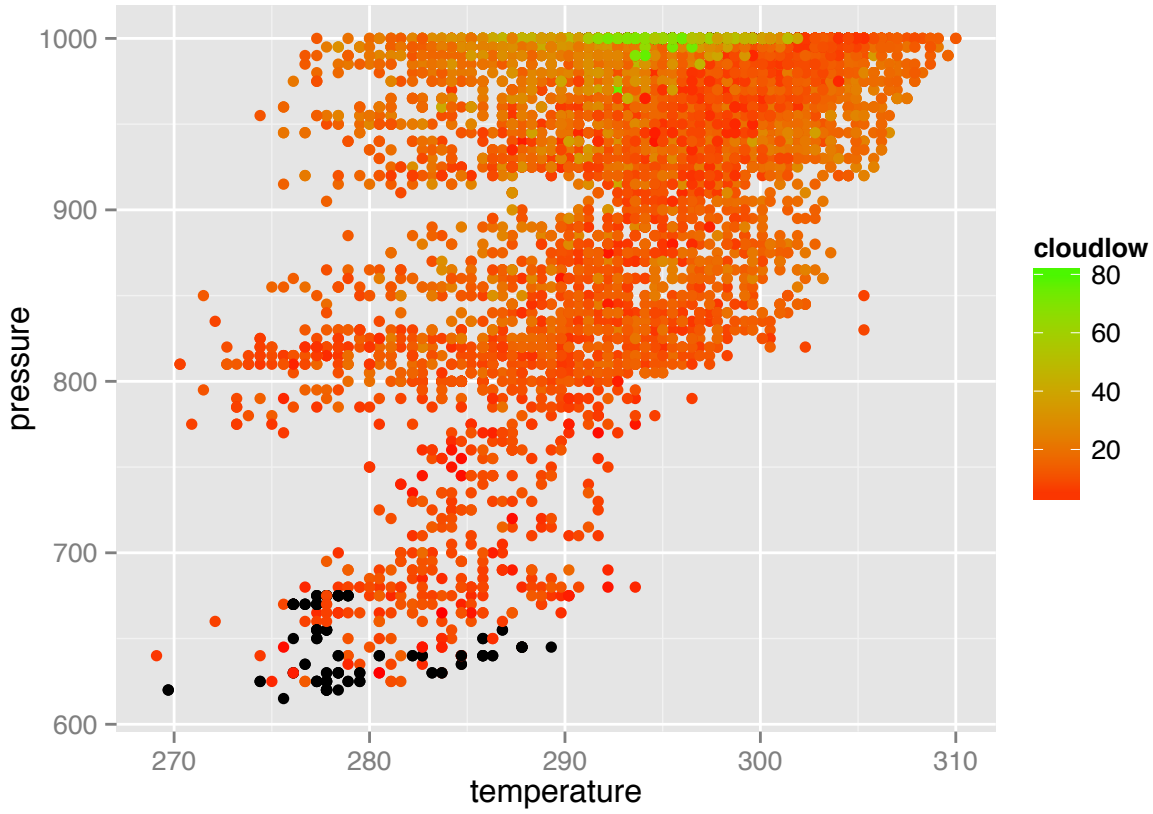
Problem 4

First load ggplot

```r
library(ggplot2)
```

Part 1

```r
ggplot(dataNASAInt, aes(x = temperature, y = pressure)) + geom_point(aes(color = cloudlow)) +
    scale_color_gradient(low = "red", high = "green", na.value = "black")
```



Part 2

First I need to find the lowest and highest value for the longitude and latitude. let's see the levels for latitude

```
levels(dataNASAInt$lat)
```

```
##  [1] "1.2N"  "1.2S"  "11.2N" "11.2S" "13.8N" "13.8S" "16.2N" "16.2S" "18.8N"
## [10] "18.8S" "21.2N" "21.2S" "23.8N" "26.2N" "28.8N" "3.8N"  "3.8S"  "31.2N"
## [19] "33.8N" "36.2N" "6.2N"  "6.2S"  "8.8N"  "8.8S"
```

```
# After observing, I see that the corner ones are '113.8W' and '56.2W'
lon_high <- "113.8W"
lon_low <- "56.2W"
```

Let's see the levels for longitude:

```
levels(dataNASAInt$lon)
```

```
##  [1] "101.2W" "103.8W" "106.2W" "108.8W" "111.2W" "113.8W" "56.2W"  "58.8W"
##  [9] "61.2W"  "63.8W"  "66.2W"  "68.8W"  "71.2W"  "73.8W"  "76.2W"  "78.8W"
## [17] "81.2W"  "83.8W"  "86.2W"  "88.8W"  "91.2W"  "93.8W"  "96.2W"  "98.8W"
```

```
# After observing, I see that the corner ones are '36.2N' and '21.2S'
lat_high <- "36.2N"
lat_low <- "21.2S"
```

Now subset data with 2 * 2 grids

```
lat_high_lon_high <- subset(dataNASAInt, lat == lat_high & lon == lon_high)
lat_high_lon_low <- subset(dataNASAInt, lat == lat_high & lon == lon_low)
lat_low_lon_high <- subset(dataNASAInt, lat == lat_low & lon == lon_high)
lat_low_lon_low <- subset(dataNASAInt, lat == lat_low & lon == lon_low)
```

Combine them in a list:

```
lat_lon_corner <- list(lat_high_lon_high, lat_high_lon_low, lat_low_lon_high, lat_low_lon_low)
```

Give them a type variable indicating the lon and lat:

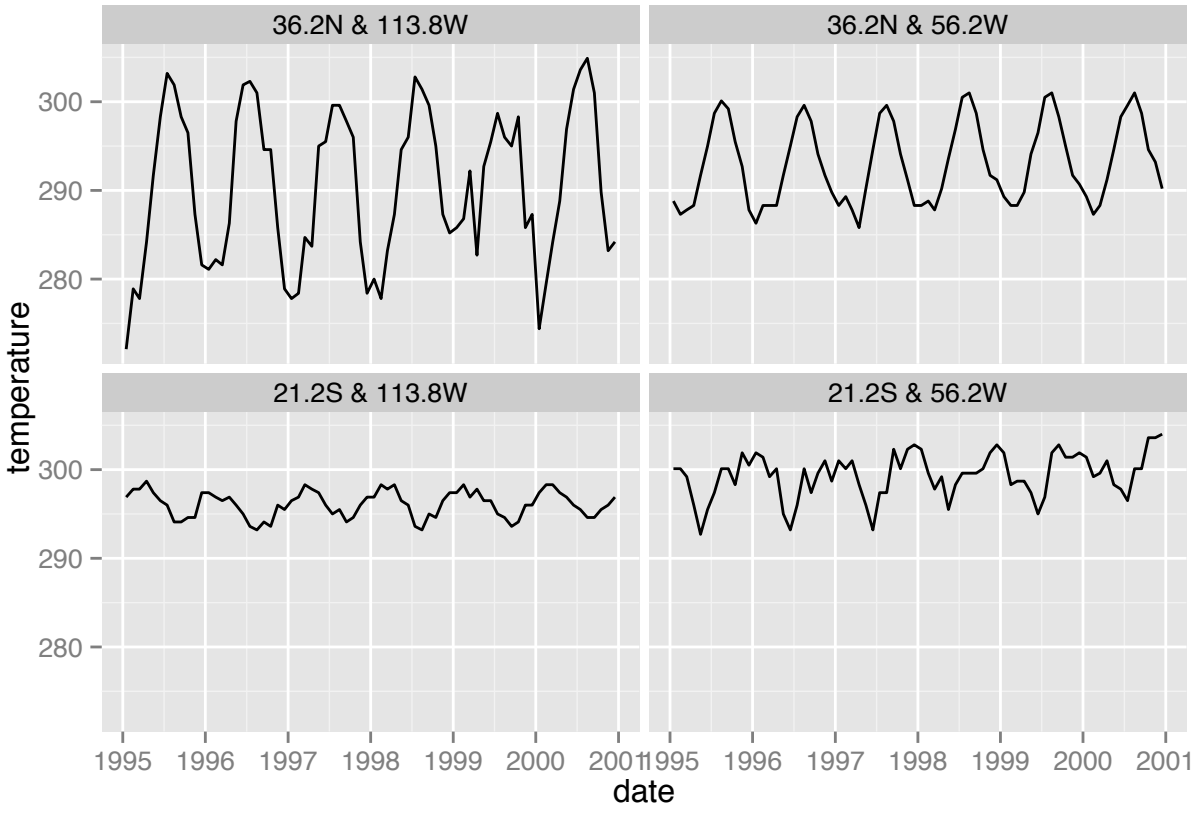
```
lat_lon_corner <- lapply(lat_lon_corner, function(x) {
    x <<- data.frame(x, cornerType = paste(x[1, 1], "&", x[1, 2]))
})
```

Combine them into one dataframe:

```
lat_lon_corner_combine <- do.call(rbind, lat_lon_corner)
```

Call ggplot by `facet_wrap`:

```
ggplot(lat_lon_corner_combine, aes(x = date, y = temperature)) + geom_line() + facet_wrap(~cornerType)
```



It shows strong pattern of seasonality. Also, region in the south corner varies not as much as that of the north corner.

Part 3

The first step is to create a summary statistic that can generate mean and sd simultaneously.

```
summary_func <- function(x) {
    mean_sd_func <- c(mean, sd)
    unlist(lapply(mean_sd_func, function(y) {
        y(x, na.rm = T)
    }))
}
```

Then split the data by lon and lat, and apply to each column to calculate the mean and variance by the summary function defined in the previous step. After modify their rownames and colnames, merge them together

```
mean_sd_dataNASAInt <- t(sapply(split(dataNASAInt[, 4:11], dataNASAInt[, 1:2]), function(x) {
    apply(x, 2, summary_func)
}))
colnames(mean_sd_dataNASAInt) <- unlist(lapply(c(nameNASA, "elevation"), function(x) {
    lapply(c("_mean", "_sd"), function(y) paste0(x, y))
}))
```

Convert rownames to lat and lon:

```
lat_lon_split <- strsplit(rownames(mean_sd_dataNASAInt), "[.]")
lat_split <- unlist(lapply(lat_lon_split, function(i) paste(i[1], i[2], sep = ".")))
lon_split <- unlist(lapply(lat_lon_split, function(i) paste(i[3], i[4], sep = ".")))
row.names(mean_sd_dataNASAInt) <- NULL
```

Add them to mean_sd_dataNASAInt:

```
mean_sd_dataNASAInt <- data.frame(lat = lat_split, lon = lon_split, mean_sd_dataNASAInt)
head(mean_sd_dataNASAInt)
```

```
##     lat    lon cloudhigh_mean cloudhigh_sd cloudlow_mean cloudlow_sd
## 1  1.2N 101.2W      4.7013889     8.575538      37.17361   12.814538
## 2  1.2S 101.2W      3.6319444     8.377953      29.79167    8.358916
## 3 11.2N 101.2W     22.3541667    18.372215      18.43056    5.636467
## 4 11.2S 101.2W      0.8472222     1.888883      50.50000   11.008639
## 5 13.8N 101.2W     18.8055556    15.715910      18.86111    6.000130
## 6 13.8S 101.2W      0.7222222     1.460486      50.70139    9.386332
##   cloudmid_mean cloudmid_sd ozone_mean ozone_sd pressure_mean pressure_sd
## 1      8.909722    6.737705   255.8889 9.081099          1000           0
## 2      6.159722    6.421699   257.0000 9.452625          1000           0
## 3     13.763889    7.683883   257.6389 8.197055          1000           0
## 4      3.784722    3.172090   258.9722 8.125724          1000           0
## 5     12.680556    7.413494   259.5556 8.690109          1000           0
## 6      4.902778    3.378306   260.7222 8.326476          1000           0
##   surftemp_mean surftemp_sd temperature_mean temperature_sd elevation_mean
## 1      296.6250    2.910677         298.6292       2.233764              0
```

```
## 2      295.8250    3.136462         298.0944       2.331166             0
## 3      300.9417    1.265106         301.4514       1.025999             0
## 4      295.9931    1.706188         296.7986       1.733554             0
## 5      301.4764    1.100511         301.7389       1.027174             0
## 6      295.3389    1.436404         296.0083       1.627990             0
##   elevation_sd
## 1            0
## 2            0
## 3            0
## 4            0
## 5            0
## 6            0
```

Part 4

We need to first convert coordinate to numbers:

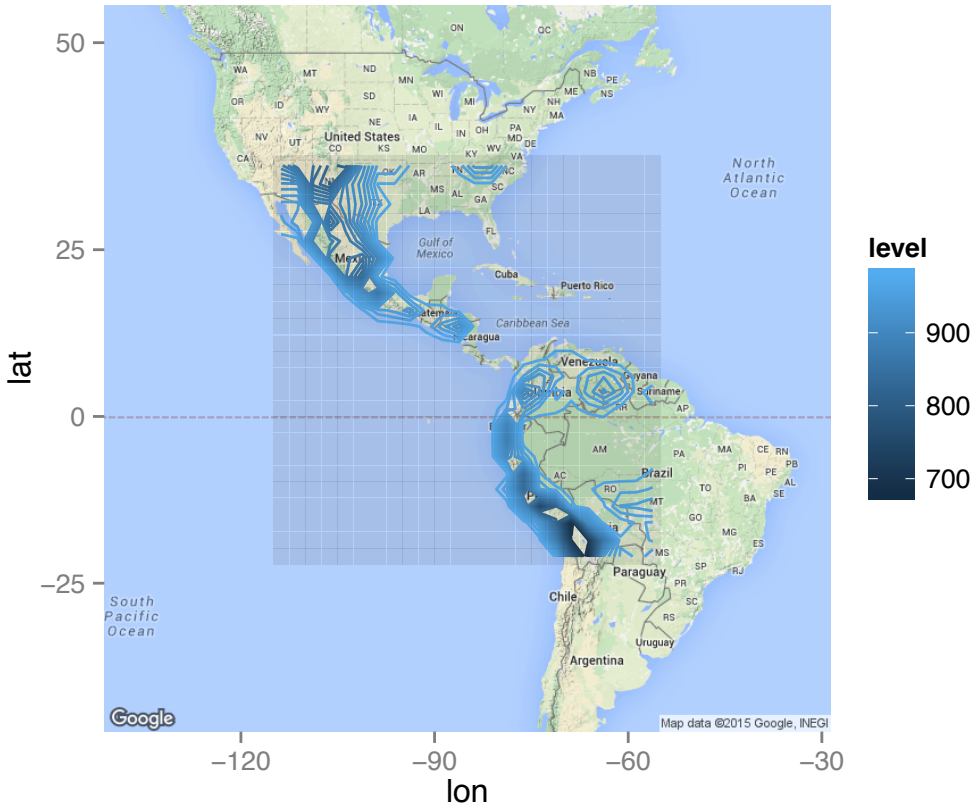
```
coordinateNum <- function(i, lon = F) {
    if (lon == T) {
        return(0 - as.numeric(substr(i, 1, nchar(i) - 1)))
    } else {
        if (substr(i, nchar(i), nchar(i)) == "S") {
            return(0 - as.numeric(substr(i, 1, nchar(i) - 1)))
        } else {
            return(as.numeric(substr(i, 1, nchar(i) - 1)))
        }
    }
}
```

Convert to coordinate number format:

```
mean_sd_dataNASAInt_coordinate_num <- mean_sd_dataNASAInt
mean_sd_dataNASAInt_coordinate_num$lon <- unlist(lapply(as.character(mean_sd_dataNASAInt$lon),
    coordinateNum, lon = T))
mean_sd_dataNASAInt_coordinate_num$lat <- unlist(lapply(as.character(mean_sd_dataNASAInt$lat),
    coordinateNum))
```

Plot map:

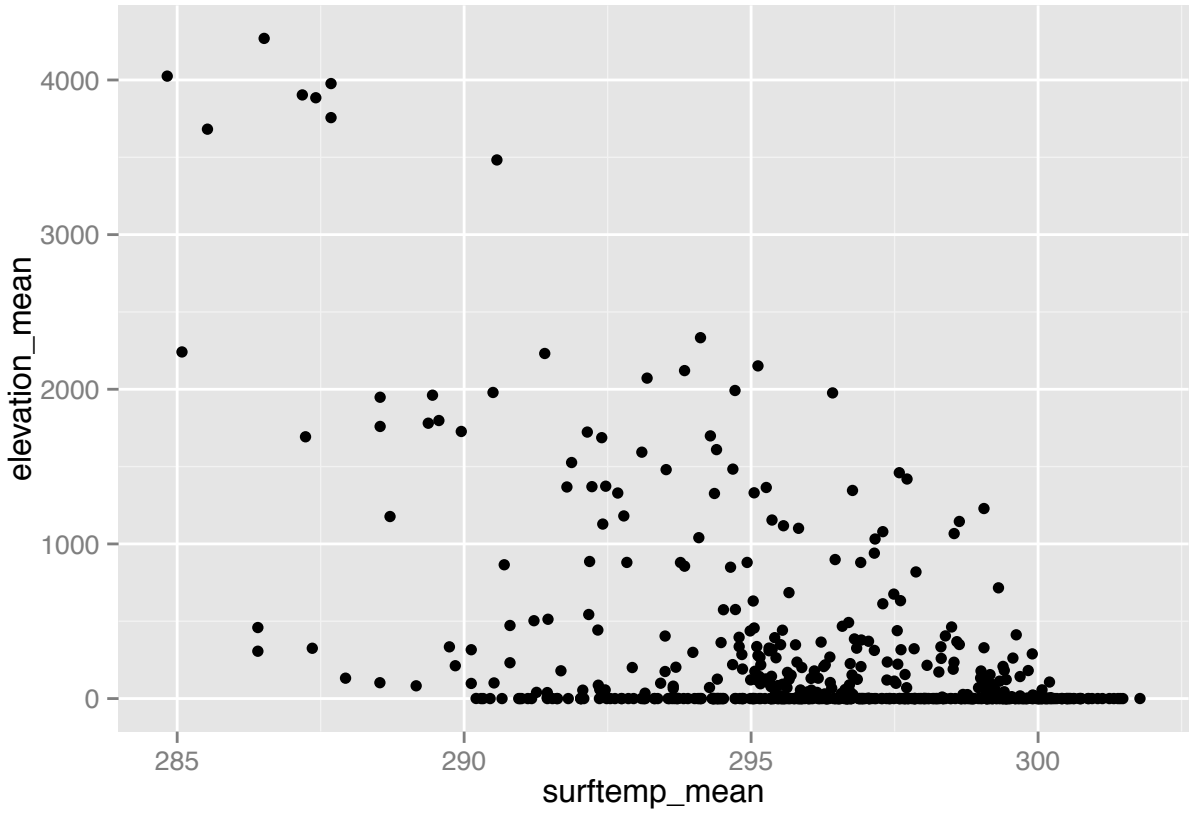
```
library(ggmap)
ggmap(get_googlemap(center = c(mean(unique(mean_sd_dataNASAInt_coordinate_num$lon)),
    mean(unique(mean_sd_dataNASAInt_coordinate_num$lat))), zoom = 3)) + stat_contour(data = mean_sd_data
    aes(lon, lat, z = pressure_mean, color = ..level..), bins = 30) + geom_tile(data = mean_sd_dataNASA
    alpha = 0.1)
```



It is clear that the regions in the off shore areas are with more pressure than others.

Part 5

```
ggplot(mean_sd_dataNASAInt, aes(x = surftemp_mean, y = elevation_mean)) + geom_point()
```


4000
3000
2000
1000
0
elevation_mean
285
290
295
300
surftemp_mean