Analyzing Weather Data

Theo Dimitrasopoulos

2019-12-29

General Information

"50% chance of rain" says the Weather Channel. How did they get this prediction?

Modeling the weather is one of the most complex and difficult problems in data analysis, due in part to the vast amount of data collected and the incredibly complicated system it comes from.

The file `weather_proj.txt` contains meteorogical observations for 1000 weather stations in the US for the year 2012 provided by NOAA (National Oceanic and Atmospheric Administration). Items measured include precipitation, snowfall, minimum / maximum temperature, etc. The data are already in a tidy format.

A note on units from the `readme.txt` provided by NOAA at ftp://ftp.ncdc.noaa.gov/pub/data/ghcn/daily/readme.txt:

- `PRCP = Precipitation (tenths of mm)`
- `SNOW = Snowfall (mm)`
- `SNWD = Snow depth (mm)`
- `TMAX = Maximum temperature (tenths of degrees C)`
- `TMIN = Minimum temperature (tenths of degrees C)`

Many stations do not have all measurements for all days; some record only temperature, or only precipitation. Additionally, some stations do not have any measurements on some days.

The file `stations_proj.txt` contains latitude, longitude, and elevation for the 1000 weather stations found in the previous file. Note that the longitude field is not the actual longitude, but instead a transform of it, so that the western-most point in the data is at longitude 0. This ensures that plotting with longitude does not create a mirror image of the US.

The file `stations_with_loc.txt` contains a field `location` that has the name of the station's location.

General skills used in this project are the following:

- Reading in `.txt` files
- Manipulating data with `dplyr`
  - `separate` from `tidyr`
  - `inner_join`
  - Standard commands: `filter`, `mutate`, `summarize`, etc
- Plotting with `ggplot2`
- Critical thinking about data and transformations
- `dcast` to prepare data for model-fitting
- Fitting a linear model and using ANOVA to compare models
- t-tests
- χ^2 tests

1. Read In Data and Merge

```r
options(stringsAsFactors = FALSE)
library(dplyr)
library(stringr)
library(ggplot2)
library(RColorBrewer)
library(broom)
library(magrittr)
library(reshape2)
```

First, read in `weather_proj.txt` and `stations_proj.txt` and store then as data frames.

```r
weatherProj <- read.table("weather_proj.txt", header=TRUE)
head(weatherProj)
```

```
##       station     date obs_type obs_value
## 1 US10chey021 20120101     PRCP         0
## 2 US10chey021 20120101     SNOW         5
## 3 US10chey021 20120101     SNWD         5
## 4 US10chey021 20120101     WESF         0
## 5 US10chey021 20120102     PRCP         0
## 6 US10chey021 20120102     SNOW         0
```

```r
tail(weatherProj)
```

```
##             station     date obs_type obs_value
## 1231433 USW00094224 20121231     AWND        32
## 1231434 USW00094224 20121231     WDF2        60
## 1231435 USW00094224 20121231     WDF5        60
## 1231436 USW00094224 20121231     WSF2        67
## 1231437 USW00094224 20121231     WSF5        89
## 1231438 USW00094224 20121231     WT16         1
```

```r
stationsProj <- read.table("stations_proj.txt", header=TRUE)
head(stationsProj)
```

```
##       station     lat    long elevation
## 1 US10chey021 41.1554 62.4556    1282.9
## 2 US10chey028 41.0317 62.5123    1296.9
## 3 US10clay009 40.6115 67.1627     570.9
## 4 US10cumi003 41.8664 68.6405     431.0
## 5 US10cumi012 42.0149 68.8644       0.0
## 6 US10keit008 41.1234 63.6888     990.0
```

```r
tail(stationsProj)
```

```
##          station     lat    long elevation
## 995  USW00094012 48.5428 55.6767     787.9
## 996  USW00094040 40.2064 64.8486     771.1
## 997  USW00094084 48.9675 63.2697     561.4
## 998  USW00094128 41.7872 53.5867    1357.6
## 999  USW00094178 42.4819 50.9531    1265.2
## 1000 USW00094224 46.1569 41.5575       2.7
```

Then, I merge the two datasets

```r
proj <- inner_join(weatherProj, stationsProj, by="station")
head(proj)
```

```
##       station     date obs_type obs_value     lat    long elevation
## 1 US10chey021 20120101     PRCP         0 41.1554 62.4556    1282.9
## 2 US10chey021 20120101     SNOW         5 41.1554 62.4556    1282.9
## 3 US10chey021 20120101     SNWD         5 41.1554 62.4556    1282.9
## 4 US10chey021 20120101     WESF         0 41.1554 62.4556    1282.9
## 5 US10chey021 20120102     PRCP         0 41.1554 62.4556    1282.9
## 6 US10chey021 20120102     SNOW         0 41.1554 62.4556    1282.9
```

```r
tail(proj)
```

```
##             station     date obs_type obs_value     lat    long elevation
## 1231433 USW00094224 20121231     AWND        32 46.1569 41.5575       2.7
## 1231434 USW00094224 20121231     WDF2        60 46.1569 41.5575       2.7
## 1231435 USW00094224 20121231     WDF5        60 46.1569 41.5575       2.7
## 1231436 USW00094224 20121231     WSF2        67 46.1569 41.5575       2.7
## 1231437 USW00094224 20121231     WSF5        89 46.1569 41.5575       2.7
## 1231438 USW00094224 20121231     WT16         1 46.1569 41.5575       2.7
```

I Extract the day and month from the date field

```r
dateToDay <- function(date) {
  date%%100
}
dateToMonth <- function(date) {
  as.integer(date/100)%%100
}

projDayMonth <- proj %>%
  mutate(day = dateToDay(date), month = dateToMonth(date)) %>%
  select(-date)
head(projDayMonth)
```

```
##       station obs_type obs_value     lat    long elevation day month
## 1 US10chey021     PRCP         0 41.1554 62.4556    1282.9   1     1
## 2 US10chey021     SNOW         5 41.1554 62.4556    1282.9   1     1
## 3 US10chey021     SNWD         5 41.1554 62.4556    1282.9   1     1
## 4 US10chey021     WESF         0 41.1554 62.4556    1282.9   1     1
## 5 US10chey021     PRCP         0 41.1554 62.4556    1282.9   2     1
## 6 US10chey021     SNOW         0 41.1554 62.4556    1282.9   2     1
```

```r
tail(projDayMonth)
```

```
##             station obs_type obs_value     lat    long elevation day month
## 1231433 USW00094224     AWND        32 46.1569 41.5575       2.7  31    12
## 1231434 USW00094224     WDF2        60 46.1569 41.5575       2.7  31    12
## 1231435 USW00094224     WDF5        60 46.1569 41.5575       2.7  31    12
## 1231436 USW00094224     WSF2        67 46.1569 41.5575       2.7  31    12
## 1231437 USW00094224     WSF5        89 46.1569 41.5575       2.7  31    12
## 1231438 USW00094224     WT16         1 46.1569 41.5575       2.7  31    12
```

2. Some Summaries

What was the hottest temperature measured in the US in 2012? Where was this temperature measured? Repeat this for the coldest temperature.

```
stationsWithLoc <- read.table("stations_with_loc.txt", sep="\t",
                              header=TRUE, quote="")
head(stationsWithLoc)
```

```
##       station     lat     long elevation               location
## 1 ACW00011604 17.1167 -61.7833      10.1  ST JOHNS COOLIDGE FLD
## 2 ACW00011647 17.1333 -61.7833      19.2               ST JOHNS
## 3 AE000041196 25.3330  55.5170      34.0    SHARJAH INTER. AIRP
## 4 AEM00041194 25.2550  55.3640      10.4             DUBAI INTL
## 5 AEM00041217 24.4330  54.6510      26.8         ABU DHABI INTL
## 6 AEM00041218 24.2620  55.6090     264.9            AL AIN INTL
```

```
tail(stationsWithLoc)
```

```
##            station    lat   long elevation          location
## 100300 ZI000067965 20.017 28.617      1326  BULAWAYO AIRPORT
## 100301 ZI000067969 21.050 29.367       861    WEST NICHOLSON
## 100302 ZI000067975 20.067 30.867      1095          MASVINGO
## 100303 ZI000067977 21.017 31.583       430     BUFFALO RANGE
## 100304 ZI000067983 20.200 32.616      1132          CHIPINGE
## 100305 ZI000067991 22.217 30.000       457        BEITBRIDGE
```

```
tempProj <- proj %>%
  filter(obs_type == "TMIN" | obs_type == "TMAX")

getStationLoc <- function(stationRaw) {
  tempLoc <- stationsWithLoc %>%
    filter(station == stationRaw)
  tempLoc[1,]$location
}

tempMax <- tempProj[which.max(tempProj$obs_value),]
sprintf("Hottest temp (Celsius): %f; recorded in %s", tempMax$obs_value/10,
        getStationLoc(tempMax$station))
```

```
## [1] "Hottest temp (Celsius): 47.600000; recorded in  LAKE HAVASU CITY 19 SE"
```

```
tempMin <- tempProj[which.min(tempProj$obs_value),]
sprintf("Hottest temp (Celsius): %f; recorded in %s", tempMin$obs_value/10,
        getStationLoc(tempMin$station))
```

```
## [1] "Hottest temp (Celsius): -54.400000; recorded in  FORT YUKON "
```

Which station got the most rain over the course of 2012?

```
rainProj <- proj %>%
  filter(obs_type == "PRCP") %>%
  group_by(station) %>%
  summarize(rainTot = sum(obs_value)) %>%
  select(station, rainTot)

station <- rainProj[which.max(rainProj$rainTot),]$station
sprintf("Station that got the most rain: %s in %s", station,
        getStationLoc(station))
```

```
## [1] "Station that got the most rain: USS0021A09S in  Marten Ridge"
```

Which date has the most recorded observations?

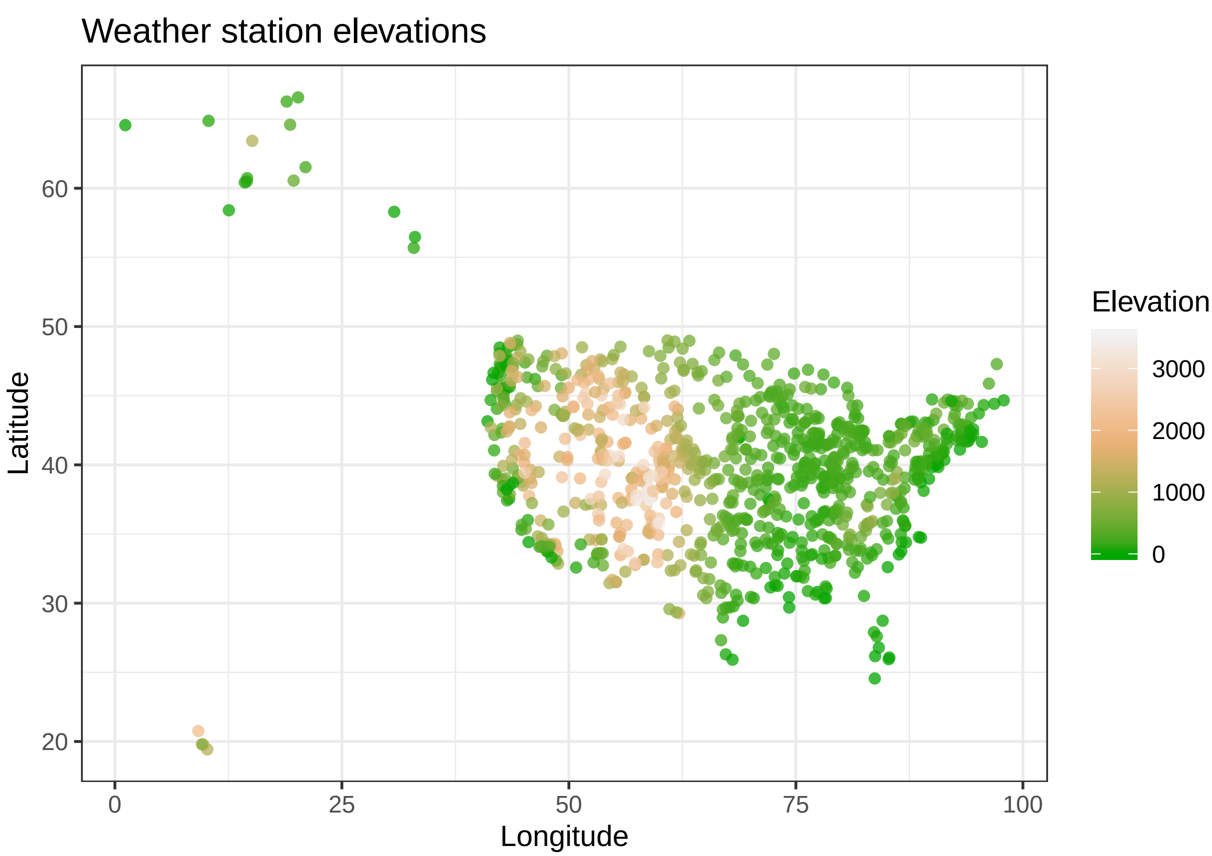
```
obsProj <- proj %>%
  group_by(date) %>%
  summarize(obsTot = n()) %>%
  select(date, obsTot)

obsProj[which.max(obsProj$obsTot),]$date
```

```
## [1] 20120213
```

3. Visualization

This plot shows the elevations of weather stations in the US in geographical space.

```
stationElevation <- stationsProj %>%
  filter(elevation >= 0) # Excluding station with bad value

ggplot(data=stationElevation, mapping=aes(x=long,y=lat,color=elevation)) +
  geom_point(alpha=0.75) +
  scale_color_gradientn(colors=terrain.colors(3), name="Elevation") +
  theme_bw() +
  ggtitle("Weather station elevations") +
  labs(x="Longitude", y="Latitude")
```

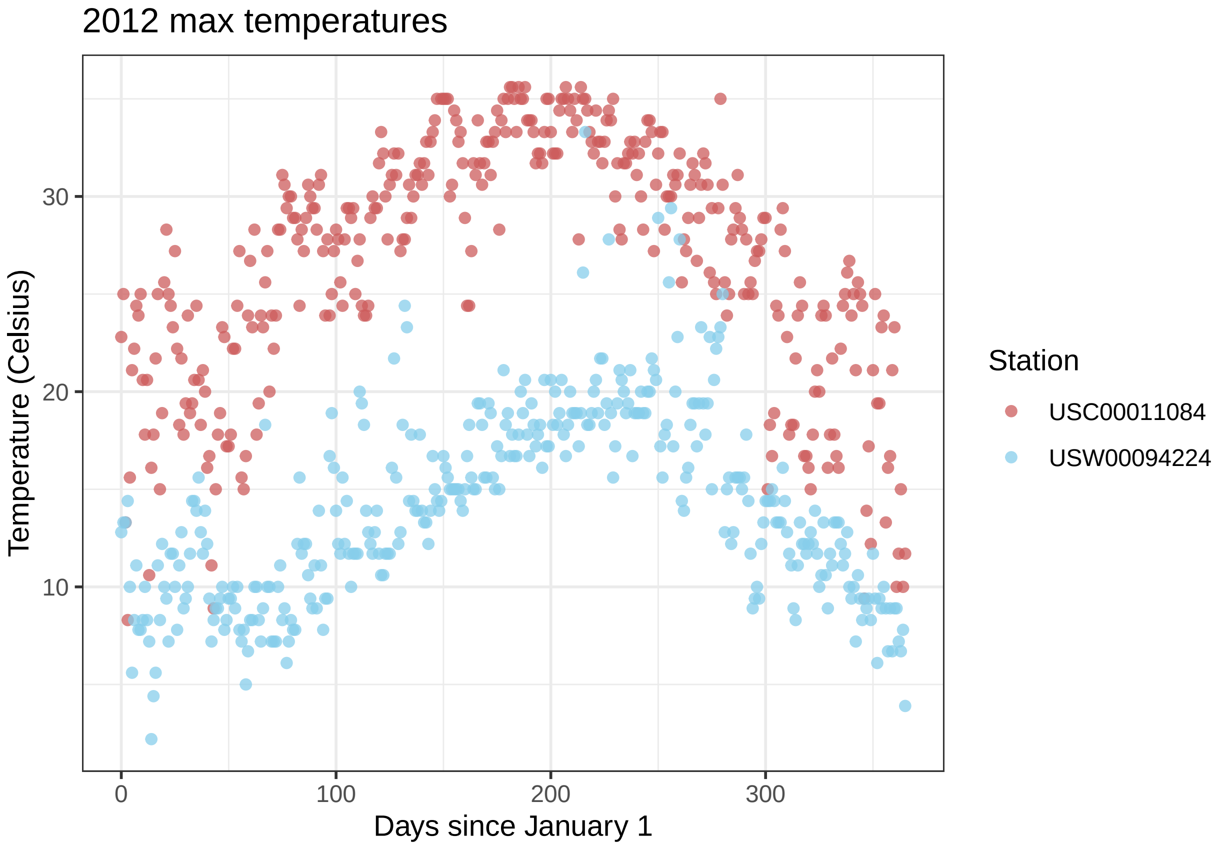


The map is sightly compressed on the x axis and that is because the longitude scale is larger than the latitude scale and is therefore slightly skewed. In order to solve this we can give more room in the y axis and normalize the bed so that the plot has the right aspect ratio.

Part 2

I plot a temperature measure over the course of the year for two weather stations that recorded temperature data.

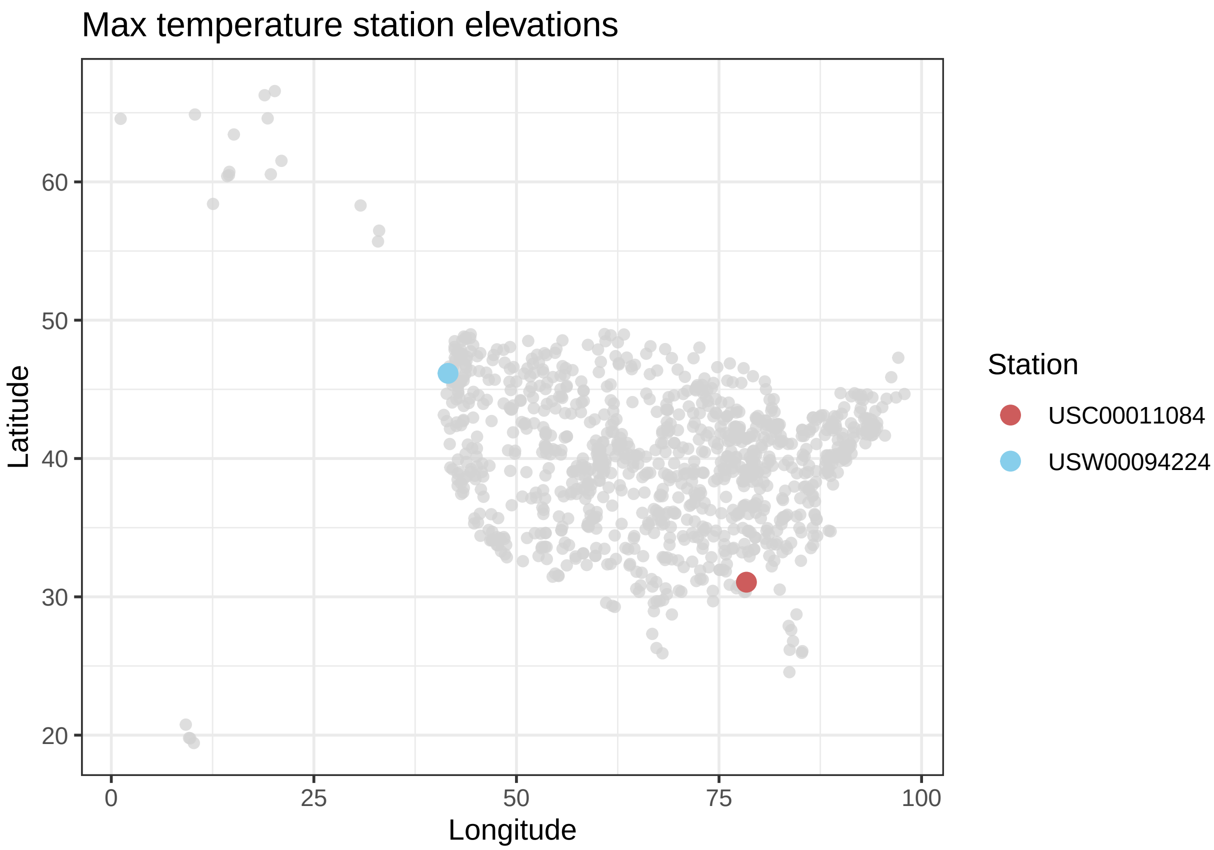
```
# Calculates the number of days since the beginning of the year
calcDays <- function(month, day) {
  as.integer(difftime(ISOdate(2012, month, day), ISOdate(2012,1,1), units="days"))
}

chosenStations <- c("USC00011084", "USW00094224")

stationTemps <- projDayMonth %>%
  filter(obs_type == "TMAX",
         station == chosenStations[1] | station == chosenStations[2]) %>%
  mutate(days = calcDays(month, day), temp = obs_value/10) %>%
  select(days, temp, station)

ggplot(data=stationTemps, mapping=aes(x=days, y=temp, color=station)) +
  geom_point(alpha=0.75) +
  theme_bw() +
  scale_color_manual(values=c("indianred", "skyblue"), name="Station") +
  labs(x="Days since January 1", y="Temperature (Celsius)") +
  ggtitle("2012 max temperatures")
```



Part 3

Here I illustrate the two stations that I chose in the previous step.

```r
chosenStationsProj <- stationsProj %>%
  filter(station == chosenStations[1] | station == chosenStations[2])

ggplot(data=stationsProj, mapping=aes(x=long,y=lat)) +
  geom_point(alpha=0.75, color="lightgray") +
  theme_bw() +
  ggtitle("Max temperature station elevations") +
  labs(x="Longitude", y="Latitude") +
  geom_point(data=chosenStationsProj, mapping=aes(x=long, y=lat, color=station),
             size=3) +
  scale_color_manual(values=c("indianred", "skyblue"), name="Station")
```

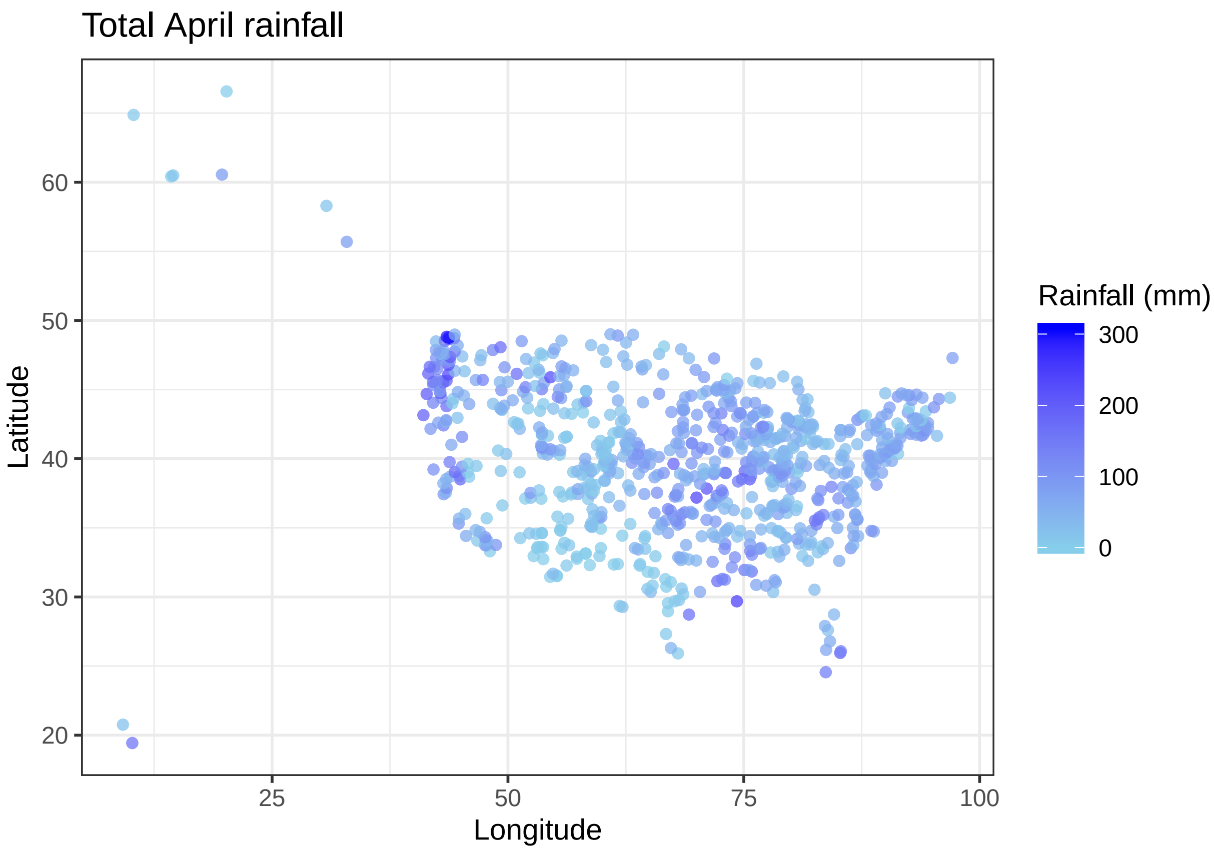


Part 4

To examine rainfall data, I make a spatial plot of the US using the total rainfall over a particular month.

```r
aprilShowers <- projDayMonth %>%
  filter(month == 4, obs_type == "PRCP") %>%
  group_by(lat, long) %>%
  summarize(totRain = sum(obs_value)/10)

ggplot(data=aprilShowers, mapping=aes(x=long, y=lat, color=totRain)) +
  geom_point(alpha=0.75) +
  scale_color_gradient(low="skyblue", high="blue", name="Rainfall (mm)") +
  theme_bw() +
  ggtitle("Total April rainfall") +
```

```r
  labs(x="Longitude", y="Latitude")
```



4. Inference

Part 1

After choosing a random weather station, I construct a 90% CI for the probability there is precipitation on any given day.

```r
statPrcp <- weatherProj %>%
  filter(station == "USC00401790", obs_type == "PRCP") %>%
  mutate(prcp = obs_value > 0) %>%
  select(date, prcp)

# Built-in test (random p, as we only want to find the CI)
n <- nrow(statPrcp)
x <- sum(statPrcp$prcp)
binom.test(x=x, n=n, p=0.5, conf.level=0.90)$conf.int # Choose one?
```

```
## [1] 0.2932131 0.3773287
## attr(,"conf.level")
## [1] 0.9
```

```r
prop.test(x=x, n=n, p=0.5, conf.level=0.90)$conf.int
```

```
## [1] 0.2935056 0.3775447
## attr(,"conf.level")
## [1] 0.9
```

```r
# Calculating CI by hand
p.hat <- x/n
z <- qnorm(0.95)
stderr <- sqrt(p.hat*(1-p.hat)/n)
lowerBound <- p.hat - z*stderr
upperBound <- p.hat + z*stderr
sprintf("(%f, %f)", lowerBound, upperBound)
```

```
## [1] "(0.293472, 0.375036)"
```

Here, I compute the p-value via simulation for the following hypothesis test involving stations USW00004725 (Binghampton, NY), USW00014765 (Providence, RI), and USW00014860 (Erie, PA):

- H_0: Precipitation occurring at all three stations on any given day is probabilistically independent. Mathematically, this means:

  $$\Pr(\text{precip. at all three stations}) = \Pr(\text{precip. at station 1}) \cdot \Pr(\text{precip. at station 2}) \cdot \Pr(\text{precip. at station 3})$$

- H_1: Precipitation occurring at all three stations is probabilistically dependent.

```r
rainStat <- c("USW00004725", "USW00014765", "USW00014860")

prcpAllStat <- weatherProj %>%
  filter(obs_type == "PRCP", station %in% rainStat) %>%
  mutate(rain = obs_value > 0) %>%
  select(date, station, rain) %>%
  dcast(date ~ station, value.var = "rain") %>%
  mutate(ALL_STATIONS = USW00004725 & USW00014765 & USW00014860)

days <- nrow(prcpAllStat)
probPrcpStat1 <- sum(prcpAllStat$USW00004725)/days
probPrcpStat2 <- sum(prcpAllStat$USW00014765)/days
probPrcpStat3 <- sum(prcpAllStat$USW00014860)/days
probPrcpAllStat <- sum(prcpAllStat$ALL_STATIONS)/days

# Create some fake data with the real probabilities. I have to generate one triad of data for every sim

multProbRainSim <- function(probPrcpStat1, probPrcpStat2, probPrcpStat3) {
  # Simulate rain over a year using sample probabilities
  prcpSimStat1 <- rbinom(days,1,probPrcpStat1)
  prcpSimStat2 <- rbinom(days,1,probPrcpStat2)
  prcpSimStat3 <- rbinom(days,1,probPrcpStat3)

  # Calculate the probabilities of the data above
  probPrcpSimStat1 <- sum(prcpSimStat1)/days
  probPrcpSimStat2 <- sum(prcpSimStat2)/days
  probPrcpSimStat3 <- sum(prcpSimStat3)/days

  # Calculate the probability of rain in all three stations
  probPrcpSimStat1 * probPrcpSimStat2 * probPrcpSimStat3
}

expNum = 1e4
simPHat <- replicate(expNum, multProbRainSim(probPrcpStat1,
                                              probPrcpStat2,
```

```
                                                  probPrcpStat3))
mu0 <- mean(simPHat)
pVal <- 2*sum(abs(simPHat-mu0) >= abs(probPrcpAllStat-mu0))/expNum
pVal
```

```
## [1] 0
```

Here, I compute the p-value using simulation of the same hypothesis test for the following stations: USC00195984 (Norton, MA), USS0008T01S (Signal Peak Trail, NM), and USC00228374 (Michigan State University), which lie at larger distances between one another.

```
rainStat <- c("USC00195984", "USS0008T01S", "USC00228374")

prcpAllStat <- weatherProj %>%
  filter(obs_type == "PRCP", station %in% rainStat) %>%
  mutate(rain = obs_value > 0) %>%
  select(date, station, rain) %>%
  dcast(date ~ station, value.var = "rain") %>%
  mutate(ALL_STATIONS = USC00195984 & USS0008T01S & USC00228374)

probPrcpStat1 <- sum(prcpAllStat$USC00195984)/days
probPrcpStat2 <- sum(prcpAllStat$USS0008T01S)/days
probPrcpStat3 <- sum(prcpAllStat$USC00228374)/days
probPrcpAllStat <- sum(prcpAllStat$ALL_STATIONS)/days
probPrcpAllStatVect <- replicate(expNum, probPrcpAllStat)

simPHat <- replicate(expNum, multProbRainSim(probPrcpStat1,
                                              probPrcpStat2,
                                              probPrcpStat3))
mu0 <- mean(simPHat)
pVal <- 2*sum(abs(simPHat-mu0) >= abs(probPrcpAllStatVect-mu0))/expNum
pVal
```

```
## [1] 0.8216
```

The p-value is very small when the stations are geographically close to each other, so we can reject H_0; this suggests that the precipitation occurring at all three stations is probabilistically dependent. On the other hand, the p-value is very large when the stations are farther from each other, so we fail to reject H_0; this suggests that the precipitation occurring at all three stations is probabilistically independent. This is not surprising, as one storm could lead to rain at three stations close to each other (leading to dependence), but weather patterns vary more across the country.

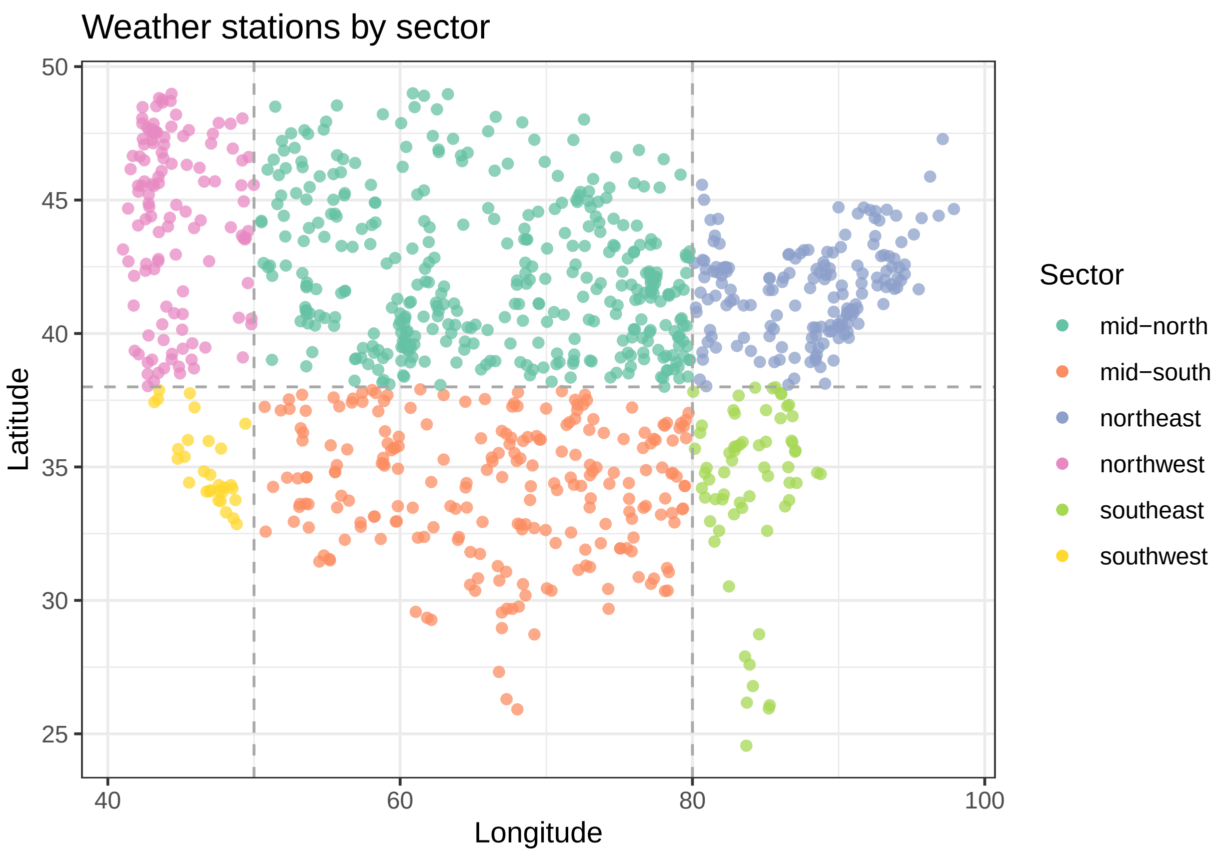
Part 2

One longitude to delineate the east coast, one longitude to delineate the west coast, and one latitude to delineate the northern half of the country from the southern half is chosen to separate the country into sectors. Alaska and Hawaii have been assigned a sector "none", so as to be excluded from further analysis.

```
nsLine <- 38
wLine <- 50
eLine <- 80
noneLine <- 40

getSector <- function(long, lat) {
  ns <- ifelse(lat > nsLine, "north", "south")
```

```
  wme <- ifelse(long > wLine & long < eLine, "mid-",
                ifelse(long < wLine, "west", "east"))
  defSector <- ifelse(wme == "mid-", paste(wme, ns, sep=""),
                      paste(ns, wme, sep=""))
  sector <- ifelse(long < noneLine, "none", defSector)
}

allProjSectors <- stationsProj %>%
  group_by(lat, long) %>%
  summarize(sector = getSector(long, lat))
```

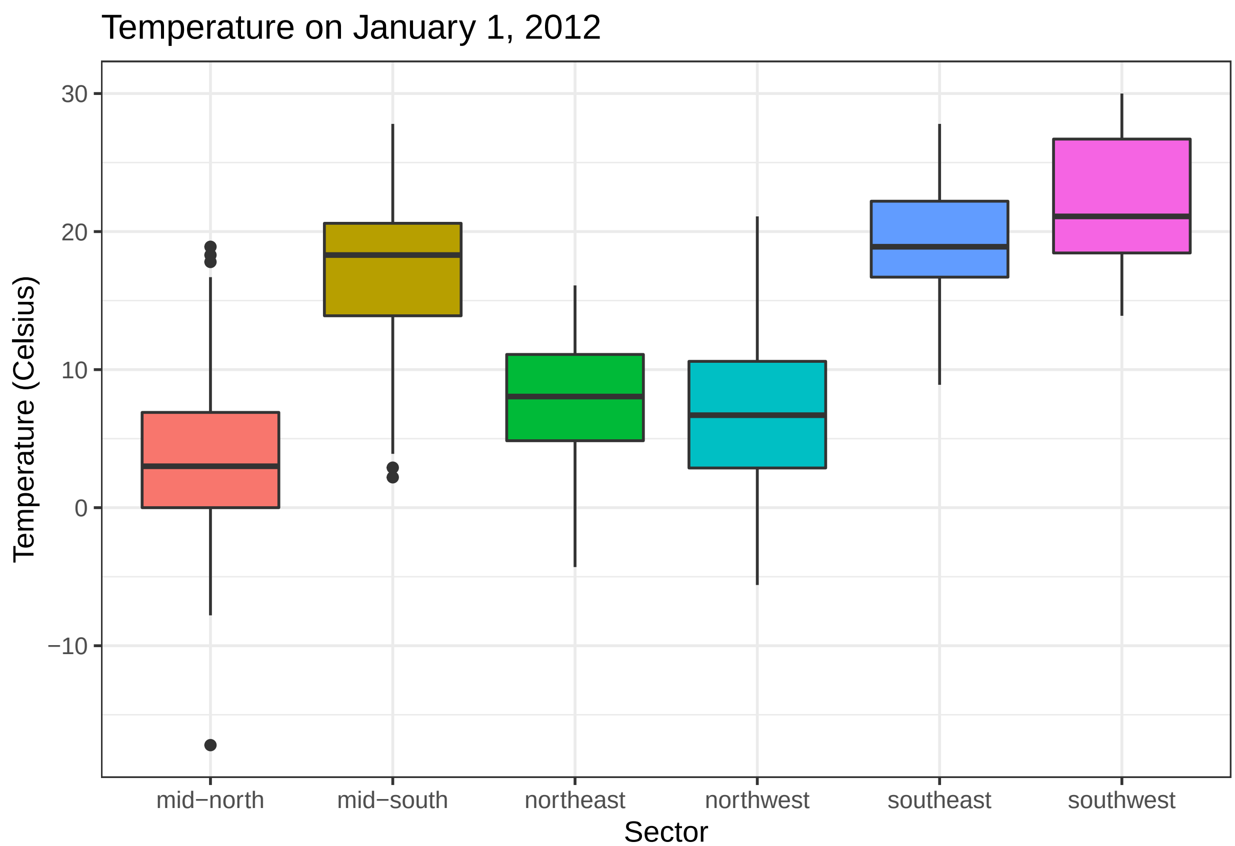
```
projSectors <- allProjSectors %>%
  filter(sector != "none")

ggplot(data=projSectors, mapping=aes(x=long, y=lat, color=sector)) +
  geom_point(alpha=0.75) +
  geom_hline(yintercept=nsLine, color="darkgray", linetype="dashed") +
  geom_vline(xintercept=wLine, color="darkgray", linetype="dashed") +
  geom_vline(xintercept=eLine, color="darkgray", linetype="dashed") +
  scale_color_manual(values=brewer.pal(6, "Set2"), name="Sector") +
  guides(color = guide_legend(override.aes = list(alpha = 1))) +
  labs(x="Longitude", y="Latitude") +
  ggtitle("Weather stations by sector") +
  theme_bw()
```



This boxplot shows the distributions of temperatures in 6 sectors: northwest, northeast, mid-north, southwest, southeast, and mid-south, given the differences in weather patterns across

sectors.

```
tempProjJan1 <- projDayMonth %>%
  filter(day == 1, month == 1, obs_type == "TMAX") %>%
  mutate(sector = getSector(long, lat), obs_value = obs_value/10) %>%
  filter(sector != "none")

sectTempGraph <- ggplot(data=tempProjJan1,
                        mapping=aes(x=sector, y=obs_value, fill=sector)) +
  geom_boxplot() +
  scale_color_manual(values=brewer.pal(6, "Set2")) +
  theme_bw() +
  theme(legend.position="none") +
  labs(x="Sector", y="Temperature (Celsius)") +
  ggtitle("Temperature on January 1, 2012")
sectTempGraph
```



They appear approximately Normal as the whiskers are of similar length on each side, and the median is near the center of the box; there is no dramatic skew.

For all pairs of sectors, I test the null hypothesis that the mean temperatures are equal vs. the alternative that they are unequal.

```
sectors <- c("northwest", "northeast", "mid-north", "southwest", "southeast",
             "mid-south")
getTemps <- function(sect) {
  tempProjJan1 %>%
    filter(sector == sect) %>%
    select(obs_value)
```

```
}
sectorsPValue <- function(sector1, sector2) {
  tidy(t.test(x=getTemps(sector1), y=getTemps(sector2)))
}
sectorTempTests <- as.data.frame(expand.grid(sectors, sectors)) %>%
  rename(sector1 = Var1, sector2 = Var2) %>%
  filter(sector1 != sector2) %>%
  group_by(sector1, sector2) %>%
  do(sectorsPValue(.$sector1, .$sector2))
sectorTempTests %>%
  select(sector1, sector2, p.value)
```

```
## # A tibble: 30 x 3
## # Groups:   sector1, sector2 [30]
##    sector1   sector2    p.value
##    <fct>     <fct>        <dbl>
##  1 northwest northeast 7.40e- 1
##  2 northwest mid-north 1.17e- 5
##  3 northwest southwest 7.58e-18
##  4 northwest southeast 1.05e-18
##  5 northwest mid-south 2.88e-21
##  6 northeast northwest 7.40e- 1
##  7 northeast mid-north 1.83e- 7
##  8 northeast southwest 5.87e-17
##  9 northeast southeast 1.51e-18
## 10 northeast mid-south 7.53e-23
## # ... with 20 more rows
```

The p-value for the northwest/northeast is very high (0.798007), so we cannot reject H_0 under any reasonable cut-off. It is also quite high for the southwest/southeast (0.16539), and somewhat high for the southeast-mid-south (0.0564719). All other p-values are far below a 5% cut-off. Therefore, p-value cut-offs should be catered to the data.

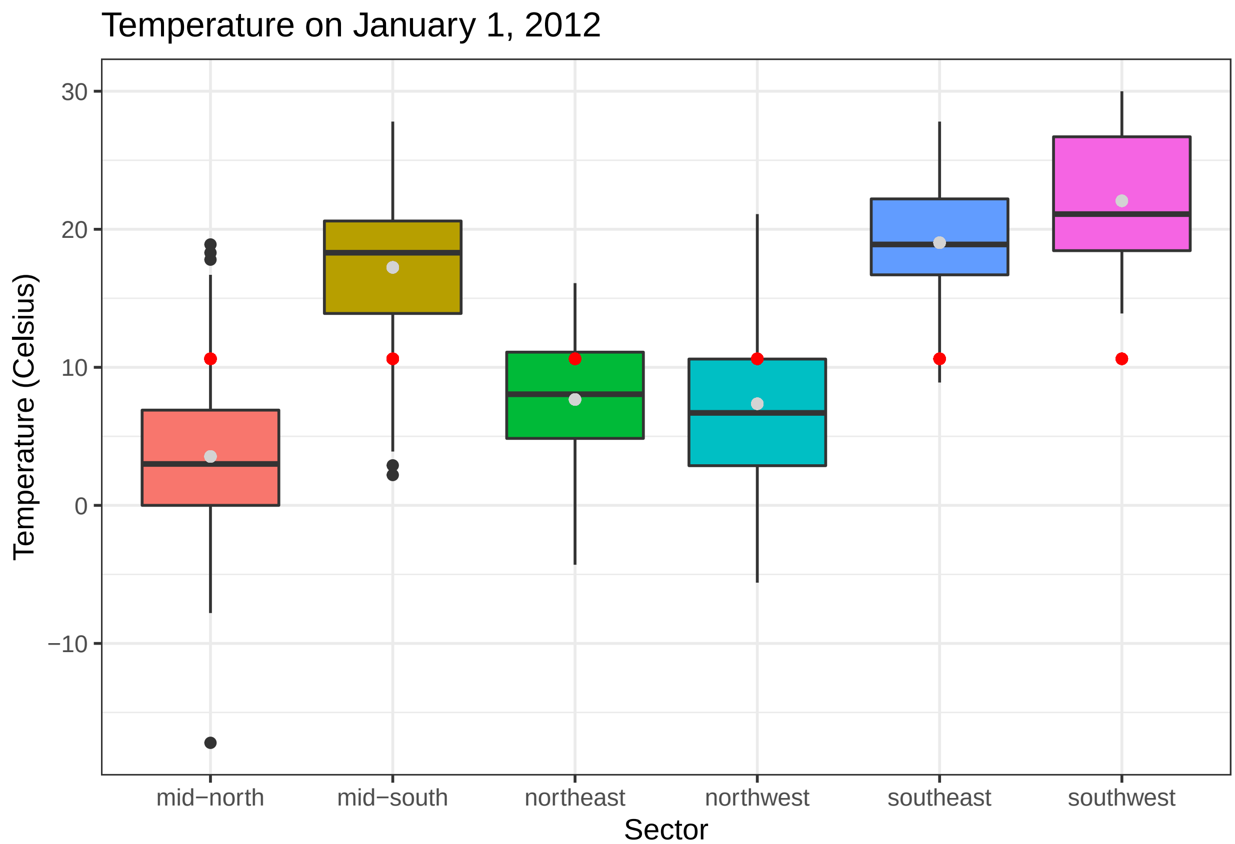
To further confirm the statistical inference, I built a linear regression model that helps test the null hypothesis that all six sectors have equal mean temperature.

```
tempProjMean <- tempProjJan1 %>%
  group_by(sector) %>%
  summarize(mean_temp = mean(obs_value), n = n())
tempProjMean
```

```
## # A tibble: 6 x 3
##   sector    mean_temp     n
##   <chr>         <dbl> <int>
## 1 mid-north      3.54   153
## 2 mid-south     17.2    125
## 3 northeast      7.67    52
## 4 northwest      7.36    72
## 5 southeast     19.0     41
## 6 southwest     22.1     26
```

```
# If all sectors have the same mean
fitConst <- lm(obs_value ~ 1, data=tempProjJan1)
# If sectors have different means
fitSect <- lm(obs_value ~ 1 + sector, data=tempProjJan1)
fitDf <- data.frame(sector=tempProjJan1$sector,
```

```
                    f1=fitConst$fitted.values,
                    f2=fitSect$fitted.values)
sectTempGraph +
  geom_point(aes(x=sector, y=f1), data=fitDf, color="red") +
  geom_point(aes(x=sector, y=f2), data=fitDf, color="lightgray")
```



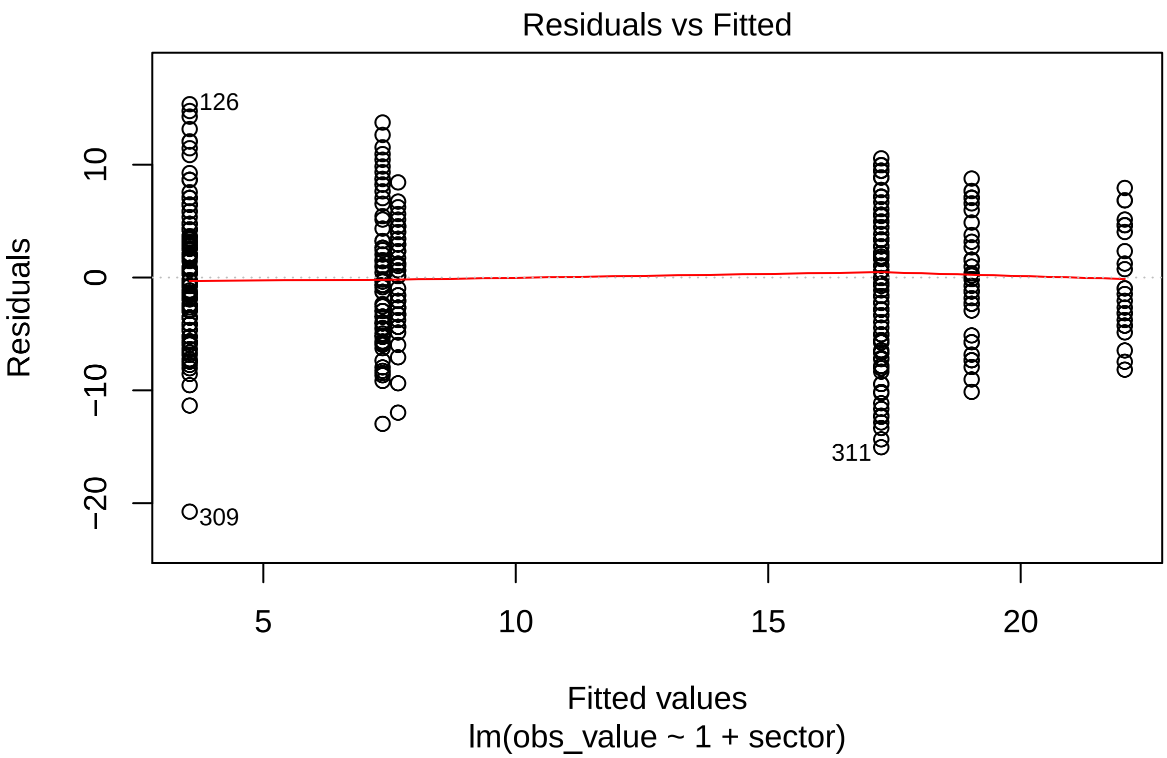

```
anova(fitConst, fitSect)
```

```
## Analysis of Variance Table
##
## Model 1: obs_value ~ 1
## Model 2: obs_value ~ 1 + sector
##   Res.Df   RSS Df Sum of Sq      F    Pr(>F)
## 1    468 34912
## 2    463 14252  5     20660 134.24 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

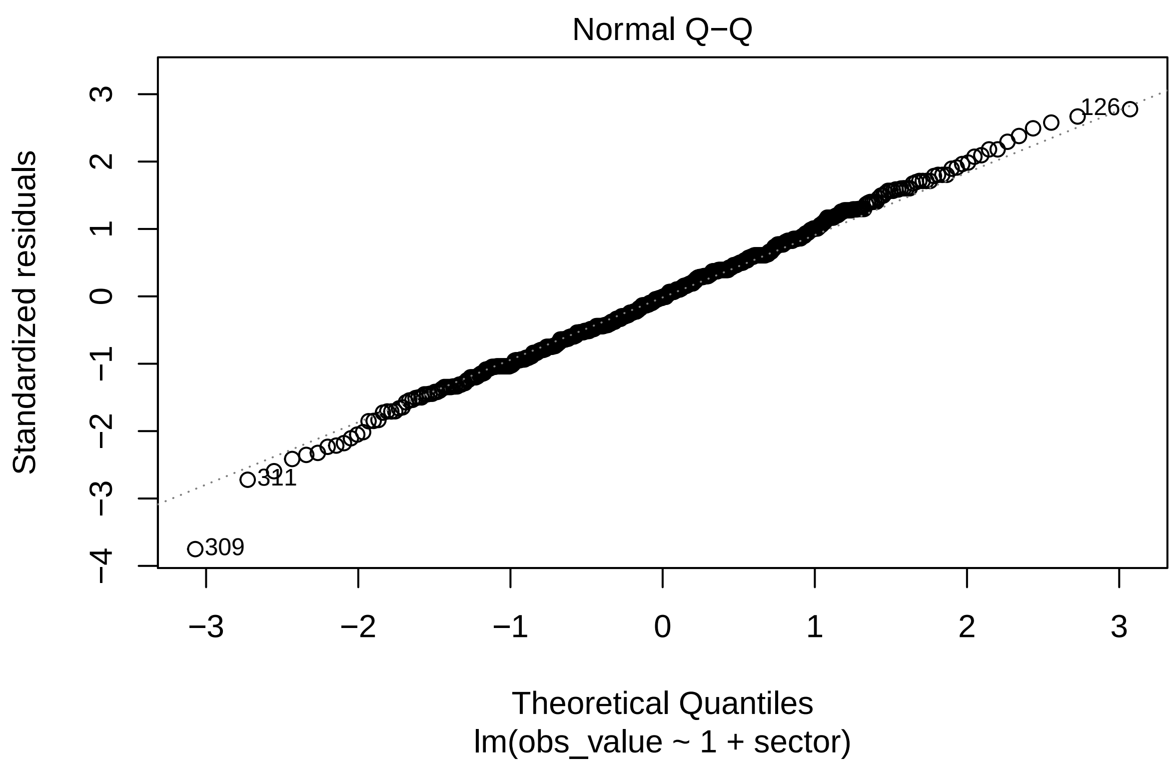
As the F-statistic is very big and its p-value is very small, we reject the null hypothesis; the six sectors do not have an equal mean temperature (visualized by the red and gray dots being significantly far apart from each other).

Assumptions for a linear hypothesis may be partially confirmed with the following two graphs, which show that the fitted values and residuals show no trends with respect to each other and that the residuals are distributed approximately normally. However, we cannot assume that there are no lurking variables.

```
plot(fitSect, which=1)
```




```r
plot(fitSect, which=2)
```

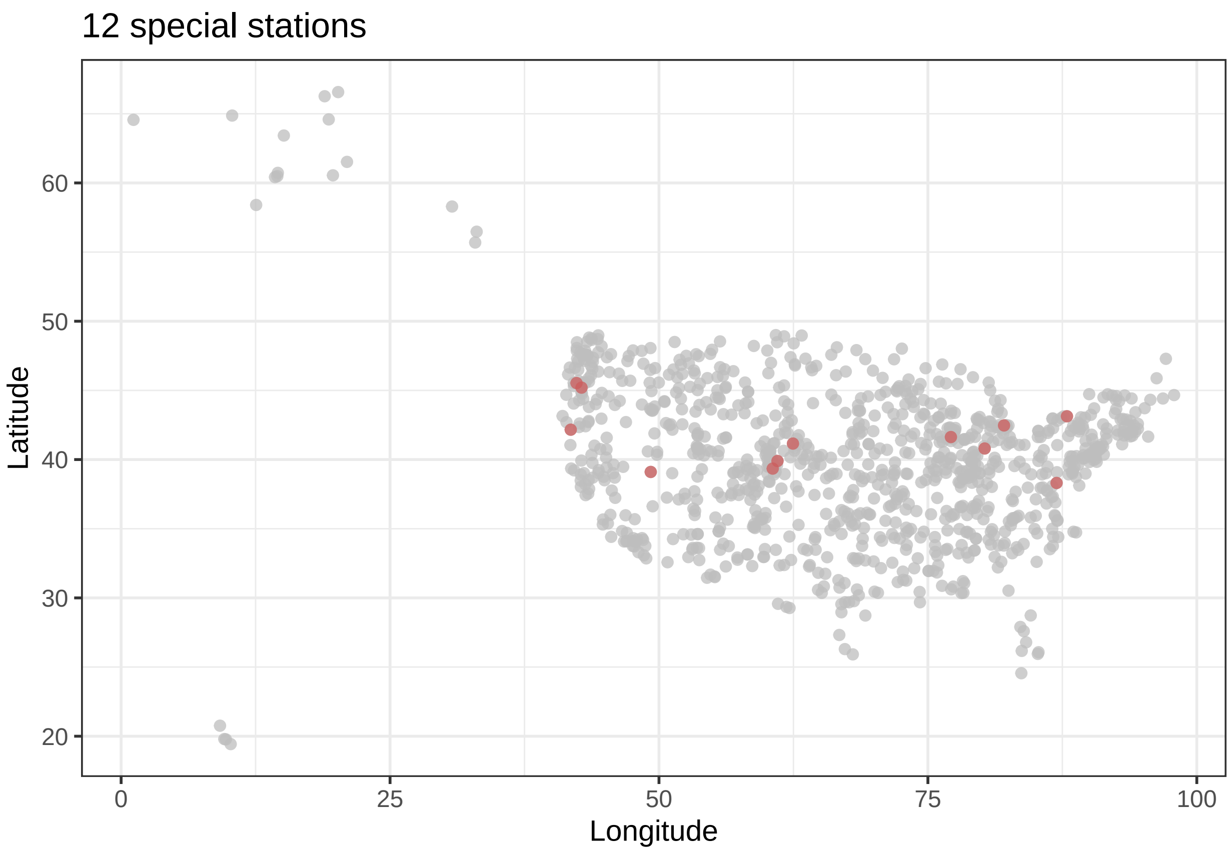


Part 3

Twelve stations are provided below in the `rain_stations` vector, four from each geographical sector northwest, mid-north, and northeast.

```
rain_stations = c("US1NYMR0018", "US1VAGN0001", "USC00202691",
                  "US1INWL0002", "US1CODG0062", "US10chey021",
                  "US1ILKD0024", "US1COAD0135", "USC00263964",
                  "US1ORCC0003", "US1ORWS0037", "USC00351448")
```

```
rainStatProj <- stationsProj %>%
  filter(station %in% rain_stations)
ggplot(data=stationsProj, mapping=aes(x=long, y=lat)) +
  geom_point(color="gray", alpha=0.75) +
  geom_point(data=rainStatProj, color="indianred", alpha=0.75) +
  theme_bw() +
  labs(x="Longitude", y="Latitude") +
  ggtitle("12 special stations")
```



To utilize snow data, I compute the total number of snows days and non-snow days for each of these stations and create a table of these values.

```
snowDays <- proj %>%
  filter(station %in% rain_stations, obs_type == "SNOW") %>%
  group_by(station, long, lat) %>%
  summarize(snow_days = n(), non_snow_days = 366 - snow_days)
snowDays
```

```
## # A tibble: 12 x 5
## # Groups:   station, long [12]
##    station      long   lat snow_days non_snow_days
##    <chr>       <dbl> <dbl>     <int>         <dbl>
##  1 US10chey021  62.5  41.2       366             0
```

```
##   2 US1COAD0135  61.0  39.9       273          93
##   3 US1CODG0062  60.6  39.3       286          80
##   4 US1ILKD0024  77.1  41.6       278          88
##   5 US1INWL0002  80.3  40.8       237         129
##   6 US1NYMR0018  87.9  43.1        40         326
##   7 US1ORCC0003  42.8  45.2       160         206
##   8 US1ORWS0037  42.3  45.5       132         234
##   9 US1VAGN0001  87.0  38.3       234         132
## 10 USC00202691  82.1  42.5        54         312
## 11 USC00263964  49.2  39.1       238         128
## 12 USC00351448  41.8  42.2         9         357
```

Then, I perform a hypothesis test to determine if snow days are independently distributed across the three sectors.

```r
snowDaysSector <- snowDays %>%
  mutate(sector = getSector(long, lat)) %>%
  group_by(sector) %>%
  summarize(snow_days = sum(snow_days), non_snow_days = sum(non_snow_days))
snowDaysSector
```

```
## # A tibble: 3 x 3
##   sector    snow_days non_snow_days
##   <chr>         <int>         <dbl>
## 1 mid-north      1203           261
## 2 northeast       565           899
## 3 northwest       539           925
```

```r
chisq.test(select(snowDaysSector, -sector))
```

```
##
##  Pearson's Chi-squared test
##
## data:  select(snowDaysSector, -sector)
## X-squared = 774.85, df = 2, p-value < 2.2e-16
```

In this case, our H_0 is that snow days are independently distributed across the three sectors; our H_0 is that they are dependent. The p-value from our chi-squared test is very small (below 0.05), so we can reject H_0 with a confidence level of 99.5%. This suggests that snow days are probabilistically dependent across the sectors.

5. Fitting A Model

Transforming Data

Now we want to build a linear model to model variation in the `TMAX` temperature variable in terms of other variables.

The date in the original data set cannot be used as the number of days per month is not from 0-99; as a result, jumps occur as there are no dates between the end of one month and the start of the next.

The days from January 1 transformation that I applied is not appropriate for linear modeling either, as the relationship is parabolic.

In order to do this, I identify and apply a transformation to the date variable that is appropriate for fitting a linear model of temperature on date.

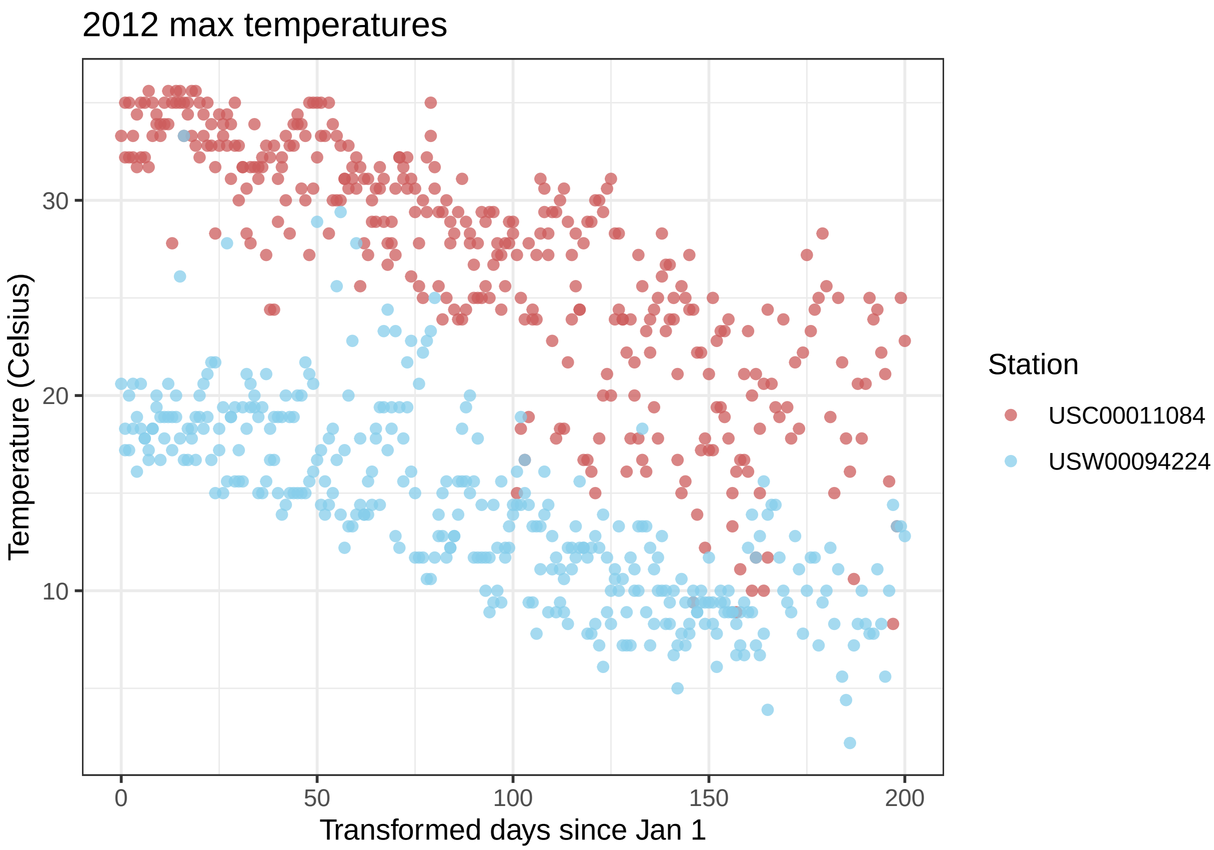
```r
chosenStations <- c("USC00011084", "USW00094224")
```

```r
stationTempsByDays <- projDayMonth %>%
  filter(obs_type == "TMAX") %>%
  mutate(days = calcDays(month, day), temp = obs_value/10) %>%
  select(-c(day, month, obs_type, obs_value))

transformedStationTempsByDays <- stationTempsByDays %>%
  mutate(transformed_days = abs(days-200))

head(transformedStationTempsByDays)
```

```
##       station     lat   long elevation days temp transformed_days
## 1 USC00011084 31.0583 78.385      25.9    0 22.8              200
## 2 USC00011084 31.0583 78.385      25.9    1 25.0              199
## 3 USC00011084 31.0583 78.385      25.9    2 13.3              198
## 4 USC00011084 31.0583 78.385      25.9    3  8.3              197
## 5 USC00011084 31.0583 78.385      25.9    4 15.6              196
## 6 USC00011084 31.0583 78.385      25.9    5 21.1              195
```

To confirm the viability of the transofmration, I plot temperature on the y-axis vs. the transformed date variable on the x-axis for two random stations.

```r
chosenStationTempsByDays <- transformedStationTempsByDays %>%
  filter(station == chosenStations[1] | station == chosenStations[2])

ggplot(data=chosenStationTempsByDays,
       mapping=aes(x=transformed_days, y=temp, color=station)) +
  geom_point(alpha=0.75) +
  theme_bw() +
  scale_color_manual(values=c("indianred", "skyblue"), name="Station") +
  labs(x="Transformed days since Jan 1", y="Temperature (Celsius)") +
  ggtitle("2012 max temperatures")
```



The transformation translates the date into the number of days since January 1, and then finds its distance from day 200 (the peak around which the data seemed to be symmetrically linear). This is appropriate as the resulting data appears linear, which is better for a linear fit.

Building a Model

```r
tempfitAll <- lm(temp ~ transformed_days + lat + long + elevation,
                 data=transformedStationTempsByDays)
summary(tempfitAll)
```

```
##
## Call:
## lm(formula = temp ~ transformed_days + lat + long + elevation,
##     data = transformedStationTempsByDays)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -37.753  -3.910   0.149   4.233  23.154
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)       6.712e+01  1.363e-01  492.63   <2e-16 ***
## transformed_days -1.480e-01  2.690e-04 -550.23   <2e-16 ***
## lat              -7.618e-01  2.377e-03 -320.51   <2e-16 ***
## long             -3.594e-02  9.809e-04  -36.64   <2e-16 ***
## elevation        -2.823e-03  1.891e-05 -149.28   <2e-16 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.11 on 175009 degrees of freedom
## Multiple R-squared:  0.7155, Adjusted R-squared:  0.7155
## F-statistic: 1.1e+05 on 4 and 175009 DF,  p-value: < 2.2e-16
```

```r
tidy(tempfitAll)
```

```
## # A tibble: 5 x 5
##   term             estimate std.error statistic   p.value
##   <chr>               <dbl>     <dbl>     <dbl>     <dbl>
## 1 (Intercept)      67.1     0.136         493.  0.
## 2 transformed_days -0.148   0.000269     -550.  0.
## 3 lat              -0.762   0.00238      -321.  0.
## 4 long             -0.0359  0.000981      -36.6 8.85e-293
## 5 elevation        -0.00282 0.0000189    -149.  0.
```

```r
anova(tempfitAll)
```

```
## Analysis of Variance Table
##
## Response: temp
##                      Df   Sum Sq  Mean Sq   F value    Pr(>F)
## transformed_days      1 11200283 11200283 300035.84 < 2.2e-16 ***
## lat                   1  4375694  4375694 117217.13 < 2.2e-16 ***
## long                  1    21340    21340    571.66 < 2.2e-16 ***
## elevation             1   831868   831868  22284.28 < 2.2e-16 ***
## Residuals        175009  6533054       37
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The model manages to linearly fit the temperatures by nature of the transformed date variable. Each term is extremely significant, as the F-Statistic results show us: The transformed date is the one that contributes to the fit the most, followed by the latitude, elevation and longitude. However, I test the importance of longtitude next, despite its relative significance to the other variables.

```r
# Remove longittude (smallest F value/largest p-value)
tempfitNoLong <- lm(temp ~ transformed_days + lat + elevation,
                 data=transformedStationTempsByDays)
summary(tempfitNoLong)
```

```
##
## Call:
## lm(formula = temp ~ transformed_days + lat + elevation, data = transformedStationTempsByDays)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -36.354  -3.974   0.107   4.259  24.018
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)       6.350e+01  9.418e-02   674.3   <2e-16 ***
## transformed_days -1.480e-01  2.700e-04  -548.1   <2e-16 ***
## lat              -7.336e-01  2.257e-03  -325.0   <2e-16 ***
## elevation        -2.551e-03  1.746e-05  -146.1   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 6.133 on 175010 degrees of freedom
## Multiple R-squared:  0.7133, Adjusted R-squared:  0.7133
## F-statistic: 1.451e+05 on 3 and 175010 DF,  p-value: < 2.2e-16
```

```
tidy(tempfitNoLong)
```

```
## # A tibble: 4 x 5
##   term            estimate std.error statistic p.value
##   <chr>              <dbl>     <dbl>     <dbl>   <dbl>
## 1 (Intercept)      63.5     0.0942         674.       0
## 2 transformed_days -0.148   0.000270      -548.       0
## 3 lat              -0.734   0.00226       -325.       0
## 4 elevation        -0.00255 0.0000175     -146.       0
```

```
anova(tempfitNoLong)
```

```
## Analysis of Variance Table
##
## Response: temp
##                      Df   Sum Sq  Mean Sq F value    Pr(>F)
## transformed_days      1 11200283 11200283  297754 < 2.2e-16 ***
## lat                   1  4375694  4375694  116326 < 2.2e-16 ***
## elevation             1   803095   803095   21350 < 2.2e-16 ***
## Residuals        175010  6583167       38
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
anova(tempfitNoLong, tempfitAll)
```

```
## Analysis of Variance Table
##
## Model 1: temp ~ transformed_days + lat + elevation
## Model 2: temp ~ transformed_days + lat + long + elevation
##   Res.Df     RSS Df Sum of Sq      F    Pr(>F)
## 1 175010 6583167
## 2 175009 6533054  1     50113 1342.4 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

I would use the first model, as the ANOVA models showed that the elevation model term was significant; it is also worth noting that the R-squared value decreases (the fit is worse) when the term is removed. Removing the longitude variable in the second model leads to a biased outcome because of its importance in the calculation - in terms of selection the data that was picked up was stripped of a significant variable. More importantly however, the response-like bias in the data is strong and should be attributed to longitude, or the lack thereof.

Session Information

Session information always included for reproducibility!

```
sessionInfo()
```

```
## R version 3.6.2 (2019-12-12)
## Platform: x86_64-apple-darwin15.6.0 (64-bit)
## Running under: macOS Mojave 10.14.6
##
```

```
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] reshape2_1.4.3     magrittr_1.5       broom_0.5.3        RColorBrewer_1.1-2
## [5] ggplot2_3.2.1      stringr_1.4.0      dplyr_0.8.3
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_1.0.3       plyr_1.8.5       pillar_1.4.3     compiler_3.6.2
##  [5] tools_3.6.2      zeallot_0.1.0    digest_0.6.23    evaluate_0.14
##  [9] lifecycle_0.1.0  tibble_2.1.3     gtable_0.3.0     nlme_3.1-142
## [13] lattice_0.20-38  pkgconfig_2.0.3  rlang_0.4.2      cli_2.0.0
## [17] yaml_2.2.0       xfun_0.11        withr_2.1.2      knitr_1.26
## [21] generics_0.0.2   vctrs_0.2.1      grid_3.6.2       tidyselect_0.2.5
## [25] glue_1.3.1       R6_2.4.1         fansi_0.4.0      rmarkdown_2.0
## [29] farver_2.0.1     purrr_0.3.3      tidyr_1.0.0      backports_1.1.5
## [33] scales_1.1.0     htmltools_0.4.0  assertthat_0.2.1 colorspace_1.4-1
## [37] labeling_0.3     utf8_1.1.4       stringi_1.4.3    lazyeval_0.2.2
## [41] munsell_0.5.0    crayon_1.3.4
```