The Effect of Daily Weather Changes on Visits to Mt. Rushmore

Faisal Mahmood

10/07/2017

Introduction

The objective of this report is to analyze the effect of weather on tourism at Mount Rushmore National Park, located in South Dakota. It will be using data of daily visitations to the park from January 1993 to May 2017, combined with daily temperature and precipitation data, to observe any possible link between the weather and the daily visits.

This analysis can be very helpful for clients who benefit from tourism to Mt. Rushmore, including the National Park Service (NPS), as well as local businesses around the Black Hills who cater to Mt. Rushmore tourists. The weather impacts the economy and the decisions people make in their daily lives, and it also affects tourism, so if a substantial link between the weather and the number of visitors to Mt. Rushmore can be found, then the results of this analysis can be used to improve predictions of tourist visits to that park based on weather forecasts. The National Park Service can assign staffers at the park based on the number of expected visits, and both the Park Service and local businesses can use weather forecasts to adjust operations if any link between weather and tourism is found, or to come up with ways to improve the experience for visitors on less desirable days. If such changes can be made, then businesses and the Park Service will be able to improve the efficiency of their operations and decisions.

The sources of data for this analysis are the National Park Service (NPS) and the National Oceanic and Atmospheric Administration (NOAA). The NPS provides data on the number of daily visits to Mt. Rushmore, starting from 1993. NOAA provides daily temperature, precipitation, snowfall, and snow depth data for the Mt. Rushmore weather station. Combining the datasets into one dataframe will enable the analysis.

Data Wrangling

The data for this analysis can be found at the following sites:

- https://irma.nps.gov/Stats/Reports/Park/MORU (NPS)
- http://xmacis.rcc-acis.org/ (NOAA)

At the National Park Service (NPS) site, there is a link, Daily Report, that leads to the daily visitation reports for Mt. Rushmore. Clicking that link will lead to a web page with each available month of data. Monthly data in CSV format can be downloaded, and every month of data starting from January 1993 is available. The CSV files of this data can be found in the Mt-Rushmore-visits-by-month folder of the Mt-Rushmore-analysis Github repository. When downloading the datasets for the visits, name them in chronological order to ensure that they are neatly organized by year and month (as shown in the Github folder), and store them all in one folder for ease of access.

Afterwards, the Mac Terminal can be used to merge all those sets into one CSV file, available as Rushmore-visits-TOTAL.csv in the Mt-Rushmore-analysis Github repository. On the Mac computer, the following procedure can be done to make the Mac Terminal readily available:

Click System Preferences, click Keyboard, and select -> Shortcuts -> Services -> New Terminal at Folder After that step, right-click the folder with the Mt. Rushmore visitation datasets, and open the Mac Terminal

for that folder. In the Terminal, enter the command `cat *.csv >merged.csv` to merge all the sets into one big CSV file titled merged.csv. Then, rename the merged set as Rushmore-visits-TOTAL.csv.

For the weather data, access the NOAA site that contains the datasets for Mt. Rushmore's weather. Under the Single-Station tab, click Daily Data Listing, and select the CSV output. Enter 1993-01-01 as the starting date, and enter 2017-05-31 as the end date. Select Max temp, Min temp, Precipitation, Snowfall, and Snow depth. Under the Station selection tab, search for Mt. Rushmore, SD, and select the station named "MT RUSHMORE NATL MEM". Click Go to view the data in CSV format.

Afterwards, select the entire dataset, and copy and paste it to a word processer in plain text (using TextEdit or a similar program). Use the command-F keyboard function (or control-F on a Windows keyboard) to convert all missing "M" values to NA values by entering "M," in the finder, and then replacing them with "NA,". Save the file as Mt-Rushmore-wx-data.txt.

Open an RStudio or another R console to begin the data wrangling. Set the working directory to the folder where related documents will be found, and then load the following packages:

```
library(readr)
library(dplyr)
library(tidyr)
library(mice)
library(ggplot2)
library(stlplus)
```

The `readr` package enables the data to be recognized by the R console, while `dplyr` and `tidyr` are used for data wrangling. The `mice` package is used for the imputation of missing values in a data set, using the available data to estimate plausible values. Finally, the `ggplot2` package is used for plots, and the `stlplus` package is used for the time series decomposition of data that has some missing values, which in this case is precipitation data that cannot be plausibly fixed via imputation.

Use the `read_csv()` function to load the weather data onto the dataframe `wx_data`. Be sure to set the date format as `%Y-%m-%d` and skip the first 2 rows.

```
wx_data <- read_csv("Mt-Rushmore-wx-data.txt",
                    col_types = cols(Date = col_date(format = "%Y-%m-%d")),
                    skip = 2)
```

Next, shorten the names of the columns depicting the maximum and minimum temperature for simplicity.

```
colnames(wx_data)[colnames(wx_data) == "MaxTemperature"] <- "MaxTemp"
colnames(wx_data)[colnames(wx_data) == "MinTemperature"] <- "MinTemp"
```

Afterwards, the dataframe can be viewed by entering `View(wx_data)`.

To load the dataset for the Mt. Rushmore visits (`Rushmore-visits-TOTAL.csv`), use the same `read_csv()` function and save that set onto a dataframe known as `visits`. Be sure to skip the first 3 rows.

```
visits <- read.csv("Rushmore-visits-TOTAL.csv", skip = 3)
```

The `visits` dataframe is messy, and it will require a substantial amount of cleaning. Many unnecessary rows need to be deleted using the `subset()` function. The date "February 29" can also be deleted for the sake of simplifying the time series decompositions.

```
visits <- subset(visits, Description == "Daily Visitation")
visits <- subset(visits, Date != "February 29")
```

```
visits <- subset(visits, Date != "February 30")
visits <- subset(visits, Date != "February 31")
visits <- subset(visits, Date != "April 31")
visits <- subset(visits, Date != "June 31")
visits <- subset(visits, Date != "September 31")
visits <- subset(visits, Date != "November 31")
```

Although the year is listed for each month in the `Date` column at the beginning of each month, a seperate column for the year will be needed so that each date is listed in a standardized format. The first step will be to create that `Year` column. Since the data begins in 1993 and ends in 2017, and each year has 365 days for the sake of this analysis, a simple equation involving a column that numbers the rows (`normdays`) can be created to form a `Year` column that progresses with each year starting with 1993. Afterwards, that number column, as well as the `PercentChange` and `SameMonthLastYear` columns, can be deleted since they are not necessary for this analysis.

```
rownames(visits) <- seq(length=nrow(visits))
visits$normdays <- as.numeric(rownames(visits))
visits$normdays <- visits$normdays - 1
visits$Year <- 1993 + visits$normdays/365
visits$Year <- floor(visits$Year)
visits$normdays <- NULL
visits$SameMonthLastYear <- NULL
visits$PercentChange <- NULL
```

The column for the number of visits needs to be converted to numeric format, which can easily be done by removing the commas in the numbers and using the `as.numeric()` function. Also, all values should be positive, and the `abs()` function will ensure that.

```
visits$ThisMonth <- gsub("," , "", visits$ThisMonth)
visits$ThisMonth <- as.numeric(visits$ThisMonth)
visits$ThisMonth <- abs(visits$ThisMonth)
```

Now that the `Date` and `Year` columns are neatly organized, they can be merged so that the dates are listed in the standard Year-Month-Day format.

```
visits <- unite(visits, Date, Date, Year, sep = ", ", remove = TRUE)
visits$Date <- as.Date(visits$Date, format = "%B %d, %Y")
visits <- subset(visits, !is.na(Date))
```

For the sake of simplicity, the column `ThisMonth` can be renamed as `DailyVisits`, and the `Description` column can be deleted.

```
colnames(visits)[colnames(visits) == "ThisMonth"] <- "DailyVisits"
visits$Description <- NULL
```

Now that the `visits` dataframe has been cleaned, it can now be saved as another file for reference using the `write.csv()` function, and it can be merged with the weather data using the `merge()` function. The merged dataset can again be written as a CSV file, `Rushmore-data.csv`, using the `write.csv()` function.

```
#Save the cleaned set as a CSV file.
write.csv(visits, "Rushmore_TOTAL-update.csv")

#Merge the weather and daily visit sets.
Rushmore <- merge(visits, wx_data, by = "Date")

#Save the merged Rushmore dataset as a CSV file.
write.csv(Rushmore, "Rushmore-data.csv")
```

Data Imputation

Now that the data for the Mt. Rushmore visits and the weather has been combined into one dataframe, it can be analyzed.

However, there is still a substantial number of days with missing data for temperatures and precipitation. Since we plan to do a time series decomposition for the temperature data, the missing values need to be filled. Fortunately, a solution is available. The `mice` (Multiple Imputation by Chained Equations) package can be downloaded and used for the multiple imputation of the missing values, especially if missing values are randomly distributed. This algorithm inserts values into the missing data points based on the non-missing values.

However, for this imputation to be effective and reasonably accurate, the data will need to be divided on a monthly basis. Average temperatures vary based on the time of year, so missing values should be filled based on other data points with similar average temperatures for the results to be plausible. Otherwise, if missing July temperatures are filled based even partially on January data, for example, the results will be very messy.

The `separate()` function will be used to create a column for each component of the date. The `ifelse()` function will be used to create a column that indicates where missing temperature values are found, so that they can be easily reviewed after the imputations are complete. Meanwhile, the `table()` function indicates exactly how many data points have missing values.

```
Rushmore$NA_temp <- ifelse((is.na(Rushmore$MaxTemp & Rushmore$MinTemp)), 1,0)
table(Rushmore$NA_temp)
```

```
##
##    0    1
## 8787  124
```

```
Rushmore <- separate(Rushmore, Date, c("Year", "Month", "Day"), sep = "-", remove = FALSE)
```

Since the imputations need to be done month by month, each month will have its own dataframe (entitled `Rushmore_01` for January, `Rushmore_02` for February, and so on). The `filter()` function will be used to create such dataframes, and then another dataframe that includes only the temperature data needs to be created for the imputations to be completed. The `set.seed()` function will be used to standardize the results of any inherently random process, and for this analysis, the number 256 will be used for each imputation.

The `Rushmore_Jan_impute <- complete(mice(Rushmore_Jan))` equation carries out the actual imputation and establishes the dataframe `Rushmore_Jan_impute` for the imputed dataset. The data for that can then be placed in the temperature columns of the `Rushmore_01` dataframe, so that the missing values are finally filled.

```
#January
Rushmore_01 <- filter(Rushmore, Month %in% c("01"))
```

```
Rushmore_Jan <- Rushmore_01[c("MaxTemp", "MinTemp")]
summary(Rushmore_Jan)
set.seed(256)
Rushmore_Jan_impute <- complete(mice(Rushmore_Jan))
summary(Rushmore_Jan_impute)
Rushmore_01$MaxTemp <- Rushmore_Jan_impute$MaxTemp
Rushmore_01$MinTemp <- Rushmore_Jan_impute$MinTemp
summary(Rushmore_01)
```

The same process will be repeated for each month, until all missing values for temperature are filled. Once this process is complete, the dataframes will be merged back together, with the dates set in order.

```
#Merge the imputed monthly sets back to one dataset, and set the dates in order.
Rushmore <- bind_rows(Rushmore_01, Rushmore_02, Rushmore_03, Rushmore_04,
                      Rushmore_05, Rushmore_06, Rushmore_07, Rushmore_08,
                      Rushmore_09, Rushmore_10, Rushmore_11, Rushmore_12)

Rushmore <- Rushmore[order(Rushmore$Date), ]
Rushmore$Year <- NULL
Rushmore$Month <- NULL
Rushmore$Day <- NULL
rownames(Rushmore) <- seq(length=nrow(Rushmore))
View(Rushmore)
str(Rushmore)
summary(Rushmore)
```

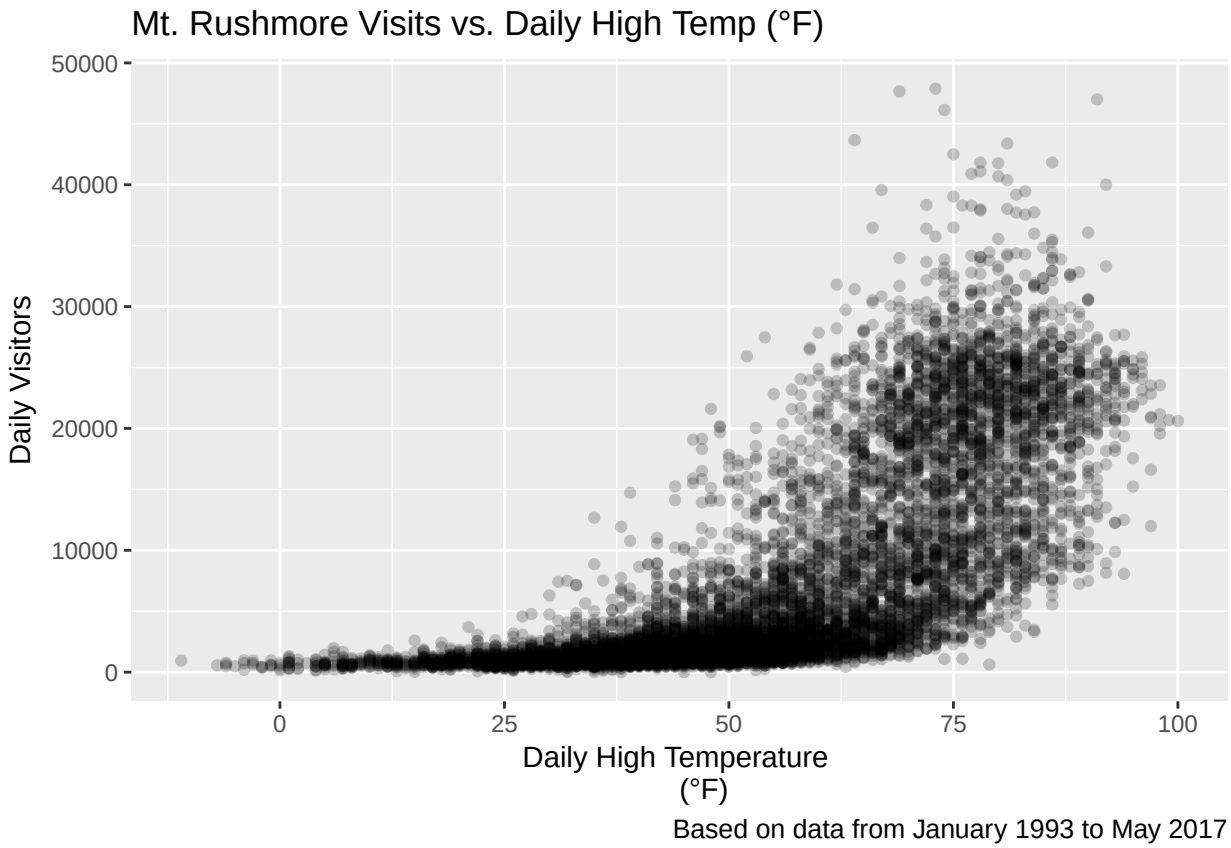
The `NA_temp` column can be viewed to ensure that the imputed values make sense, and then the column can be deleted. However, there was one particular period in early October 2013, in which the imputed values did not make sense. A significant snowstorm occurred at Mt. Rushmore, and the park closed for several days, with no data available for those days. However, the snow depth on the day the park reopened was 12“, yet the imputed temperatures on the missing days were far too warm to look plausible. Therefore, we directly inserted colder temperatures for those days that appeared more plausible, and wrote the code under the `October` imputation.

```
#October
Rushmore_10 <- filter(Rushmore, Month %in% c("10"))
Rushmore_Oct <- Rushmore_10[c("MaxTemp", "MinTemp")]
summary(Rushmore_Oct)
set.seed(256)
Rushmore_Oct_impute <- complete(mice(Rushmore_Oct))
summary(Rushmore_Oct_impute)

Rushmore_10$MaxTemp <- Rushmore_Oct_impute$MaxTemp
Rushmore_10$MinTemp <- Rushmore_Oct_impute$MinTemp
summary(Rushmore_10)

Rushmore_10$MaxTemp[which(Rushmore_10$Date == "2013-10-04")] <- 32
Rushmore_10$MaxTemp[which(Rushmore_10$Date == "2013-10-05")] <- 35
Rushmore_10$MaxTemp[which(Rushmore_10$Date == "2013-10-06")] <- 45
Rushmore_10$MinTemp[which(Rushmore_10$Date == "2013-10-04")] <- 15
summary(Rushmore_10)
```

Statistical Analysis

Much of the data for both the weather and the daily visitations has a significant seasonal variation, which makes sense because Mt. Rushmore is much more desirable to visit in the warmer months rather than during the brutal winters of the northern Plains. For example, the following plot displays the relationship between the number of daily visitors to Mt. Rushmore and the daily high temperature.

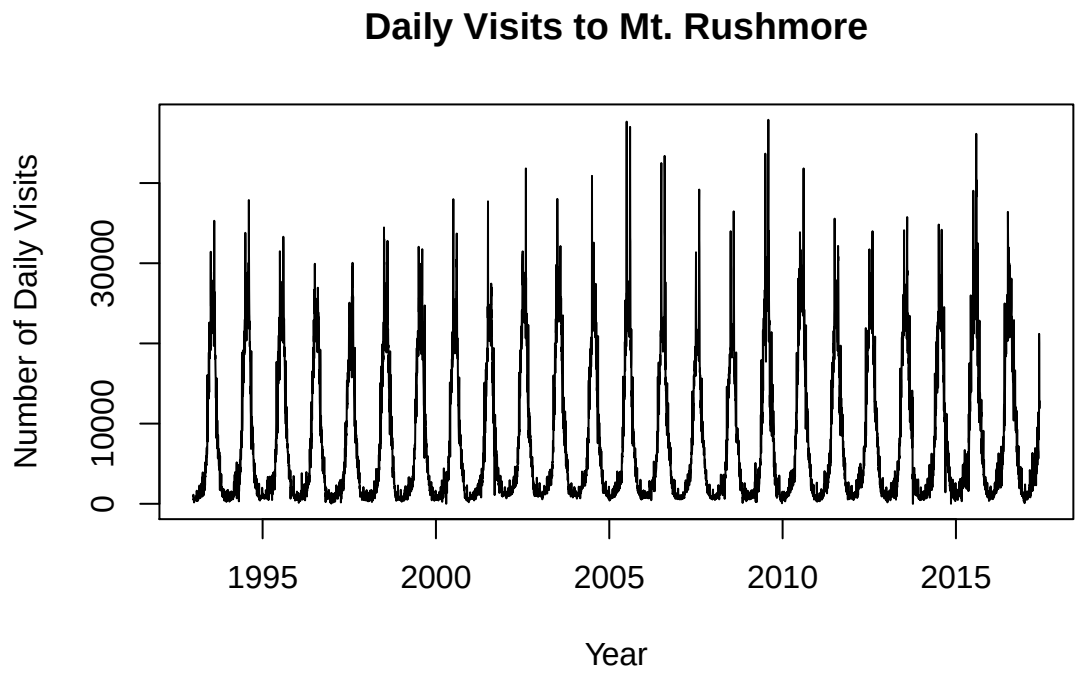
```
ggplot(Rushmore, aes(x = MaxTemp, y = DailyVisits)) + labs(x = "Daily High Temperature
      (°F)", y = "Daily Visitors", title = "Mt. Rushmore Visits vs. Daily High Temp (°F)",
      caption = "Based on data from January 1993 to May 2017") + geom_point(alpha = 0.2)
```



The above plot shows that the number of visitors exponenetially increases with the daily high temperature, with the overwhelming majority of days with 10,000 visitors or more having high temperatures above 60°F. Therefore, one can easily predict a far higher number of visitors during warmer periods of the year. Such a predictable trend reveals nothing notable, but a more effective model that accounts for the usual seasonal variation can be used.

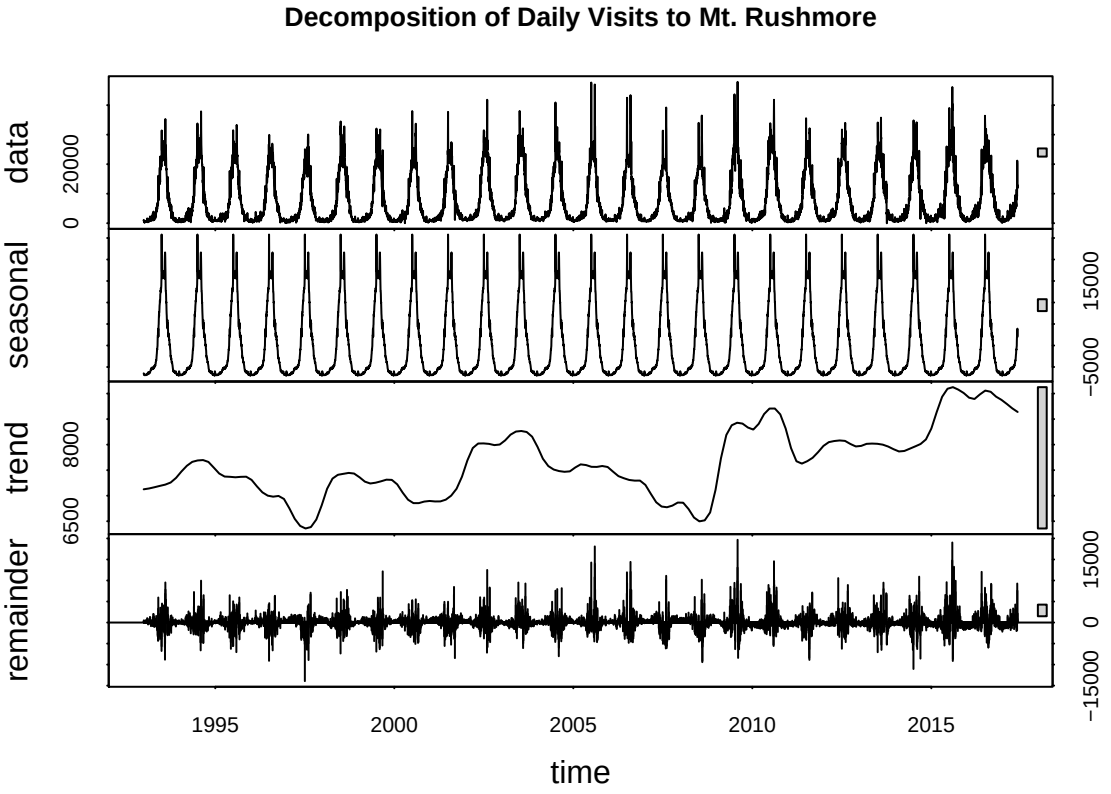
The time series plots the number of visitors to the park for every available day in chronological order, from 1993 to 2017. The `ts()` function is used, and the frequency set to 365 (since 365 is the number of data points per year), with start and end dates set at designated data points within the year, as shown below (May 31, 2017 is 151 days into the year). Those plots will make the seasonal variations very visible.

```
daily_v_ts <- ts(Rushmore$DailyVisits, frequency = 365, start=c(1993,1), end=c(2017,151))
plot(daily_v_ts, main = "Daily Visits to Mt. Rushmore", xlab = "Year",
     ylab = "Number of Daily Visits")
```

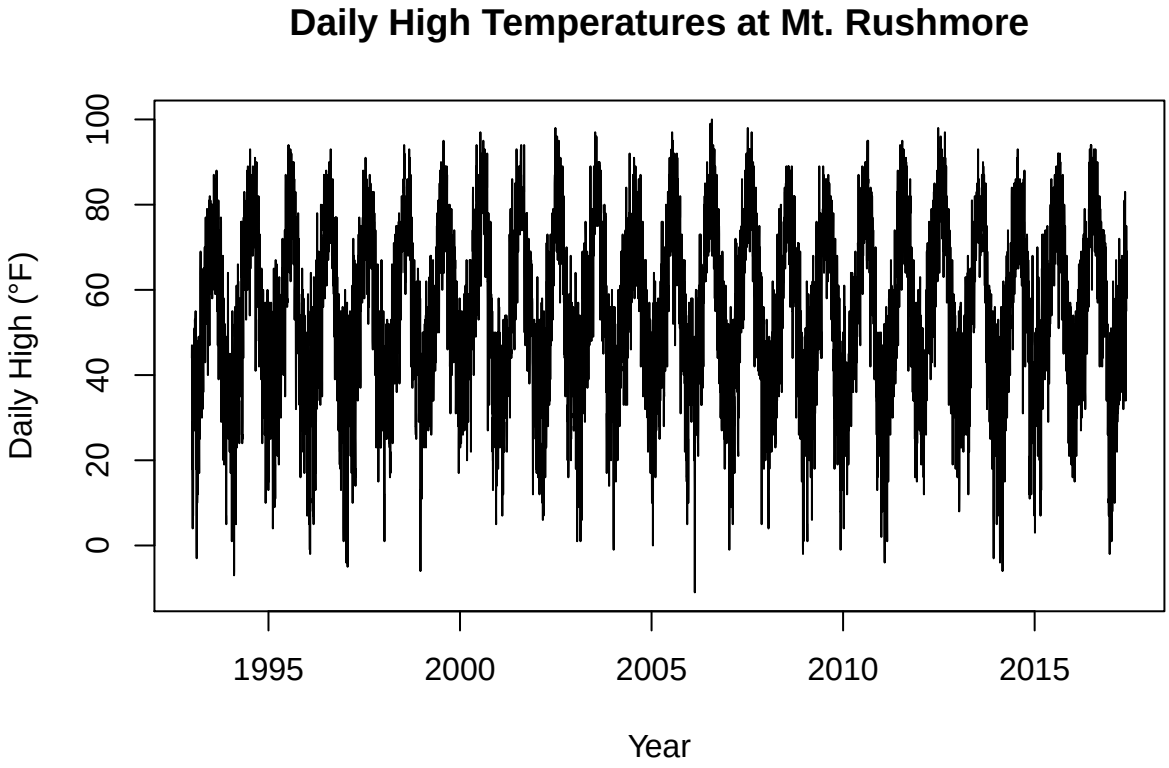


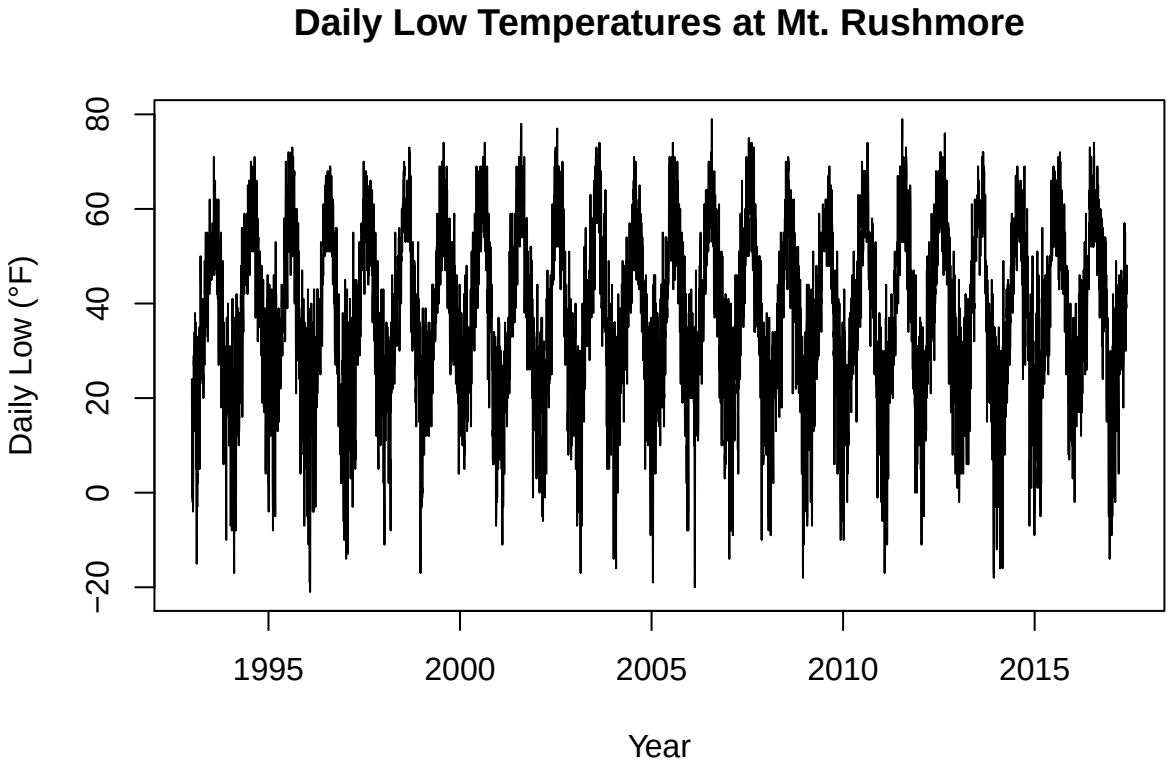
However, for the decomposing of the time series, the predictable seasonal variations, in addition to the long-term trends, are factored out of the data. The time series is decomposed using the `stl()` function, and whichever variation exists in the remaining data (bottom "remainder" plot) is variation that cannot be explained by seasonal or long term trends in the data.

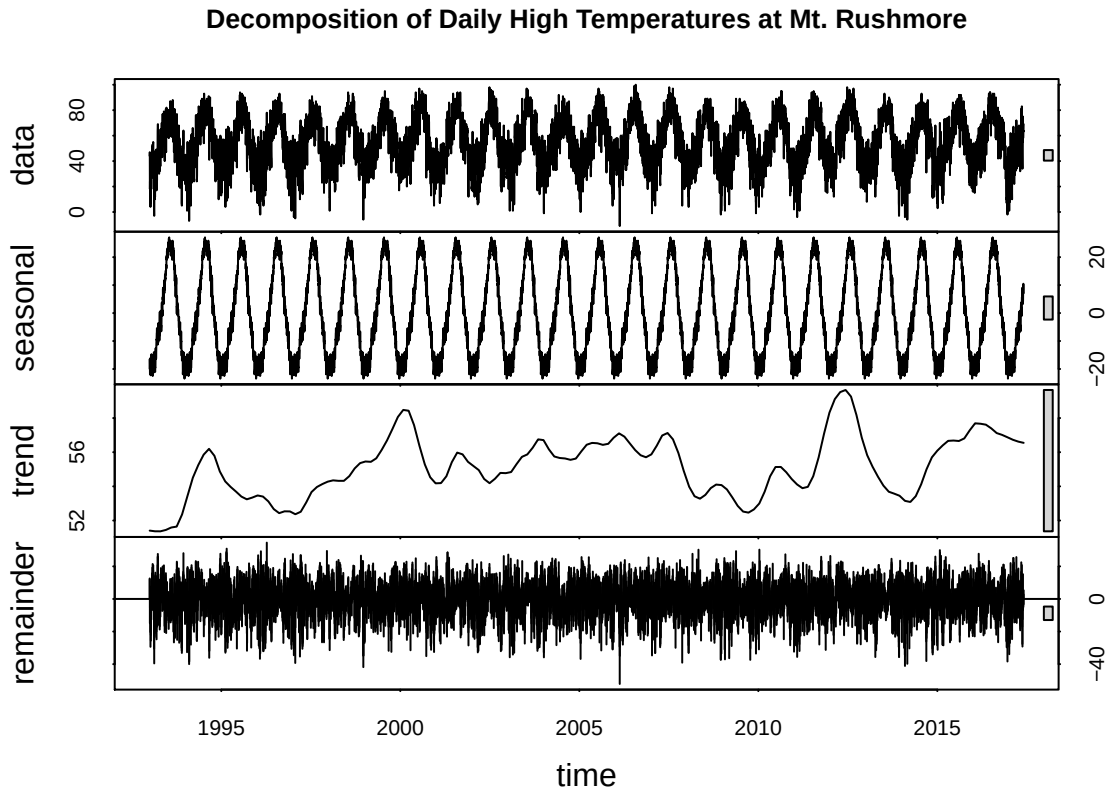
```
daily_v_stl <- stl(daily_v_ts, s.window="periodic")
plot(daily_v_stl, main = "Decomposition of Daily Visits to Mt. Rushmore")
```

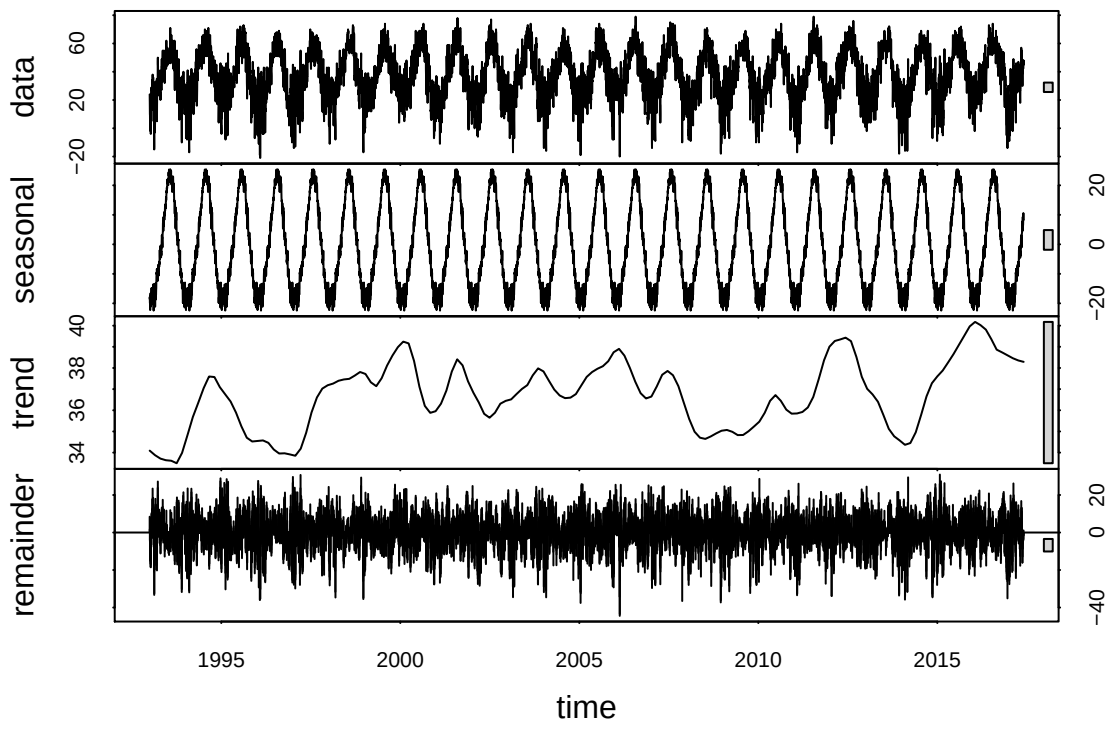


The same process will be done for the temperature and precipitation data. Both the daily high temperatures and daily low temperatures show sharp seasonal changes, and like the daily visits, they can be decomposed with seasonal and long-term trends factored out of the data.

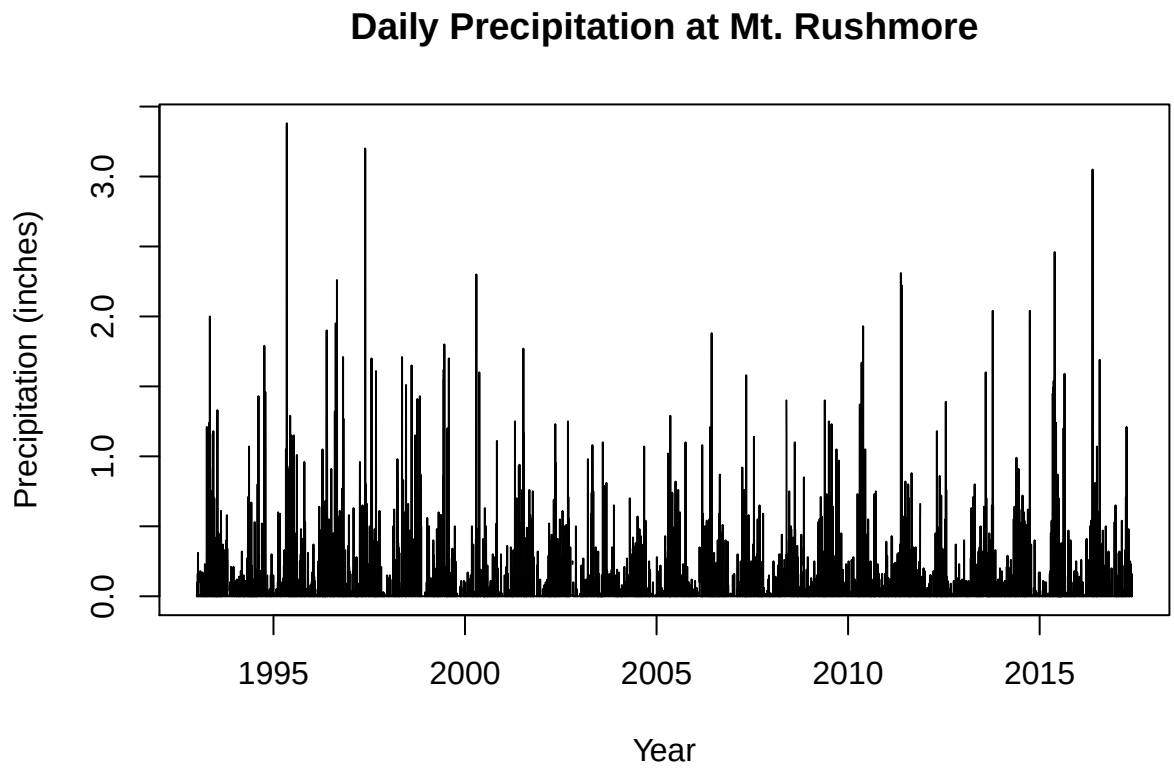




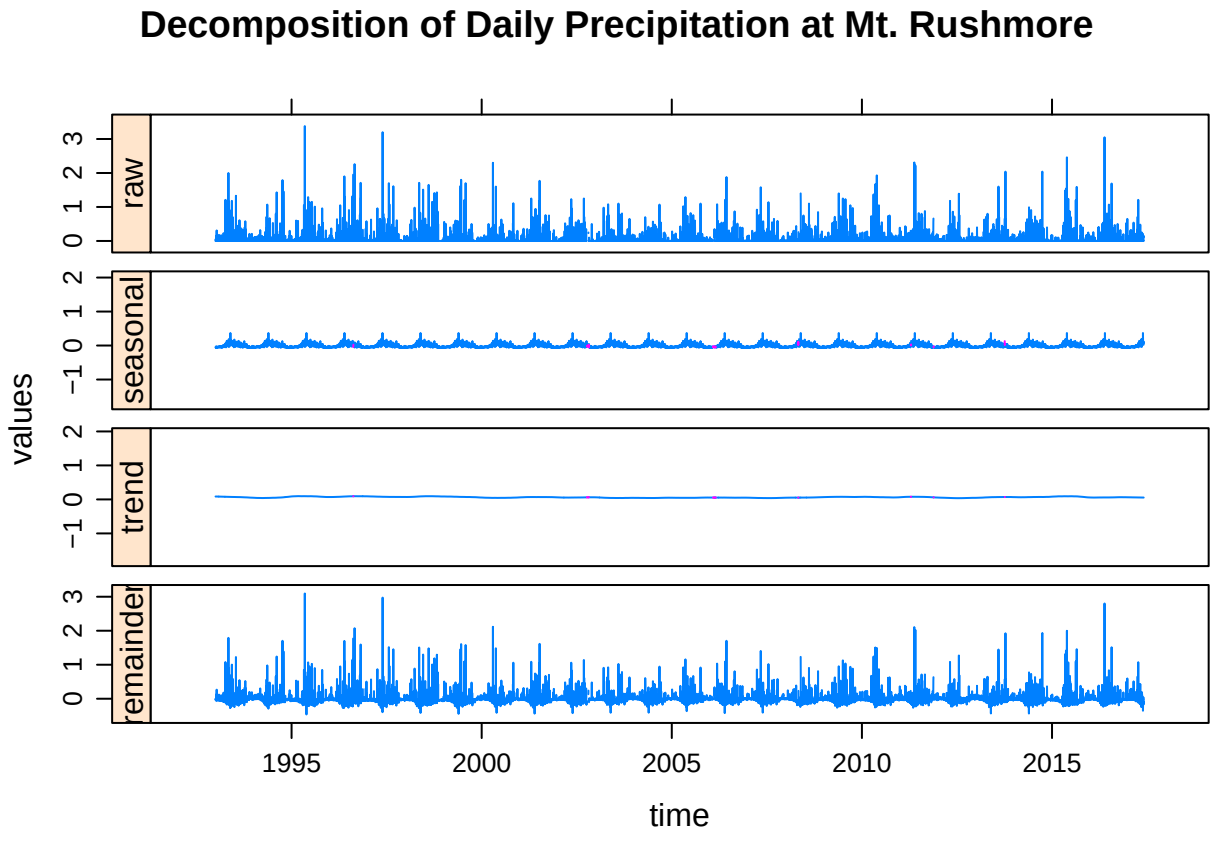




Another package, `stlplus()` (which can work with `NA` values present), will be used to decompose the precipitation time series.




```
precip_stl <- stlplus(precip_ts, s.window = "periodic")
plot(precip_stl, main = "Decomposition of Daily Precipitation at Mt. Rushmore")
```

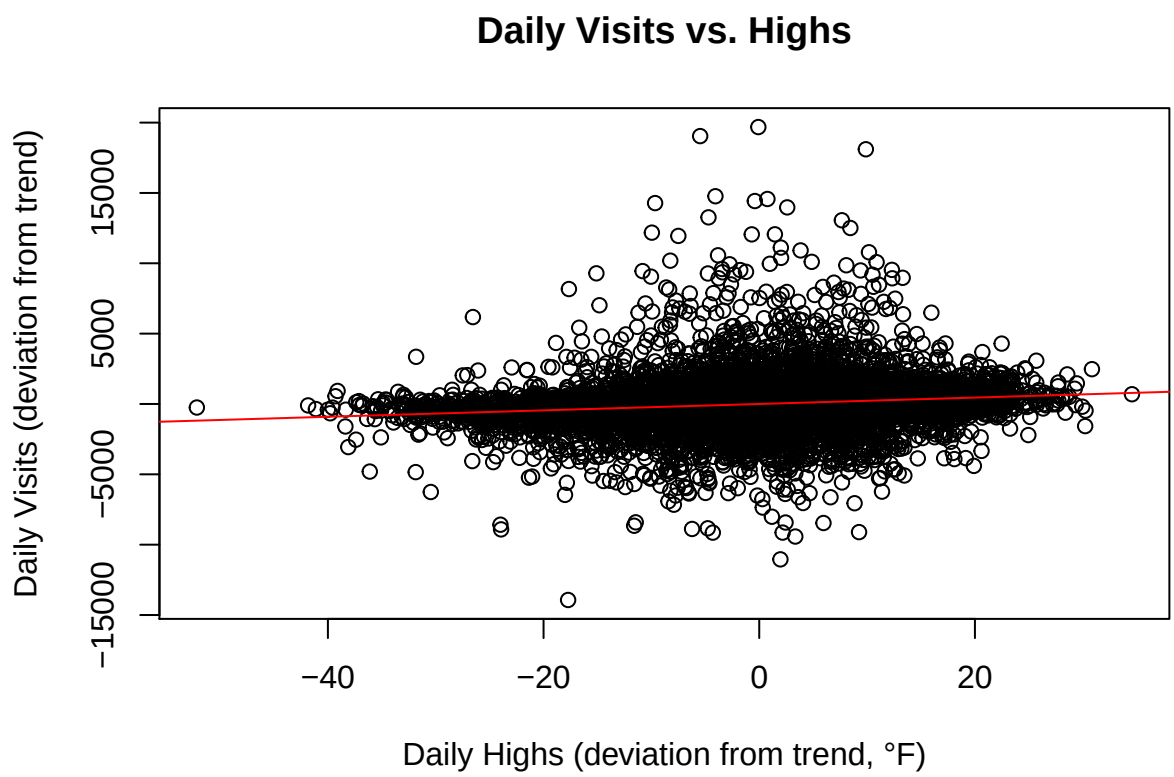


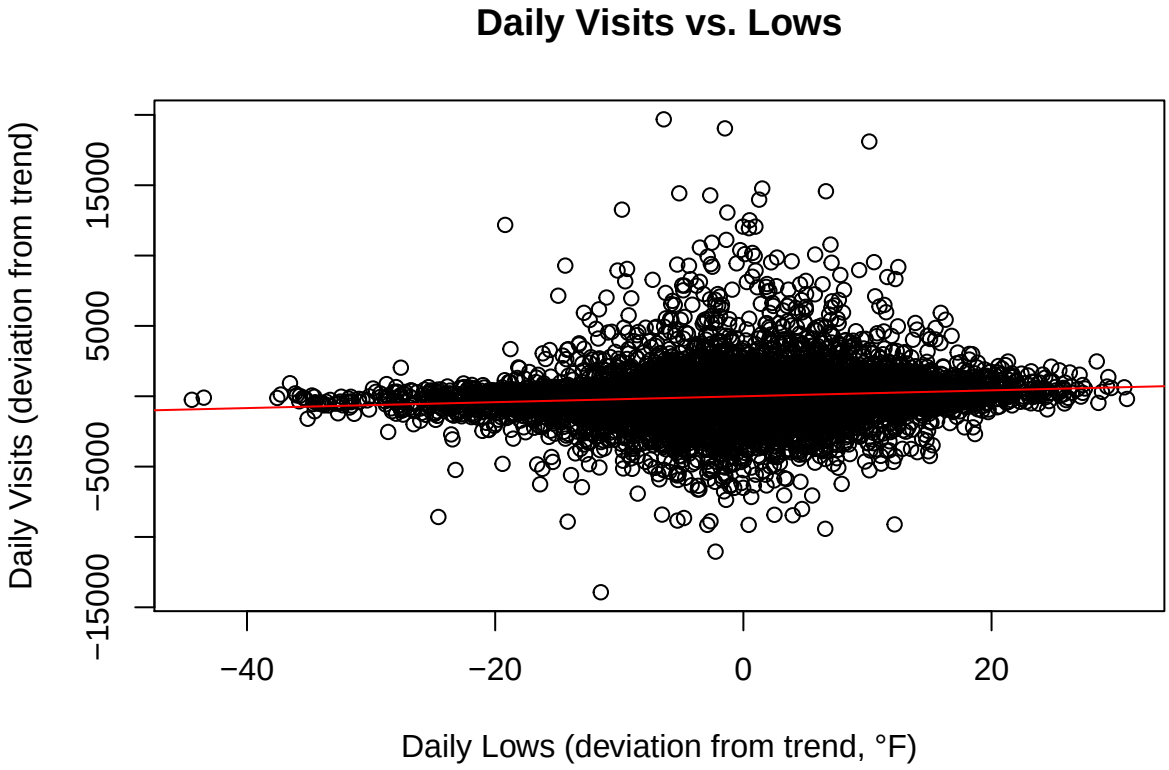
The decomposed time series data weeds out both seasonal variation and the long term trends for the daily

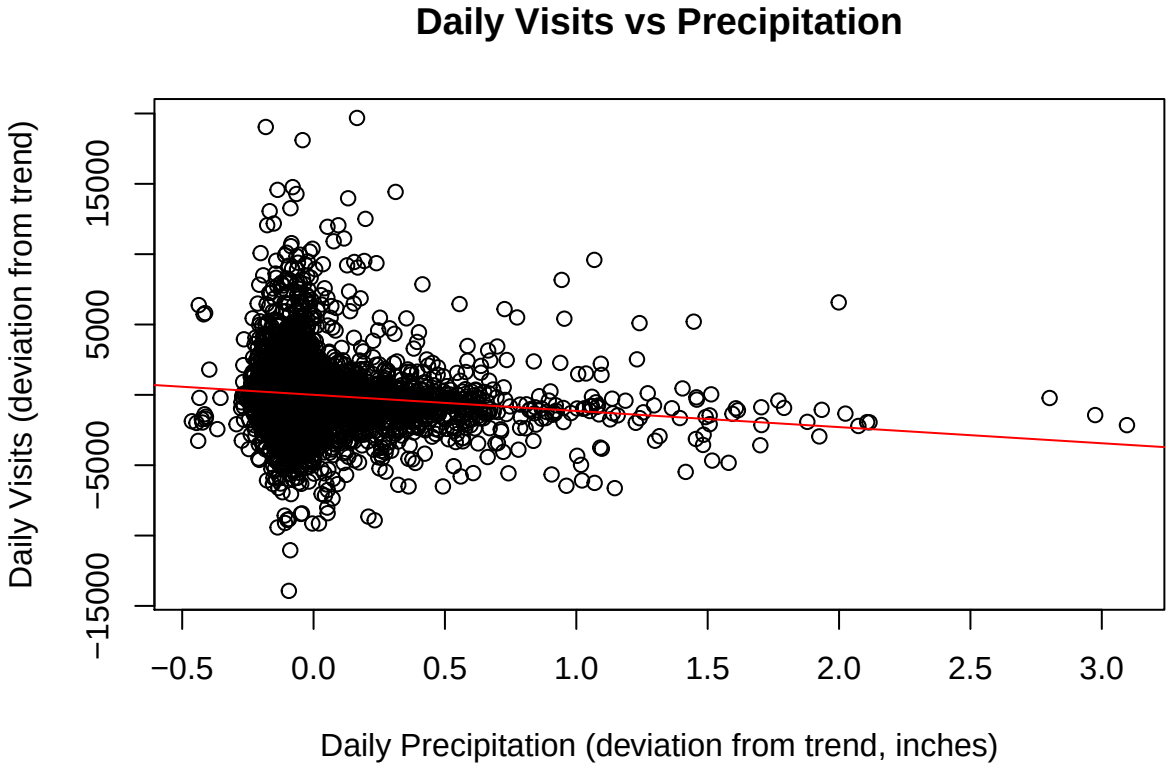
visits and the weather-related variables. Now that those predictable trends are factored out, other factors can be investigated to try explaining the short term varations in the number of visitors. In this analysis, we will attempt to find a link between the changes in the daily weather and the short-term variations in the number of visitors to Mt. Rushmore. The decomposed time series for each of the variables will be extracted and saved onto their own dataframes, and then they will be plotted to visualize their relationships.

```
#Extract the remainders, and then plot them against one another
daily_v_remain <- daily_v_stl$time.series[,"remainder"]
hi_temp_remain <- hi_temp_stl$time.series[,"remainder"]
lo_temp_remain <- lo_temp_stl$time.series[,"remainder"]
precip_remain <- precip_stl$data[,"remainder"]
```

```
plot(hi_temp_remain, daily_v_remain, main = "Daily Visits vs. Highs",
     xlab = "Daily Highs (deviation from trend, °F)", ylab = "Daily Visits (deviation from trend)")
abline(lm(daily_v_remain ~ hi_temp_remain), col="red") #regression line
```







The above plots do not seem to show strong linear correlations between the decomposed variation of daily visits and the independent variables they were plotted against. The line of best fit for each of the plots appears flat, with little predictability between each of the variables and daily visits after factoring out seasonal variation and long term trends. A linear regression analysis using the `lm()` function can quantify the linear relationship between the variables.

```r
#Complete a linear regression analysis
V_reg <- lm(daily_v_remain ~ hi_temp_remain + lo_temp_remain + precip_remain, data = Rushmore)
summary(V_reg)
```

```
##
## Call:
## lm(formula = daily_v_remain ~ hi_temp_remain + lo_temp_remain +
##     precip_remain, data = Rushmore)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -13672.6   -641.8    -43.9    540.2  19829.5
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)       2.0740    18.7863   0.110    0.912
## hi_temp_remain   19.9139     2.8936   6.882  6.3e-12 ***
## lo_temp_remain   -0.6297     3.3408  -0.188    0.850
## precip_remain  -917.9481    99.6289  -9.214  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1767 on 8847 degrees of freedom
##   (60 observations deleted due to missingness)
## Multiple R-squared:  0.02972,    Adjusted R-squared:  0.02939
## F-statistic: 90.32 on 3 and 8847 DF,  p-value: < 2.2e-16
```

The result of the linear regression analysis shows that the adjusted correlation (adjusted R-squared) between the daily visits and the weather-related variables, which is the correlation that accounts for the number of independent variables, is around 2.9%. That means that 2.9% of the day-to-day variation of the number of visitors to Mt. Rushmore can be explained by changes in precipitation or temperatures.

However, both the daily high temperature and precipitation had p-values near zero, which suggested a statistical significance between those two variables and the number of visitors to the park. The daily low temperatures displayed no statistical significance at all, so another linear regression analysis with that variable removed will be done.

```r
V_reg2 <- lm(daily_v_remain ~ hi_temp_remain + precip_remain, data = Rushmore)
summary(V_reg2)
```

```
##
## Call:
## lm(formula = daily_v_remain ~ hi_temp_remain + precip_remain,
##     data = Rushmore)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -13673.3   -641.7    -44.2    541.4  19833.6
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)       2.069     18.785   0.110    0.912
## hi_temp_remain   19.471      1.692  11.511   <2e-16 ***
## precip_remain  -918.863     99.505  -9.234   <2e-16 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1767 on 8848 degrees of freedom
##   (60 observations deleted due to missingness)
## Multiple R-squared:  0.02971,    Adjusted R-squared:  0.02949
## F-statistic: 135.5 on 2 and 8848 DF,  p-value: < 2.2e-16
```

The following linear regression analysis included only the statistically significant independent variables. However, the resulting correlation between those variables and the number of visitors remained essentially unchanged.

Conclusion

The conclusion that can be drawn from this analysis is that after adjusting for the expected seasonal and long-term trends in all variables involved, evidence of daily temperatures and precipitation being effective predictors of the daily visits to Mt. Rushmore has not been found. That would mean that on a day to day basis, there is no known indication of the weather having a meaningful impact on the number of visits to Mt. Rushmore.

However, the linear regression analysis also revealed a very low p-value for the daily high temperatures and precipitation, suggesting that although the correlation is extremely low, there may be some link between the those variables and the number of daily visits. Most likely, the high sample size of the dataset (nearly 9,000 datapoints) led to the low p-values, but the possibility of a non-linear relationship between those variables and the daily visits cannot be ruled out.

Another possibility to consider, which has not been investigated in this analysis, is the seasonal difference in the sensitivity between the weather and daily visits, which would mean that in some seasons, the weather may have a somewhat greater impact on visits to Mt. Rushmore than in other seasons. This analysis had a time series decomposition that factored out seasonal trends in the data, but it was not done under the assumption that the relationship between temperature deviations from the norm, and tourism deviations from the nrom, would differ based on the season.

Recommendations

Although evidence of daily weather changes affecting tourism has not been found, the National Park Service, in addition to local businesses around the Black Hills that benefit from tourism to Mt. Rushmore, should take an interest in further consideration of how daily high temperatures and precipitation may affect the number of tourists at Mt. Rushmore. Although a linear regression model does not appear to be an effective predictor of daily visits, more insight may be found by studying particular times of the year, perhaps on a monthly or seasonal basis, or by using other prediction models that are not based on linear trends. Further analysis will be able to provide insight about if the weather has a significant impact on tourism, and how local clients can adapt to such insights to improve the efficiency of their operations.

Also, additional studies on how other factors besides the weather impact Mt. Rushmore visits (especially in the short term) can provide further information on how the variation of the number of visits after accounting for trends can be explained, and on how to adjust operations and benefit from more knowledge about the impacts of other possible factors.