ps1

Yihuan Song

09/06/2018

question 3

(a) To download the data into the subdirectory, I first created a new directory called "temp", then I used a loop for years from 2011 to 2017, used "curl" to download data from the specified year, used "gunzip" to unzip the file, and used "wc" to count the number of observations for each year and print on the screen.

```
##creat new subdirectory
mkdir ~/temp
cd ~/temp
##download data from url and count the number of observations
for ((year=2015; year<=2017;year++))
do
curl -O -s https://www1.ncdc.noaa.gov/pub/data/ghcn/daily/by_year/{"$year.csv.gz"}
gunzip $year.csv.gz
wc -l $year.csv
done
```

```
##  35233244 2015.csv
##  35384539 2016.csv
##  34748555 2017.csv
```

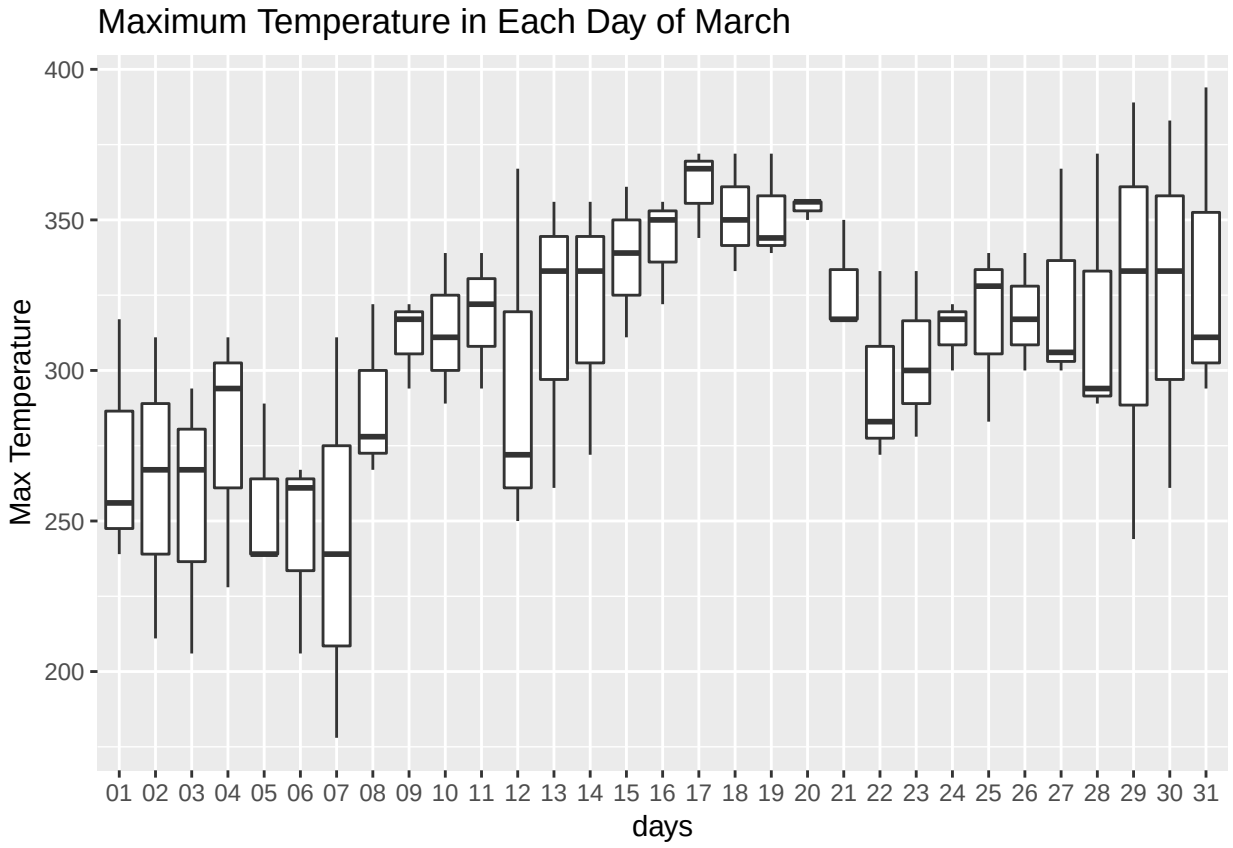
(b) To subset to the station corresponding to Death Valley, I first used "curl" tp download the file with station information. Then, I created a variable called "station", and stored the station ID in the variable using "grep" to find "DEATH VALLEY" in the file and used "cut" to specifically extract the station ID for Death Valley. Then, I used "grep" to subset the datasets to Death Valley, TMAX and March, storing the data into a new file called "dates".

```
cd ~/temp
##getting station ID of Death Valley
curl -O -s "https://www1.ncdc.noaa.gov/pub/data/ghcn/daily/ghcnd-stations.txt"
station=$(grep "DEATH VALLEY" ghcnd-stations.txt | cut -d" " -f1)
##Subset by Death Valley, TMAX and March to one dataset
grep -n $station ~/temp/*.csv | grep -n TMAX | grep -e '\<201.03..\>' >dates.csv
tail dates.csv
```

```
## 6002:/Users/yihuan/temp/2017.csv:7738656:USC00042319,20170322,TMAX,272,,,7,2400
## 6008:/Users/yihuan/temp/2017.csv:7835934:USC00042319,20170323,TMAX,300,,,7,2400
## 6014:/Users/yihuan/temp/2017.csv:7933357:USC00042319,20170324,TMAX,300,,,7,2400
## 6020:/Users/yihuan/temp/2017.csv:8030027:USC00042319,20170325,TMAX,283,,,7,2400
## 6026:/Users/yihuan/temp/2017.csv:8124503:USC00042319,20170326,TMAX,317,,,7,2400
## 6032:/Users/yihuan/temp/2017.csv:8218606:USC00042319,20170327,TMAX,306,,,7,2400
## 6038:/Users/yihuan/temp/2017.csv:8314672:USC00042319,20170328,TMAX,294,,,7,2400
## 6044:/Users/yihuan/temp/2017.csv:8410676:USC00042319,20170329,TMAX,333,,,7,2400
## 6050:/Users/yihuan/temp/2017.csv:8506997:USC00042319,20170330,TMAX,333,,,7,2400
## 6056:/Users/yihuan/temp/2017.csv:8603271:USC00042319,20170331,TMAX,294,,,7,2400
```

(c) To create the single plot of side-by-side boxplots containing the maximum daily temperature on each day in March, I created the R chunk, and read in the dates.csv file. I added a new column to the original dataset, which is the days subtracted from the column representing dates. Finally, I used ggplot2 to create the boxplot.

```r
setwd("~")
##read subseted data into R
dates <- read.table('~/temp/dates.csv',sep = ',', head = FALSE, stringsAsFactors = FALSE)
#extract days and make it a factor variable
days <- substr(dates[, "V2"], 7, 8)
dates_day <-cbind(dates, days)
dates_day$days <- factor(dates_day$days)
#make side-by-side boxplot
library(ggplot2)
dates_bp = ggplot(data = dates_day, aes(y = dates_day$V4, x = dates_day$days)) + geom_boxplot()
dates_bp = dates_bp + ggtitle("Maximum Temperature in Each Day of March")
dates_bp = dates_bp + xlab("days") + ylab("Max Temperature")
dates_bp
```



(d) The shell function "get_weather" takes 5 parameters as user input, including location, weather variable of interest, year to start, year to end and month. If user invokes the function with "-h", then the "if" statement would return instructions for the function. If user inputs wrong number of parameters, the "elif" would catch the mistake and return an error message. Then I used "curl" and "grep" to get the ID of user-specified location and to check if there is only one station that matches the location. Then by generalizing code from a) and b), the function used "curl" to download the weather data of the user-specified year and unzipped it. I used "grep" and "cut" to subset appropriate weather information. Finally, I returned the subseted information and removed downloaded raw data.

```r
function get_weather(){
##enable user to get help by invoking "get_weather -h"
  if [ "$1" = "-h" ]; then
    echo "Input "location", "weather variable of interest", "year to start",
```

```
          "year to end" and "month", in that order with quotes,
          to get weather information of interest."
##checking for the right number of parameters entered
  elif [ $# -ne 5 ]; then
    echo "Error, wrong number of parameters, 5 needed"
  else
    loc=$1
    wea=$2
    yr_start=$3
    yr_end=$4
    mon=$5
## check if the user-specified location identifies a single weather station
    curl -O -s "https://www1.ncdc.noaa.gov/pub/data/ghcn/daily/ghcnd-stations.txt"
    num_loc=$(grep "$loc" ghcnd-stations.txt | cut -d" " -f1 |wc -l)
    if [ $num_loc != 1 ]; then
      echo "Error, cannot identify a single weather station"
      rm ghcnd-stations.txt
    else
##getting weather information according to user input
      for ((yr=$yr_start; yr<=$yr_end;yr++))
      do
        curl -O -s https://www1.ncdc.noaa.gov/pub/data/ghcn/daily/by_year/{"$yr.csv.gz"}
        gunzip $yr.csv.gz
      done
##subset and return information on weather data according to user input
      station=$(grep "$loc" ghcnd-stations.txt | cut -d" " -f1)
      for ((yr=$yr_start; yr<=$yr_end;yr++))
      do
          grep -n $station $yr.csv | grep -n $wea | grep -n $yr$mon |cut -d"," -f4 >> info.csv
      done
      cat info.csv
##remove raw data
      rm *.csv ghcnd-stations.txt
    fi
 fi
}
get_weather -h
get_weather "DEATH VALLEY" "TMAX" "2017"
get_weather "DEATH VALLEY" "TMAX" "2016" "2017" "04"
```

```
## Input location, weather variable of interest, year to start,
##           year to end and month, in that order with quotes,
##           to get weather information of interest.
## Error, wrong number of parameters, 5 needed
## 339
## 350
## 356
## 372
## 372
## 383
## 361
## 294
## 294
## 306
```

```
## 311
## 333
## 333
## 339
## 278
## 333
## 367
## 361
## 378
## 389
## 394
## 367
## 317
## 317
## 278
## 317
## 300
## 322
## 333
## 256
## 344
## 344
## 333
## 311
## 328
## 356
## 333
## 294
## 267
## 306
## 311
## 356
## 306
## 311
## 328
## 378
## 350
## 361
## 339
## 344
## 344
## 372
## 394
## 339
## 339
## 356
## 350
## 306
## 328
## 356
```

question4

First I found method to have a list of each file ending in .txt in the given url. Specifically, by using "curl" to the url, I got the full HTML index of each file. Then, I used "grep" to sort out the txt files, and used "sed" to clear the format so only the "name.txt"(in the *href="name.txt"*) is left. Then I could loop over the txt files to download them and use "echo" to provide the file name as a message to the user.

```
mkdir ~/temp2
cd ~/temp2
for file in $(curl -s https://www1.ncdc.noaa.gov/pub/data/ghcn/daily/ | grep txt |
              sed 's/.*href="//' | sed 's/".*//' )
do
    curl -s -O https://www1.ncdc.noaa.gov/pub/data/ghcn/daily/$file
    echo "The file to be downloaded is named $file"
done
```

```
## The file to be downloaded is named ghcnd-countries.txt
## The file to be downloaded is named ghcnd-inventory.txt
## The file to be downloaded is named ghcnd-states.txt
## The file to be downloaded is named ghcnd-stations.txt
## The file to be downloaded is named ghcnd-version.txt
## The file to be downloaded is named mingle-list.txt
## The file to be downloaded is named readme.txt
## The file to be downloaded is named status.txt
```