Week3

2023-09-12

Question 5.1 Using crime data from the file uscrime.txt (http://www.statsci.org/data/general/uscrime.txt, description at http://www.statsci.org/data/general/uscrime.html), test to see whether there are any outliers in the last column (number of crimes per 100,000 people). Use the grubbs.test function in the outliers package in R.

```
#clean the R environment
rm(list = ls())
set.seed(10)

#load data into dataframe
crimedf <- read.table("uscrime.txt", stringsAsFactors = FALSE, header = TRUE)
head(crimedf)
```

```
##      M So   Ed  Po1  Po2    LF   M.F Pop   NW    U1  U2 Wealth Ineq     Prob
## 1 15.1  1  9.1  5.8  5.6 0.510  95.0  33 30.1 0.108 4.1   3940 26.1 0.084602
## 2 14.3  0 11.3 10.3  9.5 0.583 101.2  13 10.2 0.096 3.6   5570 19.4 0.029599
## 3 14.2  1  8.9  4.5  4.4 0.533  96.9  18 21.9 0.094 3.3   3180 25.0 0.083401
## 4 13.6  0 12.1 14.9 14.1 0.577  99.4 157  8.0 0.102 3.9   6730 16.7 0.015801
## 5 14.1  0 12.1 10.9 10.1 0.591  98.5  18  3.0 0.091 2.0   5780 17.4 0.041399
## 6 12.1  0 11.0 11.8 11.5 0.547  96.4  25  4.4 0.084 2.9   6890 12.6 0.034201
##      Time Crime
## 1 26.2011   791
## 2 25.2999  1635
## 3 24.3006   578
## 4 29.9012  1969
## 5 21.2998  1234
## 6 20.9995   682
```

```
#load library
#install.packages("outliers")
library(outliers)
library(ggplot2)
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v tibble  3.2.1     v dplyr   1.1.2
## v tidyr   1.3.0     v stringr 1.5.0
## v readr   2.1.2     v forcats 1.0.0
## v purrr   1.0.2

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

H0:There is no outlier in crime data.

```
#using type = 10 to test for 1 outlier. test the maximum value
grubbs.test(crimedf$Crime, type = 10, opposite = FALSE, two.sided = FALSE)
```

```
##
##  Grubbs test for one outlier
##
## data:  crimedf$Crime
## G = 2.81287, U = 0.82426, p-value = 0.07887
## alternative hypothesis: highest value 1993 is an outlier
```

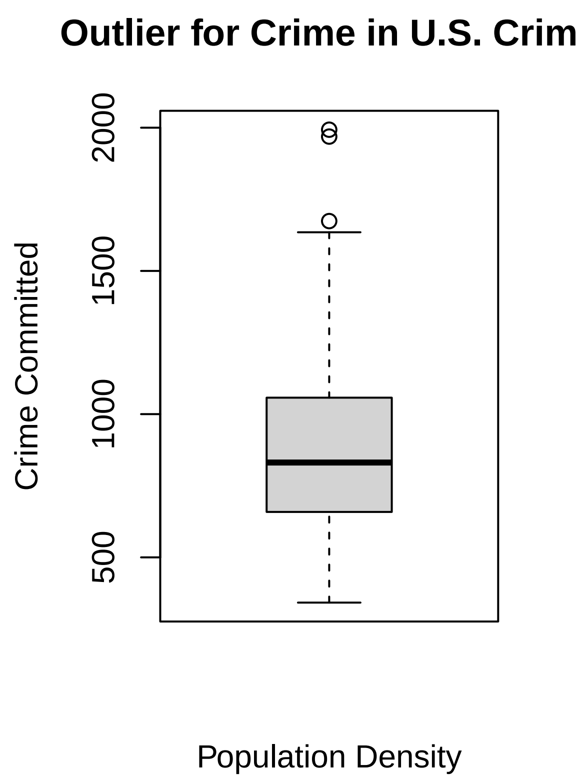
```r
#test the minimum value
grubbs.test(crimedf$Crime, type = 10, opposite = TRUE, two.sided = FALSE)
```

```
##
##  Grubbs test for one outlier
##
## data:  crimedf$Crime
## G = 1.45589, U = 0.95292, p-value = 1
## alternative hypothesis: lowest value 342 is an outlier
```

```r
#using type =11 to test for 2 outliers on opposite tails.
grubbs.test(crimedf$Crime, type = 11, opposite = FALSE, two.sided = FALSE)
```

```
##
##  Grubbs test for two opposite outliers
##
## data:  crimedf$Crime
## G = 4.26877, U = 0.78103, p-value = 1
## alternative hypothesis: 342 and 1993 are outliers
```

As the result shown, the p-value = 0.07887 > alpha = 0.05. In this case, we don't have strong evidence to reject the null hypothesis. Therefore, we cannot find outlier in uscrime data.



Let's take a look at the box graph of the crime data.

Even the graph shown there are two outliers in the data set. Based on the p-value we found previously is 0.07887. The outliers maybe occur due the random noise. In the big data, outlier may appears more often than case study. But in the uscrime data, I would keep the outliers. The outlier may depends on the population in the states. Where the city or states has higher poplulation density, the higher chance to commit a crime. That can be the reason for the outliers.

#Question 6.1 Describe a situation or problem from your job, everyday life, current events, etc., for which a Change Detection model would be appropriate. Applying the CUSUM technique, how would you choose the critical value and the threshold?

Answer We can refer the real life situation into subway maintenance, similar situation we can think of citi bike in New York City. This is a bike sharing project for city worker commute during rush hours. Using change detection model for monitor the repair for bike would be appropriate. For instance, New York City has about 2 millions bike trips everyday. We can use the change detection model to monitor the bike usage. We can calculate the approximate use time of each bike, trace the bike usage and detect the best time we should do repairs and maintenance for the bikes.

#Question 6.2 1. Using July through October daily-high-temperature data for Atlanta for 1996 through 2015, use a CUSUM approach to identify when unofficial summer ends (i.e., when the weather starts cooling off) each year. You can get the data that you need from the file temps.txt or online, for example at http://www.iweathernet.com/atlanta-weather-records or https://www.wunderground.com/history/airport/KFTY/2015/7/1/CustomHistory.html. You can use R if you'd like, but it's straightforward enough that an Excel spreadsheet can easily do the job too.

```
#install.packages("lubridate")
library(repr)
library(reshape)
```

```
##
## Attaching package: 'reshape'

## The following object is masked from 'package:dplyr':
##
##     rename

## The following objects are masked from 'package:tidyr':
##
##     expand, smiths
```

```
#load temp data
temp <- read.table("temps.txt", stringsAsFactors = FALSE, header = TRUE)
head(temp)
```

```
##     DAY X1996 X1997 X1998 X1999 X2000 X2001 X2002 X2003 X2004 X2005 X2006 X2007
## 1 1-Jul    98    86    91    84    89    84    90    73    82    91    93    95
## 2 2-Jul    97    90    88    82    91    87    90    81    81    89    93    85
## 3 3-Jul    97    93    91    87    93    87    87    87    86    86    93    82
## 4 4-Jul    90    91    91    88    95    84    89    86    88    86    91    86
## 5 5-Jul    89    84    91    90    96    86    93    80    90    89    90    88
## 6 6-Jul    93    84    89    91    96    87    93    84    90    82    81    87
##   X2008 X2009 X2010 X2011 X2012 X2013 X2014 X2015
## 1    85    95    87    92   105    82    90    85
## 2    87    90    84    94    93    85    93    87
## 3    91    89    83    95    99    76    87    79
## 4    90    91    85    92    98    77    84    85
## 5    88    80    88    90   100    83    86    84
## 6    82    87    89    90    98    83    87    84
```

```r
# average the temperature for each day across the years
avg_date_temp <- rowMeans(temp[c(2:length(temp))], na.rm=T)
# compute the mean of the average temperature by the same date from 1996-2015.
mu <- mean(avg_date_temp)
#find the standard deviation
std_temp <- sd(avg_date_temp)
# set C
C <- std_temp

# set threshold T.
T <- 4*std_temp

# create an empty column to store the data
temp[,"St"]<-NA

# starts loop from 0.

temp[1,"St"]<-0
for(i in 1:nrow(temp)){

  temp[i,"St"]<-max(0,(temp[i-1,"St"]+mu-avg_date_temp[i]-C))


}

temp$St
```

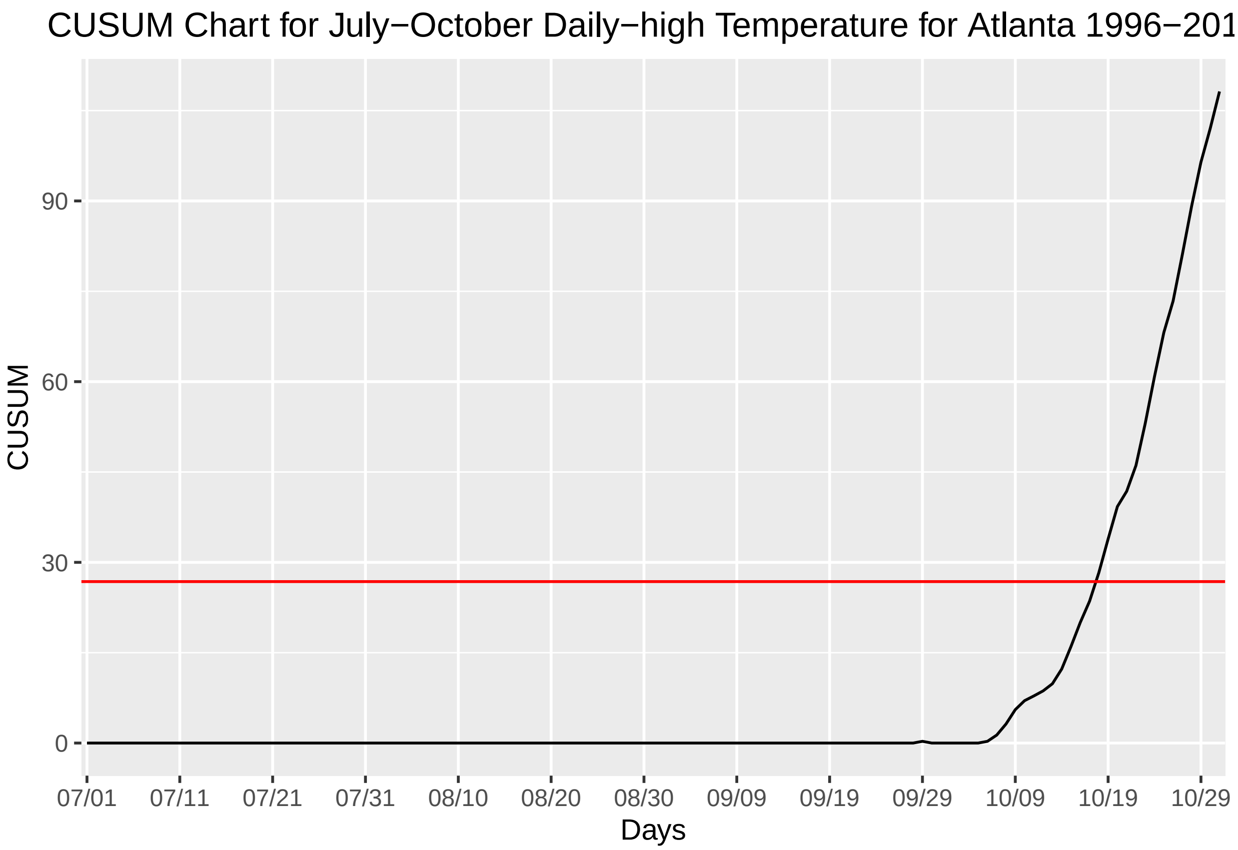
```
##   [1]   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000
##   [7]   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000
##  [13]   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000
##  [19]   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000
##  [25]   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000
##  [31]   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000
##  [37]   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000
##  [43]   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000
##  [49]   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000
##  [55]   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000
##  [61]   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000
##  [67]   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000
##  [73]   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000
##  [79]   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000
##  [85]   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000
##  [91]   0.2876435   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000
##  [97]   0.0000000   0.2876435   1.3252870   3.1629306   5.5505741   7.0382176
## [103]   7.8258611   8.6635046   9.8511481  12.2887917  16.0264352  20.0140787
## [109]  23.5517222  28.2893657  33.8770092  39.2646528  41.8022963  46.0899398
## [115]  53.0775833  60.8652268  68.1528704  73.3905139  81.1281574  89.1658009
## [121]  96.4534444 102.0410879 108.1787315
```

```r
cat("The day a change in trend is detected is:",temp[which(temp$St>T),"DAY"][1])
```

```
## The day a change in trend is detected is: 18-Oct
```

```r
#change data type for "day" as month/date
temp[,"Date"]<-as.Date(temp[,"DAY"],"%d-%B")
```

```r
temp[,"Date"]<-format(temp[,"Date"],format="%m/%d")
```



Per the control graph shown above, we can see the fall starts around 10/18 in Atlanta. The result is appropriate because geographically speaking. Atlanta located in the south part of the United States. Which means the fall starts a little later by the end of September until mid-October is applicable.

2. Use a CUSUM approach to make a judgment of whether Atlanta's summer climate has gotten warmer in that time (and if so, when).

In the approach, I used the cusum of each year to perform the model.

```r
#creating a data frame to store the cusum values for each year instead of same day on every year.
cusum_temp<-data.frame(matrix(nrow=nrow(temp),ncol=length(1996:2015)+1))

#assigning columns names to cusum data frame
colnames(cusum_temp)<-colnames(temp[,1:21])

#converting Day into Date that the loop can process easier.
cusum_temp$DAY<-temp$Date

# starts from zero
#calculating cusum values for each year
for(y in 2:ncol(cusum_temp)){
  cusum_temp[1,y]<-0 #initial St value for each column,set to zero
  mu<-mean(temp[,y]) #mean of each sample space(each year's observations)
  std<-sd(temp[,y]) #sd of each sample,also used as allowable slack
  threshold<-5*std #using 5 sd as threshold value,different T for each year
  change<-NULL # to store dates with St over threshold,first value:first day change detected
```

```
  for(i in 2:nrow(cusum_temp)){
    cusum_temp[i,y]<-max(0,cusum_temp[i-1,y]+(mu-temp[i,y]-std))
    if (cusum_temp[i,y]>=threshold){
      change<-append(change,cusum_temp[i,y])}}
  cat("In ",colnames(cusum_temp[y])," first day of Fall started on",
      cusum_temp[which(cusum_temp[,y]==change[1]),"DAY"],"\n")
}
```

```
## In  X1996  first day of Fall started on 10/06
## In  X1997  first day of Fall started on 10/19
## In  X1998  first day of Fall started on 10/22
## In  X1999  first day of Fall started on 10/23
## In  X2000  first day of Fall started on 10/09
## In  X2001  first day of Fall started on 10/28
## In  X2002  first day of Fall started on 10/18
## In  X2003  first day of Fall started on 10/11
## In  X2004  first day of Fall started on 10/14
## In  X2005  first day of Fall started on 10/25
## In  X2006  first day of Fall started on 10/23
## In  X2007  first day of Fall started on 10/26
## In  X2008  first day of Fall started on 10/23
## In  X2009  first day of Fall started on 10/17
## In  X2010  first day of Fall started on 10/04
## In  X2011  first day of Fall started on 10/23
## In  X2012  first day of Fall started on 10/29
## In  X2013  first day of Fall started on 10/23
## In  X2014  first day of Fall started on 10/22
## In  X2015  first day of Fall started on 10/29
```

As we can see from the statements shown above, from 1996 to 2015 fall generally starts in October. For the giving data, we do not have strong evidence to proof that summer climate has gotten warmer. Perhaps, the data is not large enough to show the difference. In addition, even if some year may have longer summer time, that maybe just an "outlier" through out the time series. However, we cannot ignore the outlier because the exist of outlier is meaningful to help us study the climate.