ISYE 6501 Homework 3

2023-09-06

Question 5.1

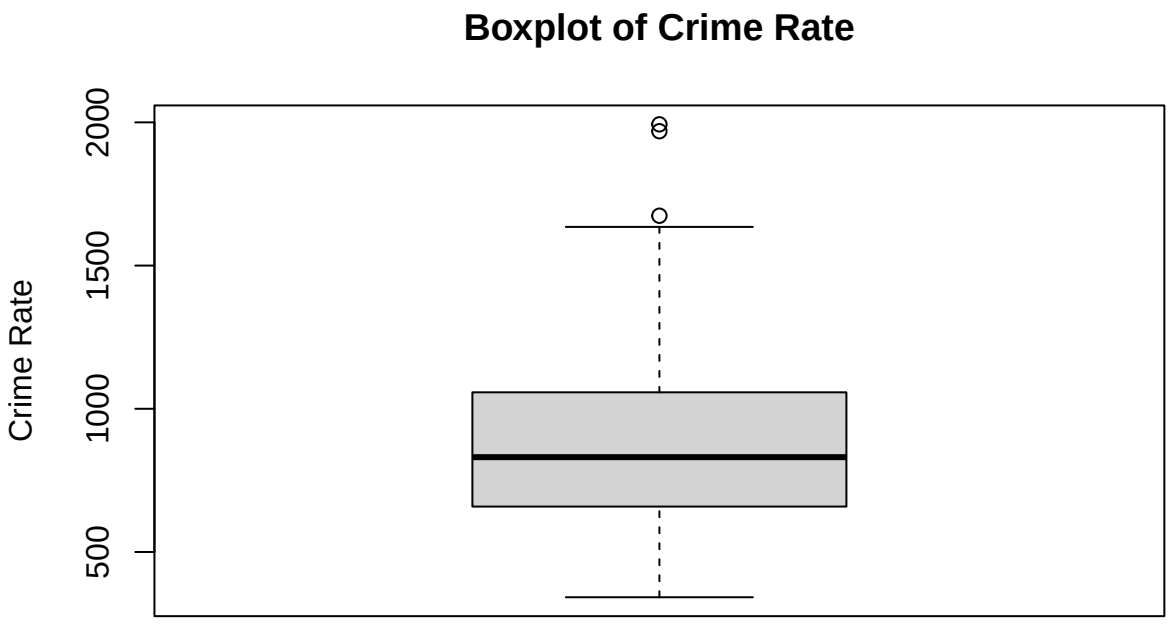
Using crime data from the file uscrime.txt, test to see whether there are any outliers in the last column (number of crimes per 100,000 people). Use the grubbs.test function in the outliers package in R.

```
#load libraries
library(outliers)

#load the data
crime_data <- read.table("http://www.statsci.org/data/general/uscrime.txt", header=TRUE)
```

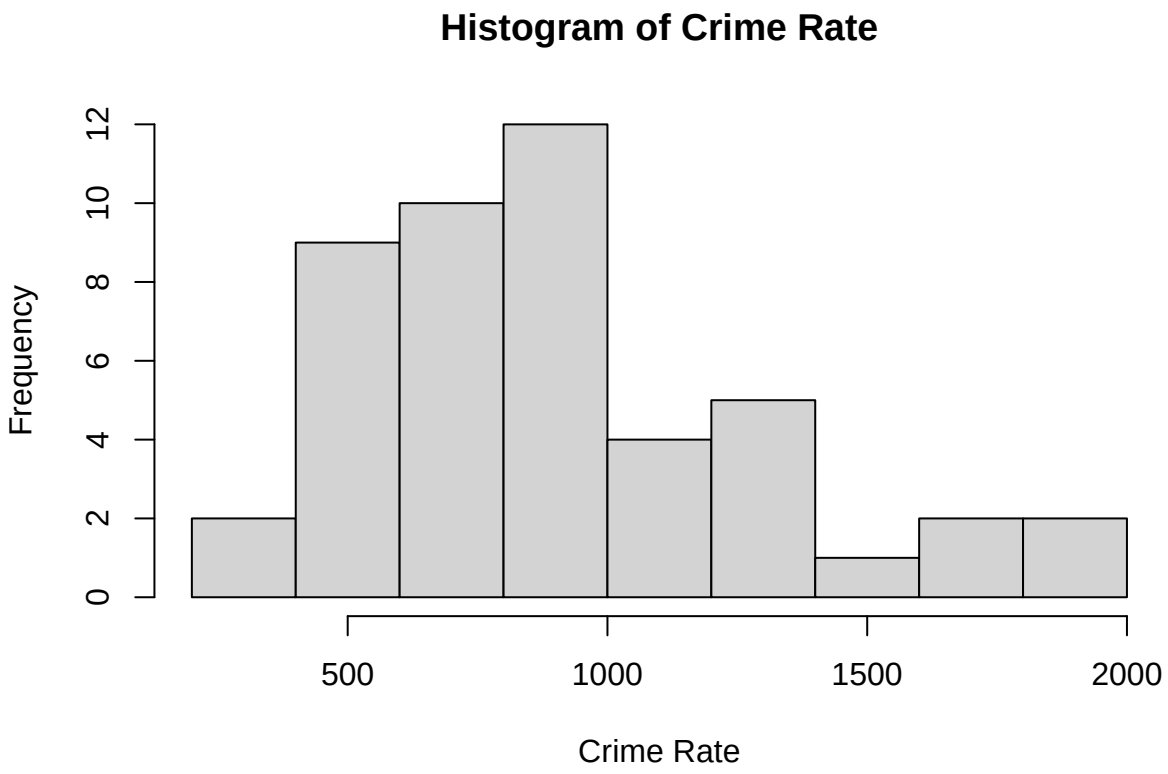
As a first step to answer this question, I created a boxplot to visualize the distribution of data. As you can see below, there are a few data points falling outside the distribution. However, we need to do further analysis to determine if these are actually outliers.

```
#visualize the distribution of data using a boxplot
boxplot(crime_data$Crime, ylab="Crime Rate", main="Boxplot of Crime Rate")
```



Next, I performed the Grubbs Test to determine if there were outliers in the data set. The grubbs.test() function assumes the data are normally distributed, so I created a histogram to further explore the distribution of data and determine if grubbs.test() is an appropriate method to check for outliers.

```
#check the normality of data using a histogram
hist(crime_data$Crime, breaks=10, xlab="Crime Rate", main="Histogram of Crime Rate")
```



Since the data appears to be normally distributed, I performed the Grubbs Test. When executing the function, it flags the value of 1993, the largest number in the data set. Although this value appears to be an outlier, it's p-value does not go below the significance threshold (p=0.05). Therefore, we fail to reject the null hypothesis (or in other words: there are no real outliers in this data set).

```
#use the grubbs.test() function to check for outliers
grubbs.test(x=crime_data$Crime,
            type=10,
            opposite=FALSE,
            two.sided=FALSE
            )
```

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

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?

A Change Detection Model would be appropriate in a manufacturing setting. Let's say a wood manufacturing company sells wood planks to be used in construction, and these planks need to adhere to certain specifications (e.g., they need to be between 4.8 and 5.2 feet long). The manufacturing company may implement the CUSUM technique to monitor small shifts in the process and make timely adjustments to bring the process back on target, as needed. Using this example, the company may align the threshold to the product specifications (e.g., a threshold that would signal alarm if the process was consistently outside of its specifications). Similarly, they will want to choose a critical value that considers the trade offs of false detection vs. real change. If the cost associated with bad products is higher than the opportunity cost of shutting down machines, the company should pick a smaller value of C.

Question 6.2

Part 1

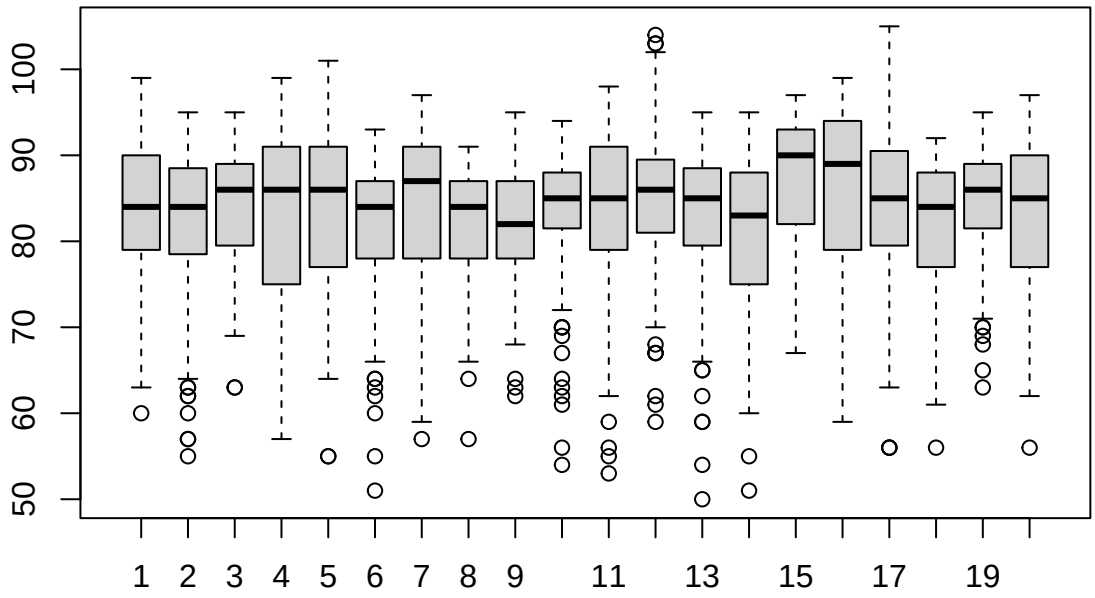
Using July through October daily-high-temperature data for Atlanta for 1996 through 2015, use a CUSUM approach to identify when unofficial summer ends (i.e., when the weather starts cooling off) each year.

Before answering this question, I visualized each year's temperature data using box plots to observe the distribution of data. As you can see in the chart below, the average temperature hovers between 80-90 degrees F, with a majority of the data points falling in the 75-95 range. We can also see quite a few outliers, most of these on the lower end.

```
#load libraries
library(ggplot2)

#load the data
temp_data <- read.table("temps.txt", header=TRUE)

#use box plots to visualize the data
boxplot(temp_data$X1996,
        temp_data$X1997,
        temp_data$X1998,
        temp_data$X1999,
        temp_data$X2000,
        temp_data$X2001,
        temp_data$X2002,
        temp_data$X2003,
        temp_data$X2004,
        temp_data$X2005,
        temp_data$X2006,
        temp_data$X2007,
        temp_data$X2008,
        temp_data$X2009,
        temp_data$X2010,
        temp_data$X2011,
        temp_data$X2012,
        temp_data$X2013,
        temp_data$X2014,
        temp_data$X2015
        )
```



Because the temperature data looks similar year-after-year, I decided to consolidate the data by taking an average temperature. This way, I would only be applying the CUSUM approach to one column of data (as opposed to repeating it over 20 columns of data).

Once I calculated the average, I created a new data frame with 3 columns: the day, the average summer temperature, and a blank column to store the St value. I used this new data frame (called "cusum") to apply the CUSUM approach.

```
#add column to data frame that calculates average temperature from 1996 to 2015
temp_data[,"Average Temp"] <- rowMeans(temp_data[, -1])

#create a new data frame
cusum <- subset(temp_data, select=c("DAY", "Average Temp"))

#add empty column to store St
cusum[,"St"] <- NA

#format dates to make plotting in ggplot easier
date_format <- "%d-%b"
cusum$DAY <- as.Date(cusum$DAY, format=date_format)
```

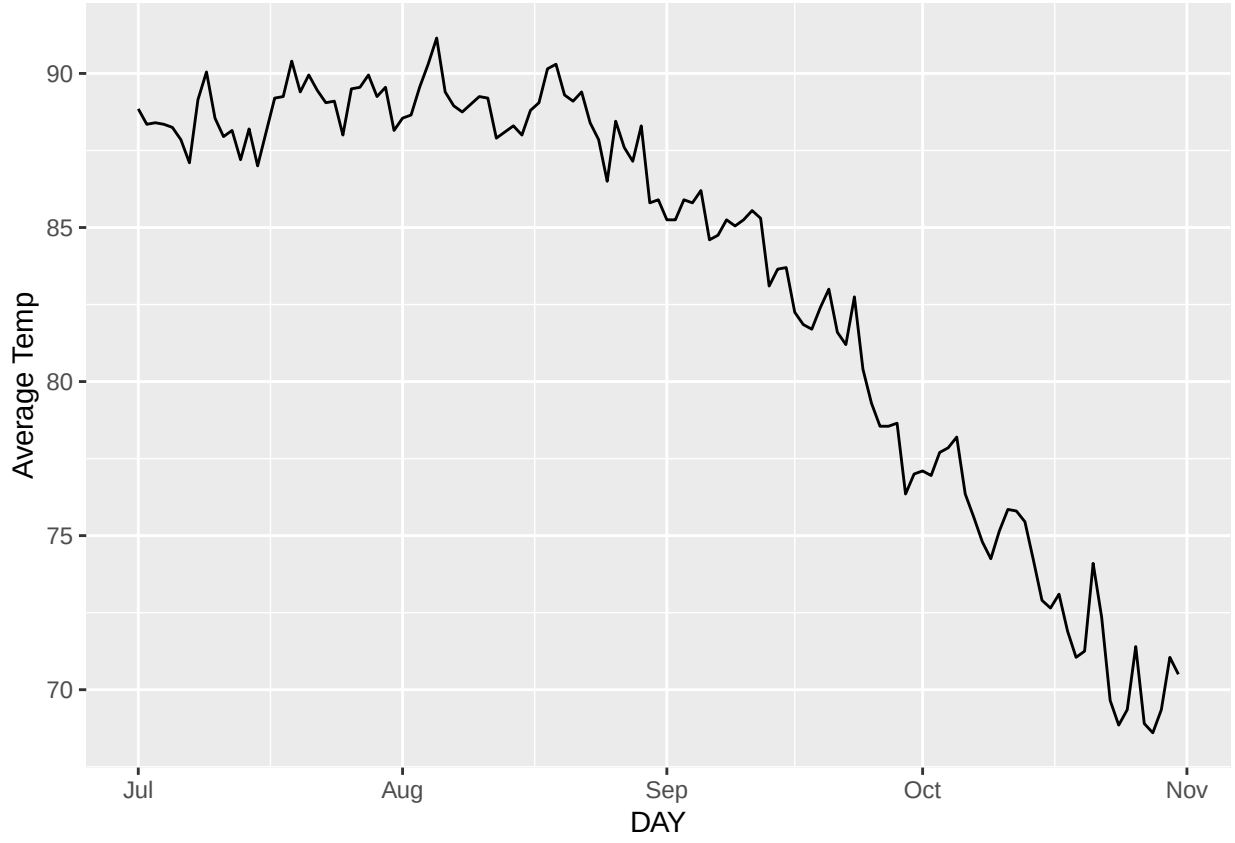
Next, I determined the mean, C-value, and threshold to be used in the CUSUM approach.

To choose an appropriate mean, I plotted key points in the time series data to see where the temperature begins to drop. As you can see in the graph below, temperature started to noticeably drop near the beginning of September. For that reason I truncated the data set from July 1 to September 1 to approximate the mean during the "steady period."

I calculated the standard deviation from July 1 to September 1 to help me choose an appropriate threshold and C-value. I chose C=2(std_dev) and T=5(std_dev) as a starting point. After trial and error with different values of C and T, I decided C=2(std_dev) and T=5(std_dev) yielded an appropriate result.

```
#plot temperatures
ggplot(cusum, aes(x=DAY, y=`Average Temp`, group=1))+
  geom_line()+
  scale_x_continuous(limit=cusum$DAY)+
  scale_x_date(date_breaks="1 month", date_labels="%b")
```

```
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
```




```
#calculate mean from July 1 to September 1
mean <- mean(cusum[1:63,'Average Temp'])

#calculate standard deviation from July 1 to September 1
st_dev <- sd(cusum[1:63,'Average Temp'])

#choose a C value
C <- st_dev*2

#choose a threshold
threshold = st_dev*5
```

Then, I applied the CUSUM approach and printed out the first value to exceed the predetermined threshold.

```
#calculate first value of St
first_St <- max(0, mean - cusum[1, "Average Temp"])
cusum[1, "St"] <- first_St

#calculate remaining values of St
for(i in 2:nrow(cusum)){

  #calculate x and St-1
  x <- cusum[i, "Average Temp"]
  previous_St <- cusum[i-1, "St"]

  #calculate St
  St <- max(0, previous_St +(mean - x - C))

  #update table with value of St
  cusum[i, "St"] <- St
}

#print table
cusum
```

```
##            DAY Average Temp          St
## 1   2023-07-01        88.85   0.0000000
## 2   2023-07-02        88.35   0.0000000
## 3   2023-07-03        88.40   0.0000000
## 4   2023-07-04        88.35   0.0000000
## 5   2023-07-05        88.25   0.0000000
## 6   2023-07-06        87.85   0.0000000
## 7   2023-07-07        87.10   0.0000000
## 8   2023-07-08        89.15   0.0000000
## 9   2023-07-09        90.05   0.0000000
## 10  2023-07-10        88.55   0.0000000
## 11  2023-07-11        87.95   0.0000000
## 12  2023-07-12        88.15   0.0000000
## 13  2023-07-13        87.20   0.0000000
## 14  2023-07-14        88.20   0.0000000
## 15  2023-07-15        87.00   0.0000000
## 16  2023-07-16        88.10   0.0000000
## 17  2023-07-17        89.20   0.0000000
## 18  2023-07-18        89.25   0.0000000
## 19  2023-07-19        90.40   0.0000000
## 20  2023-07-20        89.40   0.0000000
## 21  2023-07-21        89.95   0.0000000
## 22  2023-07-22        89.45   0.0000000
## 23  2023-07-23        89.05   0.0000000
## 24  2023-07-24        89.10   0.0000000
## 25  2023-07-25        88.00   0.0000000
## 26  2023-07-26        89.50   0.0000000
## 27  2023-07-27        89.55   0.0000000
## 28  2023-07-28        89.95   0.0000000
## 29  2023-07-29        89.25   0.0000000
## 30  2023-07-30        89.55   0.0000000
```

```
## 31  2023-07-31        88.15   0.0000000
## 32  2023-08-01        88.55   0.0000000
## 33  2023-08-02        88.65   0.0000000
## 34  2023-08-03        89.55   0.0000000
## 35  2023-08-04        90.30   0.0000000
## 36  2023-08-05        91.15   0.0000000
## 37  2023-08-06        89.40   0.0000000
## 38  2023-08-07        88.95   0.0000000
## 39  2023-08-08        88.75   0.0000000
## 40  2023-08-09        89.00   0.0000000
## 41  2023-08-10        89.25   0.0000000
## 42  2023-08-11        89.20   0.0000000
## 43  2023-08-12        87.90   0.0000000
## 44  2023-08-13        88.10   0.0000000
## 45  2023-08-14        88.30   0.0000000
## 46  2023-08-15        88.00   0.0000000
## 47  2023-08-16        88.80   0.0000000
## 48  2023-08-17        89.05   0.0000000
## 49  2023-08-18        90.15   0.0000000
## 50  2023-08-19        90.30   0.0000000
## 51  2023-08-20        89.30   0.0000000
## 52  2023-08-21        89.10   0.0000000
## 53  2023-08-22        89.40   0.0000000
## 54  2023-08-23        88.40   0.0000000
## 55  2023-08-24        87.85   0.0000000
## 56  2023-08-25        86.50   0.0000000
## 57  2023-08-26        88.45   0.0000000
## 58  2023-08-27        87.60   0.0000000
## 59  2023-08-28        87.15   0.0000000
## 60  2023-08-29        88.30   0.0000000
## 61  2023-08-30        85.80   0.5728465
## 62  2023-08-31        85.90   1.0456930
## 63  2023-09-01        85.25   2.1685395
## 64  2023-09-02        85.25   3.2913860
## 65  2023-09-03        85.90   3.7642325
## 66  2023-09-04        85.80   4.3370790
## 67  2023-09-05        86.20   4.5099255
## 68  2023-09-06        84.60   6.2827720
## 69  2023-09-07        84.75   7.9056185
## 70  2023-09-08        85.25   9.0284650
## 71  2023-09-09        85.05  10.3513116
## 72  2023-09-10        85.25  11.4741581
## 73  2023-09-11        85.55  12.2970046
## 74  2023-09-12        85.30  13.3698511
## 75  2023-09-13        83.10  16.6426976
## 76  2023-09-14        83.65  19.3655441
## 77  2023-09-15        83.70  22.0383906
## 78  2023-09-16        82.25  26.1612371
## 79  2023-09-17        81.85  30.6840836
## 80  2023-09-18        81.70  35.3569301
## 81  2023-09-19        82.40  39.3297766
## 82  2023-09-20        83.00  42.7026231
## 83  2023-09-21        81.60  47.4754696
## 84  2023-09-22        81.20  52.6483161
```

```
## 85  2023-09-23        82.75  56.2711626
## 86  2023-09-24        80.40  62.2440091
## 87  2023-09-25        79.30  69.3168556
## 88  2023-09-26        78.55  77.1397021
## 89  2023-09-27        78.55  84.9625486
## 90  2023-09-28        78.65  92.6853951
## 91  2023-09-29        76.35 102.7082417
## 92  2023-09-30        77.00 112.0810882
## 93  2023-10-01        77.10 121.3539347
## 94  2023-10-02        76.95 130.7767812
## 95  2023-10-03        77.70 139.4496277
## 96  2023-10-04        77.85 147.9724742
## 97  2023-10-05        78.20 156.1453207
## 98  2023-10-06        76.35 166.1681672
## 99  2023-10-07        75.60 176.9410137
## 100 2023-10-08        74.80 188.5138602
## 101 2023-10-09        74.25 200.6367067
## 102 2023-10-10        75.15 211.8595532
## 103 2023-10-11        75.85 222.3823997
## 104 2023-10-12        75.80 232.9552462
## 105 2023-10-13        75.45 243.8780927
## 106 2023-10-14        74.20 256.0509392
## 107 2023-10-15        72.90 269.5237857
## 108 2023-10-16        72.65 283.2466322
## 109 2023-10-17        73.10 296.5194787
## 110 2023-10-18        71.90 310.9923252
## 111 2023-10-19        71.05 326.3151718
## 112 2023-10-20        71.25 341.4380183
## 113 2023-10-21        74.10 353.7108648
## 114 2023-10-22        72.35 367.7337113
## 115 2023-10-23        69.65 384.4565578
## 116 2023-10-24        68.85 401.9794043
## 117 2023-10-25        69.35 419.0022508
## 118 2023-10-26        71.40 433.9750973
## 119 2023-10-27        68.90 451.4479438
## 120 2023-10-28        68.60 469.2207903
## 121 2023-10-29        69.35 486.2436368
## 122 2023-10-30        71.05 501.5664833
## 123 2023-10-31        70.50 517.4393298
```

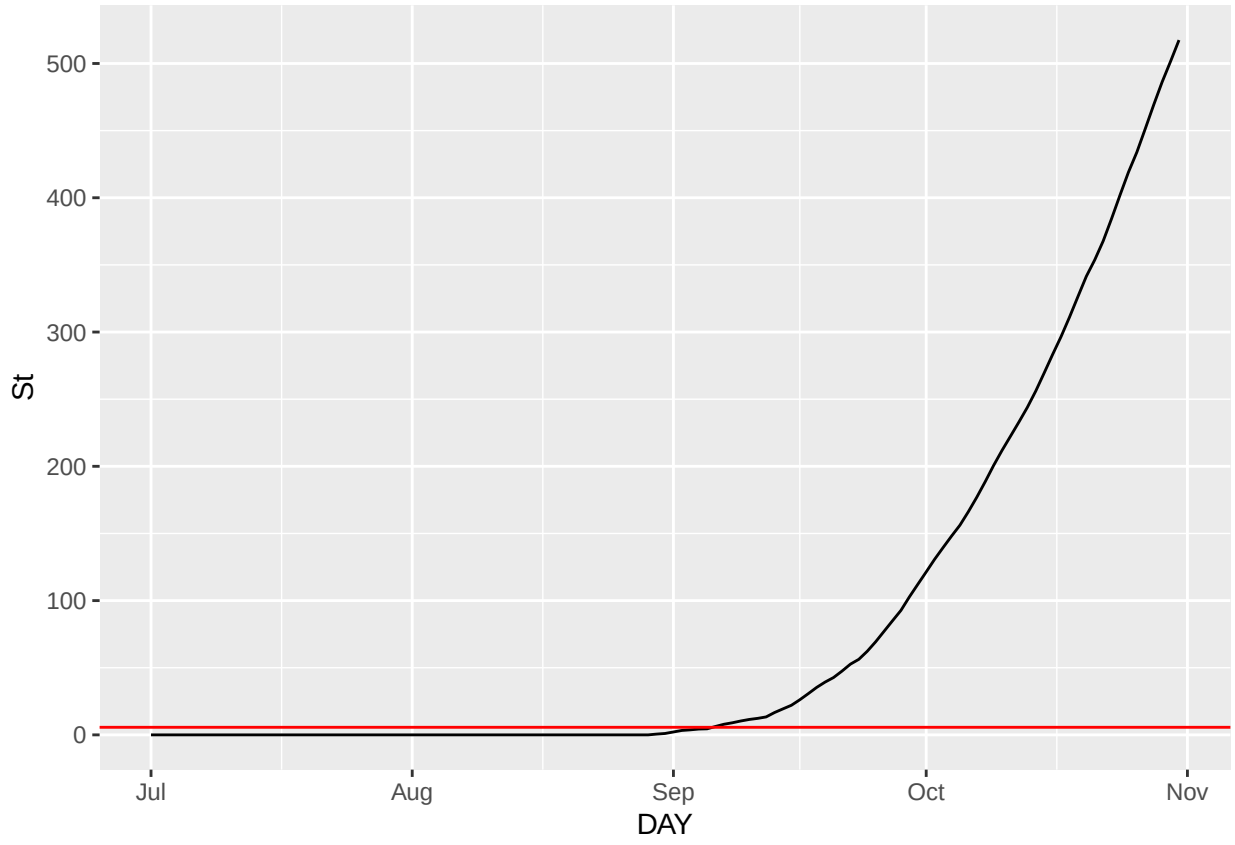
```
#print first date to equal or exceed threshold
i <- min(which(cusum$St >= threshold))
print(paste0("Official End of Summer: ", cusum$DAY[i]))
```

```
## [1] "Official End of Summer: 2023-09-06"
```

Finally, I plotted my results. You can see a change was detected where the black line intersects with the red.

```
ggplot(cusum, aes(x=DAY, y=St, group=1))+
  geom_line()+
  geom_hline(yintercept=threshold, color="red")+
  scale_x_continuous(limit=cusum$DAY)+
  scale_x_date(date_breaks="1 month", date_labels="%b")
```

```
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
```



Part 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).

I started this question by calculating average temperatures for each summer season over the 20 year period. This gave me 20 data points, each representing the average temperature for a single summer.

```
#reset to original data frame
temp_data <- read.table("temps.txt", header=TRUE)

#calculate the mean of each column (year)
col_means <- colMeans(temp_data[,2:ncol(temp_data)])
```

Similar to part 1, I created a new data frame with 3 columns: the year, the average summer temperature, and a blank column to store the St value. I used this new data frame to apply the cusum approach.

```
#add new row with calculated means to data frame
new_row <- c("Average Temp", col_means)
temp_data <- rbind(temp_data, new_row)

#transpose data frame to get years as the rows and days as the columns
```

```r
transposed_temp_data <- as.data.frame(t(temp_data))
colnames(transposed_temp_data) <- transposed_temp_data[1,]
transposed_temp_data <- transposed_temp_data[-1,]

#set up a chart to apply CUSUM approach
cusum <- subset(transposed_temp_data, select="Average Temp")
cusum["Year"] <- rownames(cusum)
cusum <- cusum[,c("Year", "Average Temp")] #change order of columns
rownames(cusum) <- 1:nrow(cusum)

#add empty column to store St
cusum[,"St"] <- NA

#make sure data is numeric
cusum$'Average Temp' <- as.numeric(cusum$'Average Temp')
```

Next, I determined the mean, C-value, and threshold to be used in the CUSUM approach.

Like above, I plotted key points in the time series data to see where/if the temperature begins to rise. As you can see in the graph below, temperature spikes in 2010, but there does not seem to be a consistent increase. For this reason, I used all the data from 1996-2015 to approximate the mean.

I calculated the standard deviation from 1996 to 2015 to help me choose an appropriate threshold and C-value. I chose C=2(std_dev) and T=5(std_dev) as a starting point. After trial and error with different values of C and T, I decided C=0 and T=5(std_dev) yielded an appropriate result. A C value of 0 makes the model more sensitive to smaller changes.

```r
#plot temperatures
ggplot(cusum, aes(x=Year, y=`Average Temp`, group=1))+
  geom_line()+
  scale_x_discrete(limit=cusum$Year)
```


```
#calculate mean
mean <- mean(cusum$`Average Temp`)

#calculate standard deviation
std <- sd(cusum$`Average Temp`)

#choose threshold
threshold = st_dev*5

#choose C value
C <- 0
```

Then, I applied the CUSUM approach and printed out the first value to exceed the predetermined threshold.

```
#calculate first value of St
first_St <- max(0, cusum[1, 'Average Temp'] - mean)
cusum[1, "St"] <- first_St

#calculate remaining values of St
#because we are trying to detect a decrease, we flip equation to mean - X
for(i in 2:nrow(cusum)){

  #calculate x and St-1
  x <- cusum[i, 'Average Temp']
  previous_St <- cusum[i-1, "St"]
```

```
  #calculate St
  St <- max(0, previous_St +(x - mean - C))

  #update table with value of St
  cusum[i, "St"] <- St
}

#print table
cusum
```

```
##     Year Average Temp         St
## 1  X1996     83.71545 0.37642276
## 2  X1997     81.67480 0.00000000
## 3  X1998     84.26016 0.92113821
## 4  X1999     83.35772 0.93983740
## 5  X2000     84.03252 1.63333333
## 6  X2001     81.55285 0.00000000
## 7  X2002     83.58537 0.24634146
## 8  X2003     81.47967 0.00000000
## 9  X2004     81.76423 0.00000000
## 10 X2005     83.35772 0.01869919
## 11 X2006     83.04878 0.00000000
## 12 X2007     85.39837 2.05934959
## 13 X2008     82.51220 1.23252033
## 14 X2009     80.99187 0.00000000
## 15 X2010     87.21138 3.87235772
## 16 X2011     85.27642 5.80975610
## 17 X2012     84.65041 7.12113821
## 18 X2013     81.66667 5.44878049
## 19 X2014     83.94309 6.05284553
## 20 X2015     83.30081 6.01463415
```

As you can see in the table above, St never equals or exceeds our threshold, even with a C value of 0. This means that even a highly sensitive CUSUM model did not identify enough change to warrant flagging this as an increase over time. Thus, I think we can draw the conclusion that Atlanta's summer climate has not gotten warmer between 1996 and 2015.