Homework 3 (ISYE 6501) — Questions 5.1 & 6.1 & 6.2

Sorry in advance to my peer reviewers! (— —) I like to write out my reasoning and can be… wordy… But I will try to be even more brief this time after taking the advice from HW2 peers!

I want to thank https://r-graphics.org/ for guiding me on how to create clear and effective graphs and plots. While I did not copy any specific lines of code, I used it as a helpful reference when creating the plots for Question 6.2.
I also want to thank R's built in help() function, which has been extremely useful for understanding things or functions I do not immediately get, and for providing examples that helped me write my own, more complex scripts.

Question 5.1 Answer

```
rm(list = ls())
library(outliers)
set.seed(16) #not needed but I added this to continue practicing adding a seed to my code blocks

crime_data <- read.table(
  "C:/Users/mstout/OneDrive - AANP/Documents/Workspace.MAIN/edX/GTx.ISYE6501/data/Homework3_ISYE_6501/da
  header = TRUE
)
```

```
dim(crime_data)
```

```
## [1] 47 16
```

```
g <- grubbs.test(na.omit(crime_data$Crime))
#type = 20 was not working for .Rmd file,
#so I had to use this after already knowing the higher value was the possible outlier
g
```

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

This was a pretty simple setup, I loaded the uscrime dataset and ran Grubbs test on the [Crime] column (after removing any missing values). Even though they tell us exactly which test to run, I still wanted to know "why this test?"
So, from my understanding, this specific kind of test is just to see if there is one unusually large or small value compared to the rest of the data, which is perfect for the question we are being asked.

ANALYSIS:
[1993] is was the most extreme value in the data, and is extreme in the sense this is the largest value.
I got [G = 2.81287], which is the Grubbs test statistic, this shows us that the largest value is [~2.81] standard deviations (distance) away from the mean.
I also got [U = 0.82426], I didn't know what this meant, so I used 'help(grubbs.test)' which essentially told me: 1. Smaller U corresponds to a more extreme observation
2. It can be ignored for this assignment
FInally, I got [p-value = 0.07887] (7.87%), This result says: If all the crime rates follow the same general pattern, results this like this would show up about 8 times out of 100 just by chance.
Before deciding my thoughts, I wanted to also make sure to say that Grubbs is really good for catching the extremes, which could be the outlier, but not so good at catching multiple large values at once. For example, the next closest value is [1969], which is pretty close! But Grubbs doesn't know that, and treats [1993] like it's all alone out there.
So, because we know the upper tail contains more than one large value, the case that [1993] is a lone, truly aberrant point is much weaker.
Therefore, in my personal and professional opinion, there is no outlier in this dataset. 8 out of 100 is not insanely rare, rare enough to notice, but not rare enough I could confidently say "this can't occur"naturally."

Question 6.1 Answer

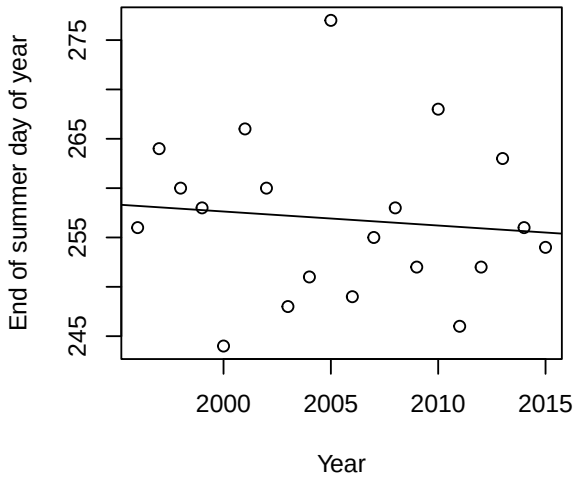
I believe a change detection model would be appropriate for monitoring my heart rate during a semi long run, since heart rate is measured continuously over time and usually stays around a normal level when my pace and conditions are consistent.
During a recent 5 mile run at a 12:00 mile pace (I know I'm slow), my average heart rate was [159bpm], with a maximum of [169bpm]. So, a meaningful change to me would be when my heart rate starts to stay above my normal running level, which could be due to increased effort, fatigue, dehydration, or changes in terrain (uphill).
Using the CUSUM technique, the baseline mean would be set to my typical heart rate of [159bpm].
The threshold or reference value would be based on the smallest sustained increase I care about, maybe something like a prolonged [5/6bpm] rise above my average rather than short spikes.
The critical value would be larger to avoid false alarms, so the model only signals when my heart rate stays high over time instead of reacting to a single high reading like [169bpm].
This lets the model catch real changes while ignoring normal short term variation.

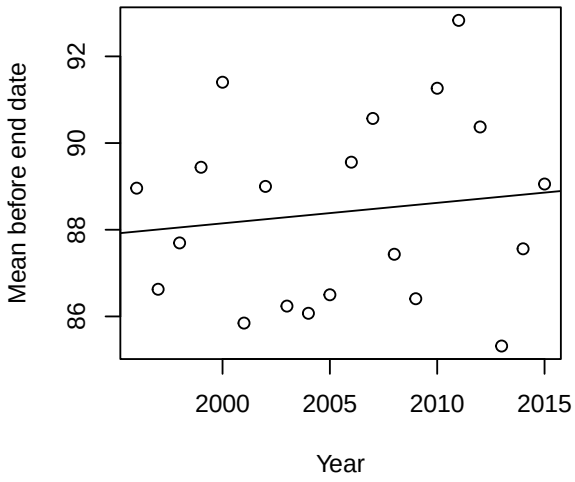
Question 6.2 Answer

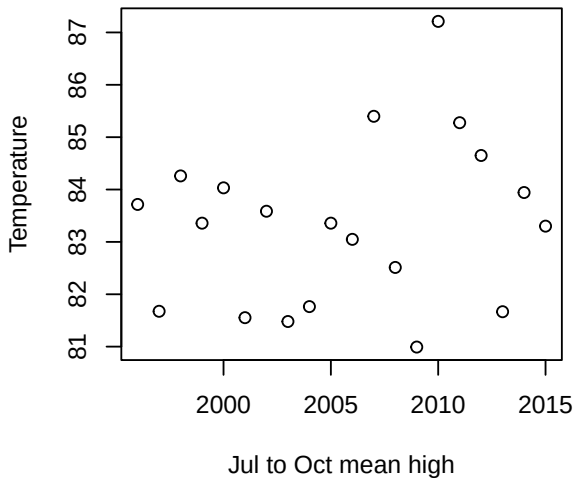
Part 1

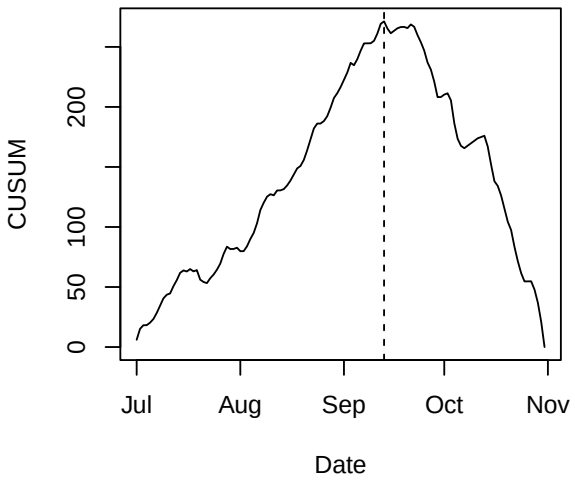
```
path <- "C:/Users/mstout/OneDrive - AANP/Documents/Workspace.MAIN/edX/GTx.ISYE6501/data/Homework3_ISYE_
w <- read.delim(path, sep = "\t", header = TRUE, check.names = FALSE)

print(dim(w))

years <- names(w)[names(w) != "DAY"]
d <- as.Date(paste0("2000-", w$DAY), format = "%Y-%d-%b")
mmdd <- format(d, "%m-%d")

L <- do.call(rbind, lapply(years, function(y) {
  data.frame(
    year = as.integer(y),
    date = as.Date(paste0(y, "-", mmdd)),
```

```
    tmax = as.numeric(w[[y]])
  )
}))
L <- L[order(L$year, L$date), ]

one_year <- function(z) {
  z <- z[order(z$date), ]
  x <- z$tmax
  s <- cumsum(x - mean(x, na.rm = TRUE))
  k <- which.max(s)
  data.frame(
    year = z$year[1],
    end_date = z$date[k],
    end_doy = as.integer(format(z$date[k], "%j")),
    mean_before = mean(x[1:k], na.rm = TRUE),
    mean_after = mean(x[(k + 1):length(x)], na.rm = TRUE)
  )
}

R <- do.call(rbind, lapply(split(L, L$year), one_year))
R$drop_after <- R$mean_after - R$mean_before
R <- R[order(R$year), ]

A <- aggregate(tmax ~ year, L, mean)
A <- A[order(A$year), ]

Z <- split(L, L$year)[["2014"]]
Z <- Z[order(Z$date), ]
S <- cumsum(Z$tmax - mean(Z$tmax, na.rm = TRUE))
k2014 <- which.max(S)

par(mfrow = c(2, 2))

plot(R$year, R$end_doy, xlab = "Year", ylab = "End of summer day of year")
abline(lm(end_doy ~ year, data = R))

plot(R$year, R$mean_before, xlab = "Year", ylab = "Mean before end date")
abline(lm(mean_before ~ year, data = R))

plot(A$year, A$tmax, xlab = "Jul to Oct mean high", ylab = "Temperature")

plot(Z$date, S, type = "l", xlab = "Date", ylab = "CUSUM")
abline(v = Z$date[k2014], lty = 2)
```










```
dim(w)
```

```
## [1] 123  21
```

```
dim(L)
```

```
## [1] 2460    3
```

```
dim(R)
```

```
## [1] 20  6
```

```r
head(R,20) #just want to ensure it all loaded as expected
```

```
##      year   end_date end_doy mean_before mean_after drop_after
## 1996 1996 1996-09-12     256    88.95946   75.79592 -13.163541
## 1997 1997 1997-09-21     264    86.62651   71.40000 -15.226506
## 1998 1998 1998-09-17     260    87.69620   78.09091  -9.605293
## 1999 1999 1999-09-15     258    89.44156   73.17391 -16.267645
## 2000 2000 2000-08-31     244    91.40323   76.54098 -14.862242
## 2001 2001 2001-09-23     266    85.84706   71.94737 -13.899690
## 2002 2002 2002-09-17     260    89.00000   73.86364 -15.136364
## 2003 2003 2003-09-05     248    86.23881   75.78571 -10.453092
## 2004 2004 2004-09-07     251    86.07246   76.25926  -9.813205
## 2005 2005 2005-10-04     277    86.50000   72.18519 -14.314815
## 2006 2006 2006-09-06     249    89.55882   75.00000 -14.558824
## 2007 2007 2007-09-12     255    90.56757   77.59184 -12.975731
## 2008 2008 2008-09-14     258    87.43421   74.55319 -12.881019
## 2009 2009 2009-09-09     252    86.40845   73.59615 -12.812297
## 2010 2010 2010-09-25     268    91.26437   77.41667 -13.847701
## 2011 2011 2011-09-03     246    92.83077   76.81034 -16.020424
## 2012 2012 2012-09-08     252    90.37143   77.09434 -13.277089
## 2013 2013 2013-09-20     263    85.31707   74.36585 -10.951220
## 2014 2014 2014-09-13     256    87.56000   78.29167  -9.268333
## 2015 2015 2015-09-11     254    89.05479   74.90000 -14.154795
```

This was the main setup for Part 1!
I used https://r-graphics.org/ and AI to help me shape this part, as I am not as experienced with graphs or plotting within R yet. As I said before, I did not copy any specific lines of code, I just used it as a helpful reference for me to better understand how best to write it.
I loaded the temps dataset, reshaped it into a long table with one row per [date] per [year] using `lapply()` and `do.call()`, and then processed each year separately using `split()`.
Within each year, I centered daily high temperatures by subtracting the yearly July to October mean and used `cumsum()` to compute the CUSUM values. I defined the end of summer as the date where the CUSUM reached its maximum, using `which.max()`.
I also included a single year (2014) example plot to make it easier to see how the CUSUM curve increases during consistently warm days and peaks when temperatures begin to drop.

Part 2

```r
summary(lm(mean_before ~ year, data = R))
```

```
##
## Call:
## lm(formula = mean_before ~ year, data = R)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.4454 -1.9256 -0.0794  1.4201  4.1629
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)  -6.47092  170.43134  -0.038    0.970
## year          0.04731    0.08498   0.557    0.585
##
## Residual standard error: 2.191 on 18 degrees of freedom
## Multiple R-squared:  0.01693,    Adjusted R-squared:  -0.03769
## F-statistic: 0.3099 on 1 and 18 DF,  p-value: 0.5846
```

```
summary(lm(end_doy ~ year, data = R))
```

```
##
## Call:
## lm(formula = end_doy ~ year, data = R)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -13.6316  -4.7803  -0.6368   3.4750  20.0789
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 541.8421   642.6503   0.843    0.410
## year         -0.1421     0.3204  -0.443    0.663
##
## Residual standard error: 8.263 on 18 degrees of freedom
## Multiple R-squared:  0.01081,    Adjusted R-squared:  -0.04415
## F-statistic: 0.1967 on 1 and 18 DF,  p-value: 0.6627
```

This was the setup for Part 2!
After I identified one end of summer date per year, I fit two simple linear regression models using `lm()`.
The first regression looked at whether average daily high temperatures before the CUSUM peak changed over time, which I used to check if summers became warmer.
The second regression looked at whether the calendar day of year for the CUSUM end of summer date changed over time, which I used to check if summers have been lasting longer.

ANALYSIS:
Using the CUSUM results from Part 1, I found that summer in Atlanta typically ends in early to mid September, most often between about [September 5] and [September 20].
Before the CUSUM peak, average daily highs were usually in the upper [80s] to low [90s], and after the peak they dropped to the mid [70s], a decrease of about [10 to 15] degrees.
The estimated temperature trend was about [0.05] degrees per year with a p value of about [0.58], and the trend in the end of summer date was about [-0.14] days per year with a p value of about [0.66].
Both p values are fairly large, meaning these trends are not statistically significant and could easily be due to random variation rather than a real change over time.
Based on these results from [1996] to [2015], I believe there is no strong statistical evidence that Atlanta's summers became warmer or lasted longer over this period (based on this data alone).