Detecting Weather Changes Using CUSUM

Eric Cai

2023-09-21

Detecting Weather Changes Using CUSUM

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.

Let's read in our file and inspect it.

```
file2 <- 'temps.txt'
## let's get rid of the x prefix in the column names
col_names <- seq(1996,2015,by=1)
temps <- read.table(file2,header=T)
colnames(temps)[-1] <- col_names
head(temps, 4)
```

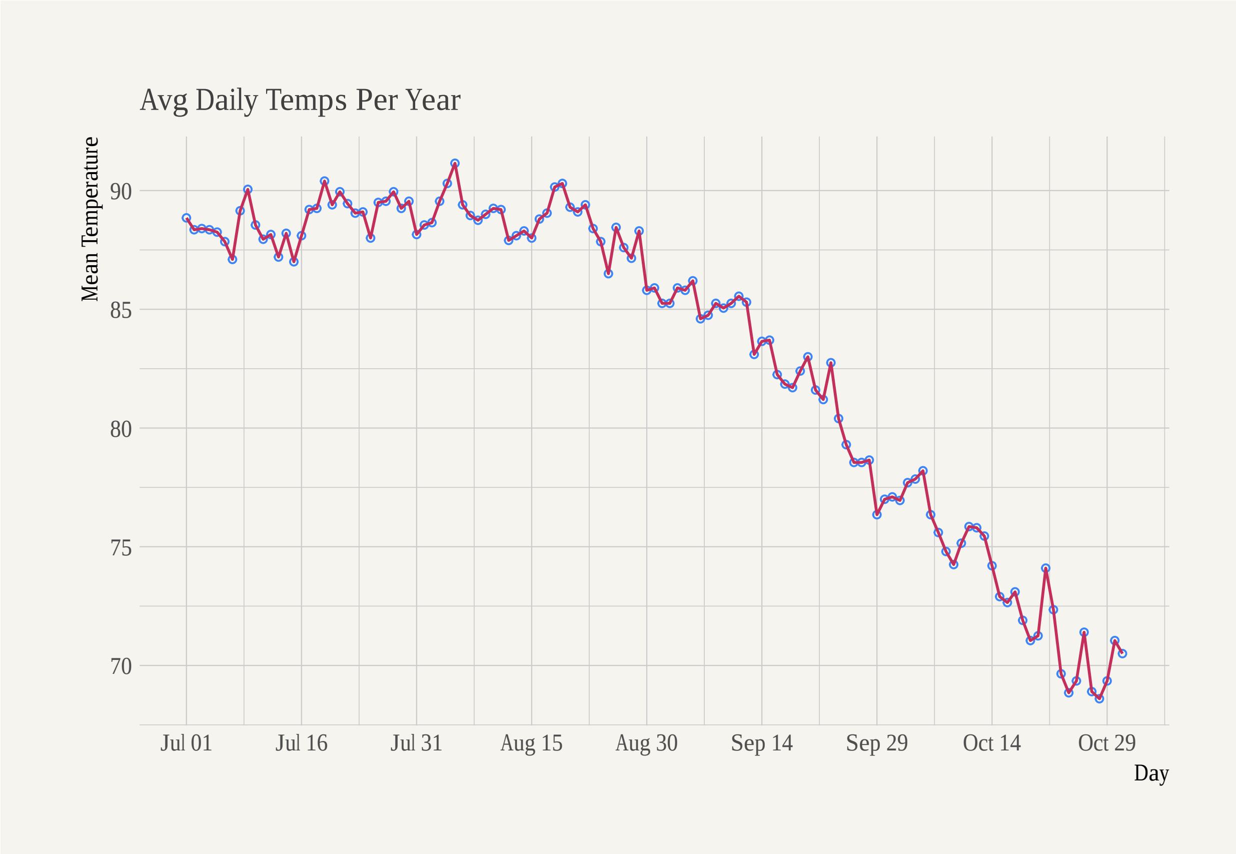
```
##     DAY 1996 1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009
## 1 1-Jul   98   86   91   84   89   84   90   73   82   91   93   95   85   95
## 2 2-Jul   97   90   88   82   91   87   90   81   81   89   93   85   87   90
## 3 3-Jul   97   93   91   87   93   87   87   87   86   86   93   82   91   89
## 4 4-Jul   90   91   91   88   95   84   89   86   88   86   91   86   90   91
##   2010 2011 2012 2013 2014 2015
## 1   87   92  105   82   90   85
## 2   84   94   93   85   93   87
## 3   83   95   99   76   87   79
## 4   85   92   98   77   84   85
```

Visualizing Our Data

Let's chart the temperatures on a graph. There are a couple of ways that we chart it. We can chart it by aggregating the averages of the temperatures of each day throughout the years or we can aggregate the temperatures per year. Both are just as easy, but since our data is already formatted such that we can take the average of each row, let's just do that.

```
## i'm just going to create a new df
df <- melt(temps,
           id.vars = c("DAY"),
           variable.name = "Year",
           value.name = "Temp")
## relabel the DAY column
df$DAY <- as.Date(df$DAY, format = "%d-%b")
## aggregate the days and take the average
```

```r
mean_temp <- df %>%
  group_by(DAY) %>%
  summarize(
    Mean_Temperature = mean(Temp, na.rm = TRUE)
  )
## the following few lines of code are formatting variables for the plot
breaks <- mean_temp$DAY[c(seq(1, length(mean_temp$DAY), by = 15))]
dates <- lubridate::ymd(breaks)
custom_labels <- format(breaks, format = "%B %d")
reset_col <- lubridate::ymd(mean_temp$DAY[c(seq(1, length(mean_temp$DAY)))])
## our plot
mean_temp %>%
        ggplot( aes(x=DAY, y=Mean_Temperature, group=1)) +
        geom_point(shape=21, color="#3D85F7", fill="#F6F5E9", size=1) +
        geom_line(color = "#C32E5A") +
        theme_minimal(base_family = "Fira Sans Compressed") +
        ggtitle("Avg Daily Temps Per Year") +
        labs(x = "Day", y = "Mean Temperature") +
        theme_ipsum() +
        theme(
                plot.title = element_text(
                        size = 12,
                        face = "bold",
                        vjust = 0,
                        color = "grey25"
    ),
                axis.text.x = element_text(size = 9),
                axis.text.y = element_text(size = 9),
                plot.background = element_rect(fill = "#F5F4EF", color = NA)
        ) +
        scale_x_date(
                breaks = breaks,
                labels = custom_labels,
                date_labels = "%b %d"
  )
```



. . . And this is what we get.

Choosing Our Range for μ

Looking at our graph, it looks like the temperatures don't really make a remarkable shift into "cooler" weather until the end of August. Average temperatures from July 1 to August 28ish vacillate between the low 90s (F) and the upper 80s. Only after August do we see a downward trend in temperature.

Since we want to detect the change in seasons from summer to fall, it seems advisable to take the average of temperatures *before* any significant temperature drop occurs (otherwise, it'll bring down our average and affect our detection of when the change in temperature actually happens). For example, we don't want to take the average from July to October because our change detection model will flag the change in temperature too late.

Looking at mean_temp, the first time the average temperature drops to 85 degrees is August 30, so I'll be taking the average of the temperatures between July 1 to August 29 as our μ variable.

```
jul_aug <- mean_temp$Mean_Temperature[1:60]
mu <- mean(jul_aug)
```

The average temperature (of average temperatures) that we get between July and August turns out to be 88.7775, which makes sense since our temperatures range from 86.5 to 91.15.

Choosing C and T

Since CUSUM works by detecting changes from the mean, determining our C and T values requires calculating how far our temperatures are from the mean. In other words, we need to calculate the standard deviation of x_t to determine C and T. Perusing through the `qcc::cusum` documentation reveals alternative ways to define the variables C and T.

C is also referred to as the reference value or allowable slack k, regulating the size of s_t by determining the number of standard deviations from μ. k then is appropriately called the slack which either makes s_t increase faster or slower. k or C is computed by multiplying the standard deviation by a scaling factor (usually 0.5 to 1). In `qcc::cusum`, the argument for C is `se.shift`, which defaults to a scaling factor of $\frac{1}{2}$.

T is alternatively referred to as the decision interval, H. (Don't ask why, I don't know lol.) Similar to C, T is a multiple of the standard deviation (σ), which is usually set to 4σ or 5σ. A larger value of t (e.g., 5σ) makes it more conservative and requires more substantial deviations from the mean to declare a change, while a smaller value of t (e.g., 4σ) makes it more sensitive to smaller deviations. `qcc::cusum` defaults to 5σ and describes it as the number of standard errors of the summary statistics at which the cumulative sum is out of control (one might say, a Ramblin' Wreck, ha ha..).

A CUSUM Approach to Temperature Changes

We're finally ready to implement a CUSUM approach to weather changes in Atlanta. Let's remember the problem that we're trying to solve:

$$s_t = max\{0, s_{t-1} + (\mu - x_t - C)\},$$

where a change is detected when

$$s_t \geq T,$$

that is, when s_t exceeds our threshold of deviation from the mean.

For the astute, you'll notice we're subtracting μ from x_t. This is intentional. Since we're tracking how *falling* temperatures differ from the mean, we subtract from μ. If we wanted to track changes in *rising* temperatures at each time period from the mean, *then* we subtract μ from x_t.

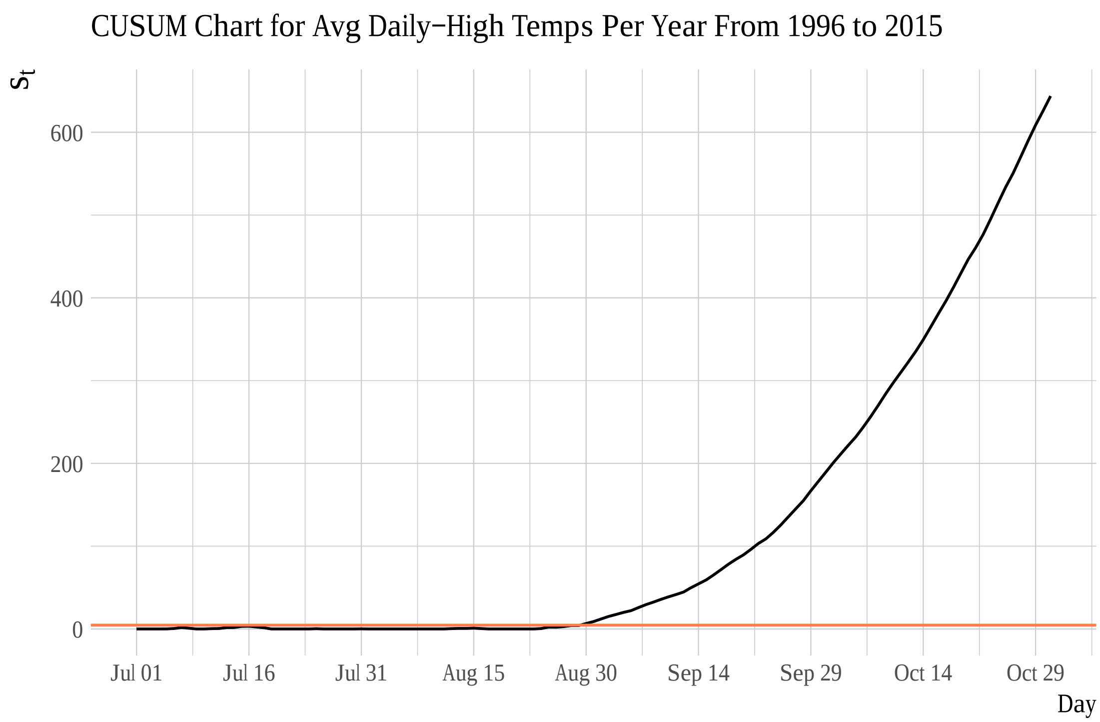
Now that we've formulated our problem, let's code it:

```r
# i'm going to create it as a function because i might be using it later on
cusum_metric <- function(x, sigma, mu, c_val, opposite = F){
        C <- c_val * sigma
        x$s_t <- 0
        ## if opposite is FALSE, then this runs (the lower bound)
        if(!opposite){
                for(i in 2:nrow(x)){
                        x[i, 's_t'] <- max(0,
                                           (x$s_t[i-1]
                                            + mu
                                            - x$Mean_Temperature[i]
                                            - C))
                }
                return(x)
        } else{ ## if opposite is TRUE, then this runs (the upper bound)
                for(i in 2:nrow(x)){
                        x[i, 's_t'] <- max(0,
                                           (x$s_t[i-1]
                                            + x$Mean_Temperature[i]
```

```
                                                  - mu
                                                  - C))
                        }
                        return(x)
                }
}
## setting sigma variable
sig <- sd(jul_aug)
## creating T
t <- 5 * sig
## creating a variable to store our metric table
first <- cusum_metric(mean_temp, sigma=sig, mu=mu, 0.5, FALSE)
change_detect <- format(first[which(first$s_t>t),][1,]$DAY, format = "%d-%b")
```

Let's see what we get:

```
first %>%
        ggplot( aes(x=DAY, y=s_t)) +
        geom_line() +
        geom_hline(yintercept = t, color='#FD814E')+
        scale_x_date(
                breaks = breaks,
                labels = custom_labels,
                date_labels = "%b %d"
        ) +
        theme_minimal() +
        ggtitle("CUSUM Chart for Avg Daily-High Temps Per Year From 1996 to 2015") +
        labs(x = "Day", y = expression(s[t])) +
        theme_ipsum() +
        theme(
                plot.title = element_text(size=12),
                axis.title.y = element_text(size=14),
                axis.title.x = element_text(size=10),
                axis.text.x = element_text(size = 9),
                axis.text.y = element_text(size = 9)
        )
```




```r
first$DAY <- format(reset_col, format = "%b %d")
```

Interpreting Our Results

Well, can't say I didn't expect this. Our T value, 4.6339355, is hilariously tiny and s_t kicks up exponentially. We'll see in just a moment, but s_t exceeds 600 by the end of October. As suspected, our CUSUM metric noticed a downward trend in temperature around Aug 30 and then the curve really trends up around mid September. I think that's fair. Let's see what actual day and temperature we get with our C and T values:

```r
kable(first[which(first$s_t>t),][1,],
      col.names=c('Change Detected',
                  'Avg Temp',
                  paste("$s_t$", collapse = "")),
      align='c',
      caption=paste('Change Detection where T=',
                    round(t, 2),
                    'and C=',
                    round(0.5 * sig, 2))
)
```

Table 1: Change Detection where T= 4.63 and C= 0.46

Change Detected	Avg Temp	s_t
Aug 30	85.8	6.548745

If you'll remember, the dates we used to calculate μ were from July 1 to August 29. CUSUM detected a change the day after on Aug 30. Atlantans can beg to differ with this chart on whether or not the end of August unofficially marks the end of summer. Perhaps if we set C to σ rather than $\frac{\sigma}{2}$, we'd make s_t less sensitive to 2-3 degree changes. Let's see what dates we get when we change the C and T values.

```
result_table <- data.frame(change=character(),
                           avg=double(),
                           c_val=numeric(),
                           t=numeric())
c_vals <- c(0.5*sig, sig, 2*sig, 3*sig, 4*sig, 5*sig)
t_vals <- c(4*sig, 5*sig, 12*sig, 15*sig, 36*sig, 45*sig)
for(i in 1:length(c_vals)){
        c_val <- c_vals[i]
        t <- t_vals[i]
        mean_temp_copy <- cusum_metric(mean_temp, sig, mu=mu, c_val, FALSE)
        first_change <- mean_temp_copy[which(mean_temp_copy$s_t>t),]
        vals <- first_change[1,c('DAY', 'Mean_Temperature')]
        result_row <- data.frame(change = format(vals$DAY, format="%B %d"),
                                 avg = vals$Mean_Temperature,
                                 c_val = round(c_val,2),
                                 t = round(t, 2))
        result_table <- rbind(result_table, result_row)
}
kable(result_table,
      col.names=c('Change Detected Date',
                  'Avg Temp',
                  'C',
                  'T'),
      align='c',
      caption='Change Detection at Different C and T Values'
)
```

Table 2: Change Detection at Different C and T Values

Change Detected Date	Avg Temp	C	T
August 28	87.15	0.46	3.71
August 31	85.90	0.93	4.63
September 06	84.60	1.85	11.12
September 13	83.10	2.78	13.90
September 24	80.40	3.71	33.36
September 27	78.55	4.63	41.71

As we observe, the larger our C and T values the more we dampen the effect of deviations, allowing for our change detection model to detect changes in temperature later than earlier.

Climate Change?

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

We're almost done. Since the question is asking whether Atlanta's summer is getting warmer over time, we no longer want to aggregate by averages per day, but per year. However, if I'm understanding the prompt correctly, it's also specifying whether Atlanta's *summer* climate has gotten warmer. So we don't want to average from months July to October, but from July to the unofficial end of summer that we calculated in 6.2.1. Though I left it up to an Atlantan's interpretation, our CUSUM signaled the end of August to be summer's end.

So this is our game plan: We narrow down our months from July to August 29, average the temperatures of these months per year, plot it, and then apply our CUSUM method to these averages.

Let's code it up and first see how the yearly averages plot on a graph.

```
cat("The first day that CUSUM noticed a change is", change_detect)
```

```
## The first day that CUSUM noticed a change is 30-Aug
```

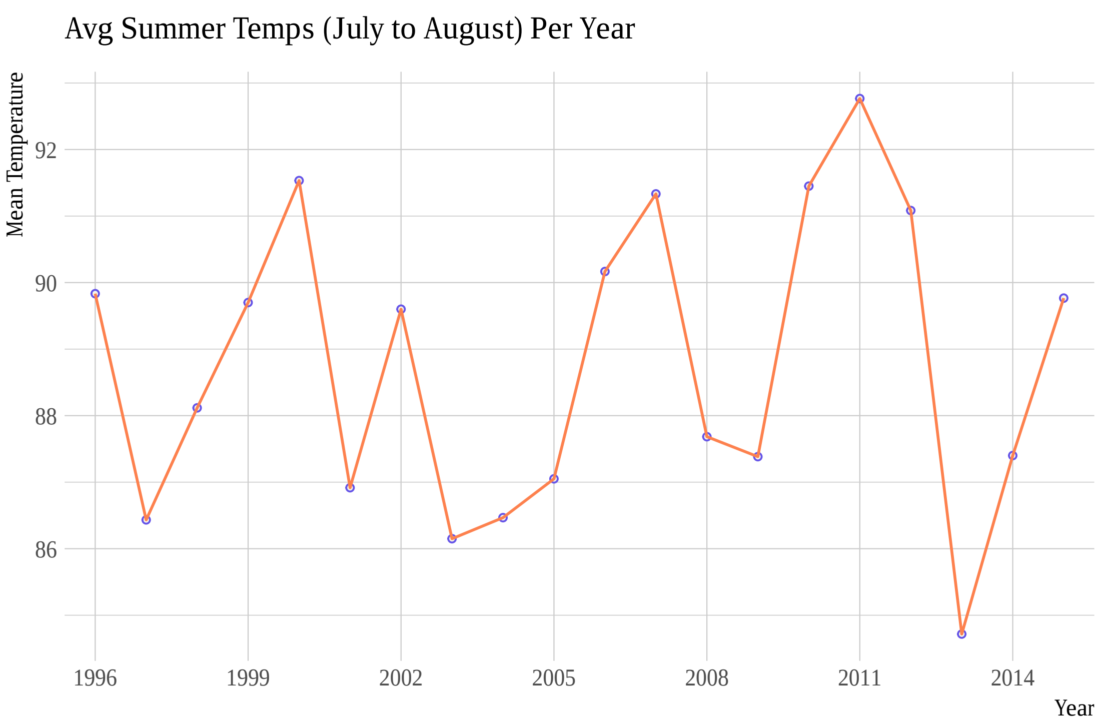
```
## then we locate which row that is in our temps data frame
row_index <- which(temps$DAY == change_detect)
## we set it to the day before
## let's see what we get
summer <- temps[1:row_index-1,]
tail(summer)
```

```
##       DAY 1996 1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009
## 55 24-Aug   91   82   88   80   92   93   93   89   87   85   83   95   83   86
## 56 25-Aug   84   84   90   82   92   90   91   88   82   84   85   94   78   87
## 57 26-Aug   88   87   91   89   90   91   88   89   86   84   88   92   83   90
## 58 27-Aug   84   90   89   88   90   91   84   90   88   86   88   88   80   83
## 59 28-Aug   86   90   90   90   92   81   82   91   90   86   90   88   86   75
## 60 29-Aug   88   91   93   91   92   86   82   89   87   85   90   89   89   86
##    2010 2011 2012 2013 2014 2015
## 55   90   93   86   84   88   89
## 56   89   95   85   82   84   84
## 57   90   99   90   82   86   86
## 58   89   95   90   86   88   85
## 59   87   95   80   90   91   83
## 60   84   93   86   92   92   81
```

The tail of our data shows us exactly what we want to see: the day before summer "ends". So now we take the average of the temperatures per year. This is easy–we can use colMeans().

```
## didn't realize our columns weren't numeric lol
numeric_cols <- sapply(summer, is.numeric)
## calculate the mean of our year columns
yearly_means <- colMeans(summer[, numeric_cols], na.rm = TRUE)
## create a new df
yearly_df <- data.frame(Year = as.factor(names(yearly_means)),
                        Mean_Temperature = yearly_means)
## reset the index
```

```
rownames(yearly_df) <- 1:nrow(yearly_df)
## then we plot
yearly_df %>%
        ggplot( aes(x=Year, y=Mean_Temperature, group=1)) +
        geom_point(shape=21, color="#6554E8", fill="#F6F5E9", size=1) +
        geom_line(color = "#FD814E") +
        theme_minimal() +
        ggtitle("Avg Summer Temps (July to August) Per Year") +
        labs(x = "Year", y = "Mean Temperature") +
        theme_ipsum() +
        theme(
                plot.title =element_text(size=12),
                axis.text.x = element_text(size = 9),
                axis.text.y = element_text(size = 9)
        ) +
        scale_x_discrete(
                breaks = yearly_df$Year[c(seq(1,
                                             length(yearly_df$Year),
                                             by = 3))]
  )
```



The yearly temperatures from July to August fluctuate between mid 80s to low 90s, just as we saw when we averaged the daily highs. As a result, we shouldn't see too much of a difference in our averages below.

Setting Variables

Next, we set our variables:

μ

```
yearly_mu <- mean(yearly_df$Mean)
cat(sprintf("Yearly mu: %.2f\nDaily mu: %.2f", yearly_mu, mu))
```

```
## Yearly mu: 88.78
## Daily mu: 88.78
```

And sure enough, our yearly_mu is virtually the same as the first mu we computed.

σ and T

```
yearly_sig <- sd(yearly_df$Mean)
## creating T
yearly_t <- 5 * yearly_sig
```

Finally, we use CUSUM to see if there are any, note, *upward* trends in temperature. So, now, we subtract μ from x_t, rather than the other way around in 6.2.1:

$$s_t = max\{0, s_{t-1} + (x_t - \mu - C)\}.$$

Anticipating this, we can use the function we created earlier with the proper arguments inputted.

CUSUM

```
## we set our c_val to 0.5 which is 0.5*sigma
## and we set opposite to TRUE for the upper bound
yearly_cusum <- cusum_metric(x=yearly_df,
                             sigma=yearly_sig,
                             mu=yearly_mu,
                             c_val=0.5,
                             opposite=TRUE)
```

Before we actually apply $s_t \geq t$, let's just see this in a table first.

```
kable(yearly_cusum,
      col.names=c('Year',
                  'Avg Temp',
                  paste("$s_t$", collapse = "")),
      align='c',
      caption='CUSUM to Yearly Avg'
)
```

Table 3: CUSUM to Yearly Avg

Year	Avg Temp	s_t
1996	89.83333	0.0000000
1997	86.43333	0.0000000
1998	88.11667	0.0000000
1999	89.70000	0.0000000
2000	91.53333	1.6425093
2001	86.91667	0.0000000
2002	89.60000	0.0000000
2003	86.15000	0.0000000
2004	86.46667	0.0000000
2005	87.05000	0.0000000
2006	90.16667	0.2758427
2007	91.33333	1.7183520
2008	87.68333	0.0000000
2009	87.38333	0.0000000
2010	91.45000	1.5591760
2011	92.76667	4.4350187
2012	91.08333	5.6275280
2013	84.71667	0.4533707
2014	87.40000	0.0000000
2015	89.76667	0.0000000

This table is, at least, consistent with our understanding of s_t. s_t tracks changes when there are upward bumps in temperature. Let's see if s_t is significant enough to trigger anything.

```
kable(yearly_cusum[which(yearly_cusum$s_t>yearly_t),][1,],
      col.names=c('Change Detected',
                  'Avg Temp',
                  paste("$s_t$", collapse = "")),
      align='c',
      caption=paste('Change Detection where T=',
                    round(yearly_t, 2),
                    'and C=',
                    round(0.5 * yearly_sig, 2))
)
```

Table 4: Change Detection where T= 11.13 and C= 1.11

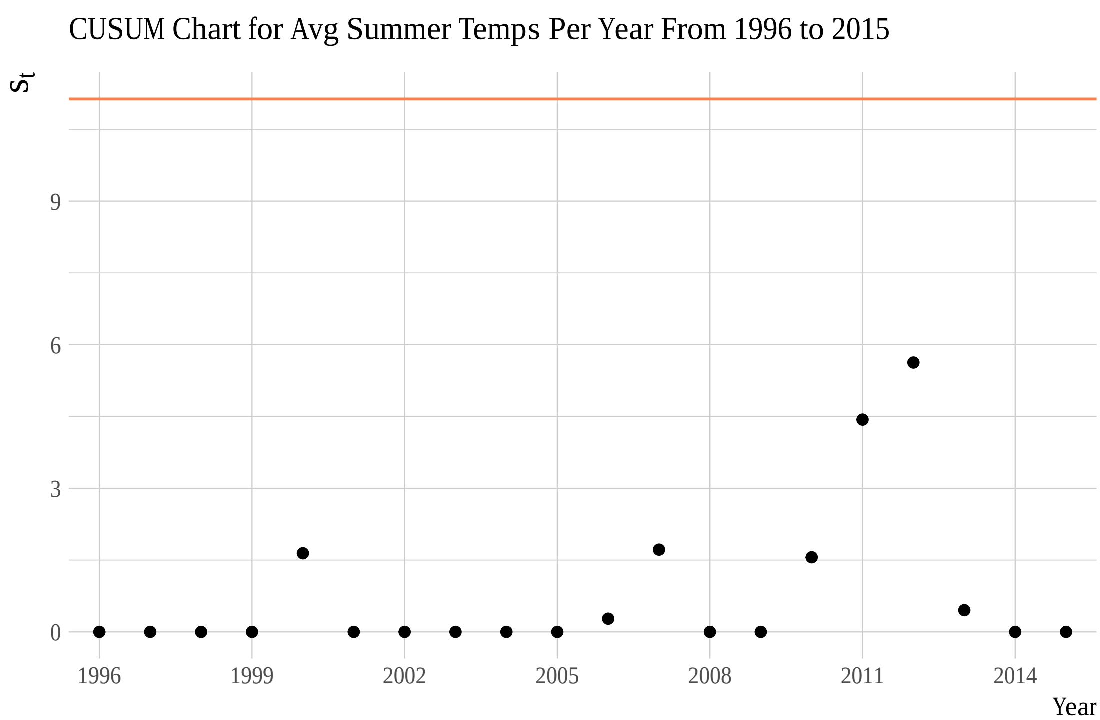
	Change Detected	Avg Temp	s_t
NA	NA	NA	NA

Even though we noticed a sizable spike in average temperature from 2009-2012, it wasn't large enough to cross our T threshold, which is why we didn't notice any change detection where $s_t \geq T$.

In fact, if we make a chart, this is what we see:

```
yearly_cusum %>%
        ggplot( aes(x=Year, y=s_t),group=1) +
        geom_point() +
```

```
        geom_hline(yintercept = yearly_t, color='#FD814E')+
        scale_x_discrete(
                breaks = yearly_cusum$Year[c(seq(1,
                                               length(yearly_cusum$Year),
                                               by = 3))]
) +
        theme_minimal() +
        ggtitle("CUSUM Chart for Avg Summer Temps Per Year From 1996 to 2015") +
        labs(x = "Year", y = expression(s[t])) +
        theme_ipsum() +
        theme(
                plot.title = element_text(size=12),
                axis.title.y = element_text(size=14),
                axis.title.x = element_text(size=10),
                axis.text.x = element_text(size = 9),
                axis.text.y = element_text(size = 9)
        )
```



As our chart shows, all of our s_t points are below our T threshold. Let's lower our C and T values to see if there's any difference.

```
## this is just modified duplicate code from above
yearly_table <- data.frame(change=character(),
                           avg=double(),
                           s_t=numeric(),
                           c_val=numeric(),
```

```
                              t=numeric())
c_vals <- c(0.1*yearly_sig, 0.2 * yearly_sig, 0.3*yearly_sig, .4*yearly_sig, .5*yearly_sig)
t_vals <- c(.1 * yearly_sig, .2 * yearly_sig, .3*yearly_sig, .4*yearly_sig, .5*yearly_sig)
for(i in 1:length(c_vals)){
        c_val <- c_vals[i]
        t <- t_vals[i]
        yearly_copy <- cusum_metric(yearly_df,
                                   yearly_sig,
                                   mu=yearly_mu, c_val, TRUE)
        first_change <- yearly_copy[which(yearly_copy$s_t>t),][1,]
        vals <- first_change[1,c('Year', 'Mean_Temperature', 's_t')]
        result_row <- data.frame(change = vals$Year,
                                 avg = vals$Mean_Temperature,
                                 s_t = vals$s_t,
                                 c_val = round(c_val,2),
                                 t = round(t, 2))
        yearly_table <- rbind(yearly_table, result_row)
}
kable(distinct(yearly_table),
      col.names=c('Change Detected Year',
                  'Avg Temp',
                  paste("$s_t$", collapse = ""),
                  'C',
                  'T'),
      align='c',
      caption='Change Detection at Different C and T Values'
)
```

Table 5: Change Detection at Different C and T Values

Change Detected Year	Avg Temp	s_t	C	T
1999	89.70000	0.4267039	0.22	0.22
2000	91.53333	1.7642411	0.45	0.45
2000	91.53333	1.2684450	0.67	0.67
2011	92.76667	2.6952977	0.89	0.89
2011	92.76667	1.7037054	1.11	1.11

Lowering our threshold and our C actually does help us detect changes in temperature. This time it detected changes in 1999, 2000, and 2011, which makes sense since these years marked the highest spikes in average temperature per year. And of course, CUSUM detected it only because we lowered T and C significantly. We lowered the bar, T, and increased the sensitivity of variations from the mean, C. But we really had to fudge the numbers to trigger change. Even then, it doesn't detect any subsequent change after 2011.

Conclusion

What we conclude is that even with the significant change in C and T values, it doesn't seem like Atlanta's summer climate has gotten warmer. Full disclosure, I am not a climate change denier and I'm sure actual residents of Atlanta feel like each summer experienced is *the* hottest summer. But if I created our CUSUM algorithm correctly and stored our data accurately, then it appears that our algorithm just doesn't think there's enough change in our averages per year to make any significant claim on warmer Atlantan summers.

I think the moral of this analysis is that it's possible to augment our p-values, α values, C values, whatever values, to fit whatever narrative we're trying to tell. This analysis impresses upon the analyst how imperative it is to be as unbiased and honest as possible in how we present and model our data. The way we mitigate arbitrariness, or bias, in selecting our values is by understanding the data well and working with integrity.