Daily Temperature

Load Data

Using the 20 years of daily high temperature data for Atlanta, build and use an exponential smoothing model to help make a judgment of whether the unofficial end of summer has gotten later over the 20 years.

```r
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --
```

```
## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.4     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.0
```

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

```r
library(stats)

temp_data = read.table("temps.txt.",
                       sep="",
                       fill=FALSE,
                       strip.white=TRUE,
                       header = TRUE)

head(temp_data)
```

```
##     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
summary(temp_data)
```

```
##      DAY                X1996           X1997           X1998      
##  Length:123         Min.   :60.00   Min.   :55.00   Min.   :63.00  
##  Class :character   1st Qu.:79.00   1st Qu.:78.50   1st Qu.:79.50  
##  Mode  :character   Median :84.00   Median :84.00   Median :86.00  
##                     Mean   :83.72   Mean   :81.67   Mean   :84.26  
##                     3rd Qu.:90.00   3rd Qu.:88.50   3rd Qu.:89.00  
##                     Max.   :99.00   Max.   :95.00   Max.   :95.00  
##      X1999           X2000            X2001           X2002      
##  Min.   :57.00   Min.   : 55.00   Min.   :51.00   Min.   :57.00  
##  1st Qu.:75.00   1st Qu.: 77.00   1st Qu.:78.00   1st Qu.:78.00  
##  Median :86.00   Median : 86.00   Median :84.00   Median :87.00  
##  Mean   :83.36   Mean   : 84.03   Mean   :81.55   Mean   :83.59  
##  3rd Qu.:91.00   3rd Qu.: 91.00   3rd Qu.:87.00   3rd Qu.:91.00  
##  Max.   :99.00   Max.   :101.00   Max.   :93.00   Max.   :97.00  
##      X2003           X2004           X2005           X2006      
##  Min.   :57.00   Min.   :62.00   Min.   :54.00   Min.   :53.00  
##  1st Qu.:78.00   1st Qu.:78.00   1st Qu.:81.50   1st Qu.:79.00  
##  Median :84.00   Median :82.00   Median :85.00   Median :85.00  
##  Mean   :81.48   Mean   :81.76   Mean   :83.36   Mean   :83.05  
##  3rd Qu.:87.00   3rd Qu.:87.00   3rd Qu.:88.00   3rd Qu.:91.00  
##  Max.   :91.00   Max.   :95.00   Max.   :94.00   Max.   :98.00  
##      X2007           X2008           X2009           X2010      
##  Min.   : 59.0   Min.   :50.00   Min.   :51.00   Min.   :67.00  
##  1st Qu.: 81.0   1st Qu.:79.50   1st Qu.:75.00   1st Qu.:82.00  
##  Median : 86.0   Median :85.00   Median :83.00   Median :90.00  
##  Mean   : 85.4   Mean   :82.51   Mean   :80.99   Mean   :87.21  
##  3rd Qu.: 89.5   3rd Qu.:88.50   3rd Qu.:88.00   3rd Qu.:93.00  
##  Max.   :104.0   Max.   :95.00   Max.   :95.00   Max.   :97.00  
##      X2011           X2012            X2013           X2014      
##  Min.   :59.00   Min.   : 56.00   Min.   :56.00   Min.   :63.00  
##  1st Qu.:79.00   1st Qu.: 79.50   1st Qu.:77.00   1st Qu.:81.50  
##  Median :89.00   Median : 85.00   Median :84.00   Median :86.00  
##  Mean   :85.28   Mean   : 84.65   Mean   :81.67   Mean   :83.94  
##  3rd Qu.:94.00   3rd Qu.: 90.50   3rd Qu.:88.00   3rd Qu.:89.00  
##  Max.   :99.00   Max.   :105.00   Max.   :92.00   Max.   :95.00  
##      X2015     
##  Min.   :56.0  
##  1st Qu.:77.0  
##  Median :85.0  
##  Mean   :83.3  
##  3rd Qu.:90.0  
##  Max.   :97.0
```

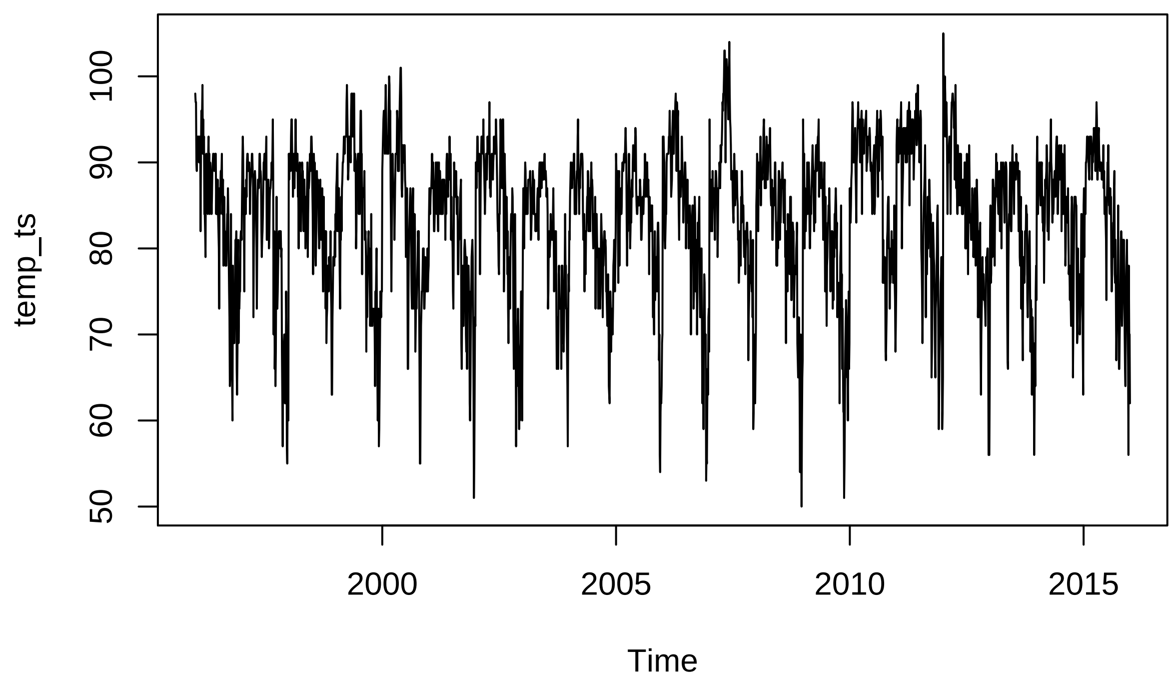
Just loading the data and the necessary packages. Also, the summary and head just give an idea of the data we are working with. Though we should be famailiar with the data because this is from last week, but never hurts to explore.

Plot the time series

```
temp_ts<-ts(as.vector(unlist(temp_data[,2:21])),start=1996,frequency=123)
summary(temp_ts)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   50.00   79.00   85.00   83.34   90.00  105.00
```

```
#plot the time series
ts.plot(temp_ts)
```



Need to convert the data into a time series data which is done with the ts function. It needs to be in a vector or a matrix so I decide to convert it into a vector. We see that the mean temperture of the data is 83.34 with a max of 105 and min of 50. The time series plot helps visualize the data and from just looking at the data there is a lot of flucation of temperature during theses months.

```
# Exponential Smoothing
temp_holt <- HoltWinters(temp_ts, seasonal = "additive")
temp_holt_ml <- HoltWinters(temp_ts, seasonal = "multiplicative")
summary(temp_holt)
```

```
##              Length Class  Mode
## fitted       9348   mts    numeric
```

```
## x            2460   ts     numeric
## alpha           1   -none- numeric
## beta            1   -none- numeric
## gamma           1   -none- numeric
## coefficients  125   -none- numeric
## seasonal        1   -none- character
## SSE             1   -none- numeric
## call            3   -none- call
```

```r
summary(temp_holt_ml)
```

```
##              Length Class  Mode
## fitted       9348   mts    numeric
## x            2460   ts     numeric
## alpha           1   -none- numeric
## beta            1   -none- numeric
## gamma           1   -none- numeric
## coefficients  125   -none- numeric
## seasonal        1   -none- character
## SSE             1   -none- numeric
## call            3   -none- call
```

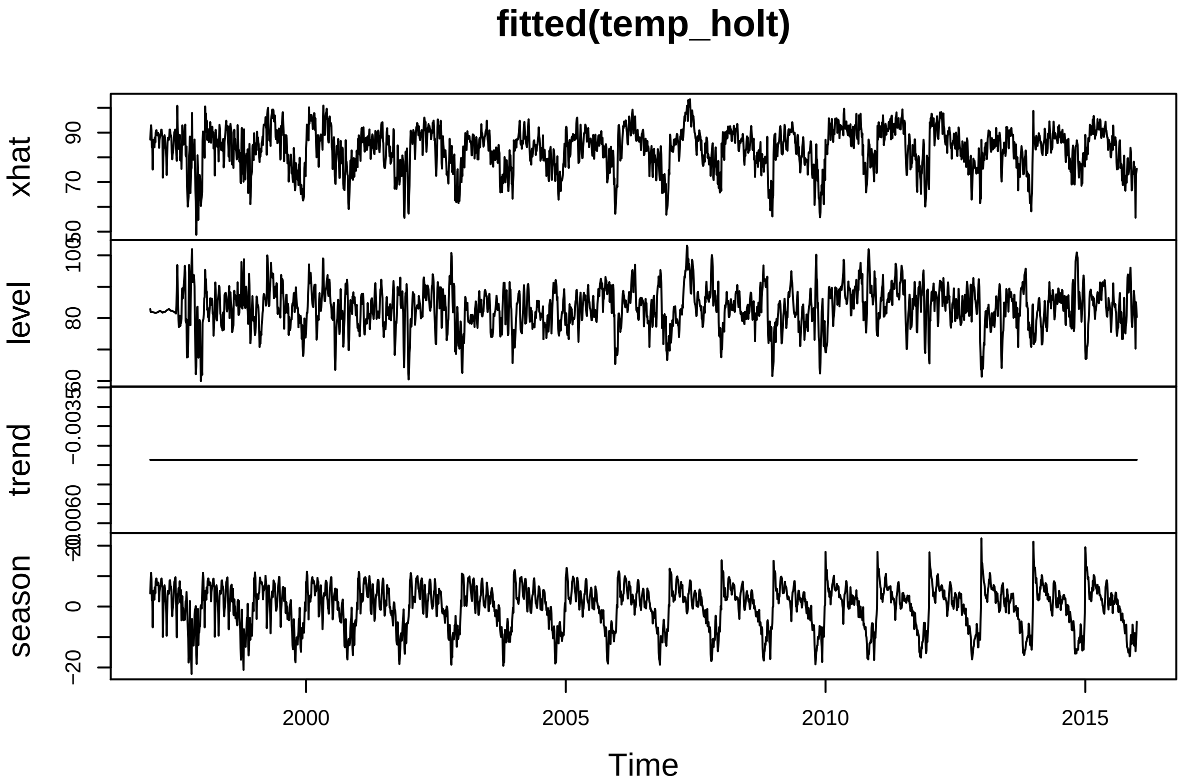
```r
temp_holt$SSE
```

```
## [1] 66244.25
```

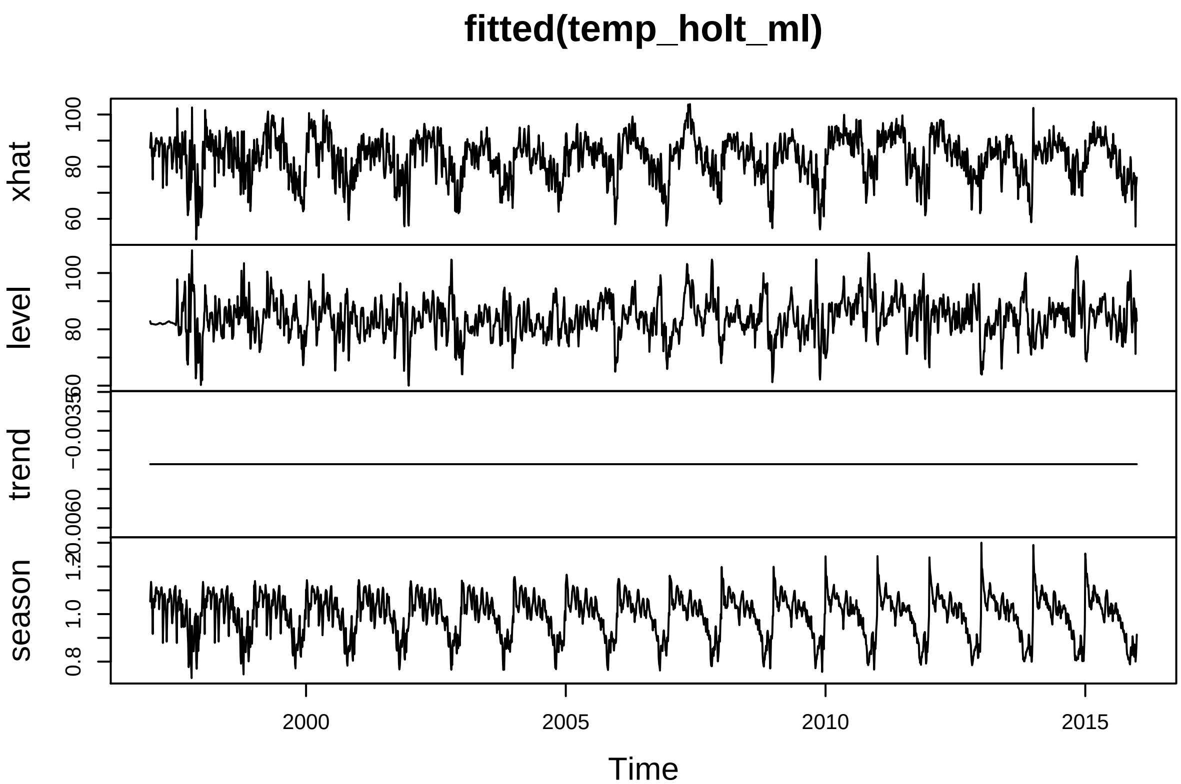
```r
temp_holt_ml$SSE
```

```
## [1] 68904.57
```

```r
plot(fitted(temp_holt))
```




```r
plot(fitted(temp_holt_ml))
```




```r
head(temp_holt$fitted)
```

```
##          xhat    level        trend    season
## [1,] 87.17619 82.87739 -0.004362918  4.303159
## [2,] 90.32925 82.09550 -0.004362918  8.238119
## [3,] 92.96089 81.87348 -0.004362918 11.091777
## [4,] 90.93360 81.89497 -0.004362918  9.042997
## [5,] 83.99752 81.93450 -0.004362918  2.067387
## [6,] 84.04358 81.93177 -0.004362918  2.116168
```

There are two different approaches to the Holtwinters function additive and multiplicative which compute the four components differently. The additive sums up the four compenents and the multiplicative uses the product of the four. We can see that additive has a smaller sum of the squared errors so we will use that for our model. If we look at the fitted model for temp_holt we see that there isnt much of a trend. The same is true for the multiplicative model so from the surface it is harder to tell if summers are getting hotter. But we can now use our computed fitted model values and use cusum to try and detect an increase in temperature.

```r
#create matrix to store season values
season <- matrix(temp_holt_ml$fitted[,4],nrow=123)

#write.csv(season, file="season.csv", row.names = F)

colnames(season) <- colnames(temp_data[,3:21])
rownames(season) <- temp_data[,1]
```

I created a matrix to hold the season values since we are interested in running those values in our cusum function. I wrote the values and explored them in Excel which lead to similiar findings. Then I add the row names and colnames to the matrix so it is easier to navigate the matrix.

```
#avg of all the years
avg_allyrs <- mean(season)
avg_allyrs
```

```
## [1] 0.9954727
```

```
#a look at an average from dates we
#know that are fall time
which(temp_data$DAY=="1-Oct")
```

```
## [1] 93
```

```
mean(season[93:123,])
```

```
## [1] 0.8751098
```

```
#Avg sf for the 1st year
##use this as the baseline to mark end of summer
avg_year1 <- mean(season[,1])
avg_year1
```

```
## [1] 1
```

I first take at the average for all the years which we see is almost one. Since we are interested in if the end of summer has gotten later then lets look at a fall day's average. October first is a fall day and the average on that day across the years is 0.87 which is about 0.12 less than the average. We need to determine a baseline of when summer ends for the cusum function so lets take the seasonal factor of the first year, which is 1.

Cusum

```
cusum_fn = function(data, avg, T, C){
  #an empty list to hold results
  results = list()
  cusum = 0 #intial 0
  Counter = 1 #a counter
  while (Counter <= nrow(data)){
    current = data[Counter,]
    #cusum equation
    cusum = max(0, cusum + (avg - current - C))
    if (cusum >= T) {
      results = Counter
      break
    }
    Counter = Counter + 1
```

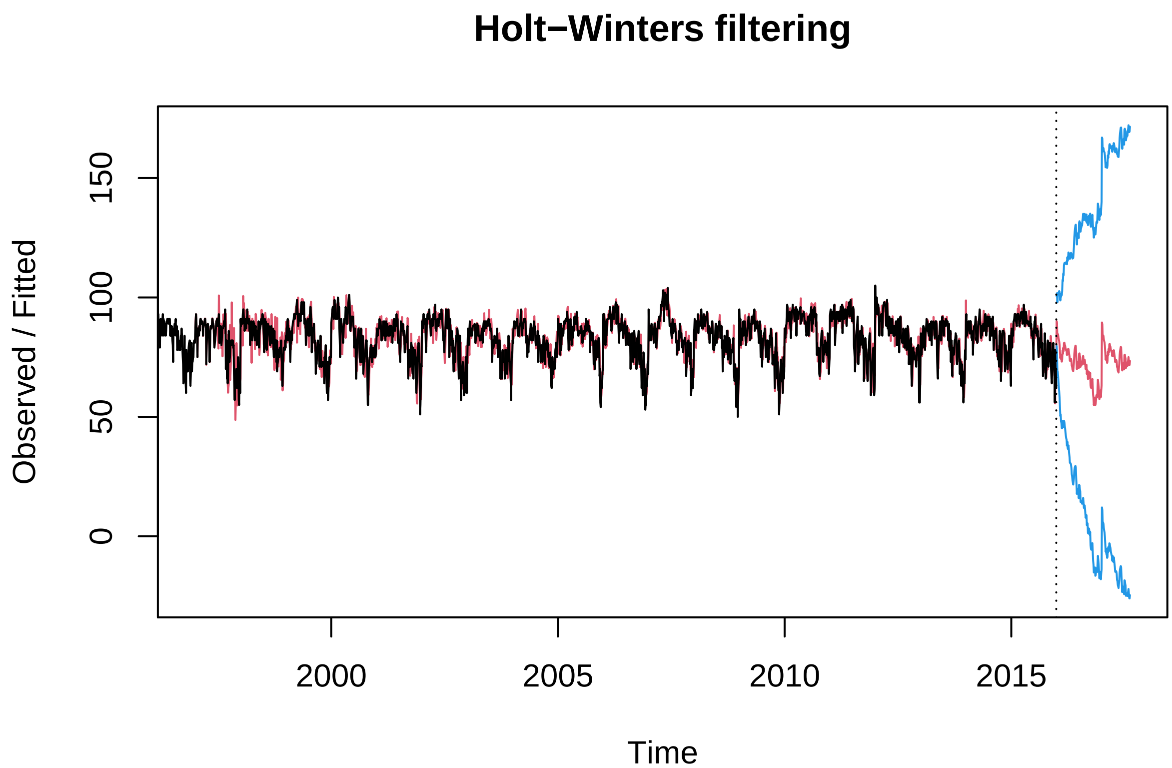
```r
    if (Counter >= nrow(data)){
      results = NA
      break
    }
  }
  return(results)
}

# C is half the std the 1st yr
# Threshold is 3 time the std
C_val = sd(season[,1])*0.5
Thres = sd(season[,1])*2

# Run for each year
#see if SF was higher than the threshold
# avg of first year
result_vector = vector()
for (x in 1:ncol(season)){
  result_vector[x] = cusum_fn(data = as.matrix(season[,x]),
                                    avg =  avg_year1,
                                    T = Thres,
                                    C = C_val)
}

#store the results in a dataframe
results = data.frame(Year = colnames(season),
                     Day = temp_data[result_vector,1])
results
```

```
##     Year    Day
## 1  X1997 30-Sep
## 2  X1998 30-Sep
## 3  X1999 30-Sep
## 4  X2000  1-Oct
## 5  X2001  1-Oct
## 6  X2002  1-Oct
## 7  X2003  2-Oct
## 8  X2004  2-Oct
## 9  X2005  2-Oct
## 10 X2006  3-Oct
## 11 X2007  3-Oct
## 12 X2008  3-Oct
## 13 X2009  3-Oct
## 14 X2010  3-Oct
## 15 X2011  2-Oct
## 16 X2012  1-Oct
## 17 X2013  1-Oct
## 18 X2014  2-Oct
## 19 X2015  3-Oct
```

Finding a good C value and T value was difficult and I did a lot of trial and error. I ran it with threshold value multipliers of 3, 4, and 5 but they all produced similiar results. Where the end of summer was slightly getting later even if you marked the end of summer later. I also changed values of C from about 0.2 to 1 but again it produced the same results but it errors detecting dates. So I decided that the multipliers for C would be 0.5 and T would be 2. The results for loop apply the C and T values to the cusum function and then is printed out in a data frame. As you can see from the results the day is slowly getting later into October. This is indicating that the average temperature is rising meaning that global warming is happening.

Predict

```
# Predicts
predicts <- predict(temp_holt, 200, prediction.interval = TRUE)
plot(temp_holt, predicts)
```



I tried to predict out just to see what the future might look like but the confidence interval is very large. I did run it through the cusum model but I did not have it set up correctly because it was only predicting one year. Though from the model it looks like temperature could trend down, but the interval is so large that it's not conclusive.