Question 7.1

Describe a situation or problem from your job, everyday life, current events, etc., for which exponential smoothing would be appropriate. What data would you need? Would you expect the value of α (the first smoothing parameter) to be closer to 0 or 1, and why?

A year or so ago, I got a one of the smart watches (Garmin Vivo active 4) that has tons of activity trackers such as pedometer, heart rate, sleep tracking, etc. Most of this gets synced with my google fit app running on my Android phone so I have access to this data. One metric I most often look at is the number of steps/miles walked in a day/week/month, etc.

I think, it would be an interesting exercise to see if there are any trends.

- How much of my predicted #steps for the day is based on historical data?
    - My initial thinking is to use a simple model such as moving averages to ignore the trend or seasonality and view the predicted vs actual levels.
    - **I would probably select an alpha value closer to 1 since I think past values don't have much influence over the forecast and there is much randomness**.
- Am I walking more or less since I got the tracker?
    - A good model to use for this would be Winter's model.
- Is my walking seasonal? Its logical to think one would walk more on the weekend than on weekdays. Is it possible to prove that? Within the weekdays, are there any "low" points and cyclical patterns?
    - We could use Holts-Winters model to fit and look for any trend and seasonality.
    - We can look at the data on different seasonality windows. eg: weekly, monthly, seasonally (eg: summer vs fall vs winter).
- Am I likely to get better or worse in future ?
    - We could use any of the models with forecast to look at the data and near term future trends. If using moving averages, we can make use of trailing window instead of centered window for forecasting.

Garmin does expose this data via Garmin Connect portal. The portal does have some decent charts and stats, it would still be an interesting and useful exercise to get more personal insights into the personal data!

Question 7.2

Using the 20 years of daily high temperature data for Atlanta (July through October) from Question 6.2 (file `temps.txt`), 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.

Basic Exploration

First, lets take a look at the temperature data and run some basic R functions and plots to examine the data and see if we can spot any trends in it.

Summary

```
datafile <- 'data/temps.txt'
temps <- read.table(datafile,
            stringsAsFactors = FALSE,
            header = TRUE)
summary(temps)
```

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

```
 3rd Qu.:88.00   3rd Qu.:93.00   3rd Qu.:94.00   3rd Qu.: 90.50   3rd Qu.:88.00   3rd Qu.:89.00   3rd
Qu.:90.0
 Max.   :95.00   Max.   :97.00   Max.   :99.00   Max.   :105.00   Max.   :92.00   Max.   :95.00
Max.   :97.0
```

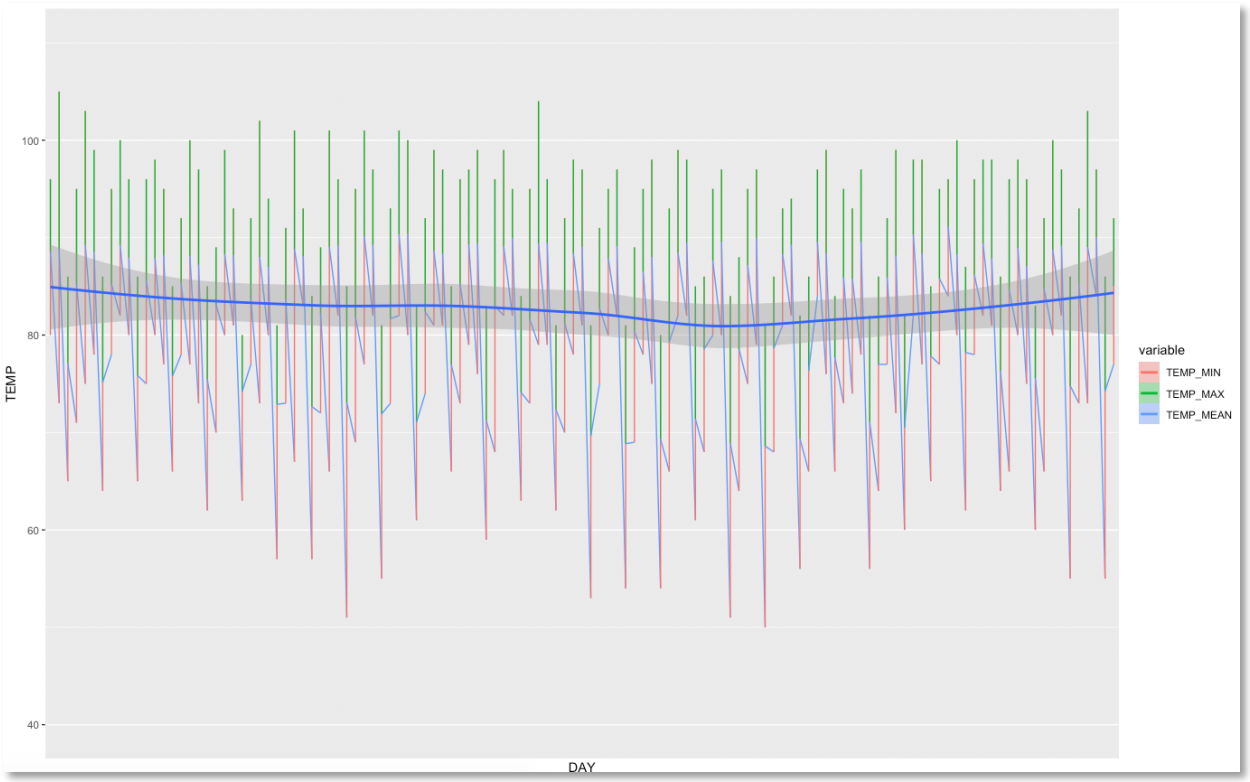
Plot

Before we can plot all the data, its useful to “melt” the data to narrow format.

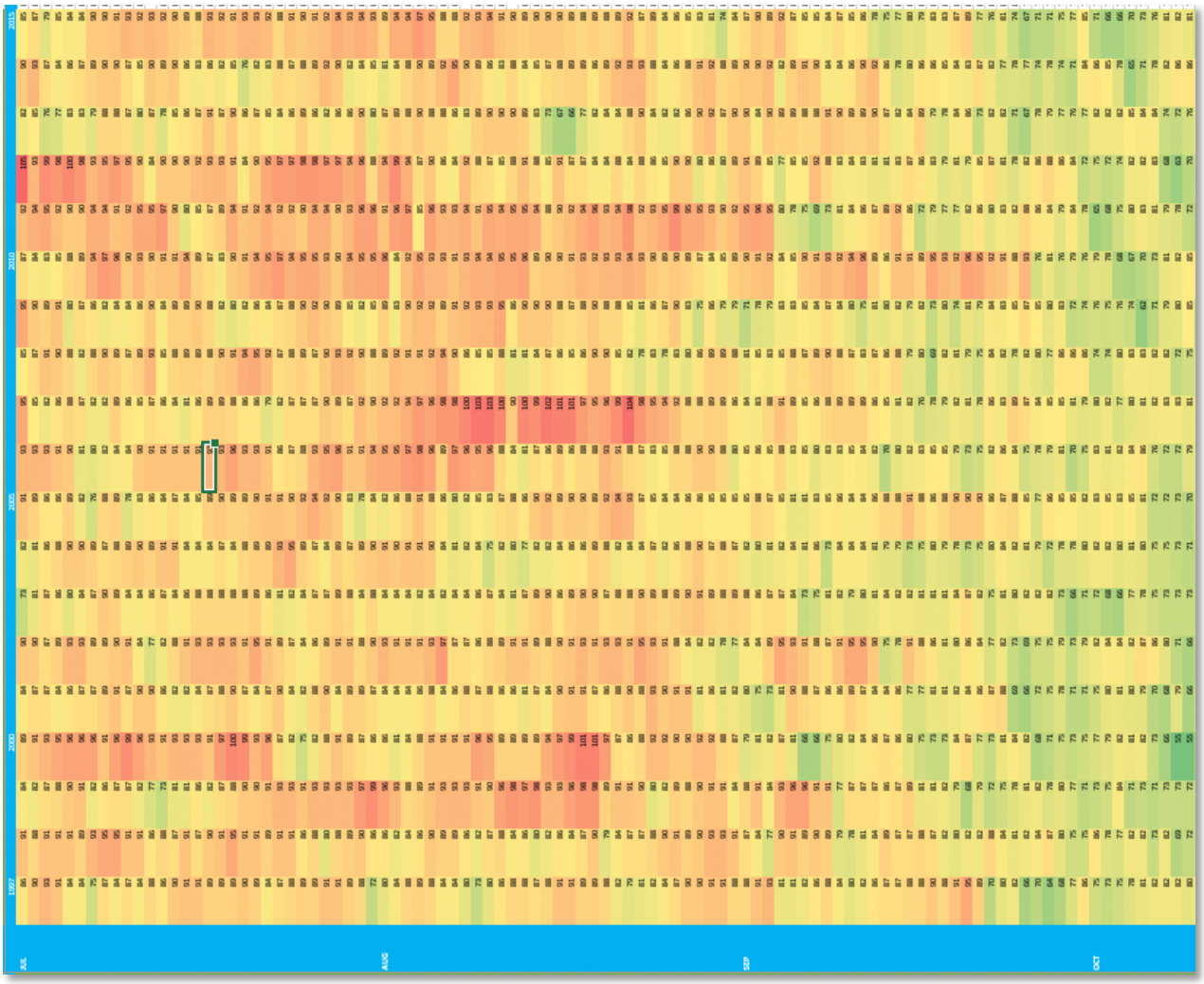
```
temps_m <- melt(temps, id.vars = "DAY", variable.name = 'YEAR', value.name = 'TEMP')
dates <- temps$DAY
temps_m$DAY <- factor(temps_m$DAY, labels = dates, ordered = T)

# avg temp over the seasions
temp_mean2 <- aggregate(TEMP~DAY, data=temps_m, mean); colnames(temp_mean2)[2] <- 'TEMP_MEAN'
temp_max2 <- aggregate(TEMP~DAY, data=temps_m, max) ; colnames(temp_max2)[2] <- 'TEMP_MAX'
temp_min2 <- aggregate(TEMP~DAY, data=temps_m, min) ; colnames(temp_min2)[2] <- 'TEMP_MIN'
t <- merge(temp_min2, temp_max2, by='DAY')
temp_stats2 <- merge(t, temp_mean2, by='DAY')
temp_stats_m2 <- melt(temp_stats2, id.vars = "DAY", value.name = 'TEMP')

# Plot data as well as smoothed avg values
ggplot(temp_stats_m2,
  aes(x=DAY, y=TEMP, fill=variable, color=variable, group = 1)) +
  geom_line() +
  ylim(40,110)+
  geom_smooth() +
  scale_x_discrete(labels = abbreviate, breaks=10) +
theme(axis.text.x=element_text(angle=90, hjust=1))
```



Here is a heatmap of temperature distribution over the years, from simple Excel conditional formatting (with apologies for poor capture resolution).



Looking at the summary and plot of all data, it is hard to see any patterns visually and hard to make any predictions. **The heatmap does seem to indicate hotter summer in 2007, but necessarily longer summers, except for 2010**. Though its common knowledge that temperatures are higher in summer months than winter, **the heatmap above shows clearly that the data has seasonality**.

Exponential smoothing

Background

There are several exponential smoothing methods available to try for both visualization and forecasting data. Without going into discussing all technical details (which are covered in the course videos and elsewhere in the literature), briefly, here are few that are of interest.

1. Simple exponential smoothing models such as moving average. These are generally good for analyzing data with no trend or seasonality
2. Holts exponential smoothing: good for analyzing data with trend, but no seasonality
3. Holts-Winters model: good for analyzing data with trend as well as seasonality.

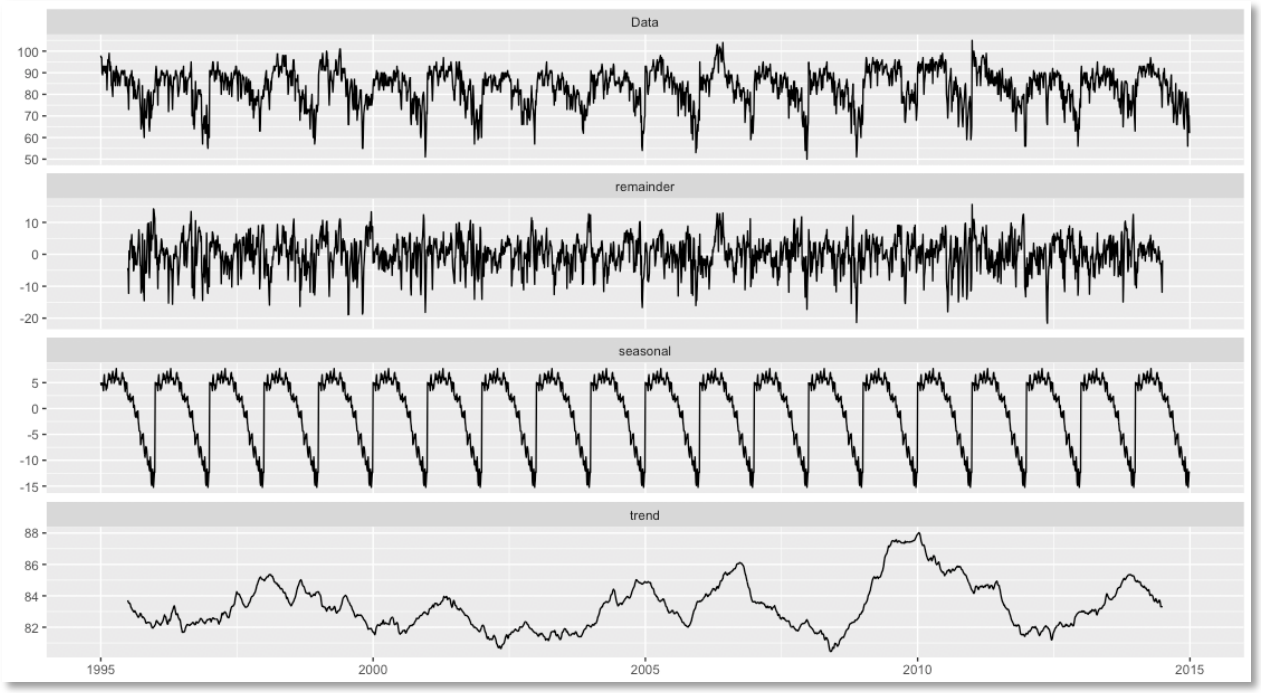
Since we are looking to analyze if there is any trend in the data and the data does have seasonality (as demonstrated in the heatmap above), we will pick Holts-Winter's exponential smoothing.

Decompose Plot

Before we run any models, lets first try to look at the temperature data more closely and try to extract the trend and seasonality from it. Its generally a good idea to run this step even when we know that trends and seasonality exists in the data to ensure that our assumptions are right. As discussed above, different smoothing models expect different data properties (eg: if we want to use simple exponential model, we will have to first de-trend and de-seasonalize the data as the simple exponential models such as moving average doesn't handle them).

For our purposes, we will first "melt" the data to narrow form, take only the temperature data and run it thru ts() and decompose() with appropriate arguments. This is fed into ggplot's autoplot()

```
#
# plot decomposed data with ggplot
# as before, melt data to narrow form first
#
temps_m <- melt(temps, id.vars = "DAY", variable.name = 'YEAR', value.name = 'TEMP')
dates <- temps$DAY
temps_m$DAY <- factor(temps_m$DAY, labels = dates, ordered = T)
# 123 is the number of data points per year. same as #rows in temps
temps_m$TEMP %>%
  ts (start = 1995, frequency = 123) %>%
  decompose() %>%
  autoplot ()
```



We can infer the following from the above plot:

- The data has quite a bit of variation (data plot)
- **Data does have some randomness** in it (remainder plot)
- As expected, the data is very seasonal (seasonal plot). Further **the seasonal variation is additive** and not multiplicative.
- There is lot of variation in the trend, but **we can't conclude there is clear upward or downward trend purely based on the trend plot**.

Holt-Winters

Holts-Winter model is an advanced smoothing model that considers the levels, trend and seasonality. Since data we are dealing with seem to have them all, we can use Holt-Winters model to visualize / predict the data.

Starting from our melted (narrow) data, we can create time-series object as before with ts(). I made slight changes to HWplot borrowed from elsewhere to create ggplot plot.

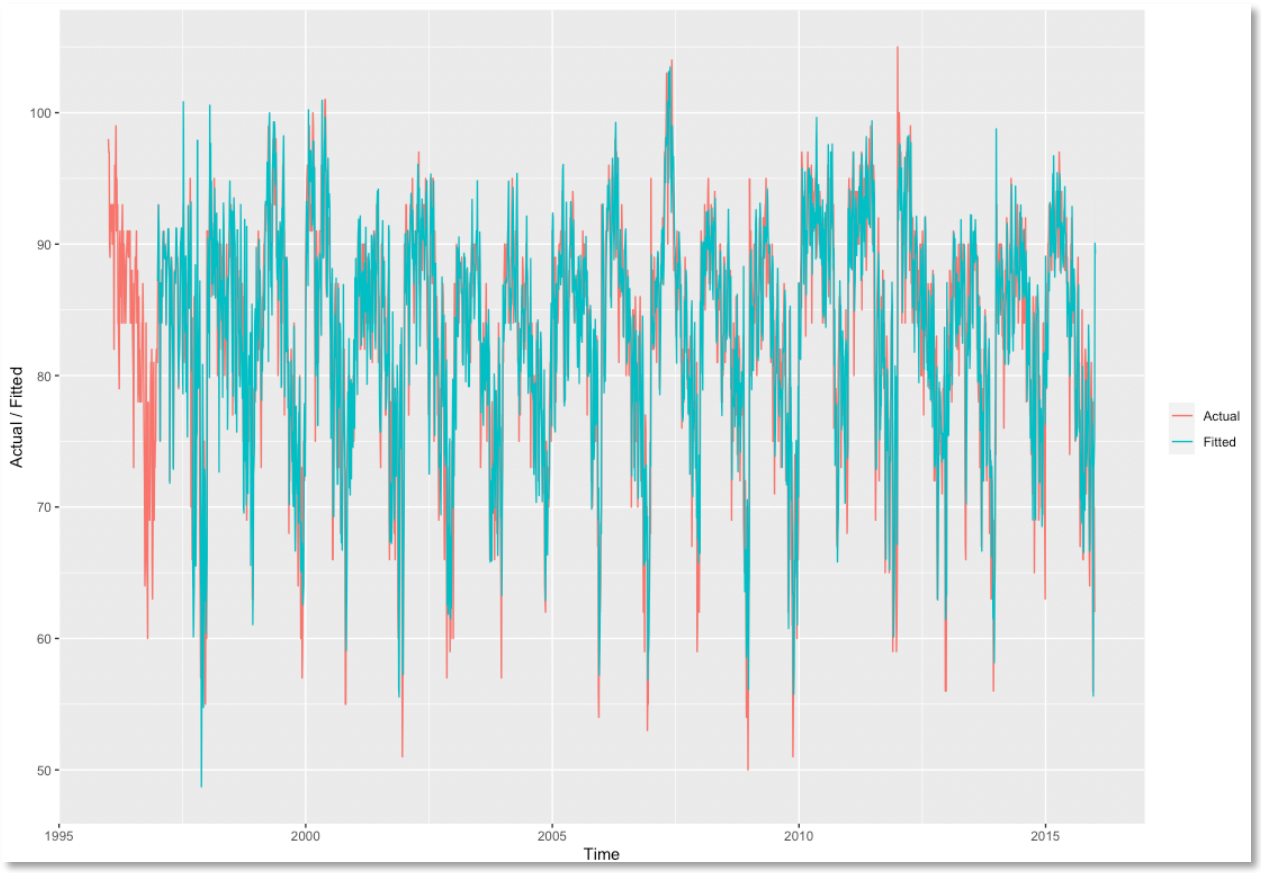
Visualization

Lets fit the data with Holt-Winters with additive and multiplicative seasonality.

Additive:

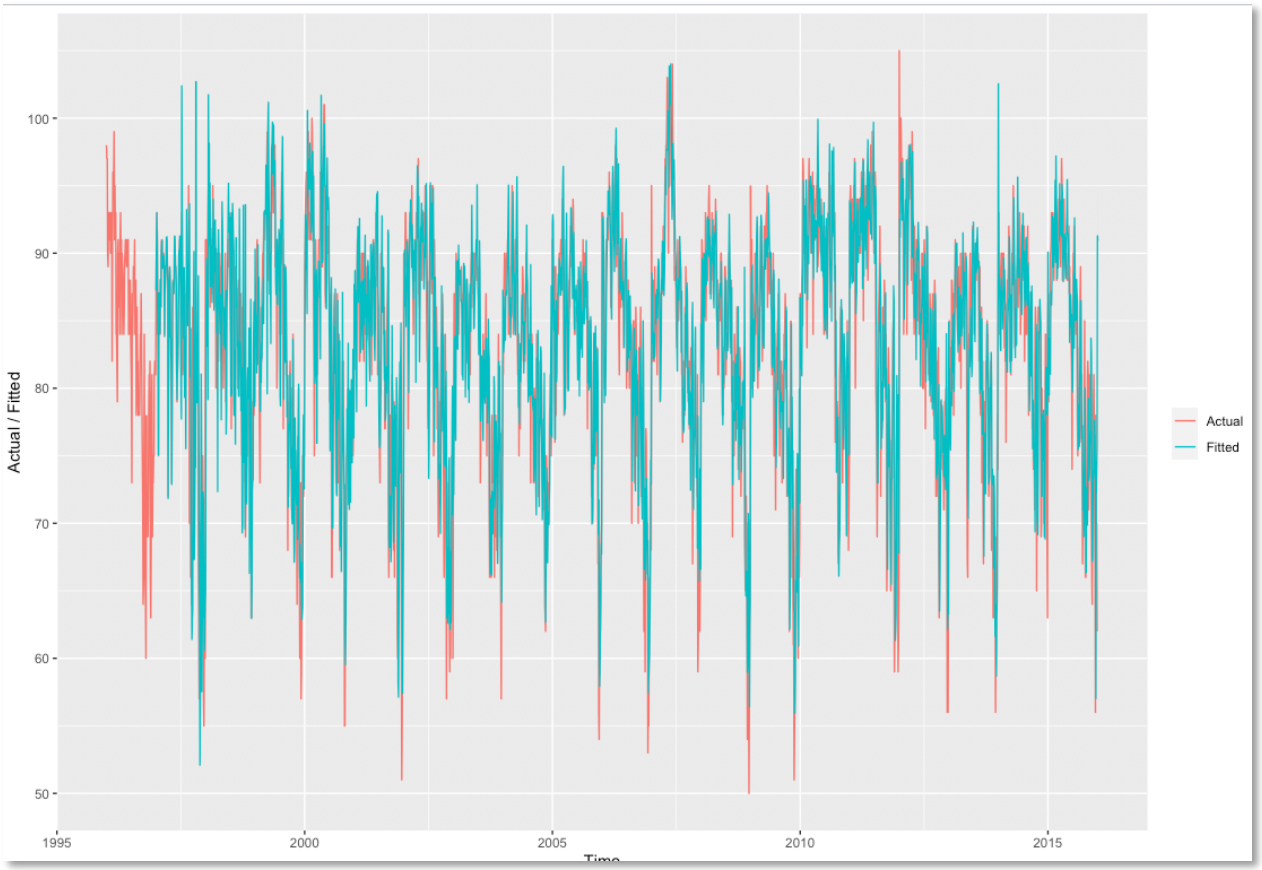
```
ts_v <- ts(temps_m$TEMP, start = 1996, frequency = 123)
```

```
hw_a <- HoltWinters(ts_v, seasonal = 'add')
hw_a
# Visualization with ggplot2
HWplot2 (hw_a)
```



Multiplicative:

```
hw_m <- HoltWinters(ts_v, seasonal = 'mult')
hw_m
# Visualization with ggplot2
HWplot2 (hw_m)
```



Analysis

Comparing the coefficients for additive and multicative, additive seem to be better fit since the additive resulted in smaller sum of squared errors (SSE)

```
> cat ('Additive      : alpha (base):', hw_a$alpha, 'beta (trend):', hw_a$beta,
'seasonal:', hw_a$gamma, 'SSE', hw_a$SSE)
Additive      : alpha (base): 0.6610618 beta (trend): 0 seasonal: 0.6248076 SSE
66244.25
> cat ('Multiplicative: alpha (base):', hw_m$alpha, 'beta (trend):', hw_m$beta,
'seasonal:', hw_m$gamma, 'SSE', hw_m$SSE)
Multiplicative: alpha (base): 0.615003 beta (trend): 0 seasonal: 0.5495256 SSE
68904.57
```

Observations

- The plot shows the fitted values following the actual value very closely, except in the peaks and lows.
- As expected, there are no fitted values for first few years as the model is trying to learn from the data and there isn't enough data to learn from in the beginning.

- We can also observe the fitted plot is under-fitted in the beginning and doesn't follow the actual data very closely, indication there is not sufficient data to model this correctly.

Conclusion

- There is no clear trend seen based on visual examination of various plots.
- The duration of warm days does fluctuate, but there is no sufficient evidence to conclude that they are progressively getting longer. So I conclude that **unofficial end of summer has not gotten later over the 20 years**

Prediction

Though, not asked in the assignment, its possible to extend the plot to include prediction. Here is a sample plot with prediction.

```
# Prediction
HWplot2 (hw, n.ahead = 250 )
```

Based on this plot, in the short term (next 10 years after tail end of data) the temperature distribution across season (Jul – Oct) will remain more or less the same 2015

References

- https://a-little-book-of-r-for-time-series.readthedocs.io/en/latest/
- https://semba-blog.netlify.app/02/22/2019/exploring-time-series-data-in-r/
- https://www.youtube.com/watch?v=mrLiC1biciY
- https://pkg.robjhyndman.com/forecast/reference/autoplot.seas.html
- https://www.r-bloggers.com/2012/07/holt-winters-forecast-using-ggplot2/
- https://rstudio-pubs-static.s3.amazonaws.com/366011_3ee069277eb84547824f8f4022973823.html

Appendix

```
# HWplot2 function
# Based on source from
# https://www.r-bloggers.com/2012/07/holt-winters-forecast-using-ggplot2/
HWplot2 <-function(hw_object, n.ahead=4, CI=.95, error.ribbon='green',
line.size=1) {
  forecast<-predict(hw_object, n.ahead=n.ahead, prediction.interval=T,
level=CI)
  for_values<-data.frame(time=round(time(forecast), 3),
                         value_forecast=as.data.frame(forecast)$fit,
                         dev=as.data.frame(forecast)$upr-
as.data.frame(forecast)$fit)
  fitted_values<-data.frame(time=round(time(hw_object$fitted), 3),
                            value_fitted=as.data.frame(hw_object$fitted)$xhat)
  actual_values<-data.frame(time=round(time(hw_object$x), 3),
                            Actual=c(hw_object$x))
  graphset<-merge(actual_values, fitted_values, by='time', all=TRUE)
  graphset<-merge(graphset, for_values, all=TRUE, by='time')
  graphset[is.na(graphset$dev), ]$dev<-0
  graphset$Fitted<-c(rep(NA, NROW(graphset)-(NROW(for_values) +
NROW(fitted_values))),
                     fitted_values$value_fitted,
                     for_values$value_forecast)
  graphset.melt<-melt(graphset[, c('time', 'Actual', 'Fitted')], id='time')
  p<-ggplot(graphset.melt, aes(x=time, y=value)) +
    geom_ribbon(data=graphset, aes(x=time, y=Fitted, ymin=Fitted-dev,
ymax=Fitted + dev), alpha=.1, fill=error.ribbon) +
    geom_line(aes(colour=variable), size=line.size) +
    xlab('Time') +
    ylab('Actual/Fitted') +
    scale_colour_hue('')
  return(p)
}
```