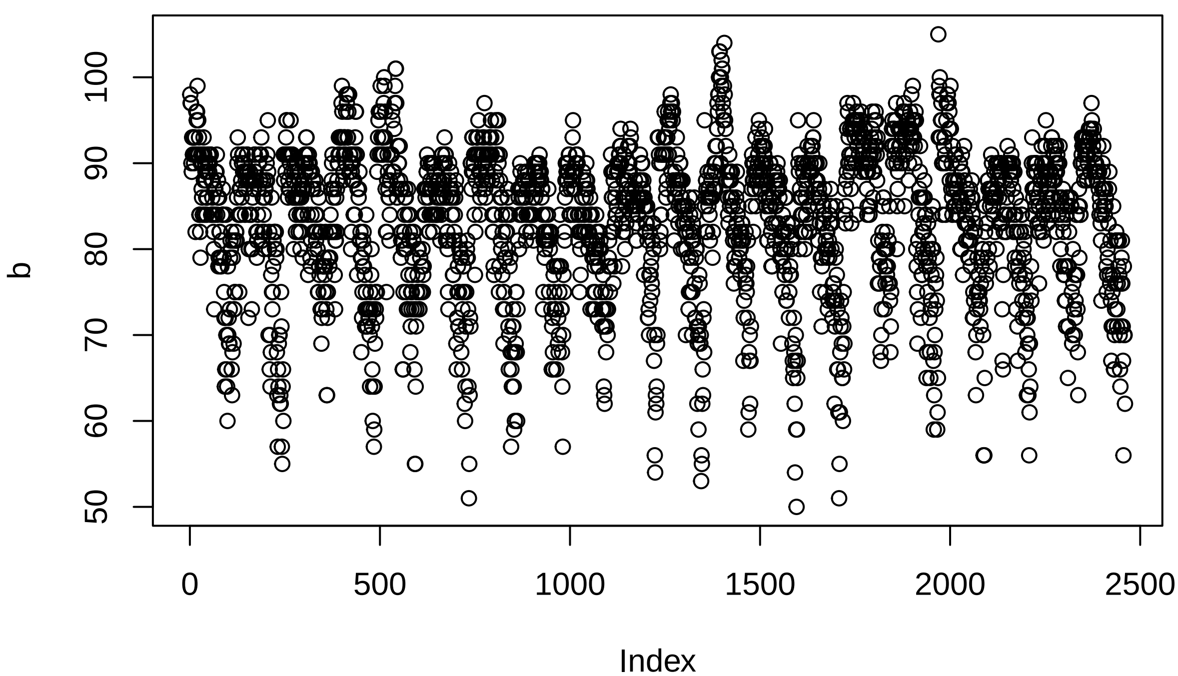
Exponential Smoothing Model

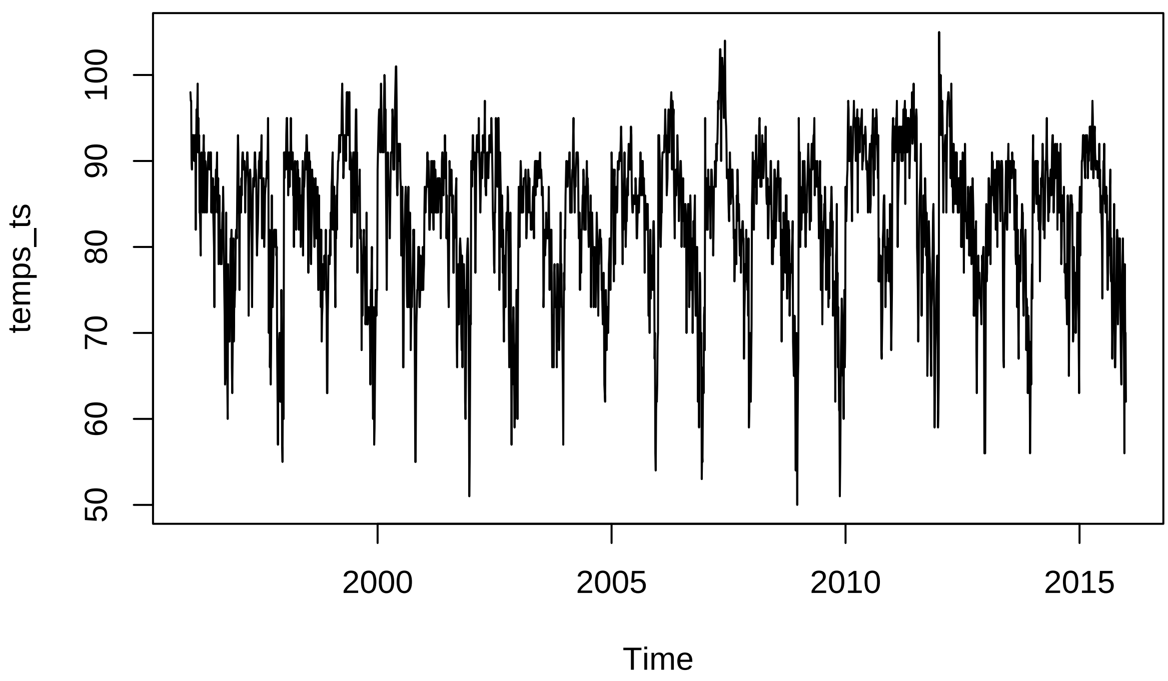
2024-02-07

HoltWinters approach on Atlanta Summer Temperature Data

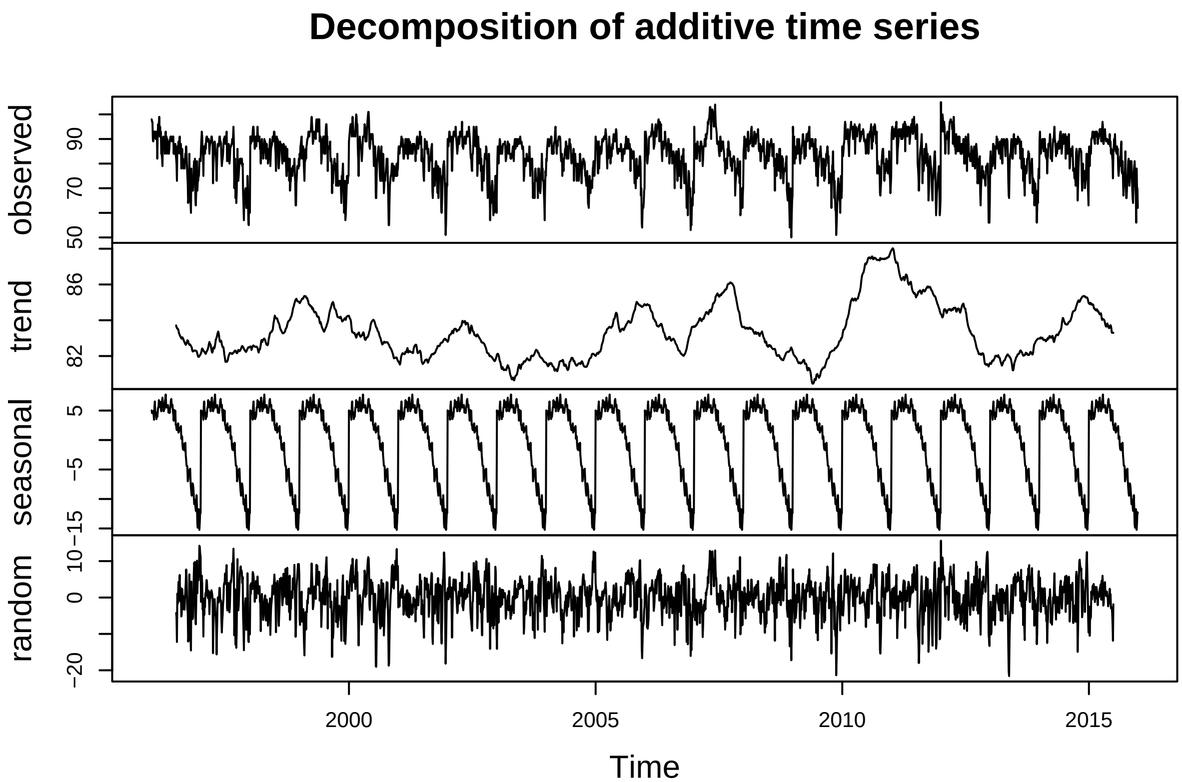
```
# Clear environment, variables.
rm(list =ls())

# Import data 'temps.txt' into table with headers.
temps <- read.table("temps.txt", stringsAsFactors = FALSE, header = TRUE)
# temps

# Make vector with all the combined years temps.
b <- as.vector(unlist(temps[,2:21]))
# b

# Plot the raw temp data from the vector 'b'.
plot(b)
```



```r
# Convert 'b' into time series data with frequency of each year's summer days data.
temps_ts <- ts(b, start = 1996, frequency = 123)
# temps_ts

# Plot the time series data.
plot(temps_ts)
```




```r
# Plot the time series data decomposition.
plot(decompose(temps_ts))
```




```r
# Use HoltWinters exponential smoothing with multiplicative seasonal.
temps_HW <- HoltWinters(temps_ts, alpha = NULL, beta = NULL, gamma = NULL, seasonal = "multiplicative")

# Show the SSE or Sum of Squared Errors.
temps_HW
```

```
## Holt-Winters exponential smoothing with trend and multiplicative seasonal component.
##
## Call:
## HoltWinters(x = temps_ts, alpha = NULL, beta = NULL, gamma = NULL,     seasonal = "multiplicative")
##
## Smoothing parameters:
##  alpha: 0.615003
##  beta : 0
##  gamma: 0.5495256
##
## Coefficients:
##              [,1]
## a    73.679517064
## b    -0.004362918
## s1    1.239022317
## s2    1.234344062
## s3    1.159509551
## s4    1.175247483
## s5    1.171344196
## s6    1.151038408
```

```
## s7    1.139383104
## s8    1.130484528
## s9    1.110487514
## s10   1.076242879
## s11   1.041044609
## s12   1.058139281
## s13   1.032496529
## s14   1.036257448
## s15   1.019348815
## s16   1.026754142
## s17   1.071170378
## s18   1.054819556
## s19   1.084397734
## s20   1.064605879
## s21   1.109827336
## s22   1.112670130
## s23   1.103970506
## s24   1.102771209
## s25   1.091264692
## s26   1.084518342
## s27   1.077914660
## s28   1.077696145
## s29   1.053788854
## s30   1.079454300
## s31   1.053481186
## s32   1.054023885
## s33   1.078221405
## s34   1.070145761
## s35   1.054891375
## s36   1.044587771
## s37   1.023285461
## s38   1.025836722
## s39   1.031075732
## s40   1.031419152
## s41   1.021827552
## s42   0.998177248
## s43   0.996049257
## s44   0.981570825
## s45   0.976510542
## s46   0.967977608
## s47   0.985788411
## s48   1.004748195
## s49   1.050965934
## s50   1.072515008
## s51   1.086532279
## s52   1.098357400
## s53   1.097158461
## s54   1.054827180
## s55   1.022866587
## s56   0.987259326
## s57   1.016923524
## s58   1.016604903
## s59   1.004320951
## s60   1.019102781
```

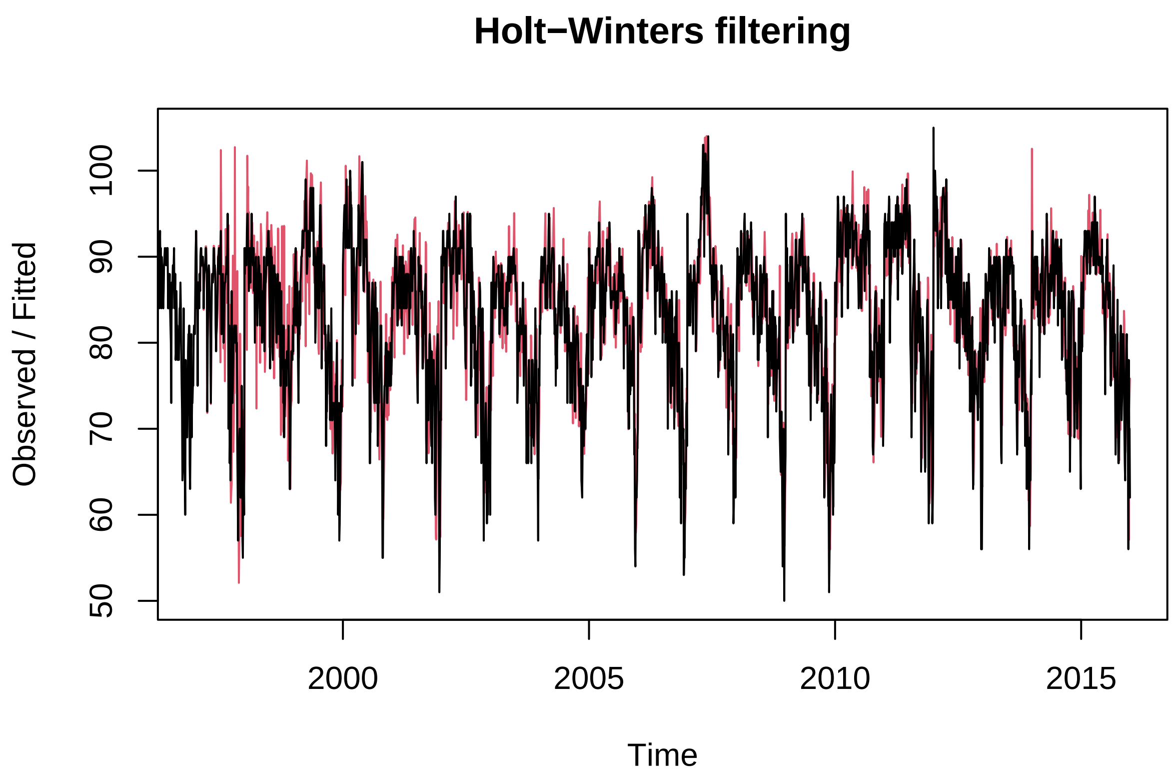
```
## s61   0.983848662
## s62   1.055888360
## s63   1.056122844
## s64   1.043478958
## s65   1.039475693
## s66   0.991019224
## s67   1.001437488
## s68   1.002221759
## s69   1.003949213
## s70   0.999566344
## s71   1.018636837
## s72   1.026490773
## s73   1.042507768
## s74   1.022500795
## s75   1.002503740
## s76   1.004560984
## s77   1.025536556
## s78   1.015357769
## s79   0.992176558
## s80   0.979377825
## s81   0.998058079
## s82   1.002553395
## s83   0.955429116
## s84   0.970970220
## s85   0.975543504
## s86   0.931515830
## s87   0.926764603
## s88   0.958565273
## s89   0.963250387
## s90   0.951644060
## s91   0.937362688
## s92   0.954257999
## s93   0.892485444
## s94   0.879537700
## s95   0.879946892
## s96   0.890633648
## s97   0.917134959
## s98   0.925991769
## s99   0.884247686
## s100  0.846648167
## s101  0.833696369
## s102  0.800001437
## s103  0.807934782
## s104  0.819343668
## s105  0.828571029
## s106  0.795608740
## s107  0.796609993
## s108  0.815503509
## s109  0.830111282
## s110  0.829086181
## s111  0.818367239
## s112  0.863958784
## s113  0.912057203
## s114  0.898308248
```

```
## s115  0.878723779
## s116  0.848971946
## s117  0.813891909
## s118  0.846821392
## s119  0.819121827
## s120  0.851036184
## s121  0.820416491
## s122  0.851581233
## s123  0.874038407
```

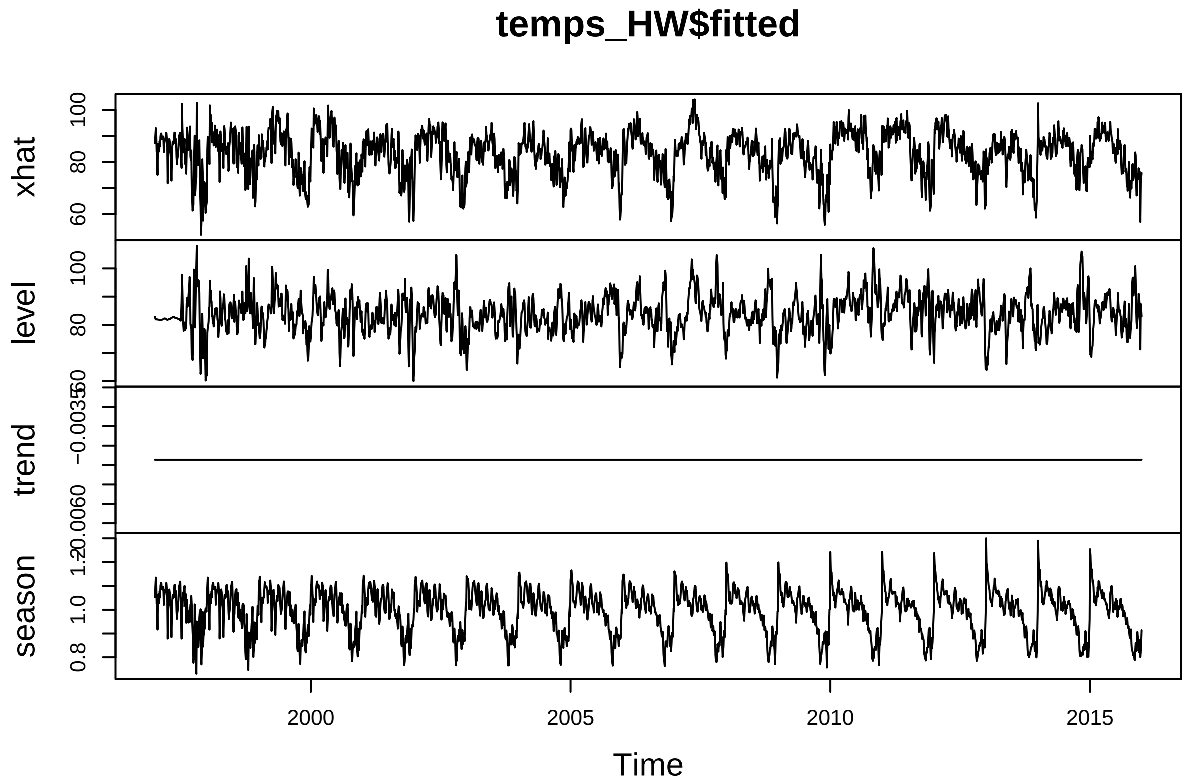
```r
temps_HW$SSE
```

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

```r
# Show the fitted values (too many values, so commented out.)
# temps_HW$fitted

# Plot the model with the original data and the fitted values.
plot(temps_HW)
```




```r
plot(temps_HW$fitted)
```




```r
# Display alpha, gamma, and SSE values.
temps_HW$alpha
```

```
##    alpha
## 0.615003
```

```r
temps_HW$gamma
```

```
##     gamma
## 0.5495256
```

In this analysis, I built and used an exponential smoothing model to examine whether the unofficial end of summer has shifted later over 20 years in Atlanta, Georgia. Analyzing temperature data from July to October between 1996 and 2015, I started with basic plots to visualize the data, then progressed to constructing an exponential smoothing model to decompose the time series data.

Using the HoltWinters' package in R, I transformed the temps.txt file's data into time-series format, adjusting the frequency to account for the data span from July to October. Initial plotting provided a basic understanding, but the decomposed time series offered deeper insights into trend, seasonal, and random components. However, the trend component did not conclusively indicate a significant shift in summer's end over the observed period.

Further analysis using Holt-Winters' model provided smoothed data visualization but still lacked clear evidence of a shift in summer's end. Despite observing a potential increase in early July temperatures in recent years, this could not be definitively attributed to a later summer end.

The alpha and gamma values from the model emphasized recent observations and seasonal trends, respectively. Yet, even with this analytical approach, the question remains inconclusive due to the limited data span and complexity of global warming effects.

This exploration into Atlanta's summer end timings through time series analysis and exponential smoothing highlights the challenges in detecting climate trends within a relatively short 20-year window.