Question 7.1 – Describe a situation for which exponential smoothing would be appropriate. What data would you need and why? Would alpha be closer to 0 or 1 and why?

I, much like many people, like to weigh myself on a daily basis. However, weight doesn't stay the same every day, and it's easy to be alarmed by a sudden spike – or be elated by a sudden dip. This would be a great use for exponential smoothing. I would only need to track my weight daily, and I would use an alpha closer to 0, because the baseline is a much closer indicator of actual weight than one day's fluctuation. I could even use trending to track overall weight gain or loss, and seasonality for the Halloween through Christmas inevitable gain! The more years of weight data that I have, the better I can get to my actual weight and predict tomorrow's weight.

Question 7.2 – Using 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.

I first started this exercise by making my data into a timeseries, and then I plotted it out to look at it. There was clear seasonality, but the trending wasn't so obvious. I ran Holt-Winters and got beta = 0. If I understood the lesson correctly, beta is the trend coefficient, and so if it is 0, then there is no trend in the data. So, no overall increasing or decreasing temperature trends. But what about the last day of summer – is it happening faster? I decomposed the model into seasonality and trend using the decompose() function, and while seasonality was obvious, trend was all over the place. If the end of summer had increased, I'd expect to see the bottom of the trend line increasing over time, and that wasn't clear. So to confirm this, I took the seasonality coefficients and put them in a matrix and ran CUSUM on this to see if there was a significant change between the seasonality coefficients that suggests a changing trend. I did this first in R, and my plot didn't show any major changes. Just to doublecheck, I then ran the CUSUM in Excel as well, and I didn't detect any change in the end of summer date – it was consistently in mid-August. Based on how I ran the models, I did not detect any change in the end of summer date.

```
library(timeSeries)

library(ggplot2)

install.packages("forecast")


#let's read the file in and call it "temp"

temp <- read.table("c:/users/kkisner/Desktop/gradclass/temps.txt", header = TRUE)

temp <- as.vector(unlist(temp[,2:21]))
```

#let's make the data a time series, and then check out what it gives us to be sure we're working properly.

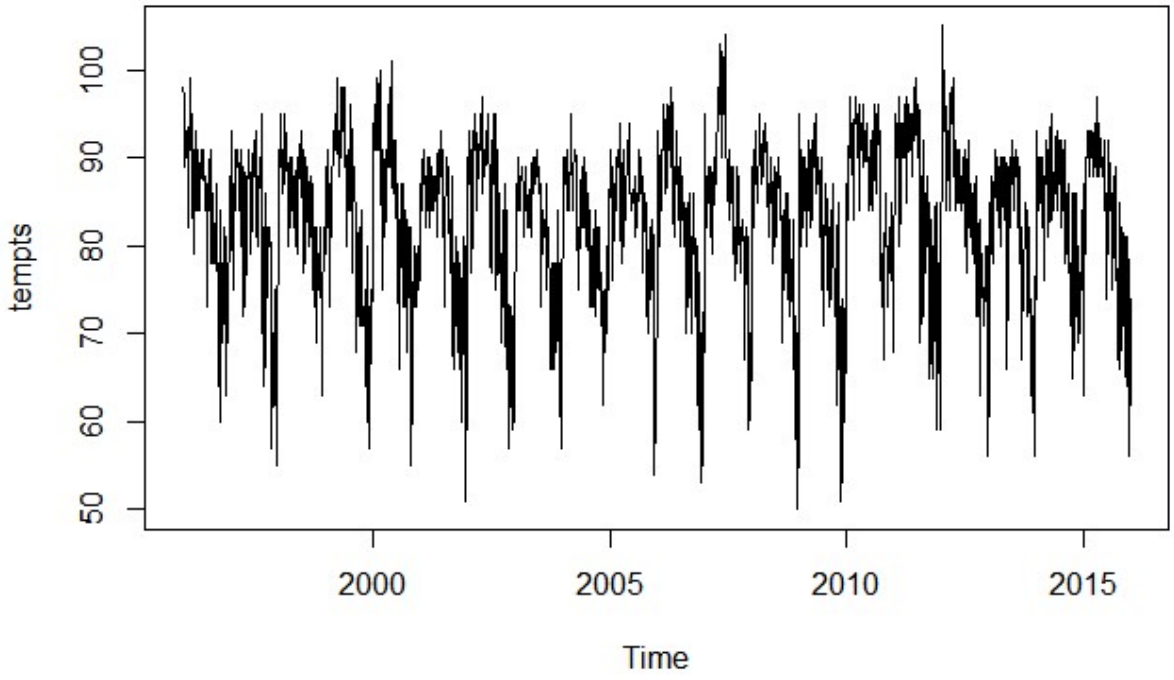
tempts <- ts(temp, start = 1996, frequency = 123)

tempts

#great! It starts in 1996 and ends is 2015... just what we wanted.

#let's see what the data looks like

plot(tempts)



#well, nothing is too obvious about the trends, but there's clearly seasonality.

#Let's do the exponential smoothing with Holt Winters. I added that seasonality is multiplicative.

#I set beta = FALSE becuase the help said that would make it do exponential smoothing.

#In practice, I got the same answer regardless of not mentioning beta, beta = NULL, and beta = FALSE.

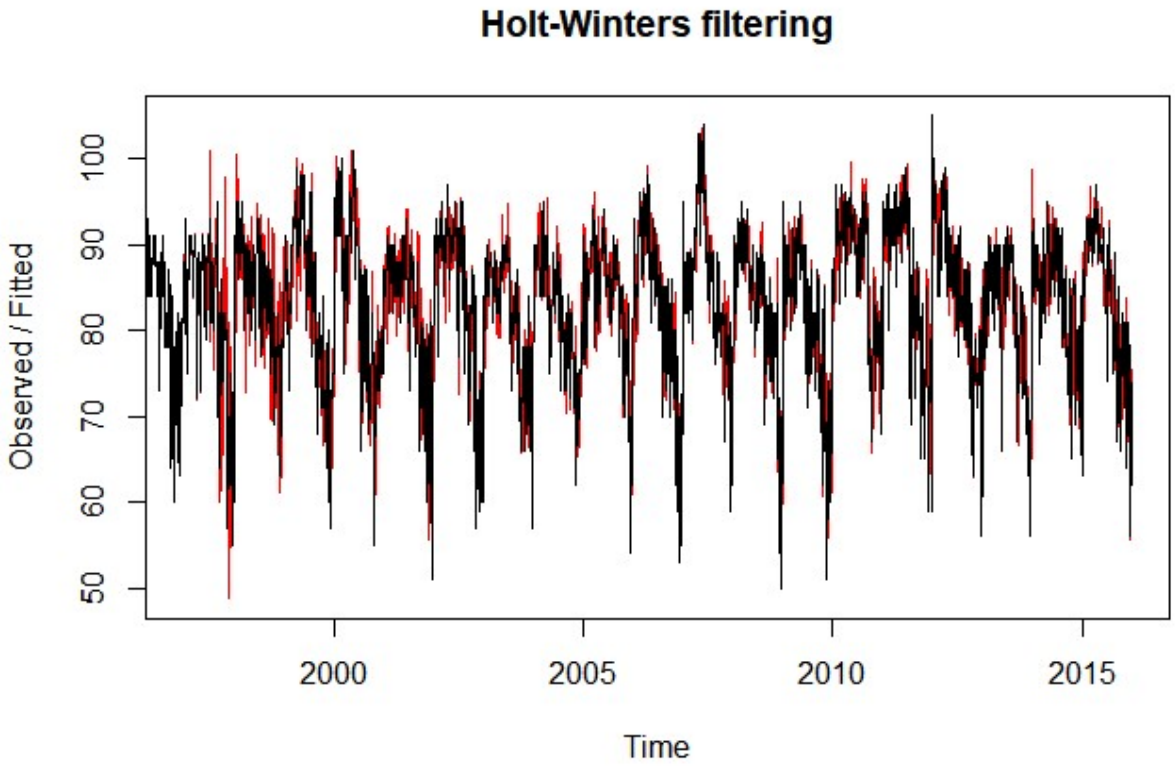
HW <- HoltWinters(tempts, beta = FALSE, seasonal = "mult")

HW

#This gives me alpha .6150, beta = 0, gamma = 0.5495

#Let's take a look at what the filtering looks like.

plot(HW)

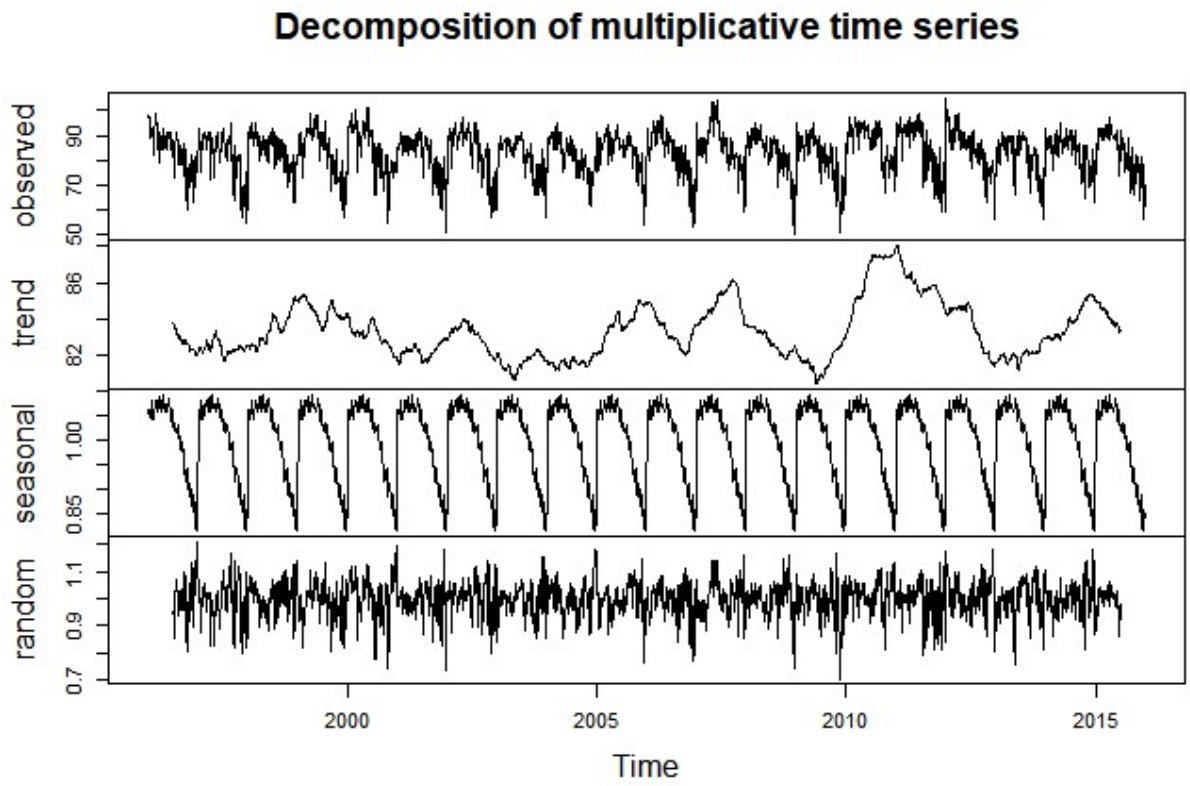


Pretty good match!

#Let's decompose our model into the trend and seasonal to see what's happening

decomposeTemp <-decompose(tempts, type = "multiplicative", filter=NULL)

plot(decomposeTemp)



#Based on this, there is clear seasonality, but there is no clear trend.

#So what are our seasonality factors? I'm going to put them in a matrix:

season <- matrix(HW$fitted[,3], nrow=123)

season

#Running CUSUM on the seasonality factors

model <- cusum(season, center = av_SF[1], std.dev = sd_SF[1], decision.interval = 2, se.shift =1, plot = TRUE)

#The R CUSUM model showed no significant changes.

#I then took this matrix and ran CUSUM in excel (attached separately) to see if I saw a change there

#Using the excel file, it seemed to confirm that there is no change in date for the end of summer.

#It was consistently 11 or 12 August.

Question 8.1 – Describe a situation for which a linear regression model would be appropriate.

To follow up on my example from 7.1, I could predict my weight using linear regression. For my predictors, I would use gender, height in inches, average calories consumed in a day, average minutes of exercise in a day, and average amount of tv time in a day.

Question 8.2 – Using crime data, use regression to predict the observed crime rate in a city with the specified data. Show the model, software output, and quality of fit.

The model is:

y=(-5984.288) + (87.83017)M + (-3.803450)So + (188.3243)Ed + (192.8043)Po1 + (-109.4219)Po2 + (-663.8261)LF + (17.40686)M.F + (-0.7330081)PoP + (4.204461)NW + (-5827.103)U1 + (167.7997)U2 + (0.09616624)Wealth + (70.67210)Ineq + (-4855.266)Prob + (-3.479018)Time

The software output is:

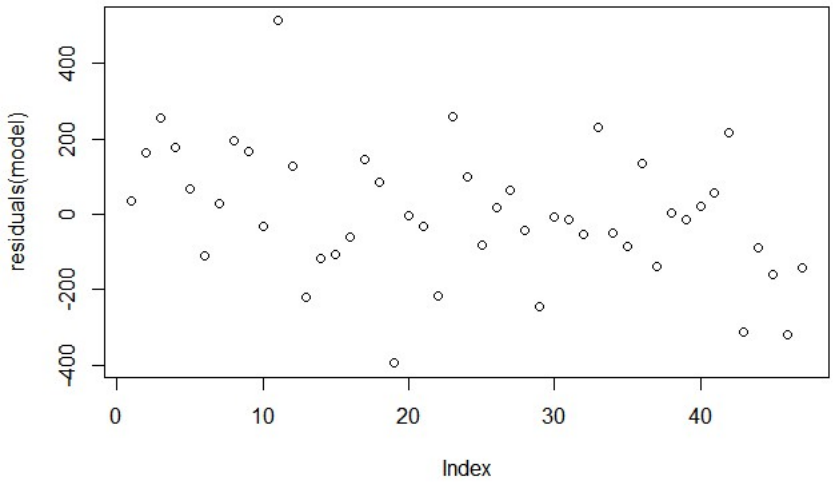
```
Call:

lm(formula = y ~ x, data = crime)


Residuals:

   Min     1Q  Median     3Q    Max

-395.74  -98.09   -6.69  112.99  512.67


Coefficients:

            Estimate Std. Error t value Pr(>|t|)

(Intercept) -5.984e+03  1.628e+03  -3.675 0.000893 ***

xM           8.783e+01  4.171e+01   2.106 0.043443 *

xSo         -3.803e+00  1.488e+02  -0.026 0.979765

xEd          1.883e+02  6.209e+01   3.033 0.004861 **
```

```
xPo1         1.928e+02  1.061e+02   1.817 0.078892 .
xPo2        -1.094e+02  1.175e+02  -0.931 0.358830
xLF         -6.638e+02  1.470e+03  -0.452 0.654654
xM.F         1.741e+01  2.035e+01   0.855 0.398995
xPop        -7.330e-01  1.290e+00  -0.568 0.573845
xNW          4.204e+00  6.481e+00   0.649 0.521279
xU1         -5.827e+03  4.210e+03  -1.384 0.176238
xU2          1.678e+02  8.234e+01   2.038 0.050161 .
xWealth      9.617e-02  1.037e-01   0.928 0.360754
xIneq        7.067e+01  2.272e+01   3.111 0.003983 **
xProb       -4.855e+03  2.272e+03  -2.137 0.040627 *
xTime       -3.479e+00  7.165e+00  -0.486 0.630708
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 209.1 on 31 degrees of freedom
Multiple R-squared:  0.8031,      Adjusted R-squared:  0.7078
F-statistic: 8.429 on 15 and 31 DF,  p-value: 3.539e-07
```

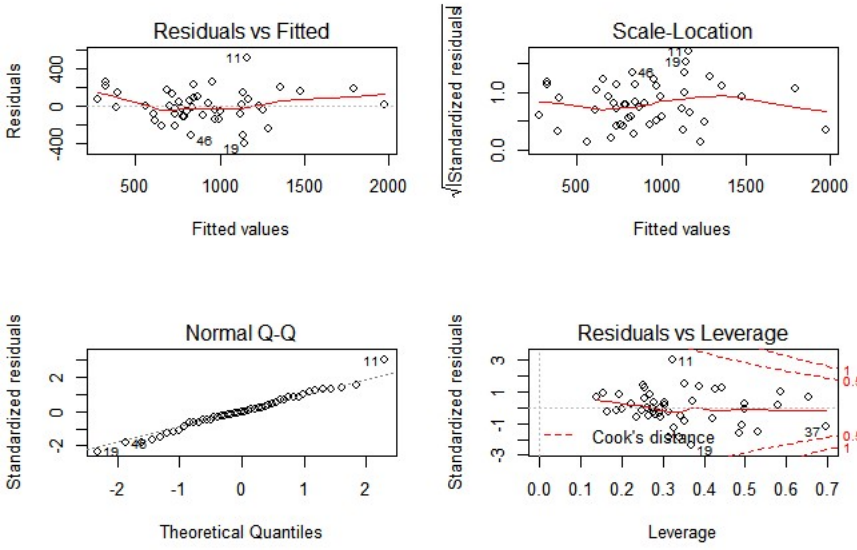
The fit is R^2 = 80.31%, with an adjusted R^2 of 70.78% because of the 15 predictors.

Regarding the number of predictors, it looks like lm() is giving us the significant ones with asterisks. Using a p-value of 0.05, only 5 were significant.

I ran a residuals plot, just to see if linear regression worked well – and it appears to be appropriate because the points are randomly dispersed.

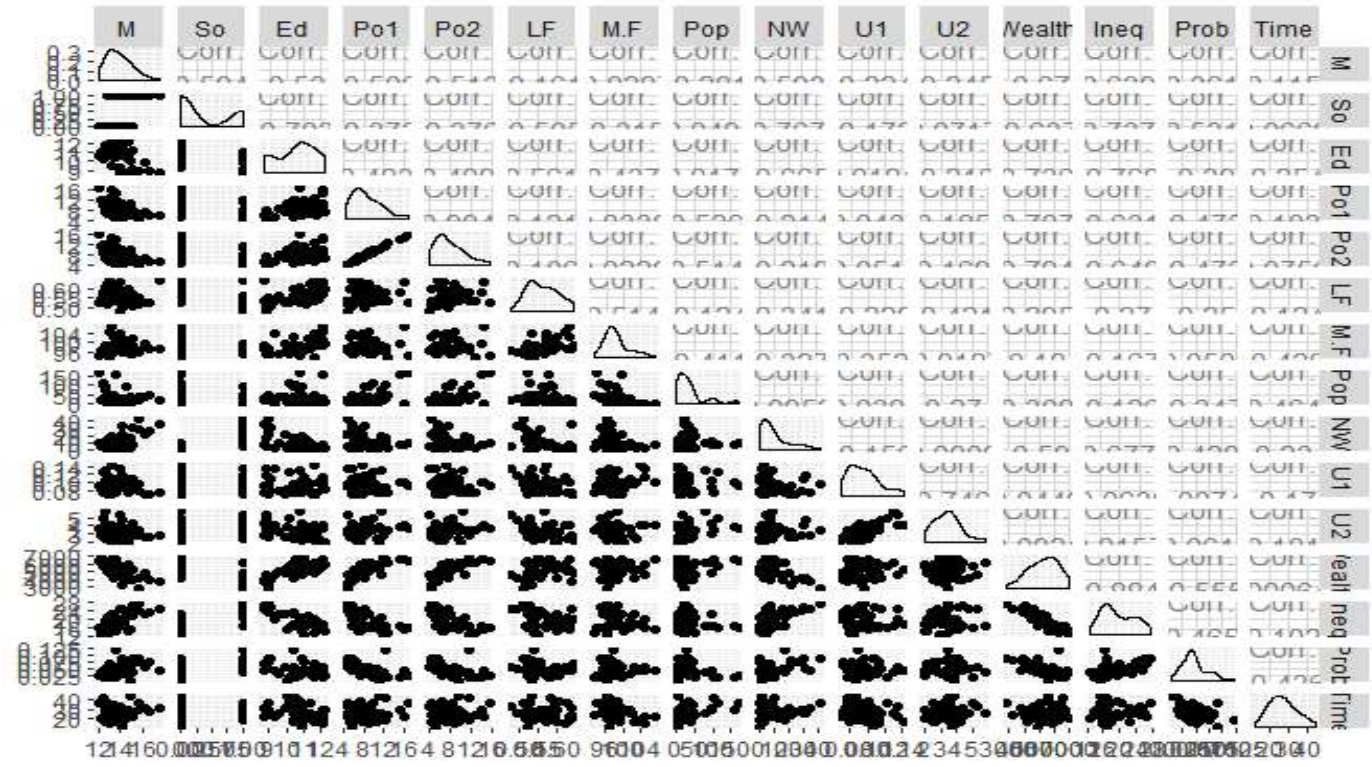


Let's check for heteroscedasticity, normality, and any influential observations



When I ran the anova for the model, I had a very low F value, indicating that all the explanatory variables together significantly explained the crime.

I looked at the pairwise correlation to examine multicollinearity:



This shows a relationship between Po1 and Po2, which may be a source of multicollinearity.

Using the values from the homework, I predicted a crime rate of 156.0724.

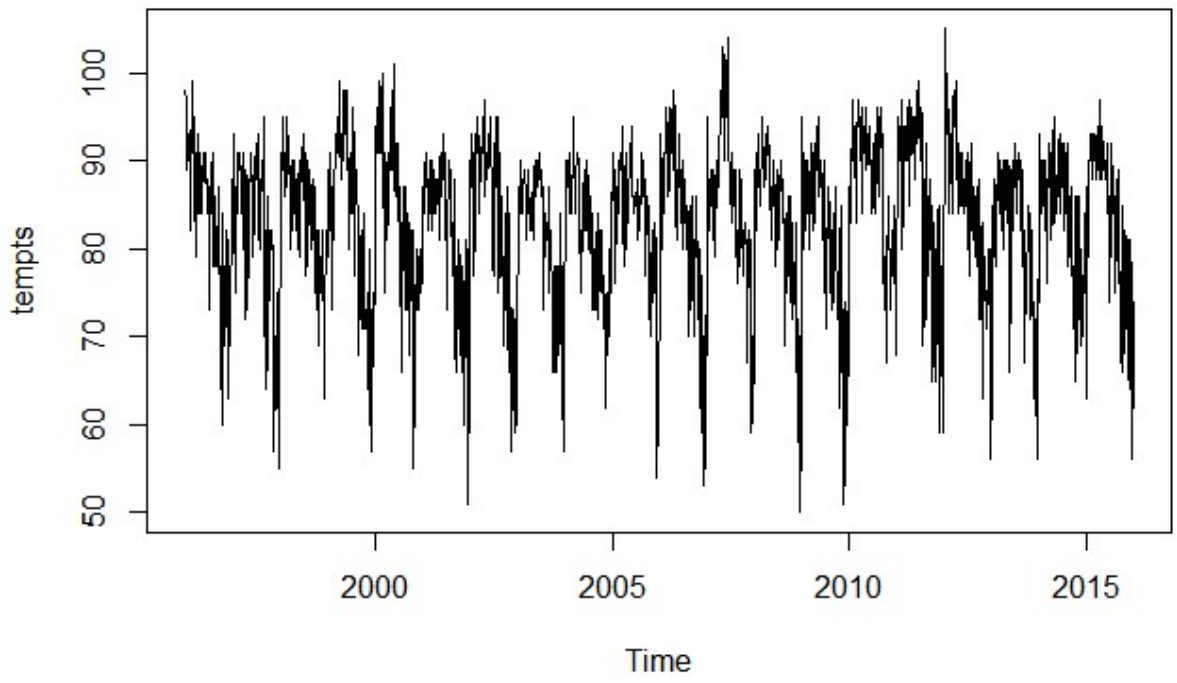
	Coefficient	HW Value	Coeff * Value
(Intercept)	-5984	1	-5984
xM	87.83	14	1229.62
xSo	-3.803	0	0
xEd	188.3	10	1883
xPo1	192.8	12	2313.6
xPo2	-109.4	15.5	-1695.7
xLF	-663.8	0.64	-424.832
xM.F	17.41	94	1636.54
xPop	-0.733	150	-109.95
xNW	4.204	1.1	4.6244
xU1	-5827	0.12	-699.24
xU2	167.8	3.6	604.08
xWealth	0.09617	3200	307.744
xIneq	70.67	20.1	1420.467
xProb	-4855	0.04	-194.2
xTime	-3.479	39	-135.681
		Sum:	156.0724

CODE (output in Console)

########################## Question 7.2 ######################################

```
> library(timeSeries)
> library(ggplot2)
> install.packages("forecast")
Installing package into 'C:/Users/kkisner/Documents/R/win-library/3.5'
(as 'lib' is unspecified)
trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.5/forecast_8.3.zip'
Content type 'application/zip' length 2082525 bytes (2.0 MB)
downloaded 2.0 MB

package 'forecast' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\kkisner\AppData\Local\Temp\Rtmpg3Ahdx\downloaded_packages
>
> #let's read the file in and call it "temp"
> temp <- read.table("c:/users/kkisner/Desktop/gradclass/temps.txt", header = TRUE)
> temp <- as.vector(unlist(temp[,2:21]))
>
> #let's make the data a time series, and then check out what it gives us to be sure we'r
> tempts <- ts(temp, start = 1996, frequency = 123)
> tempts
Time Series:
Start = c(1996, 1)
End = c(2015, 123)
Frequency = 123
   [1]  98  97  97  90  89  93  93  91  93  93  90  91  93  93  82  91  96  95  96  99  9
  [25]  84  84  82  79  90  91  87  86  90  84  91  93  88  91  84  90  89  88  86  84  8
  [49]  91  90  89  90  91  91  91  84  88  84  86  88  84  82  80  73  87  84  87  89  8
  [73]  86  88  78  79  86  82  82  78  79  79  78  81  84  84  87  84  79  75  72  64  6
  [97]  66  64  60  78  70  72  69  69  73  79  81  80  82  66  63  68  79  81  69  73  7
 [121]  82  82  81  86  90  93  91  84  84  75  87  84  87  84  88  86  90  91  91  89  8
 [145]  84  87  88  89  89  91  91  89  88  72  80  84  88  89  88  84  84  80  73  80  8
 [169]  88  91  91  89  89  88  82  79  81  82  84  87  90  90  91  91  88  88  91  93  8
 [193]  88  84  80  82  86  87  87  88  88  90  88  91  95  89  70  80  82  66  70  64  6
 [217]  73  75  78  81  82  82  82  80  82  82  79  80  68  63  57  66  64  69  70  70  6
 [241]  71  57  55  64  66  60  91  88  91  91  91  89  93  95  95  91  91  86  88  87  9
 [265]  95  91  91  89  91  91  86  88  80  88  89  90  86  86  82  84  86  90  89  89  8
 [289]  84  86  80  82  86  84  87  90  79  84  87  87  88  90  91  89  90  93  93  91  8
 [313]  91  89  90  89  79  78  81  84  89  87  87  88  87  82  80  82  82  88  84  81  8
 [337]  75  75  86  78  77  82  82  73  82  69  72  73  78  78  78  75  79  78  77  78  8
 [361]  63  72  75  79  79  79  78  82  79  84  82  87  88  90  91  82  86  87  87  82  7
 [385]  86  82  87  88  90  90  91  93  93  91  93  93  93  93  97  99  96  93  88  89  9
 [409]  91  90  96  98  97  98  93  93  96  98  98  89  91  91  90  80  82  89  88  90  9
 [433]  91  84  93  96  96  91  91  77  87  87  87  86  87  89  81  81  82  79  68  79  7
 [457]  82  78  80  77  71  73  75  84  71  73  71  73  73  72  72  73  70  64  75  73  7
 [481]  60  64  73  57  59  64  69  75  73  72  75  75  89  91  93  95  96  96  96  91  9
 [505]  91  93  93  93  91  97 100  99  93  96  87  82  75  82  88  91  89  87  86  86  8
 [529]  91  91  91  96  95  89  89  89  89  94  97  99 101 101  97  87  86  88  92  92  9
 [553]  88  87  79  81  82  87  81  66  66  75  80  82  84  86  87  86  80  75  73  73  8
 [577]  81  84  82  68  71  75  73  75  77  79  82  81  82  73  66  55  55  64  71  73  7
 [601]  80  80  73  73  75  79  75  75  78  75  78  80  75  77  78  84  87  87  84  86  8
 [625]  87  90  90  86  82  82  84  87  88  90  87  84  87  90  84  82  88  90  84  89  8
 [649]  84  86  88  84  86  88  87  88  86  86  81  87  84  90  91  91  87  86  88  90  8
 [673]  91  81  86  81  82  80  75  73  81  90  88  87  86  86  89  87  84  84  86  77  7
 [697]  84  86  87  88  69  66  72  75  78  71  71  75  80  81  80  79  70  68  79  66  7
 [721]  75  75  62  60  64  71  75  79  80  81  79  73  64  51  55  63  72  71  90  90  8
 [745]  89  89  90  91  84  77  82  88  91  93  93  93  93  91  95  91  89  87  84  86  8
 [769]  90  93  91  91  91  93  97  87  87  86  88  89  91  91  89  88  90  91  93  91  9
 [793]  93  91  88  84  82  82  78  77  84  84  89  95  93  91  88  87  91  95  95  90  7
```

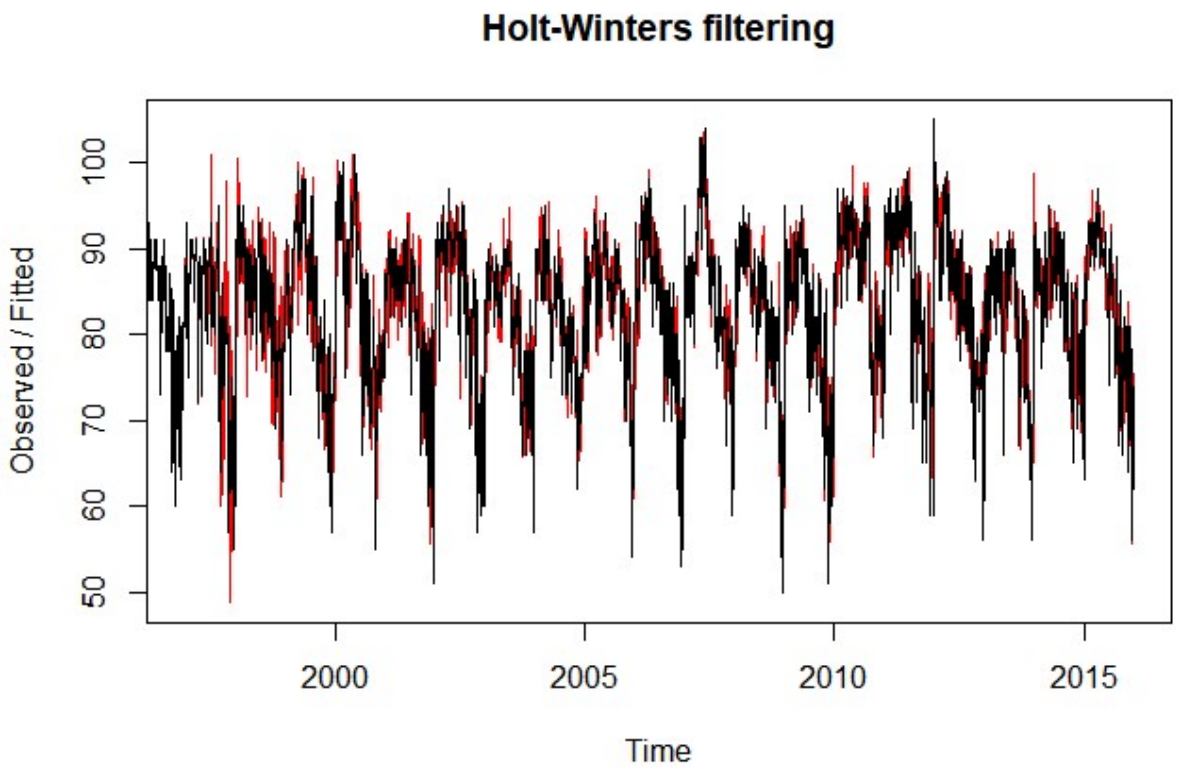
```
 [817]  86  81  80  86  84  77  82  73  69  75  75  79  73  79  82  84  84  82  87  86  8
 [841]  78  84  79  68  57  66  64  68  71  73  71  64  59  68  60  68  69  75  75  68  6
 [865]  86  80  84  87  90  89  84  84  86  87  84  86  88  88  88  88  88  89  86  81  8
 [889]  89  88  84  88  84  84  84  82  84  82  84  84  86  87  84  81  87  89  90  86  8
 [913]  88  88  90  89  88  89  90  91  89  88  89  88  86  87  87  84  73  75  81  82  7
 [937]  82  82  81  81  81  84  87  82  75  81  80  82  82  82  73  66  71  72  68  66  7
 [961]  73  73  73  66  78  78  78  69  72  68  70  75  78  84  78  78  73  73  68  64  5
 [985]  82  81  86  88  90  90  89  87  88  89  90  89  91  91  84  84
 [ reached getOption("max.print") -- omitted 1460 entries ]
> #great!  It starts in 1996 and ends is 2015... just what we wanted.
>
> #let's see what the data looks like
> plot(tempts)
```



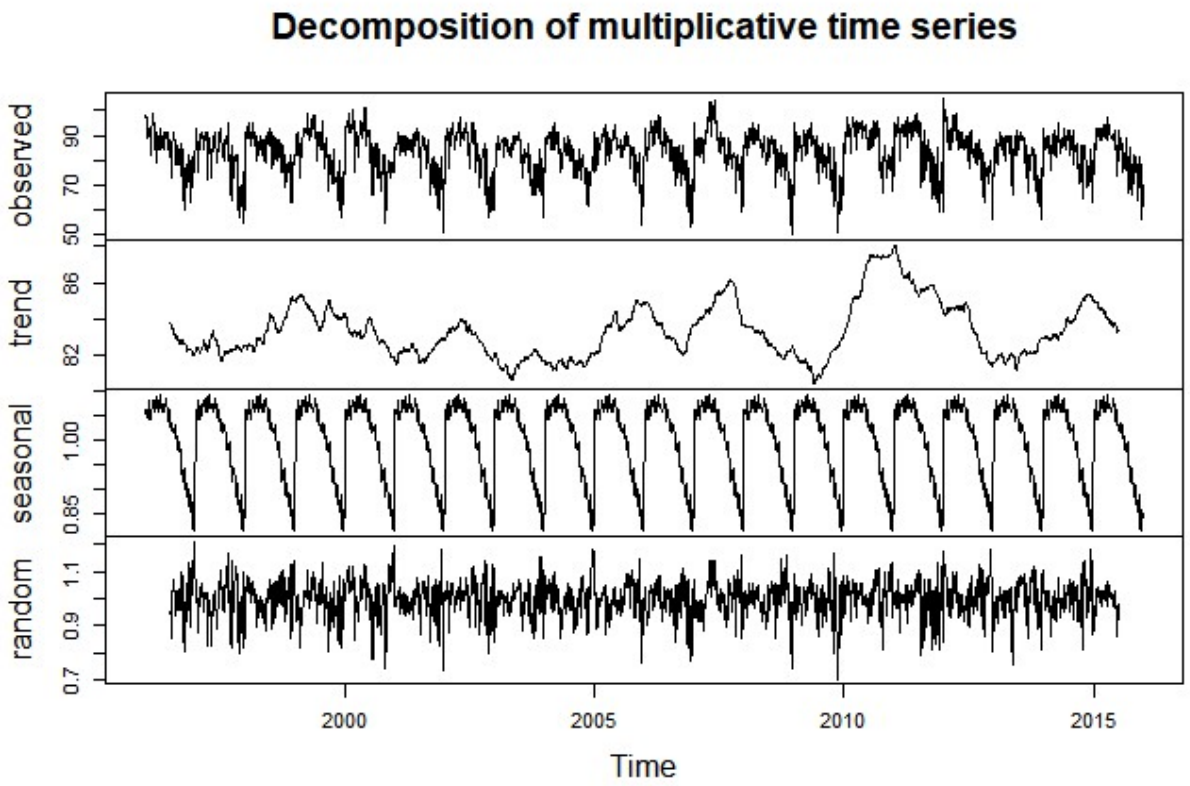

```
> #well, nothing is too obvious about the trends, but there's clearly seasonality.
>
> #Let's do the exponential smoothing with Holt Winters.  I added that seasonality is mul
> #I set beta = FALSE becuase the help said that would make it do exponential smoothing.
> #In practice, I got the same answer regardless of not mentioning beta, beta = NULL, and
> HW <- HoltWinters(tempts, beta = FALSE, seasonal = "mult")
> HW
Holt-Winters exponential smoothing without trend and with multiplicative seasonal compone

Call:
HoltWinters(x = tempts, beta = FALSE, seasonal = "mult")

Smoothing parameters:
 alpha: 0.6150515
 beta : FALSE
 gamma: 0.549585

Coefficients:
           [,1]
```

a	73.7068469
s1	1.2385831
s2	1.2338982
s3	1.1590841
s4	1.1748143
s5	1.1709114
s6	1.1506136
s7	1.1389645
s8	1.1300718
s9	1.1100862
s10	1.0758588
s11	1.0406773
s12	1.0577711
s13	1.0321410
s14	1.0359037
s15	1.0190030
s16	1.0264067
s17	1.0708084
s18	1.0544613
s19	1.0840274
s20	1.0642393
s21	1.1094422
s22	1.1122815
s23	1.1035831
s24	1.1023842
s25	1.0908828
s26	1.0841410
s27	1.0775420
s28	1.0773257
s29	1.0534278
s30	1.0790854
s31	1.0531206
s32	1.0536624
s33	1.0778522
s34	1.0697793
s35	1.0545323
s36	1.0442347
s37	1.0229427
s38	1.0254956
s39	1.0307341
s40	1.0310786
s41	1.0214910
s42	0.9978492
s43	0.9957235
s44	0.9812509
s45	0.9761939
s46	0.9676645
s47	0.9854692
s48	1.0044211
s49	1.0506206
s50	1.0721588
s51	1.0861675
s52	1.0979856
s53	1.0967853
s54	1.0544679
s55	1.0225193
s56	0.9869258
s57	1.0165824
s58	1.0162657
s59	1.0039868
s60	1.0187642
s61	0.9835214
s62	1.0555364
s63	1.0557680

```
s64   1.0431259
s65   1.0391226
s66   0.9906816
s67   1.0010961
s68   1.0018809
s69   1.0036079
s70   0.9992263
s71   1.0182906
s72   1.0261413
s73   1.0421520
s74   1.0221497
s75   1.0021580
s76   1.0042124
s77   1.0251795
s78   1.0150029
s79   0.9918296
s80   0.9790359
s81   0.9977116
s82   1.0022064
s83   0.9550989
s84   0.9706363
s85   0.9752093
s86   0.9311971
s87   0.9264481
s88   0.9582390
s89   0.9629218
s90   0.9513191
s91   0.9370425
s92   0.9539319
s93   0.8921790
s94   0.8792364
s95   0.8796457
s96   0.8903289
s97   0.9168230
s98   0.9256788
s99   0.8839515
s100  0.8463686
s101  0.8334255
s102  0.7997454
s103  0.8076803
s104  0.8190880
s105  0.8283140
s106  0.7953616
s107  0.7963619
s108  0.8152471
s109  0.8298463
s110  0.8288170
s111  0.8180966
s112  0.8636683
s113  0.9117451
s114  0.8979969
s115  0.8784171
s116  0.8486747
s117  0.8136068
s118  0.8465252
s119  0.8188350
s120  0.8507372
s121  0.8201246
s122  0.8512756
s123  0.8737193
> #This gives me alpha .6150, beta = 0, gamma = 0.5495
>
> #Let's take a look at what the filtering looks like.
> plot(HW)
```

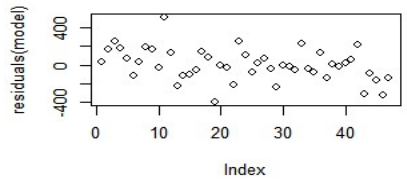



```
>
> #Let's decompose our model into the trend and seasonal to see what's happening
> decomposeTemp <-decompose(tempts, type = "multiplicative", filter=NULL)
> plot(decomposeTemp)
>
```



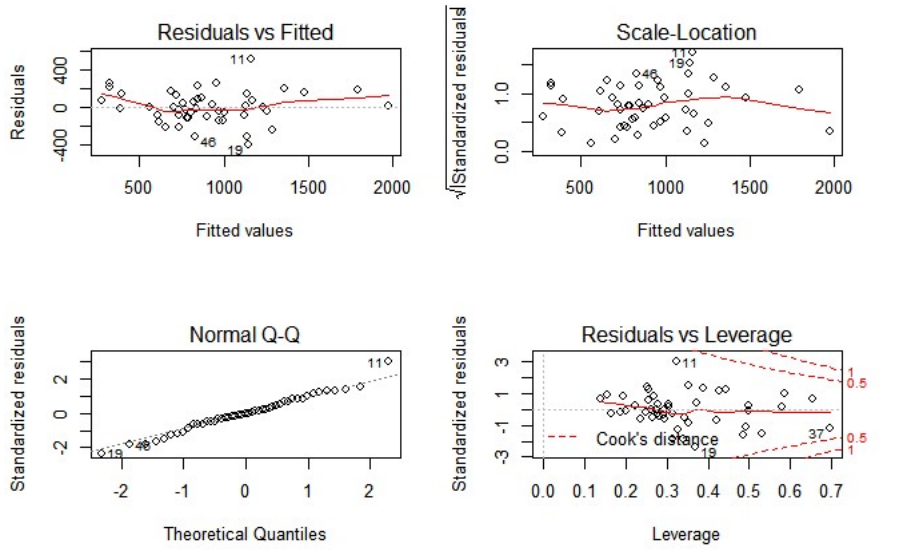

```
> #Based on this, there is clear seasonality, but there is no clear trend.
#So what are our seasonality factors?  I'm going to put them in a matrix:
season <- matrix(HW$fitted[,3], nrow=123)

#Running CUSUM on the seasonality factors
model <- cusum(season, center = av_SF[1], std.dev = sd_SF[1], decision.interval = 2, se.shift =1, plot = TRUE)
#The R CUSUM model showed no significant changes.


#I then took this matrix and ran CUSUM in excel (attached separately) to see if I saw a c
#Using the excel file, it seemed to confirm that there is no change in date for the end o
#It was consistently 11 or 12 August.



#########################   Question 8.2   ######################

> crime <- read.table("c:/users/kkisner/Desktop/gradclass/uscrime.txt", header = TRUE)
> head(crime)
     M So   Ed  Po1  Po2    LF   M.F Pop   NW    U1  U2 Wealth Ineq      Prob    Time Crim
1 15.1  1  9.1  5.8  5.6 0.510  95.0  33 30.1 0.108 4.1   3940 26.1 0.084602 26.2011   79
2 14.3  0 11.3 10.3  9.5 0.583 101.2  13 10.2 0.096 3.6   5570 19.4 0.029599 25.2999  163
3 14.2  1  8.9  4.5  4.4 0.533  96.9  18 21.9 0.094 3.3   3180 25.0 0.083401 24.3006   57
4 13.6  0 12.1 14.9 14.1 0.577  99.4 157  8.0 0.102 3.9   6730 16.7 0.015801 29.9012  196
5 14.1  0 12.1 10.9 10.1 0.591  98.5  18  3.0 0.091 2.0   5780 17.4 0.041399 21.2998  123
6 12.1  0 11.0 11.8 11.5 0.547  96.4  25  4.4 0.084 2.9   6890 12.6 0.034201 20.9995   68
>
> #let's assign x and y to clean up the lm() formula below
> y <- as.vector(crime$Crime)
> x <- as.matrix(crime[,1:15])
```

```
>
> #Now let's run the regression with the cleaned up x and y
> model <-lm(y~x, data = crime)
>
> #What does that give us?
> summary(model)

Call:
lm(formula = y ~ x, data = crime)

Residuals:
    Min      1Q  Median      3Q     Max
-395.74  -98.09   -6.69  112.99  512.67

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -5.984e+03  1.628e+03  -3.675 0.000893 ***
xM           8.783e+01  4.171e+01   2.106 0.043443 *
xSo         -3.803e+00  1.488e+02  -0.026 0.979765
xEd          1.883e+02  6.209e+01   3.033 0.004861 **
xPo1         1.928e+02  1.061e+02   1.817 0.078892 .
xPo2        -1.094e+02  1.175e+02  -0.931 0.358830
xLF         -6.638e+02  1.470e+03  -0.452 0.654654
xM.F         1.741e+01  2.035e+01   0.855 0.398995
xPop        -7.330e-01  1.290e+00  -0.568 0.573845
xNW          4.204e+00  6.481e+00   0.649 0.521279
xU1         -5.827e+03  4.210e+03  -1.384 0.176238
xU2          1.678e+02  8.234e+01   2.038 0.050161 .
xWealth      9.617e-02  1.037e-01   0.928 0.360754
xIneq        7.067e+01  2.272e+01   3.111 0.003983 **
xProb       -4.855e+03  2.272e+03  -2.137 0.040627 *
xTime       -3.479e+00  7.165e+00  -0.486 0.630708
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 209.1 on 31 degrees of freedom
Multiple R-squared:  0.8031,   Adjusted R-squared:  0.7078
F-statistic: 8.429 on 15 and 31 DF,  p-value: 3.539e-07

>
> #Let's see if linear regression was a good choice
> plot(residuals(model))
```

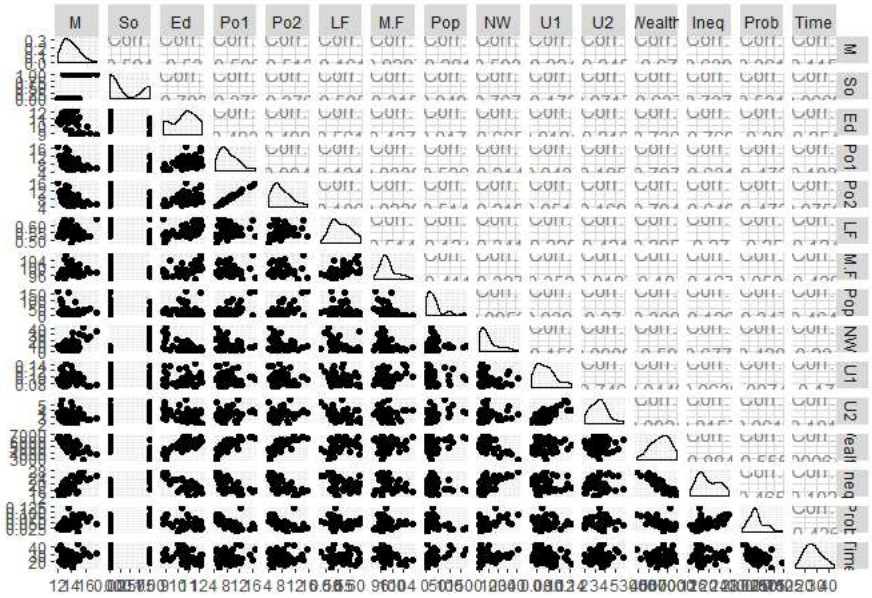



```
> #yep, the points look pretty randomly distributed... Seems like linear was a good fit.
>
```

```
> #Let's check for heteroscedasticity, normality, and any influential observations
> layout(matrix(c(1,2,3,4),2,2)) # optional 4 graphs/page
> plot(model)
```




```
>
> #how well did our variables work?
> anova(model)
Analysis of Variance Table

Response: y
          Df  Sum Sq Mean Sq F value    Pr(>F)
x         15 5525982  368399  8.4286 3.539e-07 ***
Residuals 31 1354946   43708
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
> #Great!  Look at that SUPER LOW value for F!
>

> #We have a lot of variables... we might have multicollinearity
> #So what does our pair-wise correlation look like?
> library(GGally)
> X<-crime[,1:15]
> ggpairs(X)
```




```
> # Looks like Po1 and Po2 may be related.
```