Homework 2 Solutions

Ian Cranfield [redacted]

4/21/2020 by 5PM

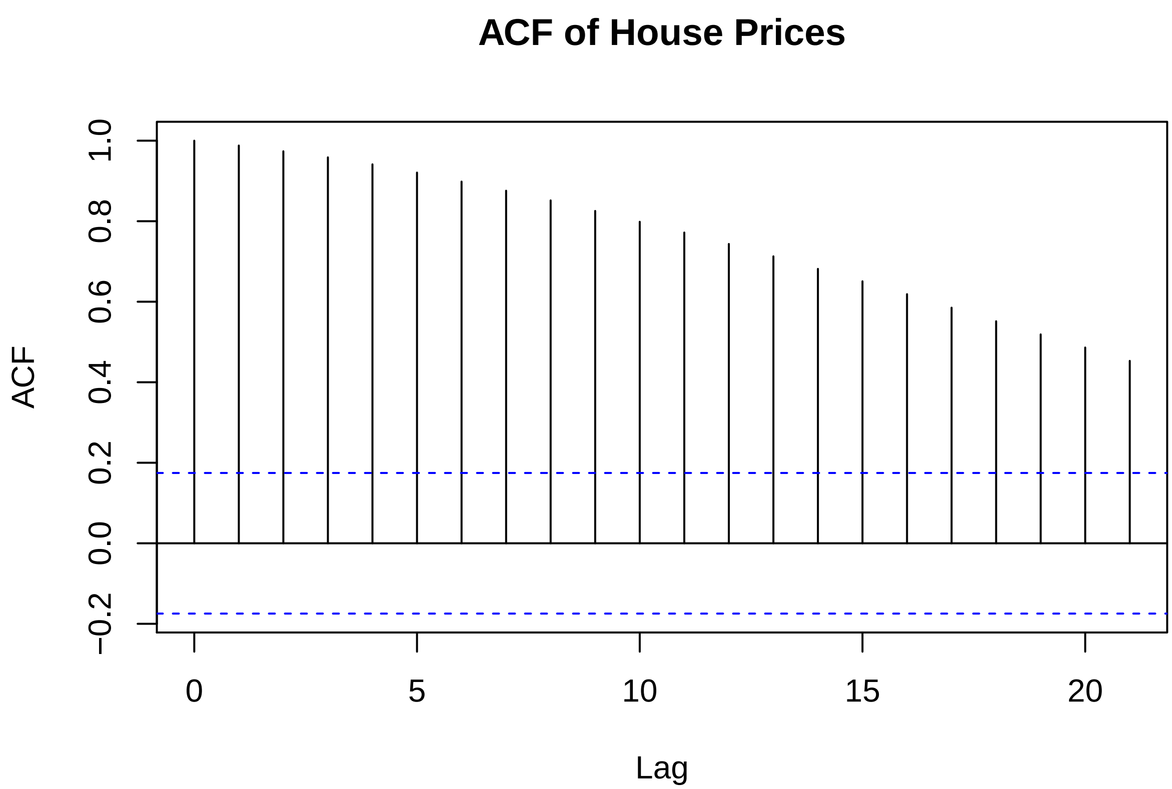
```
library(dynlm)
library(pastecs)
library(openxlsx)
library(tidyverse)
library(data.table)
library(scales)
library(TSstudio)
library("readxl")
library(ggplot2)
require(cowplot)
library(psych)
library(corrplot)
library(gtable)
library(timeDate)
library(PerformanceAnalytics)
library(fpp2)
library(seasonal)
```

Problem 4.3:

```
#### Problem 4.3 ####
Quarter_house <- read_excel("Chapter4_exercises_data.xls", sheet =2)
setnames(Quarter_house, c('Quarter', 'P', 'R'))
Quarter_house$Quarter <- as.Date( as.yearqtr(Quarter_house$Quarter, format = "%YQ%q" ))
#Price <- ts(Quarter_house$P, start = 1980.25, frequency = 4)
#Rate <- ts(Quarter_house$R, start = 1980.25, frequency = 4)

glimpse(Quarter_house)
```

```
## Rows: 126
## Columns: 3
## $ Quarter <date> 1980-04-01, 1980-07-01, 1980-10-01, 1981-01-01, 1981-04-01...
## $ P       <dbl> 42.49871, 43.27341, 43.78586, 44.35695, 45.07844, 45.55761,...
## $ R       <dbl> 14.43000, 12.65000, 14.26333, 15.14333, 16.22667, 17.42333,...
```

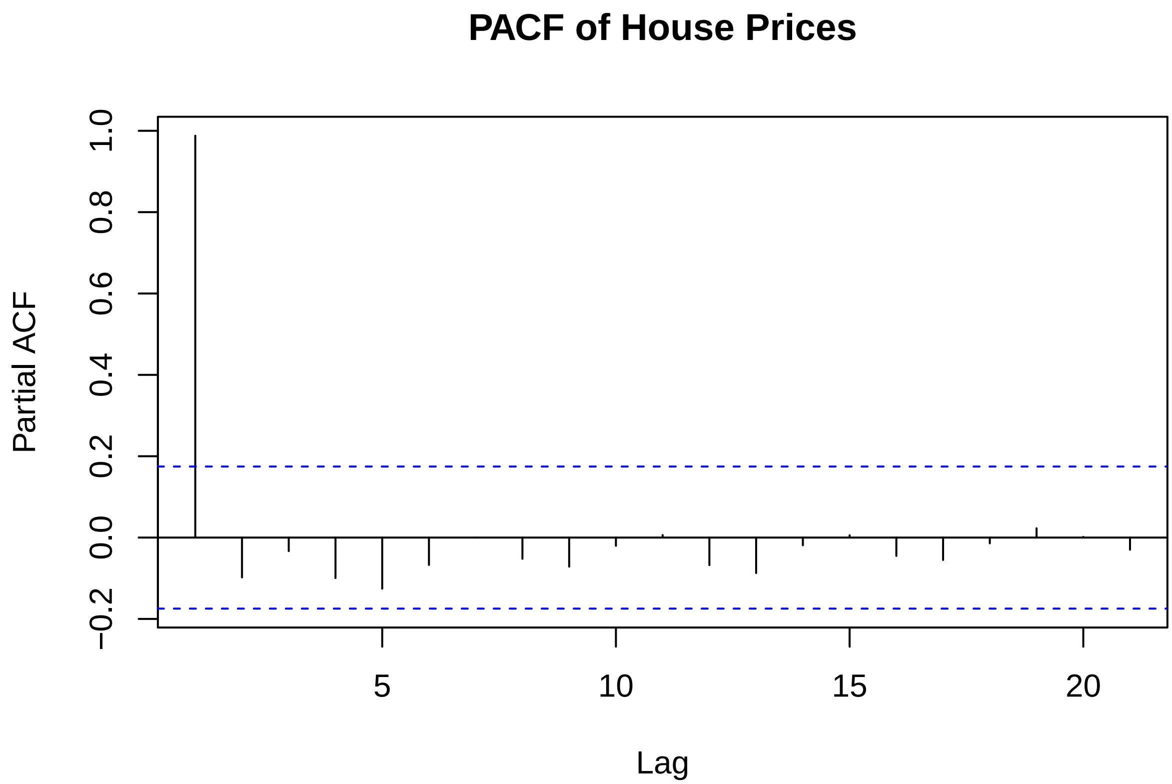
```
acf_price <- acf(Quarter_house$P, plot = T, main = "ACF of House Prices")
```




```
acf_price
```

```
##
## Autocorrelations of series 'Quarter_house$P', by lag
##
##     0     1     2     3     4     5     6     7     8     9    10    11    12
## 1.000 0.988 0.974 0.959 0.941 0.921 0.898 0.876 0.852 0.826 0.799 0.772 0.743
##    13    14    15    16    17    18    19    20    21
## 0.713 0.682 0.651 0.619 0.585 0.552 0.519 0.486 0.453
```

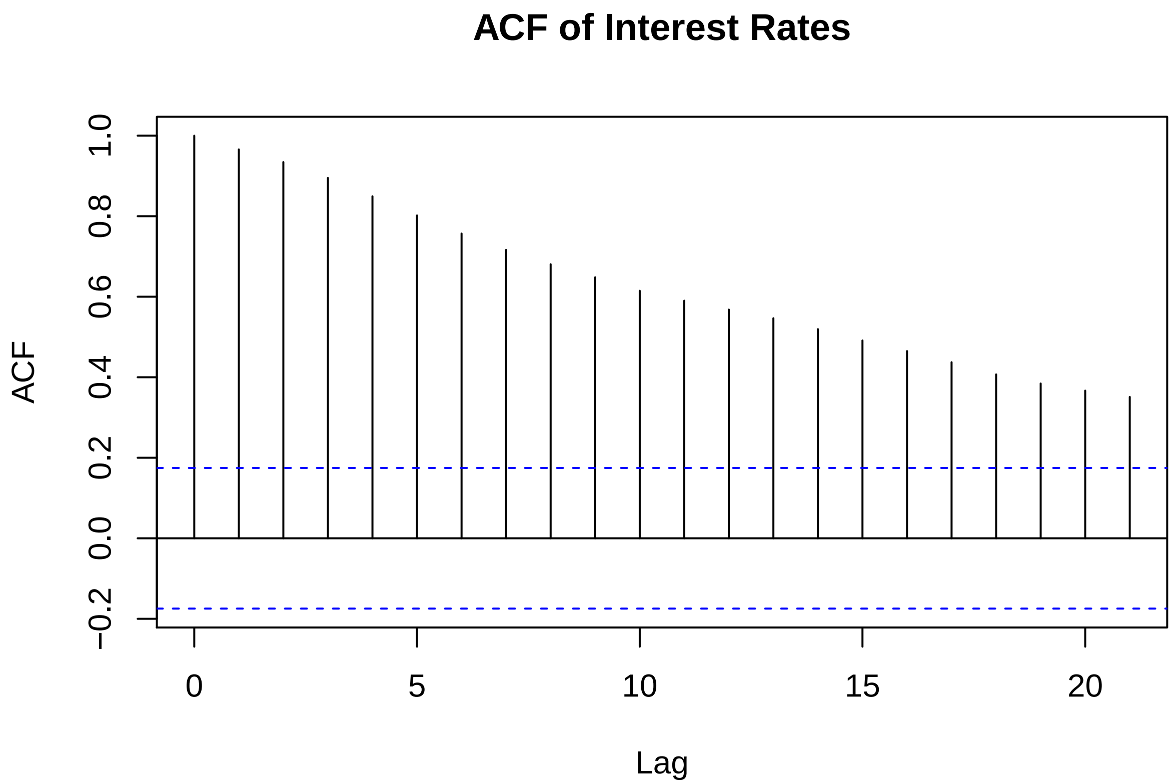
```
pacf_price <- pacf(Quarter_house$P, plot= T, main = "PACF of House Prices")
```




```r
pacf_price
```

```
##
## Partial autocorrelations of series 'Quarter_house$P', by lag
##
##      1      2      3      4      5      6      7      8      9     10     11
##  0.988 -0.098 -0.033 -0.100 -0.126 -0.067  0.000 -0.052 -0.072 -0.020  0.006
##     12     13     14     15     16     17     18     19     20     21
## -0.068 -0.087 -0.019  0.006 -0.045 -0.055 -0.014  0.023  0.002 -0.030
```

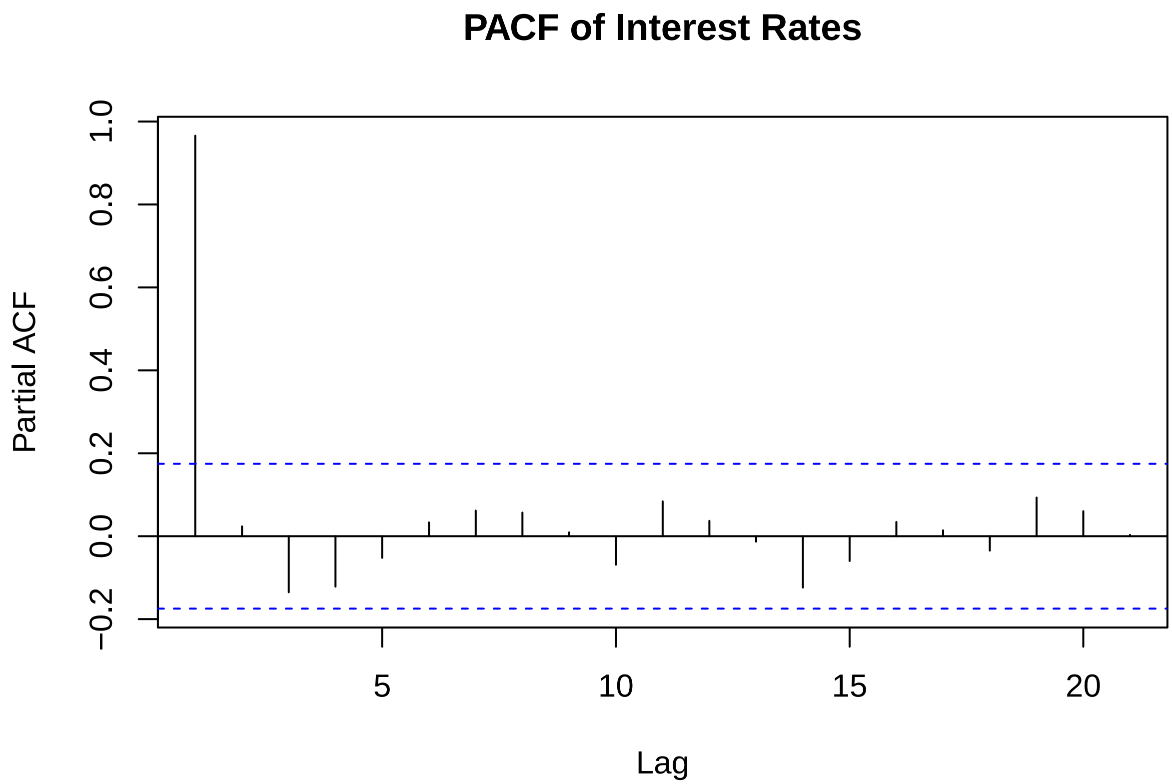
```r
acf_rate <- acf(Quarter_house$R, plot = TRUE, main = "ACF of Interest Rates")
```




```
acf_rate
```

```
##
## Autocorrelations of series 'Quarter_house$R', by lag
##
##     0     1     2     3     4     5     6     7     8     9    10    11    12
## 1.000 0.966 0.934 0.895 0.850 0.802 0.757 0.716 0.681 0.648 0.615 0.590 0.568
##    13    14    15    16    17    18    19    20    21
## 0.547 0.519 0.491 0.465 0.437 0.407 0.384 0.367 0.351
```

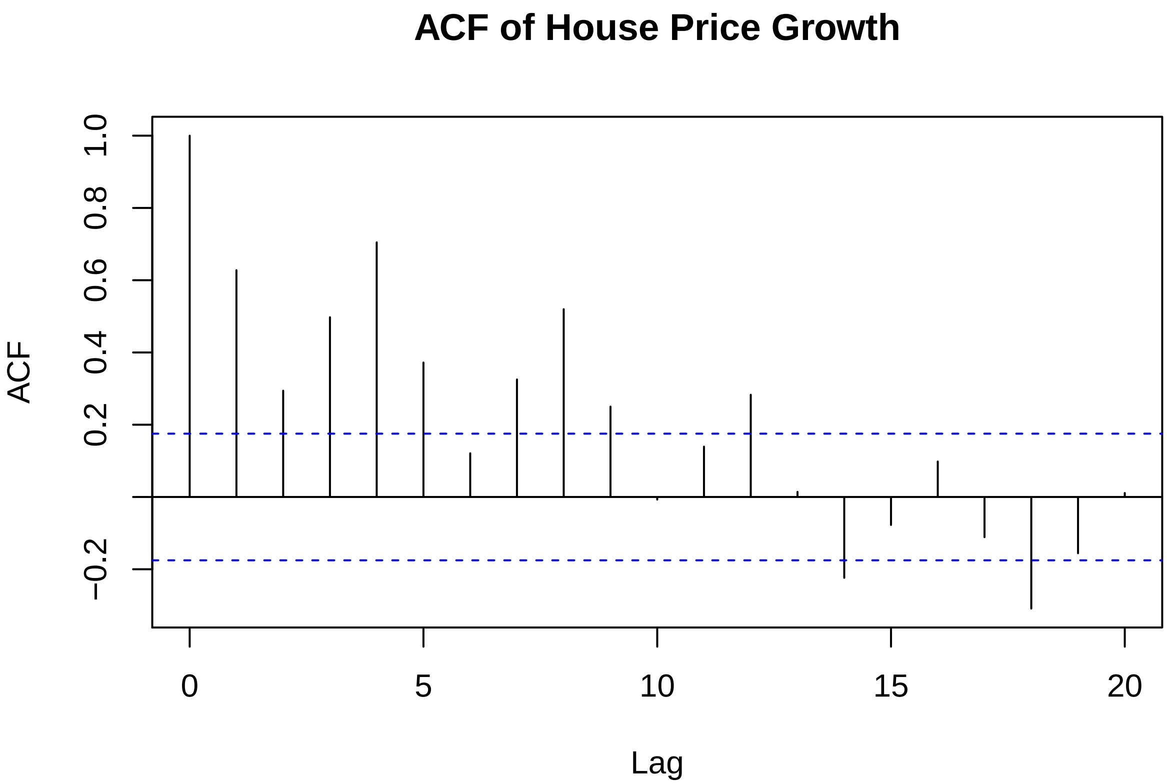
```
pacf_rate <- pacf(Quarter_house$R, plot = TRUE, main = "PACF of Interest Rates")
```




```
pacf_rate
```

```
##
## Partial autocorrelations of series 'Quarter_house$R', by lag
##
##      1      2      3      4      5      6      7      8      9     10     11
##  0.966  0.024 -0.135 -0.122 -0.052  0.033  0.062  0.057  0.009 -0.069  0.084
##     12     13     14     15     16     17     18     19     20     21
##  0.037 -0.013 -0.124 -0.060  0.034  0.014 -0.035  0.093  0.060  0.003
```

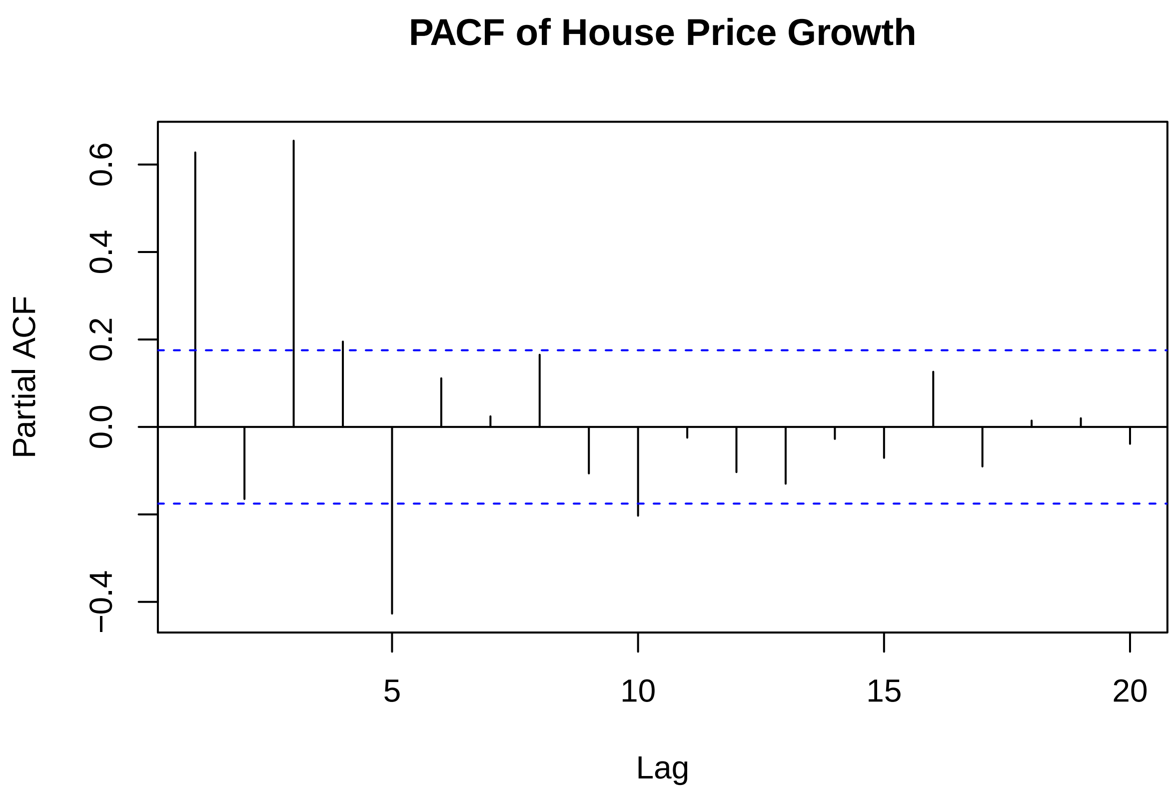
```
Quarter_house <- Quarter_house %>%
  mutate( P_growth = log(P)-log(lag(P)))


acf_price_growth <- acf(Quarter_house$P_growth, plot = TRUE, na.action = na.omit, main = "ACF of House P
```




```r
acf_price_growth
```

```
##
## Autocorrelations of series 'Quarter_house$P_growth', by lag
##
##      0      1      2      3      4      5      6      7      8      9     10
##  1.000  0.628  0.294  0.498  0.705  0.372  0.121  0.325  0.520  0.250 -0.007
##     11     12     13     14     15     16     17     18     19     20
##  0.139  0.283  0.014 -0.224 -0.078  0.098 -0.111 -0.309 -0.156  0.011
```

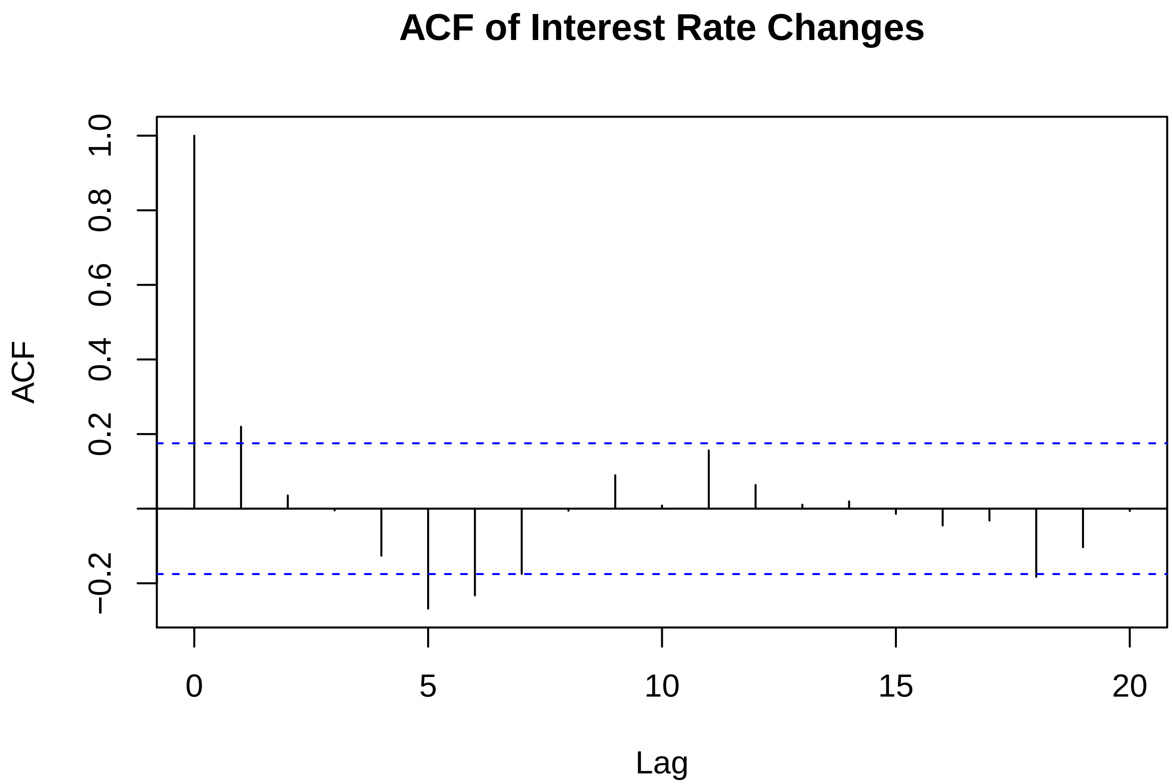
```r
pacf_price_growth <- pacf(Quarter_house$P_growth, plot = TRUE, na.action = na.omit, main = "PACF of Hou
```




```
pacf_price_growth
```

```
##
## Partial autocorrelations of series 'Quarter_house$P_growth', by lag
##
##      1      2      3      4      5      6      7      8      9     10     11
##  0.628 -0.165  0.655  0.195 -0.427  0.111  0.024  0.165 -0.106 -0.203 -0.025
##     12     13     14     15     16     17     18     19     20
## -0.103 -0.130 -0.027 -0.071  0.126 -0.090  0.015  0.020 -0.039
```

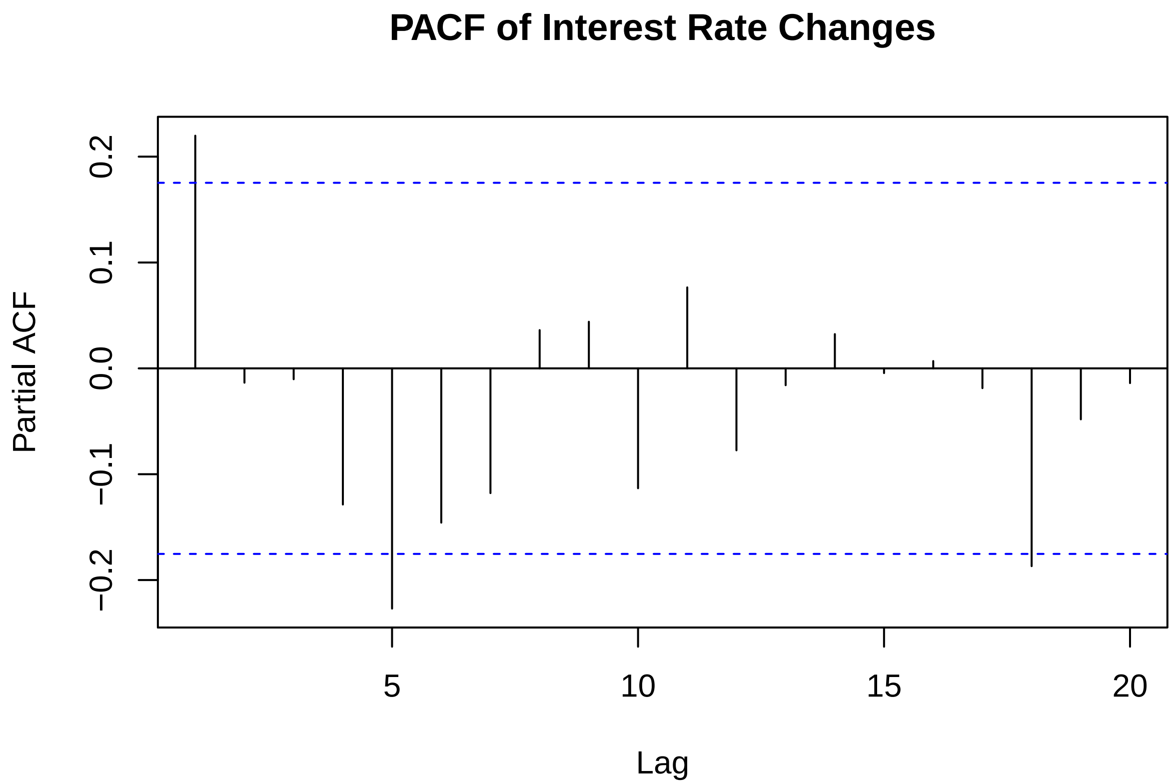
```
acf_rate_change <- acf(diff(Quarter_house$R), na.action = na.omit, main = "ACF of Interest Rate Changes
```




```
acf_rate_change
```

```
##
## Autocorrelations of series 'diff(Quarter_house$R)', by lag
##
##      0      1      2      3      4      5      6      7      8      9     10
##  1.000  0.220  0.035 -0.005 -0.126 -0.268 -0.232 -0.175 -0.006  0.090  0.008
##     11     12     13     14     15     16     17     18     19     20
##  0.156  0.063  0.011  0.020 -0.014 -0.045 -0.032 -0.183 -0.104 -0.006
```

```
pacf_rate_change <- pacf(diff(Quarter_house$R), na.action = na.omit, main = "PACF of Interest Rate Chang
```




```
pacf_rate_change
```

```
##
## Partial autocorrelations of series 'diff(Quarter_house$R)', by lag
##
##      1      2      3      4      5      6      7      8      9     10     11
##  0.220 -0.014 -0.010 -0.129 -0.227 -0.146 -0.118  0.036  0.044 -0.113  0.076
##     12     13     14     15     16     17     18     19     20
## -0.077 -0.016  0.032 -0.004  0.007 -0.019 -0.187 -0.048 -0.014
```

We observed the ACF and PACF of Quarterly House Prices, Interest Rates, House Price Growth, and Interest Rate Changes.

The ACF of House Price and Interest Rates are significantly correlated but slowly decreases to 0. But, the ACF of House Price Growth are significant correlated for the first 12 lags.

The PACF of House Price and Interest Rates remain stationary with no significant correlations meaning the past Quarter does not correlate with the present Quarter. The PACF of House Price Growth are significantly correlated for the first 5 lags. The PACF of Interest Rate Changes are significantly correlated at lags 5, and 18 where it is less than -0.2.

Problem 4.4:

```
## 4.4

# Annual data from 4.1
Annual_house <- read_excel("Chapter4_exercises_data.xls", sheet =1)
setnames(Annual_house, c("Year", "P", "R"))
```

```r
ts_P_annual <- ts(Annual_house$P, start = 1975, frequency = 1)

Ann_P_growth <- diff(log(ts_P_annual))

reg44iA <- dynlm(Ann_P_growth ~ L(Ann_P_growth,1))
summary(reg44iA)
```

```
##
## Time series regression with "ts" data:
## Start = 1977, End = 2011
##
## Call:
## dynlm(formula = Ann_P_growth ~ L(Ann_P_growth, 1))
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.086311 -0.013786  0.007275  0.017808  0.050339
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)         0.001363   0.006727   0.203    0.841
## L(Ann_P_growth, 1) 0.889768   0.099457   8.946 2.44e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.02913 on 33 degrees of freedom
## Multiple R-squared:  0.7081, Adjusted R-squared:  0.6992
## F-statistic: 80.03 on 1 and 33 DF,  p-value: 2.437e-10
```

```r
reg44iiA <- dynlm(Ann_P_growth ~ L(Ann_P_growth,1:2))
summary(reg44iiA)
```

```
##
## Time series regression with "ts" data:
## Start = 1978, End = 2011
##
## Call:
## dynlm(formula = Ann_P_growth ~ L(Ann_P_growth, 1:2))
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.075902 -0.008761  0.006968  0.015660  0.031019
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)              0.006284   0.006204   1.013  0.31894
## L(Ann_P_growth, 1:2)1  1.269847   0.157396   8.068 4.13e-09 ***
## L(Ann_P_growth, 1:2)2 -0.485851   0.161879  -3.001  0.00527 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.02556 on 31 degrees of freedom
## Multiple R-squared:  0.7791, Adjusted R-squared:  0.7649
## F-statistic: 54.68 on 2 and 31 DF,  p-value: 6.823e-11
```

```r
reg44iiiA <- dynlm(Ann_P_growth ~ L(Ann_P_growth,1:3))
summary(reg44iiiA)
```

```
##
## Time series regression with "ts" data:
## Start = 1979, End = 2011
##
## Call:
## dynlm(formula = Ann_P_growth ~ L(Ann_P_growth, 1:3))
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.070274 -0.013101  0.005806  0.017802  0.031248
##
## Coefficients:
##                        Estimate Std. Error t value Pr(>|t|)
## (Intercept)            0.002655   0.008284   0.320   0.7509
## L(Ann_P_growth, 1:3)1  1.339177   0.239772   5.585    5e-06 ***
## L(Ann_P_growth, 1:3)2 -0.679351   0.388512  -1.749   0.0909 .
## L(Ann_P_growth, 1:3)3  0.174552   0.277536   0.629   0.5343
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.02588 on 29 degrees of freedom
## Multiple R-squared:  0.7674, Adjusted R-squared:  0.7433
## F-statistic: 31.89 on 3 and 29 DF,  p-value: 2.551e-09
```

```r
reg44ivA <- dynlm(Ann_P_growth ~ L(Ann_P_growth,1:4))
summary(reg44ivA)
```

```
##
## Time series regression with "ts" data:
## Start = 1980, End = 2011
##
## Call:
## dynlm(formula = Ann_P_growth ~ L(Ann_P_growth, 1:4))
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.065867 -0.010847  0.005646  0.017617  0.035556
##
## Coefficients:
##                        Estimate Std. Error t value Pr(>|t|)
## (Intercept)            0.006745   0.011337   0.595   0.5568
## L(Ann_P_growth, 1:4)1  1.317152   0.251458   5.238 1.61e-05 ***
## L(Ann_P_growth, 1:4)2 -0.749136   0.420709  -1.781   0.0862 .
## L(Ann_P_growth, 1:4)3  0.353245   0.433174   0.815   0.4219
## L(Ann_P_growth, 1:4)4 -0.164668   0.302110  -0.545   0.5902
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.02666 on 27 degrees of freedom
## Multiple R-squared:  0.7489, Adjusted R-squared:  0.7117
## F-statistic: 20.13 on 4 and 27 DF,  p-value: 8.783e-08
```

```r
BIC(reg44iA, reg44iiA, reg44iiiA, reg44ivA)
```

```
## Warning in BIC.default(reg44iA, reg44iiA, reg44iiiA, reg44ivA): models are not
## all fitted to the same number of observations

##           df       BIC
## reg44iA    3 -139.5969
## reg44iiA   4 -141.8926
## reg44iiiA  5 -134.3214
## reg44ivA   6 -125.8103
```

```r
AIC(reg44iA, reg44iiA, reg44iiiA, reg44ivA)
```

```
## Warning in AIC.default(reg44iA, reg44iiA, reg44iiiA, reg44ivA): models are not
## all fitted to the same number of observations

##           df       AIC
## reg44iA    3 -144.2630
## reg44iiA   4 -147.9981
## reg44iiiA  5 -141.8039
## reg44ivA   6 -134.6047
```

```r
#Choose model 2
```

The second model displays the lowest values for BIC and AIC so Model 2 is preferred.

Quarterly Regression

```r
## Quarter Regression

Quarterly <- read_excel("Chapter4_exercises_data.xls", sheet =2)

ts_P_Quarter <- ts(Quarterly$P, start = 1980.25, frequency =4)

Q_P_growth <- diff(log(ts_P_Quarter))


reg44iq <- dynlm(Q_P_growth ~ L(Q_P_growth,1))
summary(reg44iq)
```

```
##
## Time series regression with "ts" data:
## Start = 1980(4), End = 2011(3)
##
## Call:
## dynlm(formula = Q_P_growth ~ L(Q_P_growth, 1))
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.048408 -0.006184  0.000542  0.006037  0.036627
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)      0.002937   0.001257   2.336   0.0211 *
## L(Q_P_growth, 1) 0.632166   0.070382   8.982 4.03e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 0.0123 on 122 degrees of freedom
## Multiple R-squared:  0.3981, Adjusted R-squared:  0.3931
## F-statistic: 80.68 on 1 and 122 DF,  p-value: 4.034e-15
```

```
reg44iiq <- dynlm(Q_P_growth ~ L(Q_P_growth,1:2))
summary(reg44iiq)
```

```
##
## Time series regression with "ts" data:
## Start = 1981(1), End = 2011(3)
##
## Call:
## dynlm(formula = Q_P_growth ~ L(Q_P_growth, 1:2))
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.048606 -0.005667  0.000845  0.007060  0.031736
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)          0.003500   0.001282   2.731  0.00728 **
## L(Q_P_growth, 1:2)1  0.743849   0.090491   8.220 2.74e-13 ***
## L(Q_P_growth, 1:2)2 -0.174611   0.090360  -1.932  0.05567 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.01222 on 120 degrees of freedom
## Multiple R-squared:  0.4162, Adjusted R-squared:  0.4065
## F-statistic: 42.78 on 2 and 120 DF,  p-value: 9.454e-15
```

```
reg44iiiq <- dynlm(Q_P_growth ~ L(Q_P_growth,1:3))
summary(reg44iiiq)
```

```
##
## Time series regression with "ts" data:
## Start = 1981(2), End = 2011(3)
##
## Call:
## dynlm(formula = Q_P_growth ~ L(Q_P_growth, 1:3))
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.033598 -0.003980  0.000351  0.004362  0.023351
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)          0.0003837  0.0009288   0.413     0.68
## L(Q_P_growth, 1:3)1  0.9018739  0.0641574  14.057  < 2e-16 ***
## L(Q_P_growth, 1:3)2 -0.7623828  0.0808082  -9.434 4.38e-16 ***
## L(Q_P_growth, 1:3)3  0.7633709  0.0664357  11.490  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.008459 on 118 degrees of freedom
```

```
## Multiple R-squared:  0.7245, Adjusted R-squared:  0.7175
## F-statistic: 103.4 on 3 and 118 DF,  p-value: < 2.2e-16
```

```r
reg44ivq <- dynlm(Q_P_growth ~ L(Q_P_growth,1:4))
summary(reg44ivq)
```

```
##
## Time series regression with "ts" data:
## Start = 1981(3), End = 2011(3)
##
## Call:
## dynlm(formula = Q_P_growth ~ L(Q_P_growth, 1:4))
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.034072 -0.003491  0.000696  0.003942  0.021627
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)          9.883e-05  9.383e-04   0.105   0.9163
## L(Q_P_growth, 1:4)1  7.815e-01  9.217e-02   8.479 8.39e-14 ***
## L(Q_P_growth, 1:4)2 -6.370e-01  1.061e-01  -6.001 2.29e-08 ***
## L(Q_P_growth, 1:4)3  6.087e-01  1.081e-01   5.630 1.28e-07 ***
## L(Q_P_growth, 1:4)4  1.745e-01  9.618e-02   1.814   0.0723 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.008413 on 116 degrees of freedom
## Multiple R-squared:  0.7316, Adjusted R-squared:  0.7223
## F-statistic: 79.04 on 4 and 116 DF,  p-value: < 2.2e-16
```

```r
BIC(reg44iq, reg44iiq, reg44iiiq, reg44ivq)
```

```
## Warning in BIC.default(reg44iq, reg44iiq, reg44iiiq, reg44ivq): models are not
## all fitted to the same number of observations
```

```
##           df       BIC
## reg44iq    3 -726.3076
## reg44iiq   4 -718.3639
## reg44iiiq  5 -798.3111
## reg44ivq   6 -789.2122
```

```r
AIC(reg44iq, reg44iiq, reg44iiiq, reg44ivq)
```

```
## Warning in AIC.default(reg44iq, reg44iiq, reg44iiiq, reg44ivq): models are not
## all fitted to the same number of observations
```

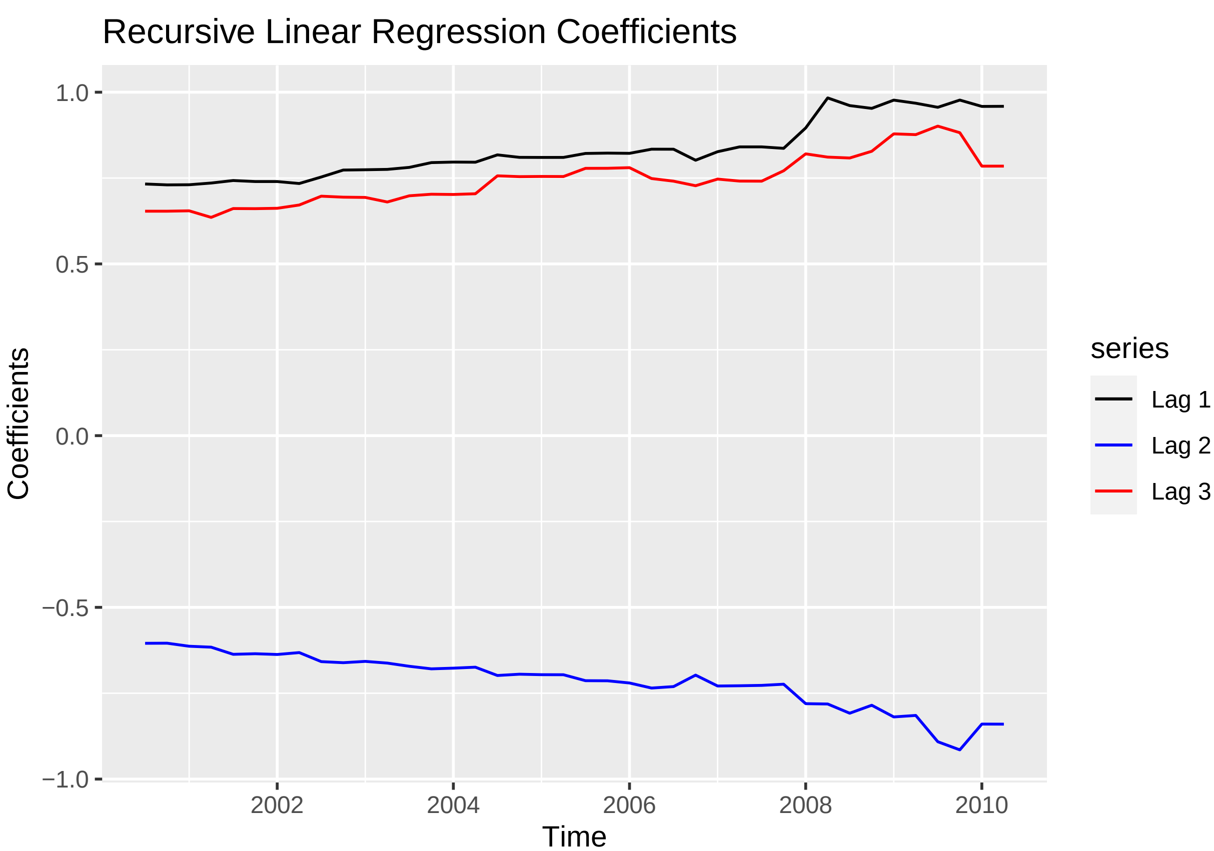
```
##           df       AIC
## reg44iq    3 -734.7684
## reg44iiq   4 -729.6127
## reg44iiiq  5 -812.3312
## reg44ivq   6 -805.9869
```

```r
# Choose model up to 3, reg44iiiq
```

Model 3 has the lowest BIC and AIC, so we would prefer using Model 3 that has up to 3 lags. All of the 3 lags are statistically significant as well. Makes more sense becuase it is quarterly data and the lags have more importance. I will be using Model 3 for the Recursive and Rolling Scheme.

Recursive Scheme*

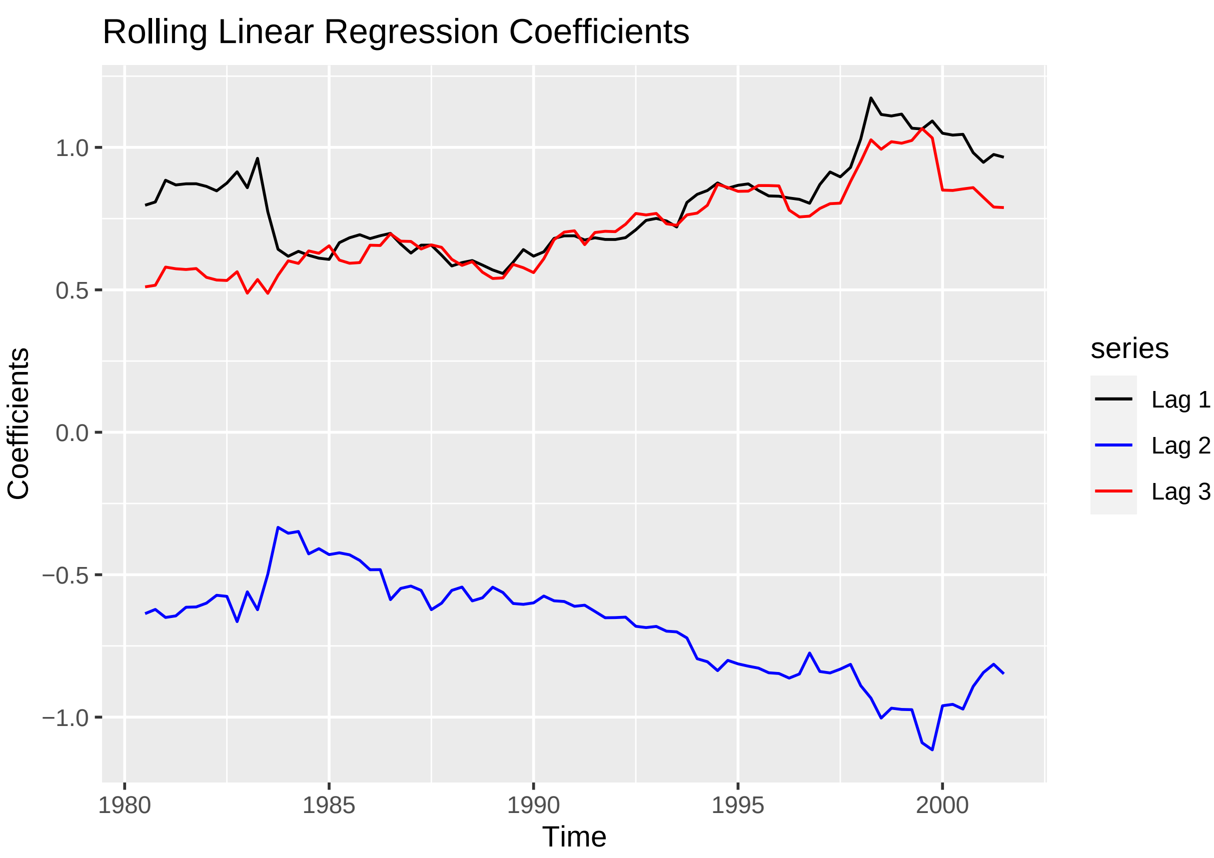
```
recursive <- list()
endwith = 2000.25
loop_length <- NROW(Q_P_growth)-10*8.5


for (i in 1:loop_length) {
  recursive[[i]] <- dynlm(Q_P_growth ~ L(Q_P_growth,1:3), start=1980.5, end=endwith)$coefficients
  endwith <- 2000.25 + 0.25*i
}

recursive_df <- ts(data.frame(matrix(unlist(recursive), nrow=length(recursive), byrow = TRUE)),
          start= 2000.5, frequency = 4)

autoplot(recursive_df[,2], series = "Lag 1") +
  autolayer(recursive_df[,3], series = "Lag 2")  +
  autolayer(recursive_df[,4], series = "Lag 3") +
  ggtitle("Recursive Linear Regression Coefficients") +
  ylab("Coefficients") +
  scale_color_manual(values = c("black", "blue", "red"), breaks = c("Lag 1", "Lag 2", "Lag 3"))
```



There is greater fluctuation in the beginning than it smooths out. There is another set of fluctuation around 2008, but this is because of the housing market crash in 2008. I used 2/3 of the data for the estimation sample so the prediction sample between 2005-2011 was the final 1/3.

Rolling Scheme*

```
rolling <- list()
startwith = 1980.25
loop_length <- NROW(Q_P_growth)-10*4

for (i in 1:loop_length) {
  rolling[[i]] <- dynlm(Q_P_growth ~ L(Q_P_growth,1:3), start=startwith, end=startwith+10)$coef
  startwith <- 1980.25 + 0.25*i
}

rolling_df <- ts(data.frame(matrix(unlist(rolling), nrow=length(rolling), byrow = TRUE)),
          start= 1980.5, frequency = 4)


autoplot(rolling_df[,2], series = "Lag 1") +
  autolayer(rolling_df[,3], series = "Lag 2")  +
  autolayer(rolling_df[,4], series = "Lag 3") +
  ggtitle("Rolling Linear Regression Coefficients") +
  ylab("Coefficients") +
  scale_color_manual(values = c("black", "blue", "red"), breaks = c("Lag 1", "Lag 2", "Lag 3"))
```



The rolling scheme is pretty similar to the recursive scheme but the estimation window is fixed at 40 observations.

***Worked with Carmel Shek**

Problem 4.8:

```r
greenbook <- read_excel("Chapter4_exercises_data.xls", sheet=3)

setnames(greenbook, c('Date', 'actual_growth', 'forecasted_growth'))

greenbook$Date <- as.Date( as.yearqtr(greenbook$Date, format = "%Y:Q%q" ))

greenbook <- greenbook %>%
  mutate(
    forecast_errors = actual_growth-forecasted_growth
  )

errors = ts(greenbook$forecast_errors, start = 1969-01-01, frequency = 4)

regress_forecast_errors <- dynlm(errors ~  L(errors, 1:3) )

summary(regress_forecast_errors)
```

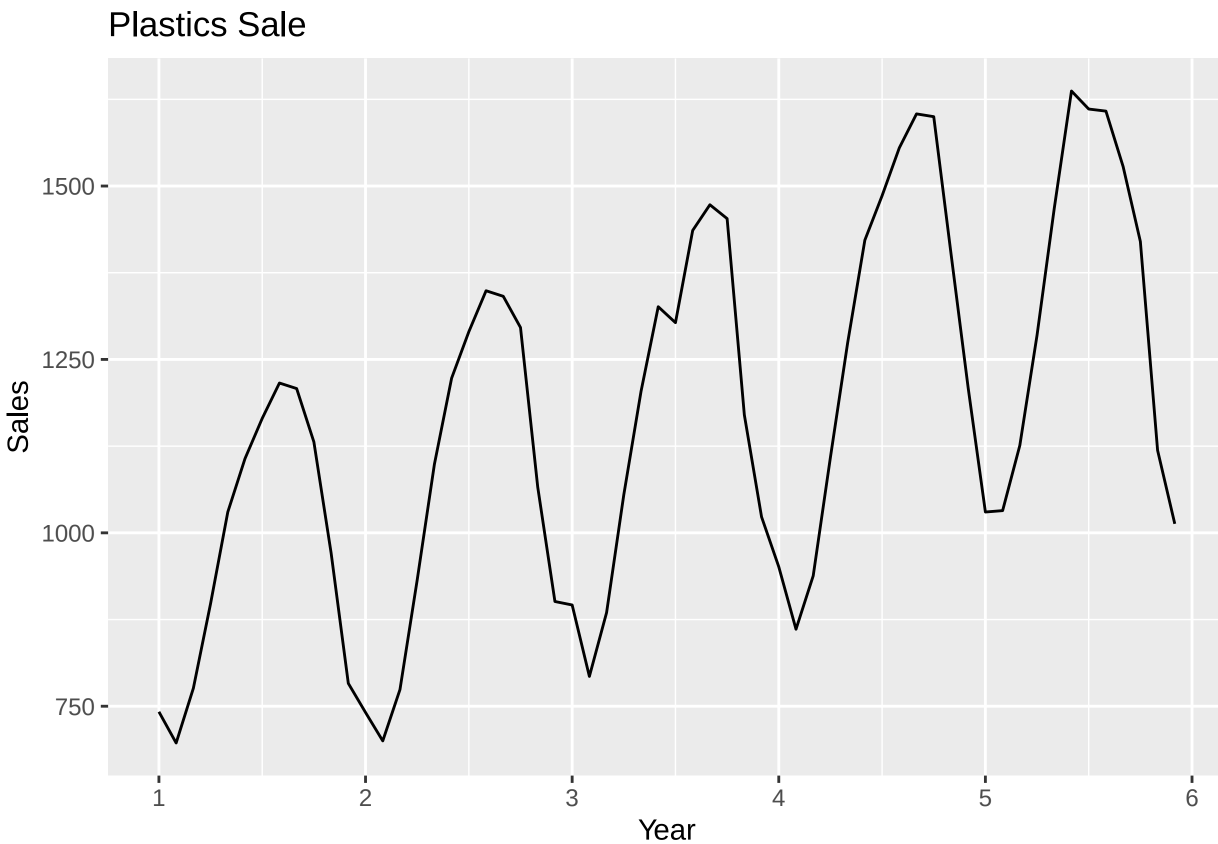
```
##
## Time series regression with "ts" data:
## Start = 1967(4), End = 2004(4)
##
## Call:
## dynlm(formula = errors ~ L(errors, 1:3))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.51624 -0.43181 -0.02828  0.35103  2.91904
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)     0.060552   0.062702   0.966    0.336
## L(errors, 1:3)1 0.083250   0.083178   1.001    0.319
## L(errors, 1:3)2 0.054550   0.082762   0.659    0.511
## L(errors, 1:3)3 0.005634   0.082664   0.068    0.946
##
## Residual standard error: 0.7539 on 145 degrees of freedom
## Multiple R-squared:  0.0108, Adjusted R-squared:  -0.009666
## F-statistic: 0.5277 on 3 and 145 DF,  p-value: 0.6639
```

Ran a regression on the forecast errors with 3 lags. The rgression that was run does a poor job to explain the forecast errors. The F-test is 0.5277 with a p-value of 66%. This shows that none of regressors can explain the forecast errors. So from this, we the forecast error is not predictable from its own past.

Problem 6.2 (Textbook C):

a. Plot the time series of sales of product A. Can you identify seasonal fluctuations and/or a trend-cycle?

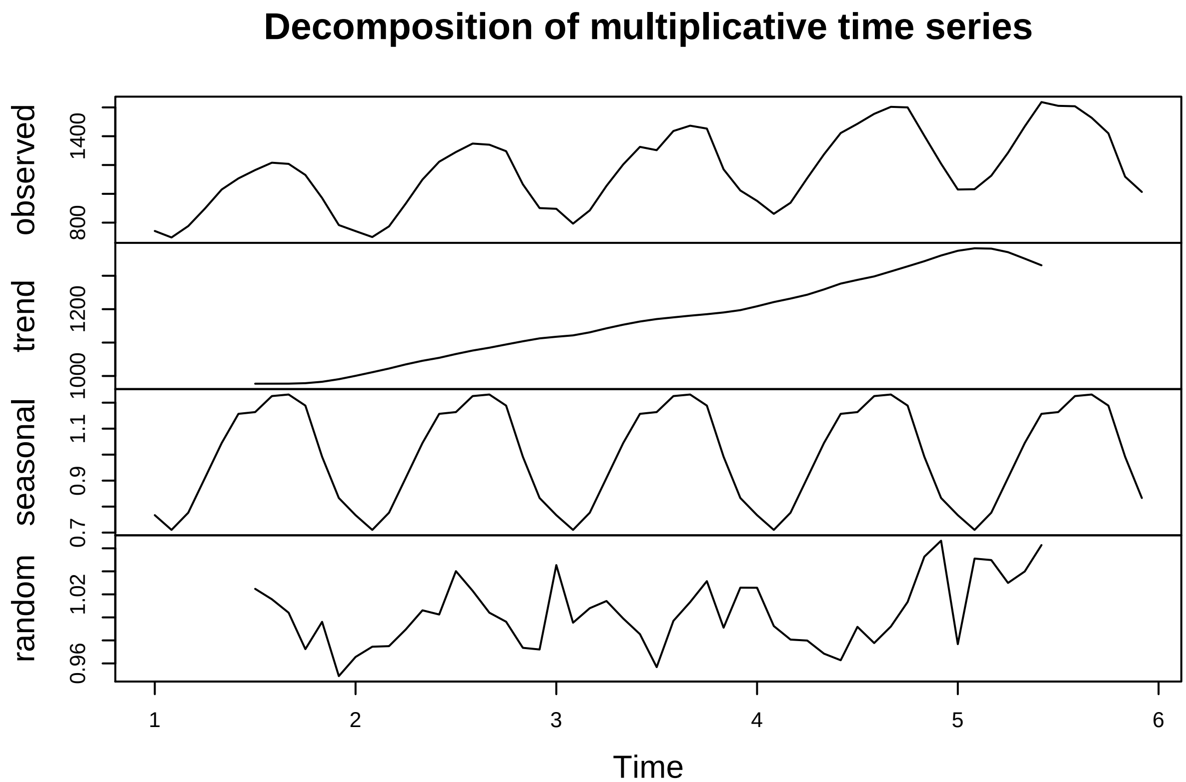
```r
autoplot(plastics) + ggtitle("Plastics Sale") + xlab("Year") + ylab("Sales")
```



I can identify an upward trend and seasonal patterns hitting peaks/ increased sales around August-September and dips/ decreased sales around January at the start of the year.

b. Use a classical multiplicative decomposition to calculate the trend-cycle and seasonal indices.

```
decompose_plastics <- decompose(plastics, "multiplicative")
plot(decompose_plastics)
```




```
# Seasonal Indices
decompose_plastics$seasonal
```

```
##         Jan       Feb       Mar       Apr       May       Jun       Jul
## 1 0.7670466 0.7103357 0.7765294 0.9103112 1.0447386 1.1570026 1.1636317
## 2 0.7670466 0.7103357 0.7765294 0.9103112 1.0447386 1.1570026 1.1636317
## 3 0.7670466 0.7103357 0.7765294 0.9103112 1.0447386 1.1570026 1.1636317
## 4 0.7670466 0.7103357 0.7765294 0.9103112 1.0447386 1.1570026 1.1636317
## 5 0.7670466 0.7103357 0.7765294 0.9103112 1.0447386 1.1570026 1.1636317
##         Aug       Sep       Oct       Nov       Dec
## 1 1.2252952 1.2313635 1.1887444 0.9919176 0.8330834
## 2 1.2252952 1.2313635 1.1887444 0.9919176 0.8330834
## 3 1.2252952 1.2313635 1.1887444 0.9919176 0.8330834
## 4 1.2252952 1.2313635 1.1887444 0.9919176 0.8330834
## 5 1.2252952 1.2313635 1.1887444 0.9919176 0.8330834
```

```
#Trend Indices
decompose_plastics$trend
```

```
##         Jan       Feb       Mar       Apr       May       Jun       Jul
## 1        NA        NA        NA        NA        NA        NA  976.9583
## 2 1000.4583 1011.2083 1022.2917 1034.7083 1045.5417 1054.4167 1065.7917
## 3 1117.3750 1121.5417 1130.6667 1142.7083 1153.5833 1163.0000 1170.3750
## 4 1208.7083 1221.2917 1231.7083 1243.2917 1259.1250 1276.5833 1287.6250
## 5 1374.7917 1382.2083 1381.2500 1370.5833 1351.2500 1331.2500        NA
##         Aug       Sep       Oct       Nov       Dec
## 1  977.0417  977.0833  978.4167  982.7083  990.4167
## 2 1076.1250 1084.6250 1094.3750 1103.8750 1112.5417
## 3 1175.5000 1180.5417 1185.0000 1190.1667 1197.0833
## 4 1298.0417 1313.0000 1328.1667 1343.5833 1360.6250
```

```
## 5         NA         NA         NA         NA         NA
```

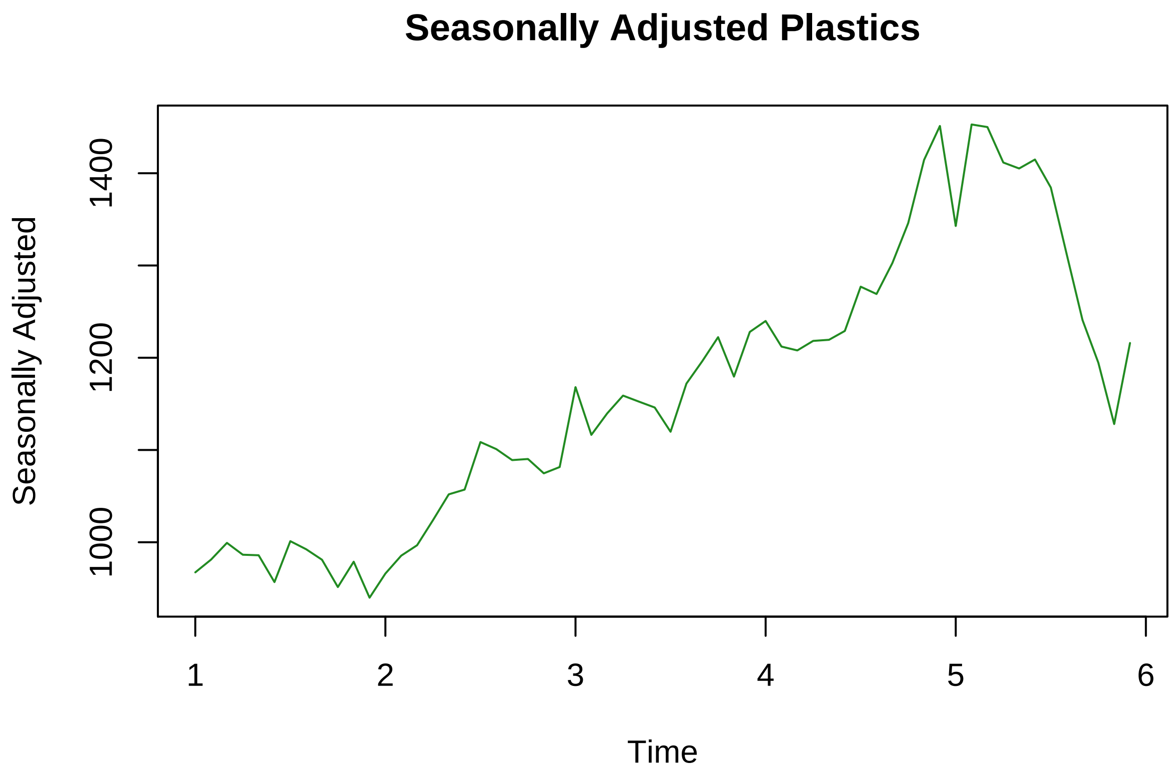
I have displayed the decomposition of the Plastics Sales. And showed the indices for Seasonal and Trend.

c. Do the results support the graphical interpretation from part a?

Yes, the results support the graphical interpretation in part A. The plot of the decomposed shows an upward trend and seasonal fluctuations

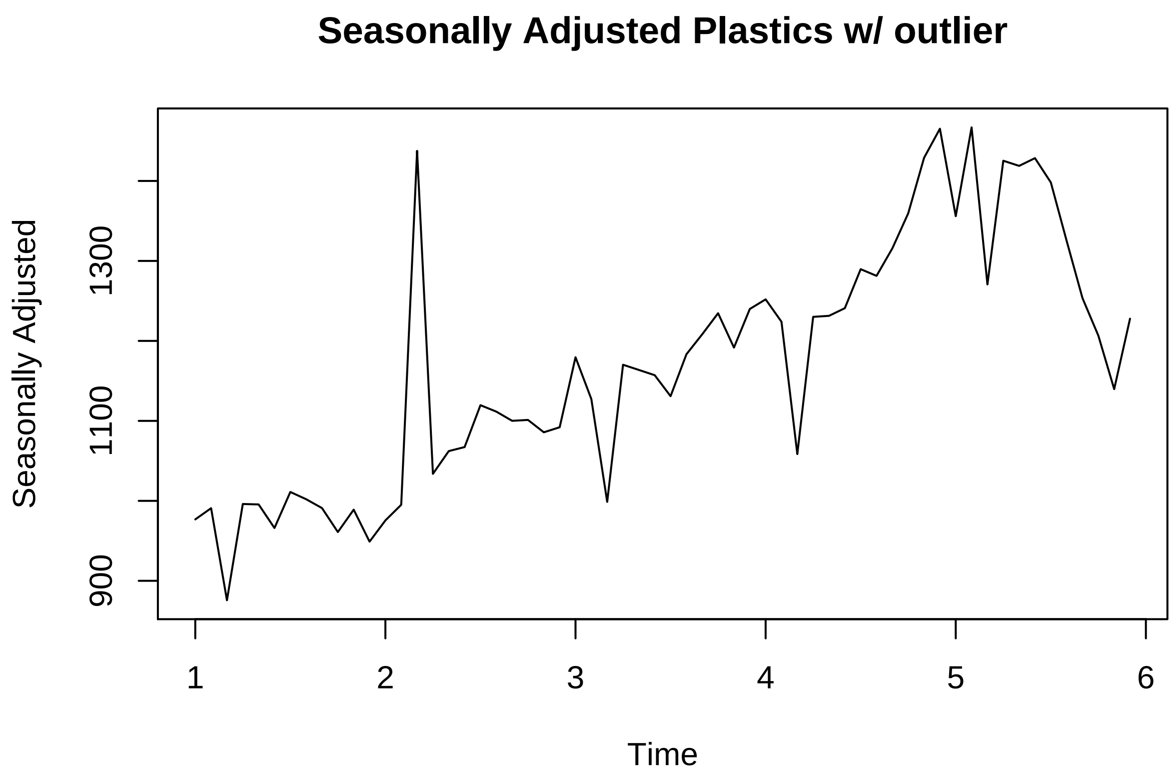
d. Compute and plot the seasonally adjusted data.

```
plastics_seas_adj <- seasadj(decompose_plastics)


plot(plastics_seas_adj, col = "forest green", ylab = "Seasonally Adjusted", xlab = "Time", main = "Seas
```



e. Change one observation to be an outlier (e.g., add 500 to one observation), and recompute the seasonally adjusted data. What is the effect of the outlier?

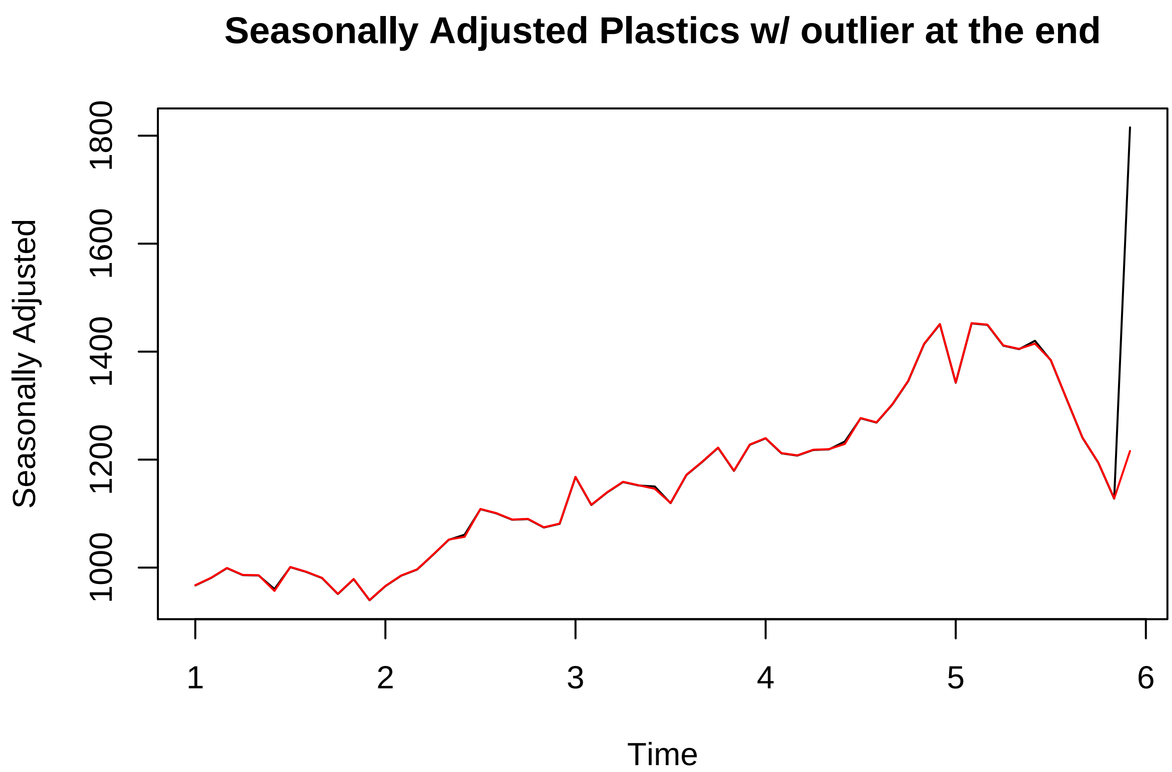
```
plastics2 <- plastics

plastics2[15] <- plastics2[15] + 500

decompose_plastics2 <- decompose(plastics2, "multiplicative")
plastics2_seas_adj <- seasadj(decompose_plastics2)
plot(plastics2_seas_adj, col = "black", ylab = "Seasonally Adjusted", xlab = "Time", main = "Seasonally
```



Added 500 to a value to create an outlier. It created a spike where the outlier was inserted.

f. Does it make any difference if the outlier is near the end rather than in the middle of the time series?

```
plastics3 <- plastics

plastics3[60] <- plastics[60] + 500

decompose_plastics3 <- decompose(plastics3, "multiplicative")
plastics3_seas_adj <- seasadj(decompose_plastics3)

plot(plastics3_seas_adj, col = "black", ylab = "Seasonally Adjusted", xlab = "Time", main = "Seasonally

lines(plastics_seas_adj, col="red", ylab="Seasonally Adjusted")
```



Added an outlier to the end of the time series, and it created a similar spike to the plot. We can see it follows the same pattern as the orignal data, but it has one spike at the end.

Problem 6.4 (Textbook C):

a. Write about 3–5 sentences describing the results of the decomposition. Pay particular attention to the scales of the graphs in making your interpretation.

When isolating the trend component from decomposition, it shows an overall upward trend indicating that the Australian Labour Force has increased over time. Around the early 1990s, it shows there were some dips to the labour force that can indicate a recession. The monthly breakdown shows that a few months had large variation and labour force increased around the months of March and December.

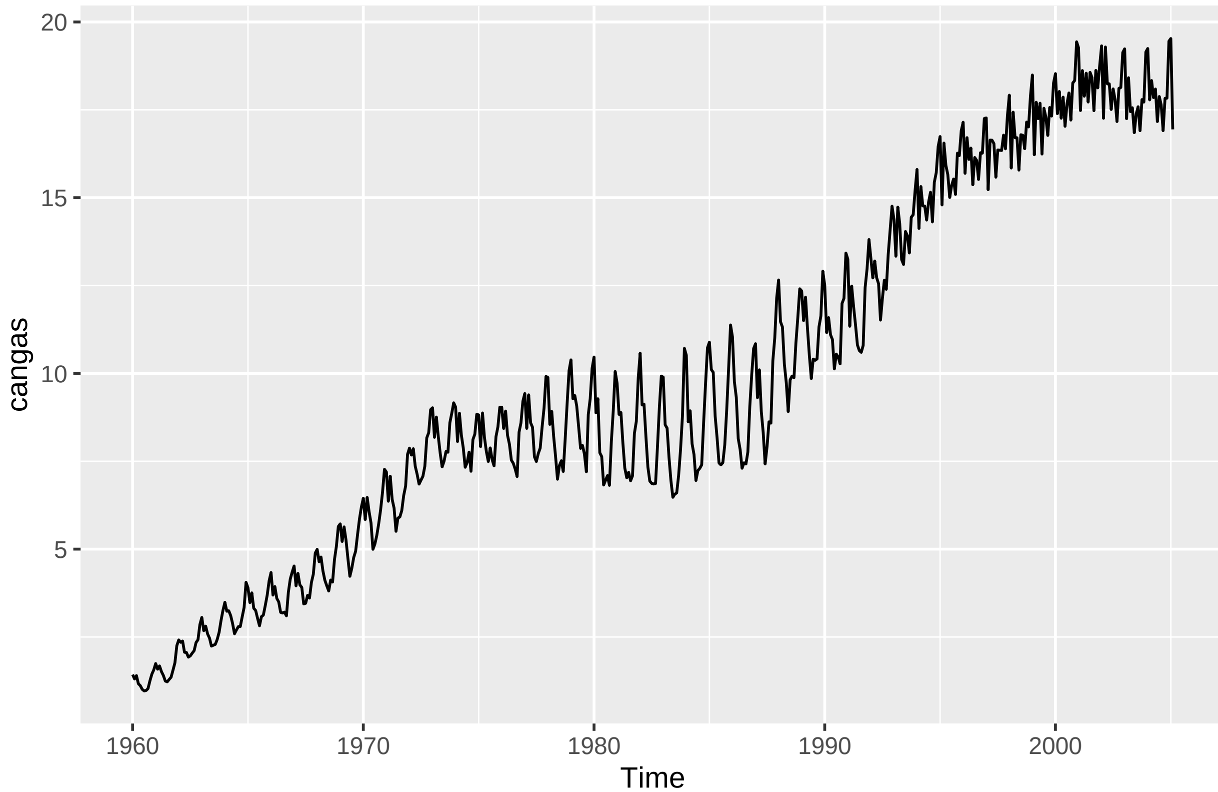
b. Is the recession of 1991/1992 visible in the estimated components?

Yes, the recessions in 1991/1992 are visible in the seasonally adjusted data.

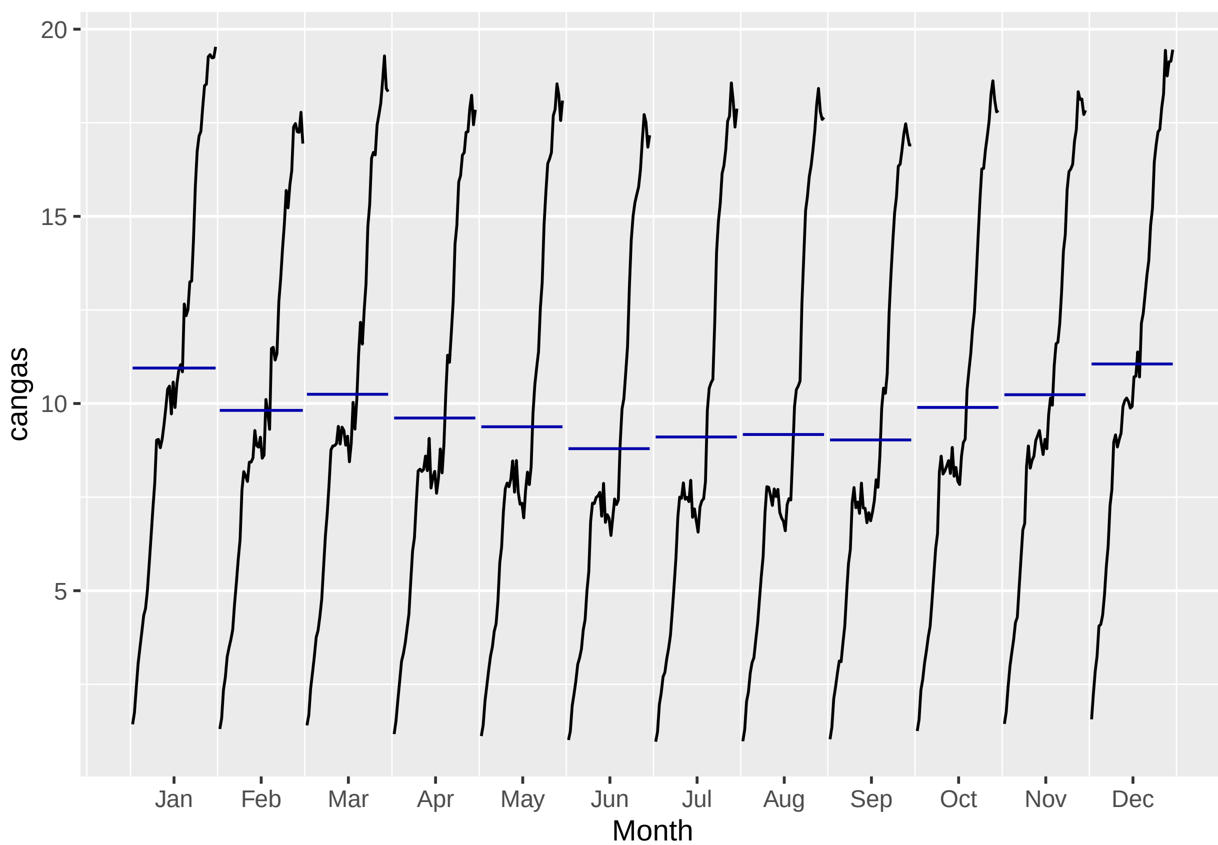
Problem 6.5 (Textbook C):

a. Plot the data using `autoplot()`, `ggsubseriesplot()` and `ggseasonplot()` to look at the effect of the changing seasonality over time. What do you think is causing it to change so much?

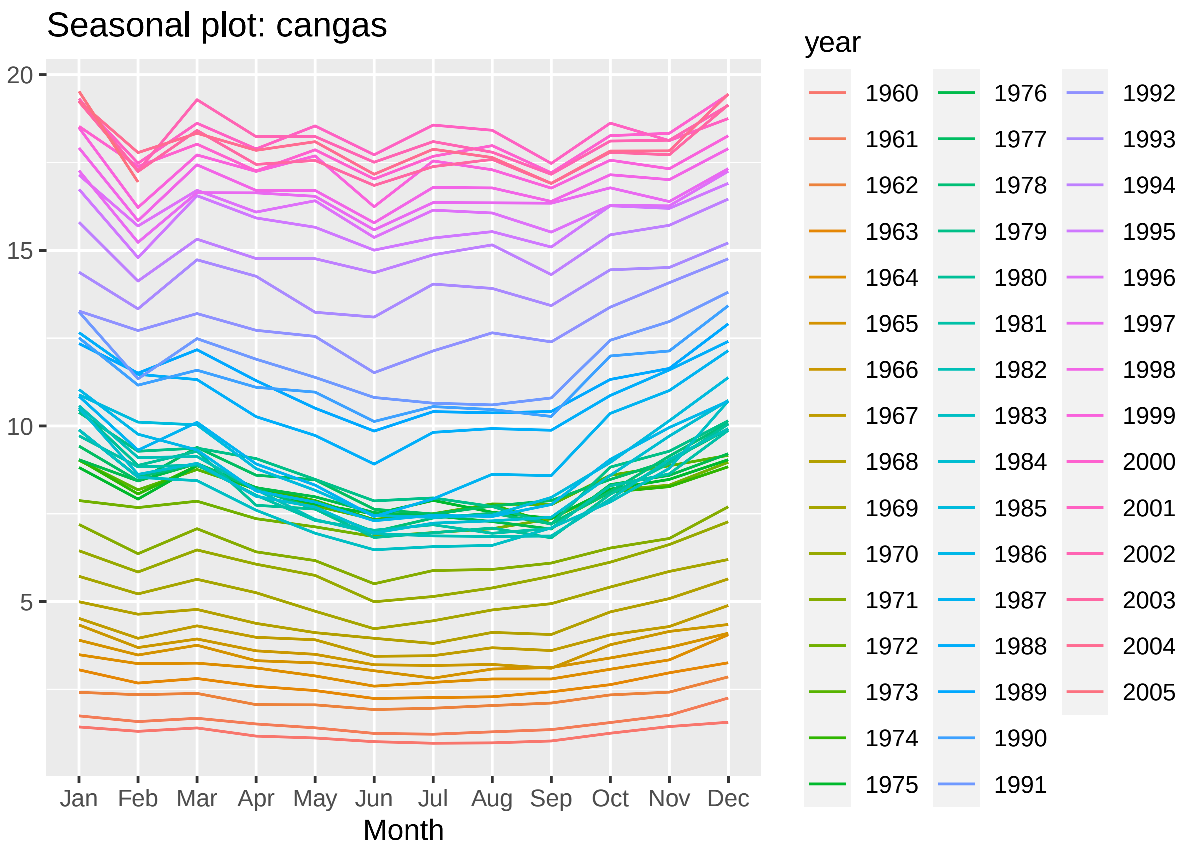
```r
autoplot(cangas)
```




```
ggsubseriesplot(cangas)
```



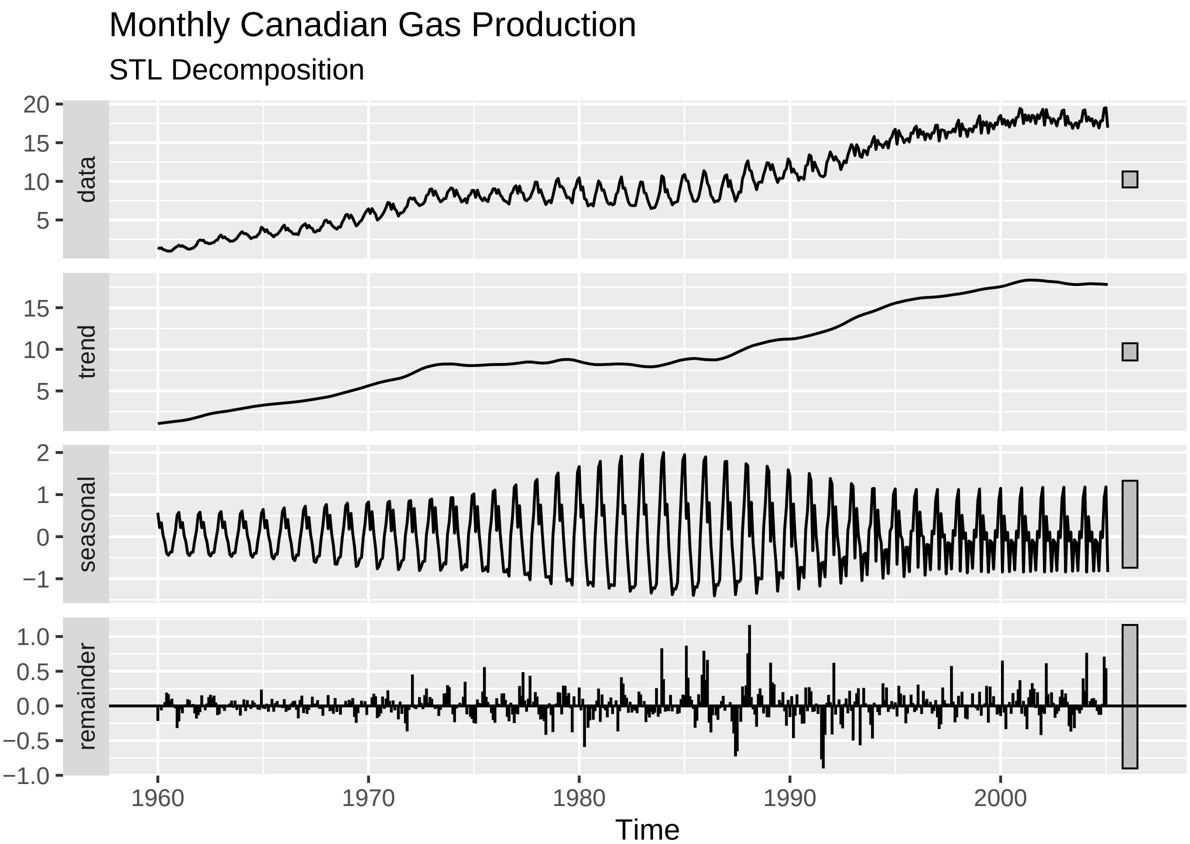
```r
ggseasonplot(cangas)
```



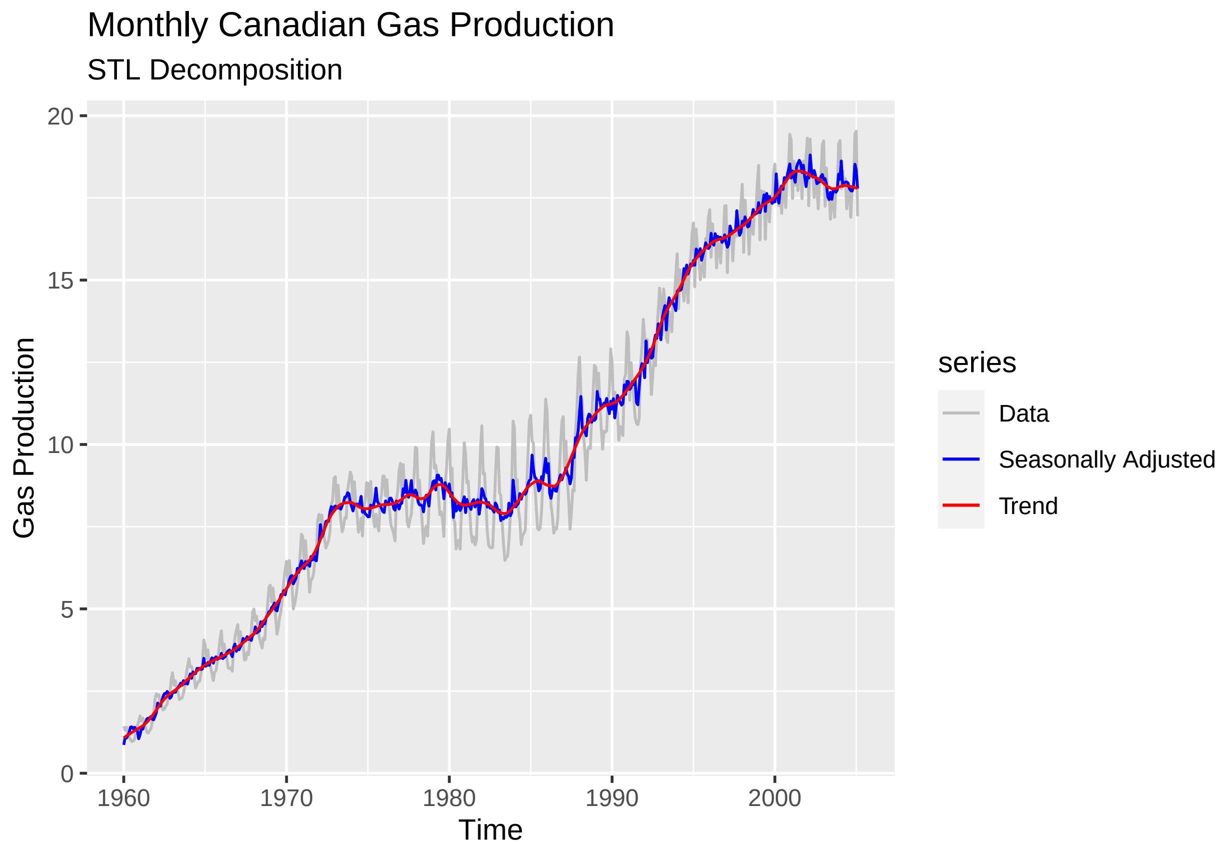
In Winter months like January and Decmeber, gas production increased, but in summer months gas production decreased. Seen by analyzing the `ggsubseriesplot()`. This happens because the demand for gas probably increases during the Winter months because gas is probably used for heat. Gas production overall increased from 1960 to 2005, seen in the `ggseasonplot()`.

b. Do an STL decomposition of the data. You will need to choose s.window to allow for the changing shape of the seasonal component.

```r
stl_cangas <- stl(cangas, s.window = 13, robust = TRUE)

autoplot(stl_cangas) + ggtitle("Monthly Canadian Gas Production", subtitle = "STL Decomposition")
```



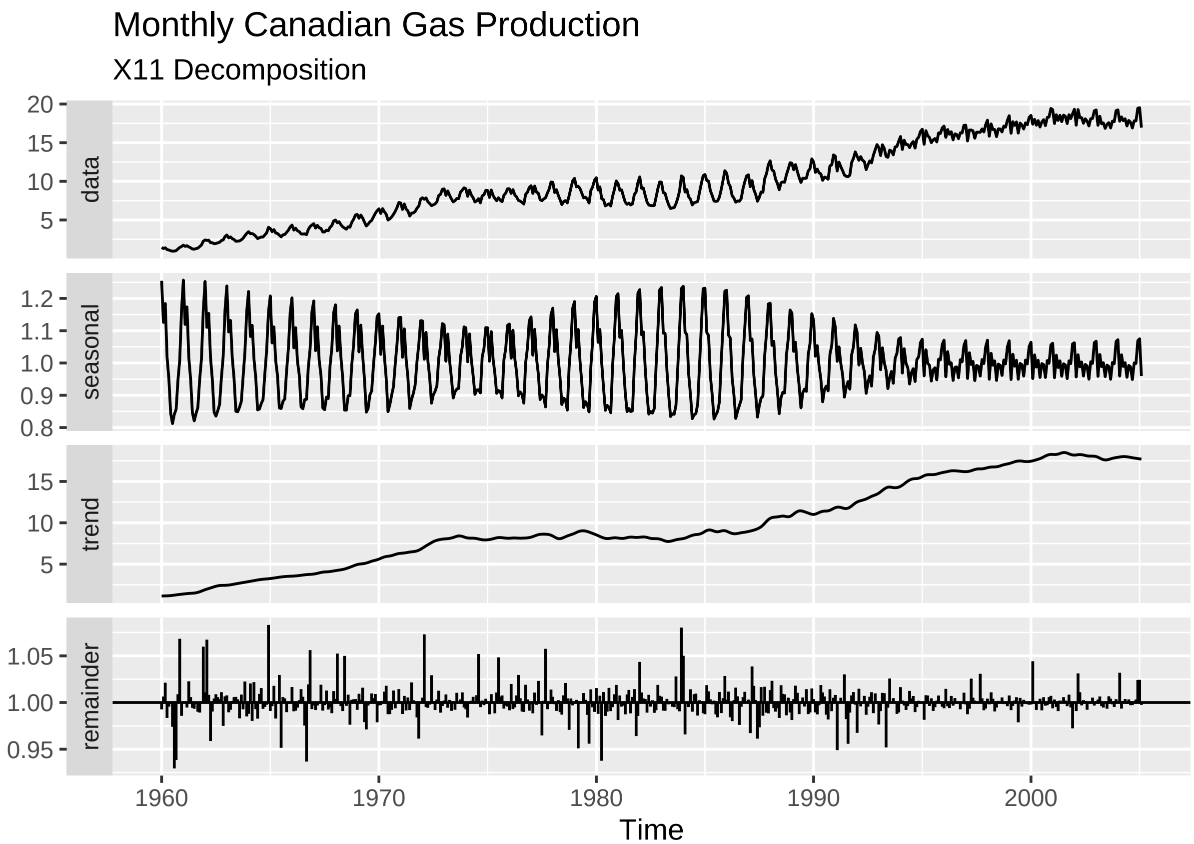

```r
autoplot(cangas, series ="Data") +
  autolayer(seasadj(stl_cangas), series ="Seasonally Adjusted") +
  autolayer(trendcycle(stl_cangas), series = "Trend" ) +
  ylab("Gas Production") +
  ggtitle("Monthly Canadian Gas Production", subtitle = "STL Decomposition") +
  scale_color_manual( values = c("grey", "blue", "red"), breaks = c("Data", "Seasonally Adjusted", "Tren
```



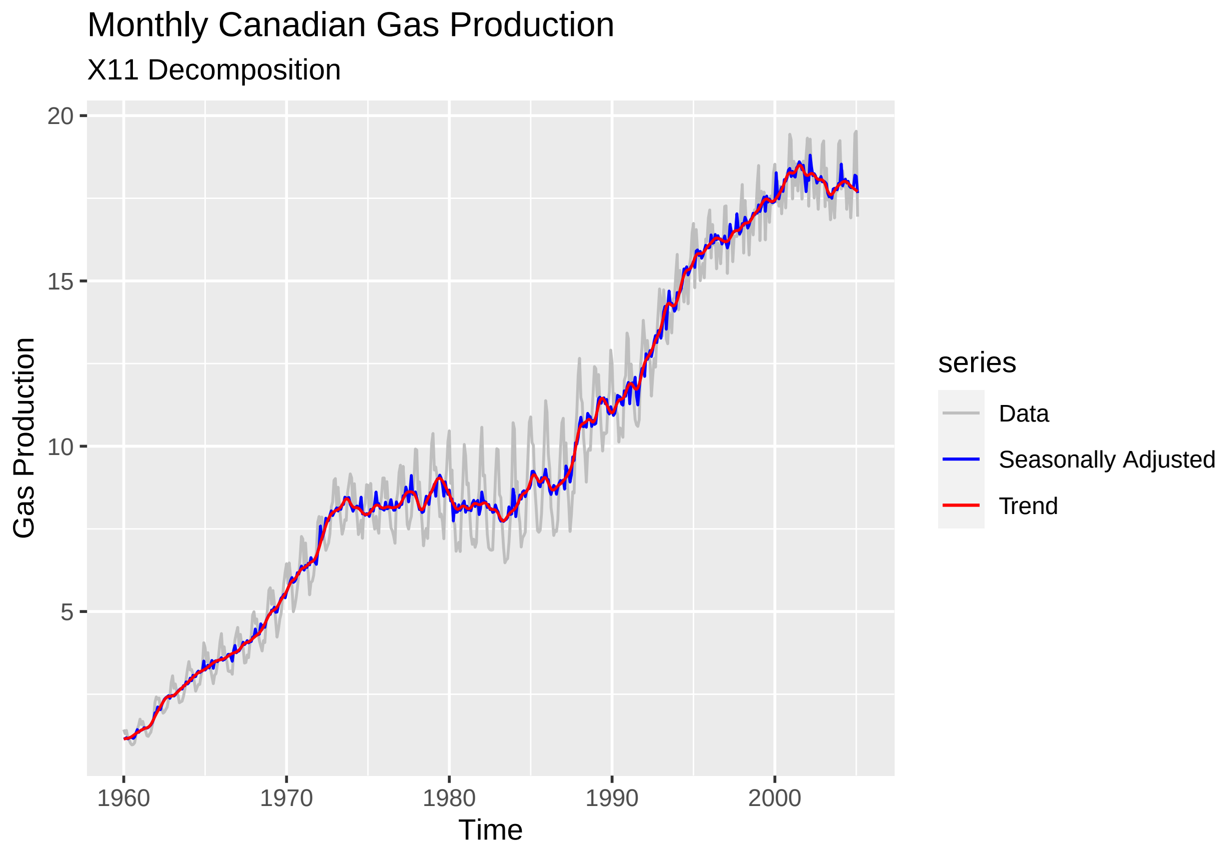
Shows an upward trend, and increase in seasonality in the 1980s.

c. Compare the results with those obtained using SEATS and X11. How are they different?

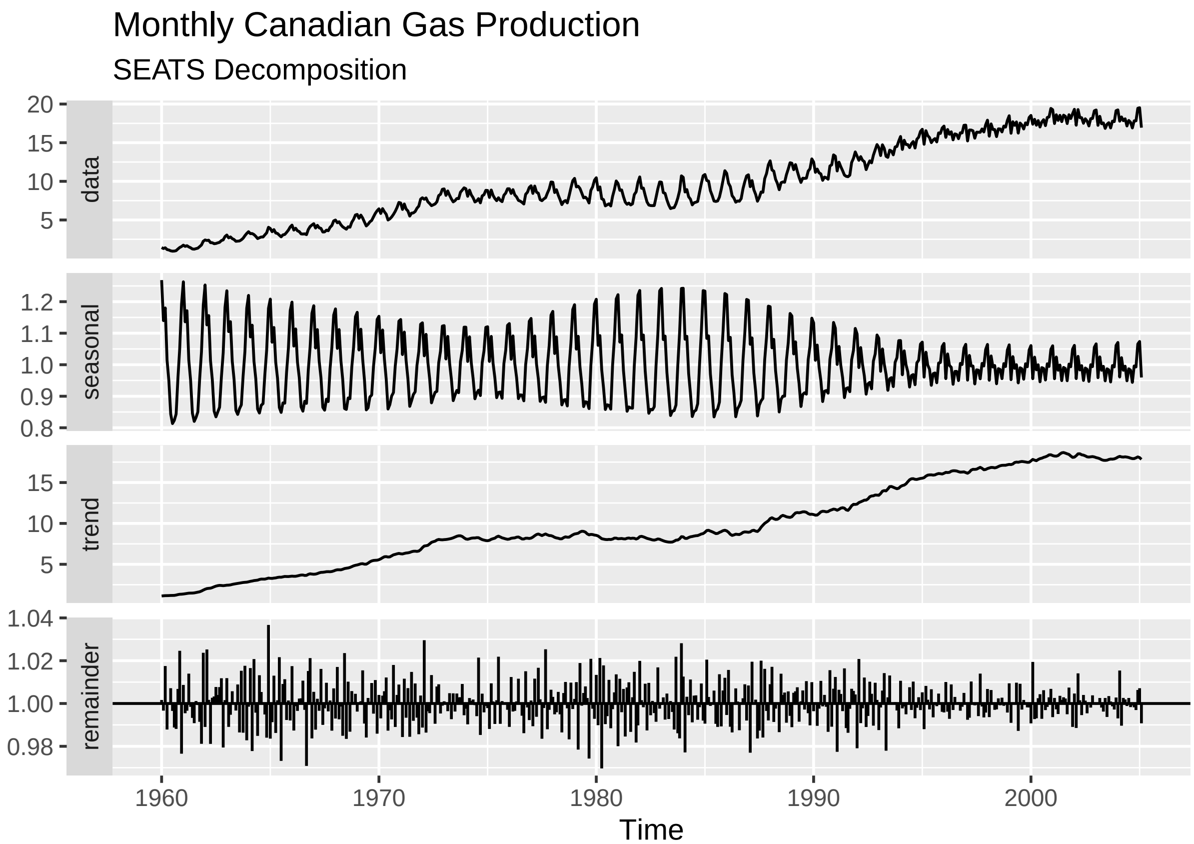
```
# X11
x11_cangas <- seas(cangas, x11 = "")
autoplot(x11_cangas) + ggtitle("Monthly Canadian Gas Production", subtitle = "X11 Decomposition")
```



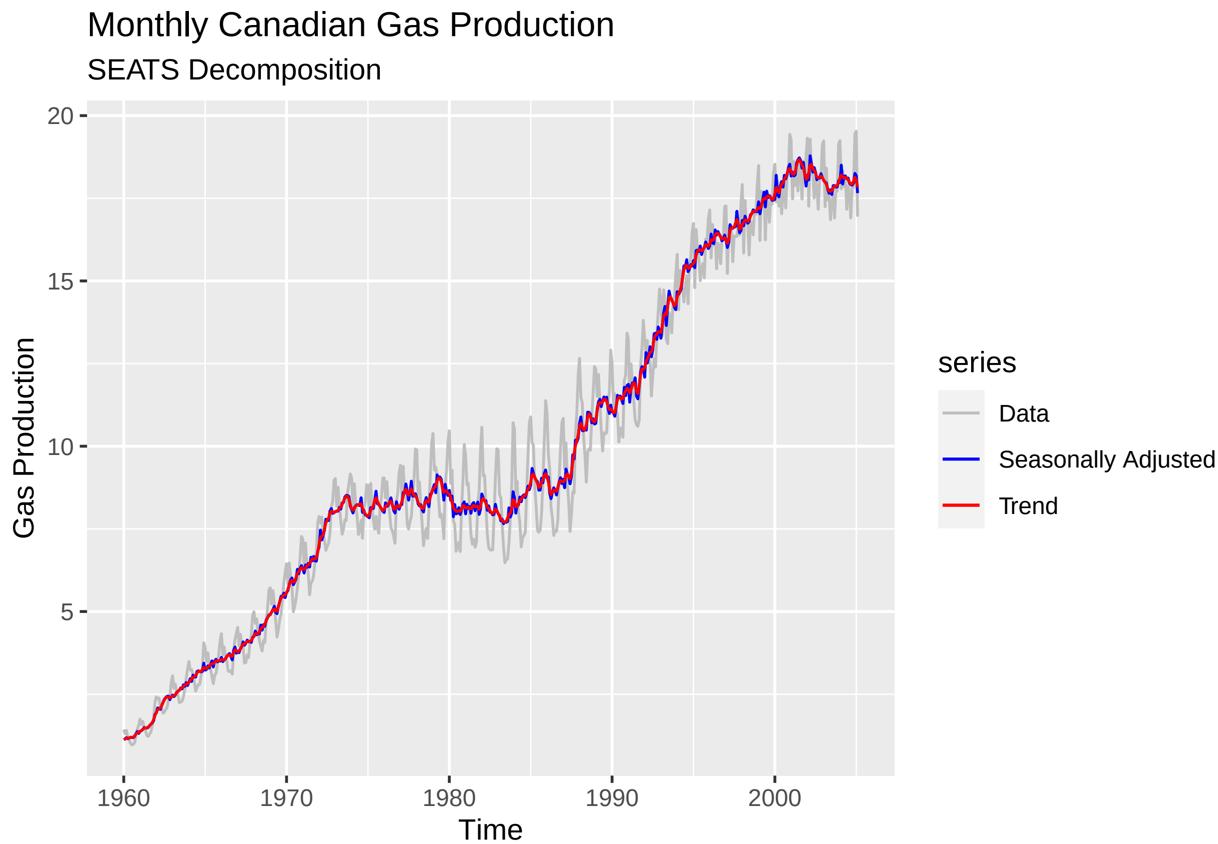

```
autoplot(cangas, series ="Data") +
  autolayer(seasadj(x11_cangas), series ="Seasonally Adjusted") +
  autolayer(trendcycle(x11_cangas), series = "Trend" ) +
  ylab("Gas Production") +
  ggtitle("Monthly Canadian Gas Production", subtitle = "X11 Decomposition") +
  scale_color_manual( values = c("grey", "blue", "red"), breaks = c("Data", "Seasonally Adjusted", "Tren
```




```
#SEATS

seats_cangas <- seas(cangas)
autoplot(seats_cangas) + ggtitle("Monthly Canadian Gas Production", subtitle = "SEATS Decomposition")
```




```r
autoplot(cangas, series = "Data") +
  autolayer(seasadj(seats_cangas), series = "Seasonally Adjusted") +
  autolayer(trendcycle(seats_cangas), series = "Trend") +
  ggtitle("Monthly Canadian Gas Production", subtitle = "SEATS Decomposition") +
  ylab("Gas Production") +
  scale_color_manual(values = c("gray", "blue", "red"), breaks = c("Data", "Seasonally Adjusted", "Trend
```

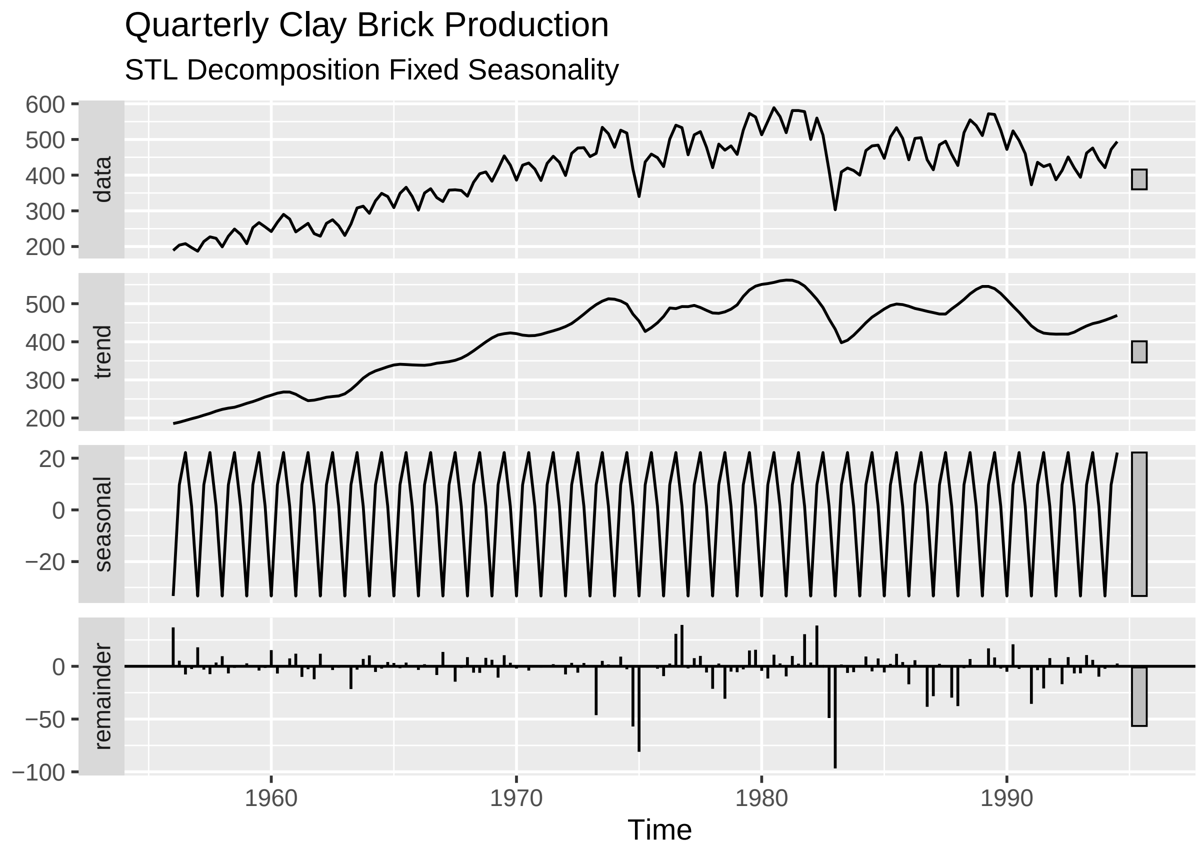


The X11 and SEATS Decompositions are similar, and in the STL decomposition, the seasonally adjusted has more variance compared the X11 and SEATS Decompositions.

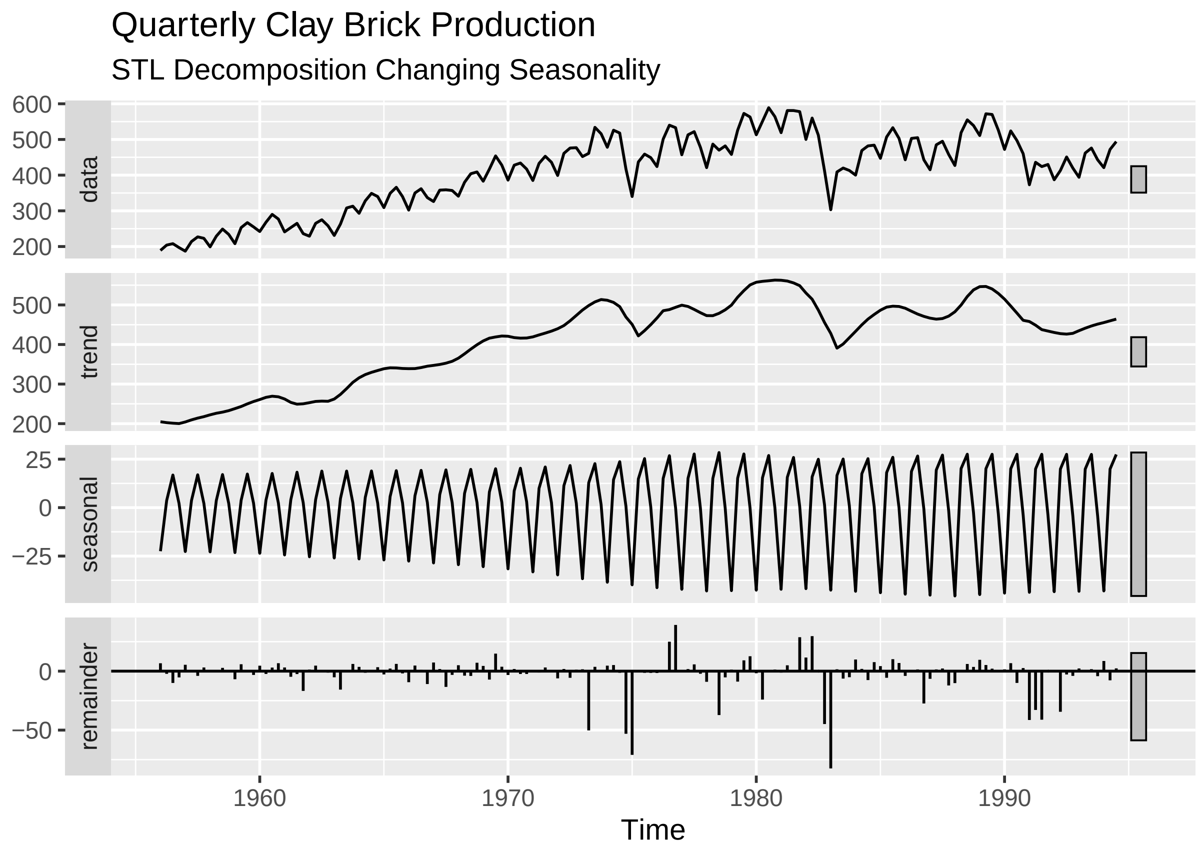
Problem 6.6 (Textbook C):

a. Use an STL decomposition to calculate the trend-cycle and seasonal indices. (Experiment with having fixed or changing seasonality.)

```
stl_fixed <- stl(bricksq, s.window = "periodic", robust = TRUE)
autoplot(stl_fixed) + ggtitle("Quarterly Clay Brick Production" , subtitle = "STL Decomposition Fixed S
```



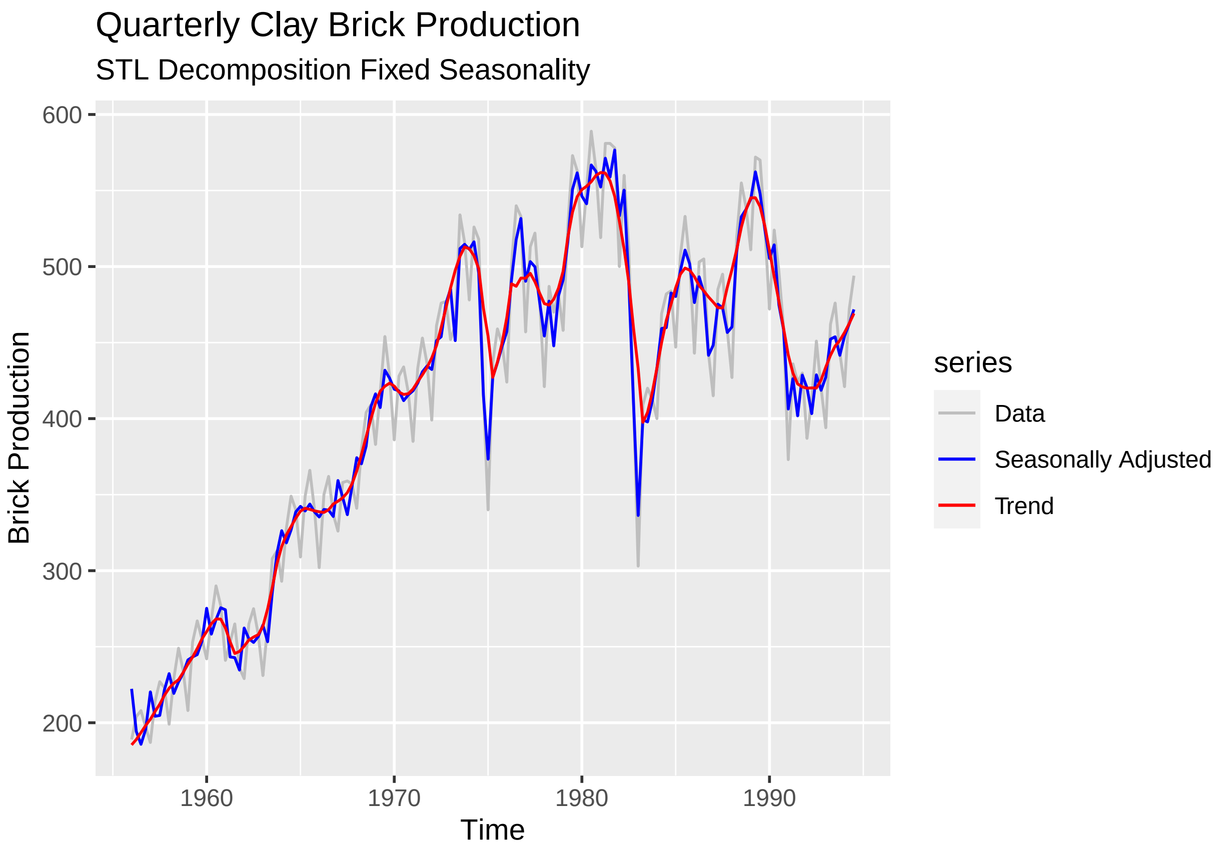

```
stl_change <- stl(bricksq, s.window = 13, robust = TRUE)
autoplot(stl_change) + ggtitle("Quarterly Clay Brick Production" , subtitle = "STL Decomposition Changi
```



It is evident that the seasonality changes when it is fixed vs when it is changing.

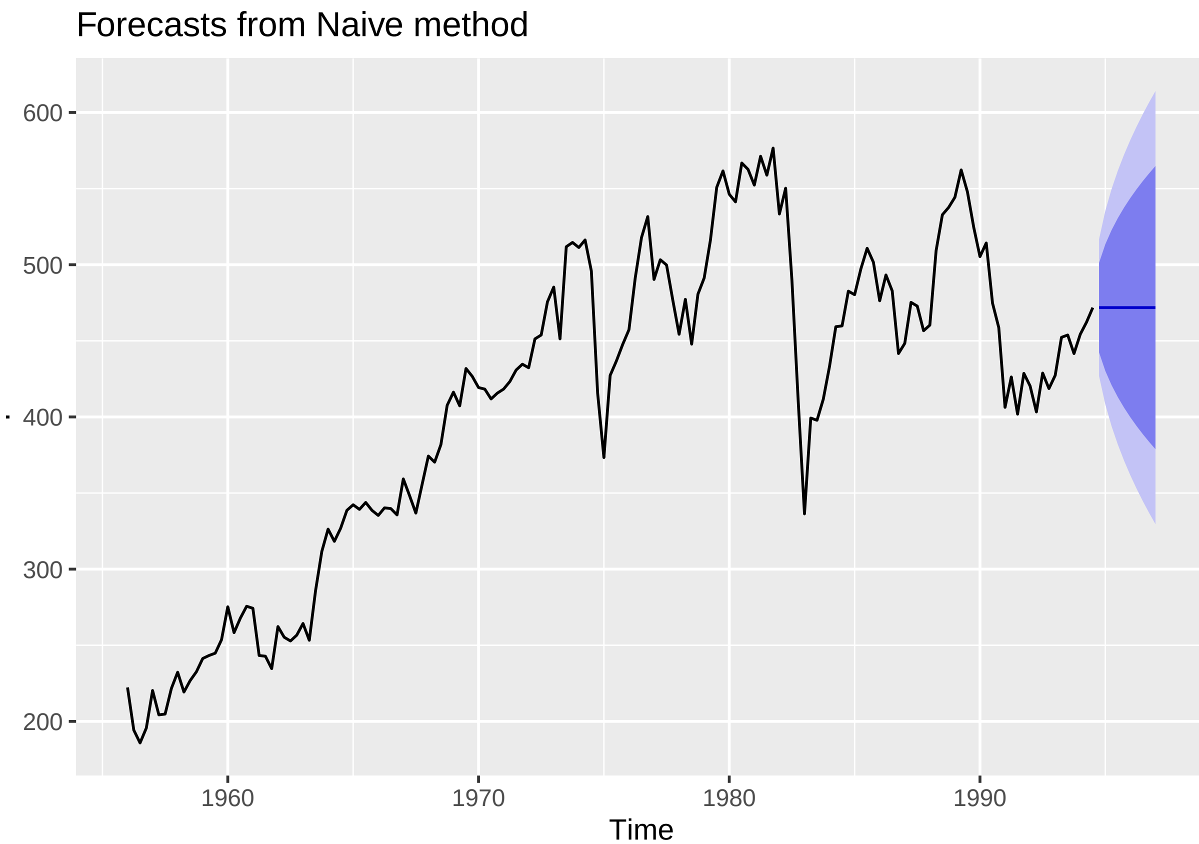
b. Compute and plot the seasonally adjusted data.

```
brick_stl_adj <- seasadj(stl_fixed)

autoplot(bricksq, series = "Data") +
  autolayer(brick_stl_adj, series = "Seasonally Adjusted") +
  autolayer(trendcycle(stl_fixed), series = "Trend") +
  ggtitle("Quarterly Clay Brick Production", subtitle = "STL Decomposition Fixed Seasonality") +
  ylab("Brick Production") +
  scale_color_manual(values = c("gray", "blue", "red"), breaks = c("Data", "Seasonally Adjusted", "Trend
```



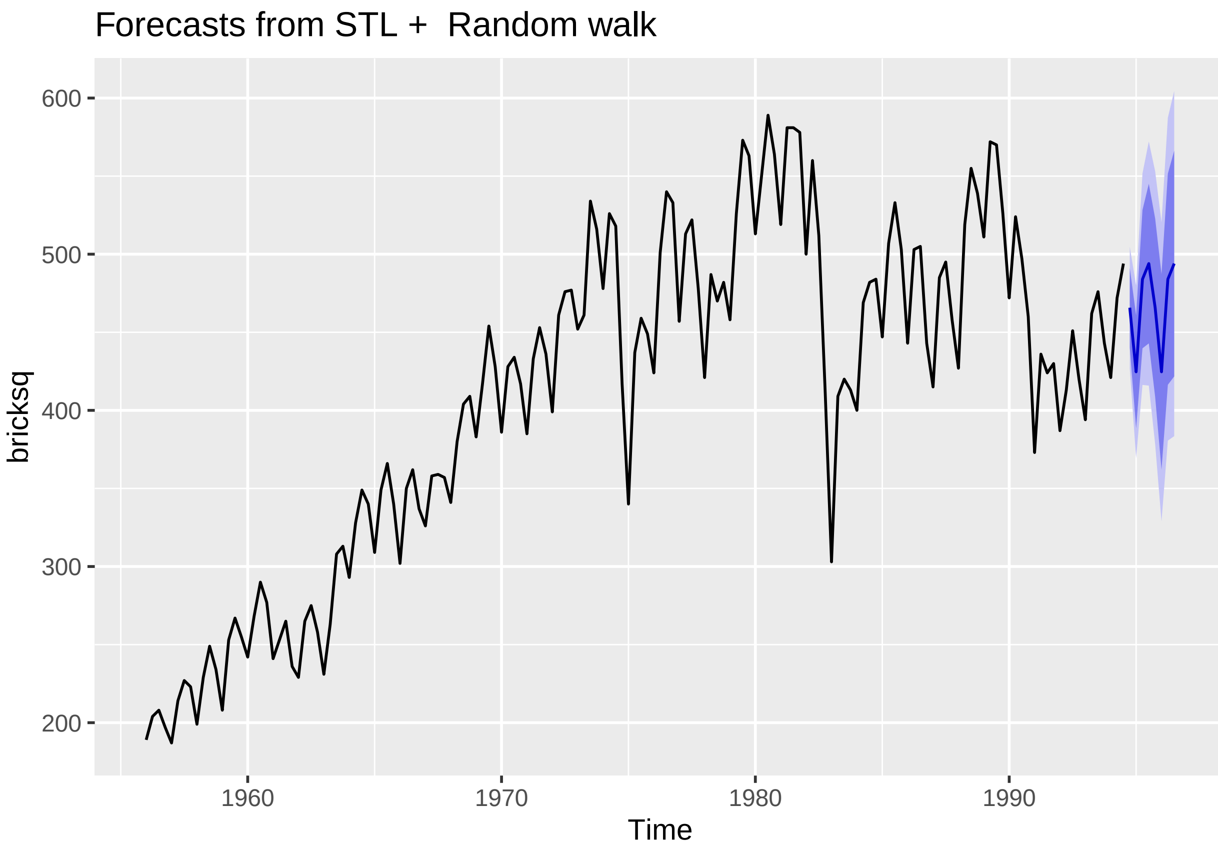
c. Use a naïve method to produce forecasts of the seasonally adjusted data.

```
brick_stl_adj %>% naive() %>% autoplot()
```



d. Use stlf() to reseasonalise the results, giving forecasts for the original data.

```
fcast_brick <- stlf(bricksq, method = "naive")
autoplot(fcast_brick)
```



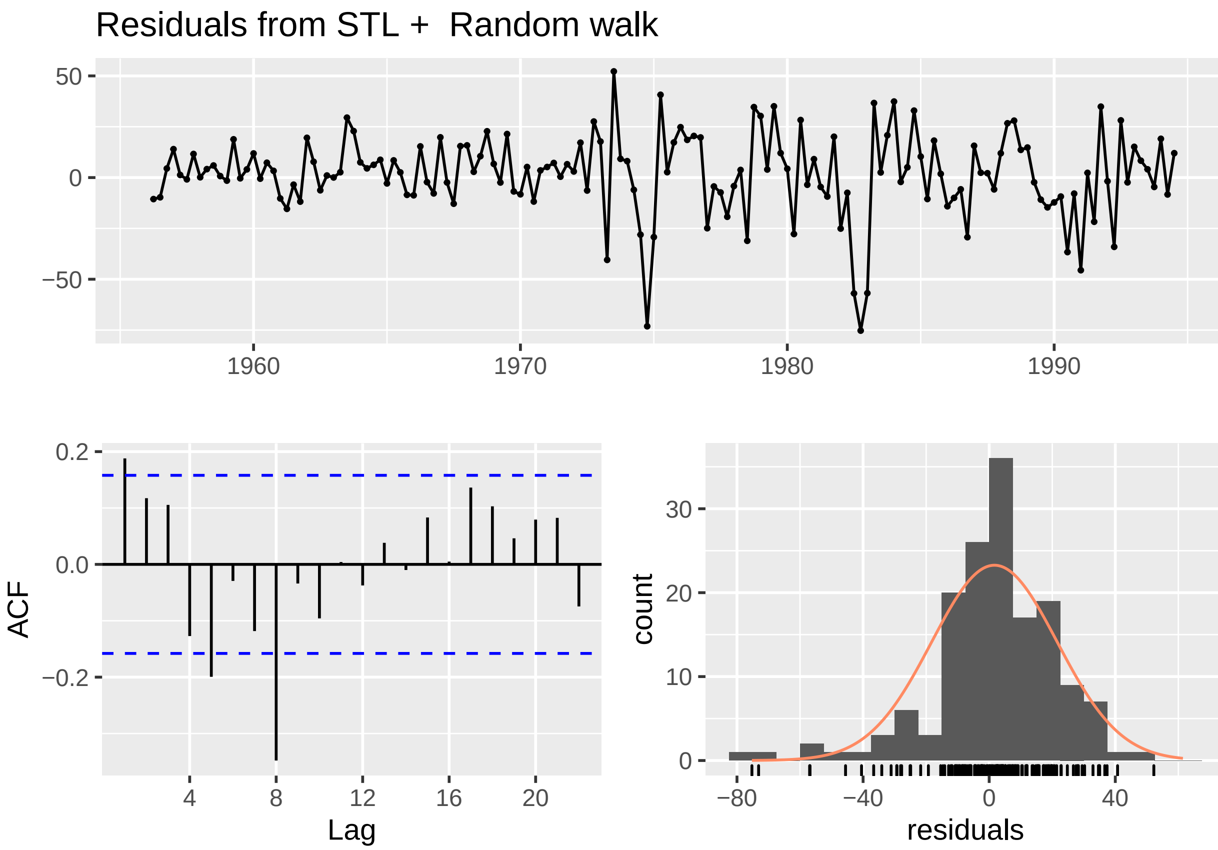

```
forecast(fcast_brick)
```

```
##         Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## 1994 Q4       465.7338 440.1878 491.2798 426.6645 504.8030
## 1995 Q1       424.6739 388.5464 460.8014 369.4217 479.9261
## 1995 Q2       483.9926 439.7456 528.2395 416.3227 551.6624
## 1995 Q3       494.0000 442.9080 545.0920 415.8616 572.1384
## 1995 Q4       465.7338 408.6112 522.8563 378.3723 553.0952
## 1996 Q1       424.6739 362.0993 487.2485 328.9742 520.3736
## 1996 Q2       483.9926 416.4042 551.5809 380.6251 587.3600
## 1996 Q3       494.0000 421.7450 566.2550 383.4956 604.5044
```

e. Do the residuals look uncorrelated?

```
checkresiduals(fcast_brick)
```

```
## Warning in checkresiduals(fcast_brick): The fitted degrees of freedom is based
## on the model used for the seasonally adjusted data.
```

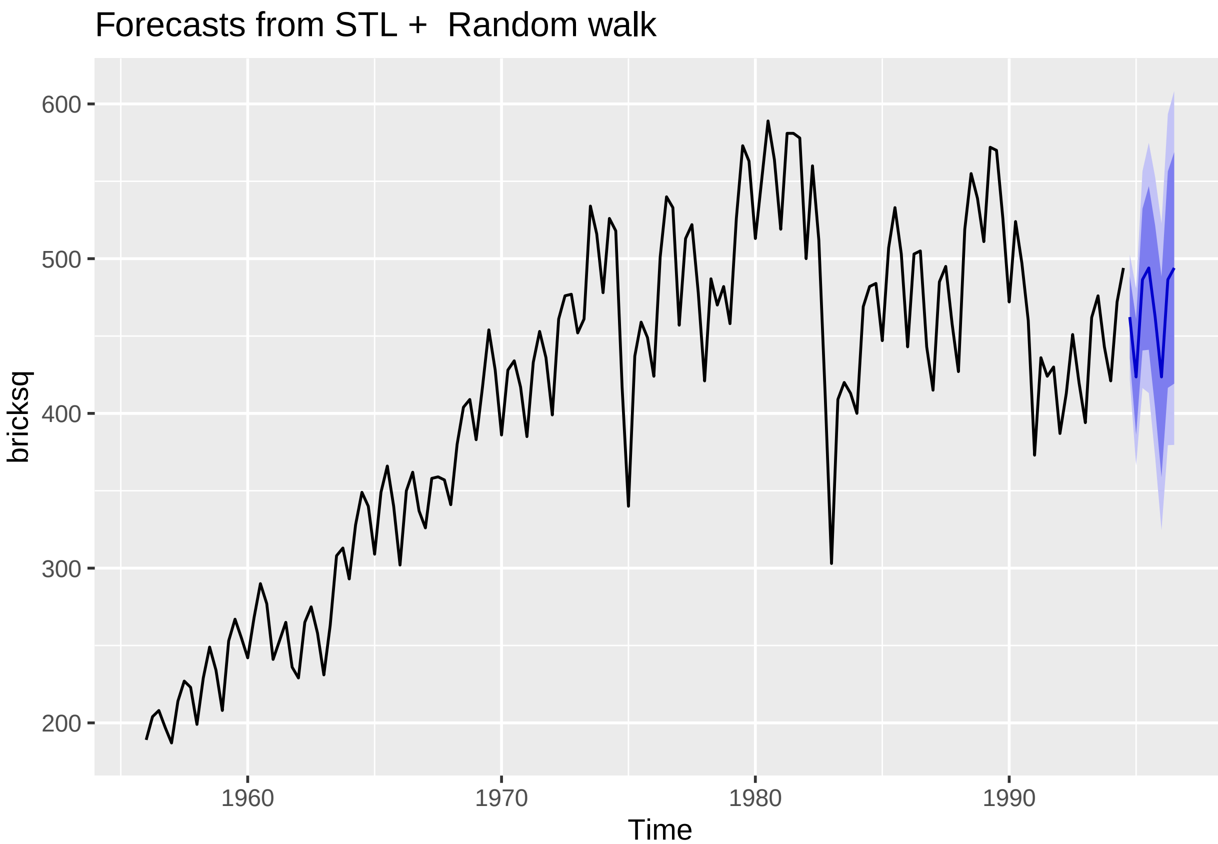



```
##
##  Ljung-Box test
##
## data:  Residuals from STL +  Random walk
## Q* = 40.829, df = 8, p-value = 2.244e-06
##
## Model df: 0.   Total lags used: 8
```

Looking at the acf and the plot, we can tell the residuals are correlated especially at lag 1, 4, and 8.

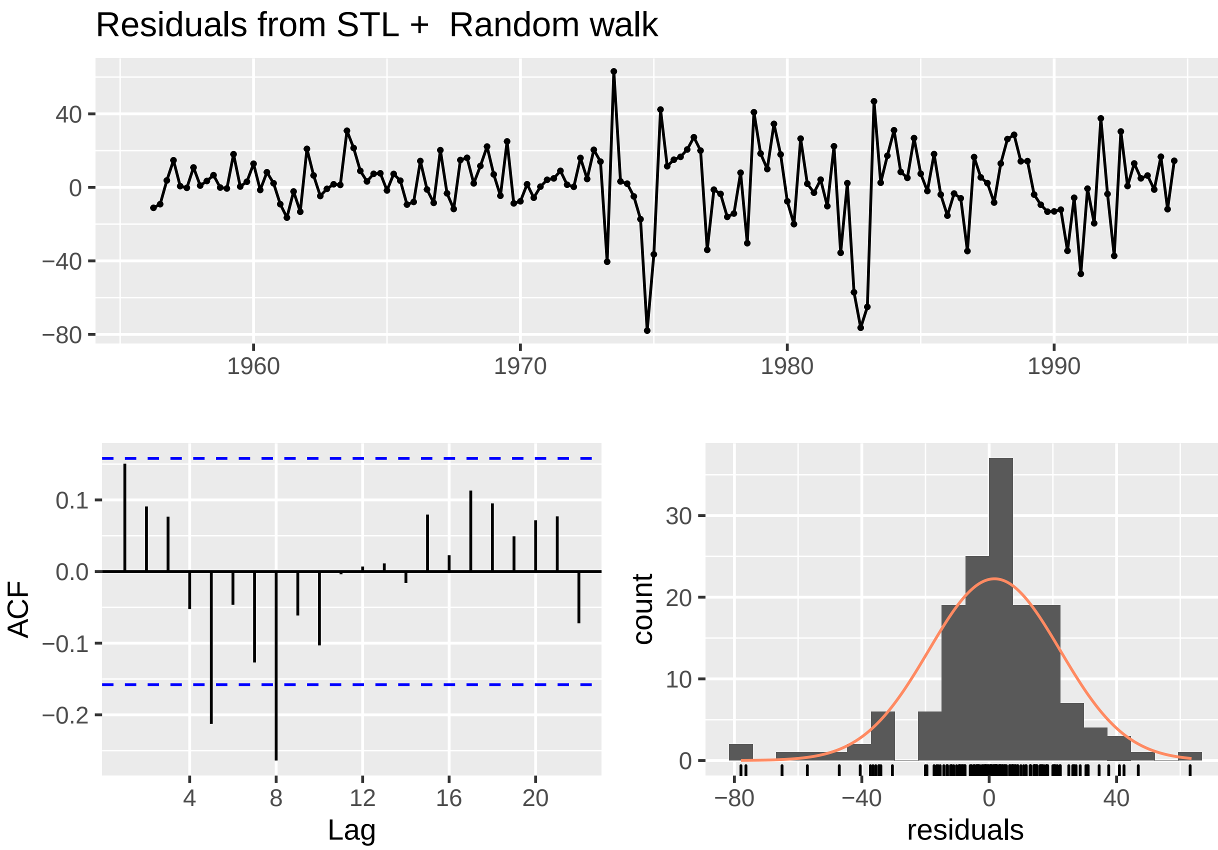
f. Repeat with a robust STL decomposition. Does it make much difference?

```
stlf_brick_robust <- stlf(bricksq, method = "naive", robust = TRUE)
autoplot(stlf_brick_robust)
```




```
checkresiduals(stlf_brick_robust)
```

```
## Warning in checkresiduals(stlf_brick_robust): The fitted degrees of freedom is
## based on the model used for the seasonally adjusted data.
```




```
##
##  Ljung-Box test
##
## data:  Residuals from STL +  Random walk
## Q* = 27.961, df = 8, p-value = 0.0004817
##
## Model df: 0.   Total lags used: 8
```

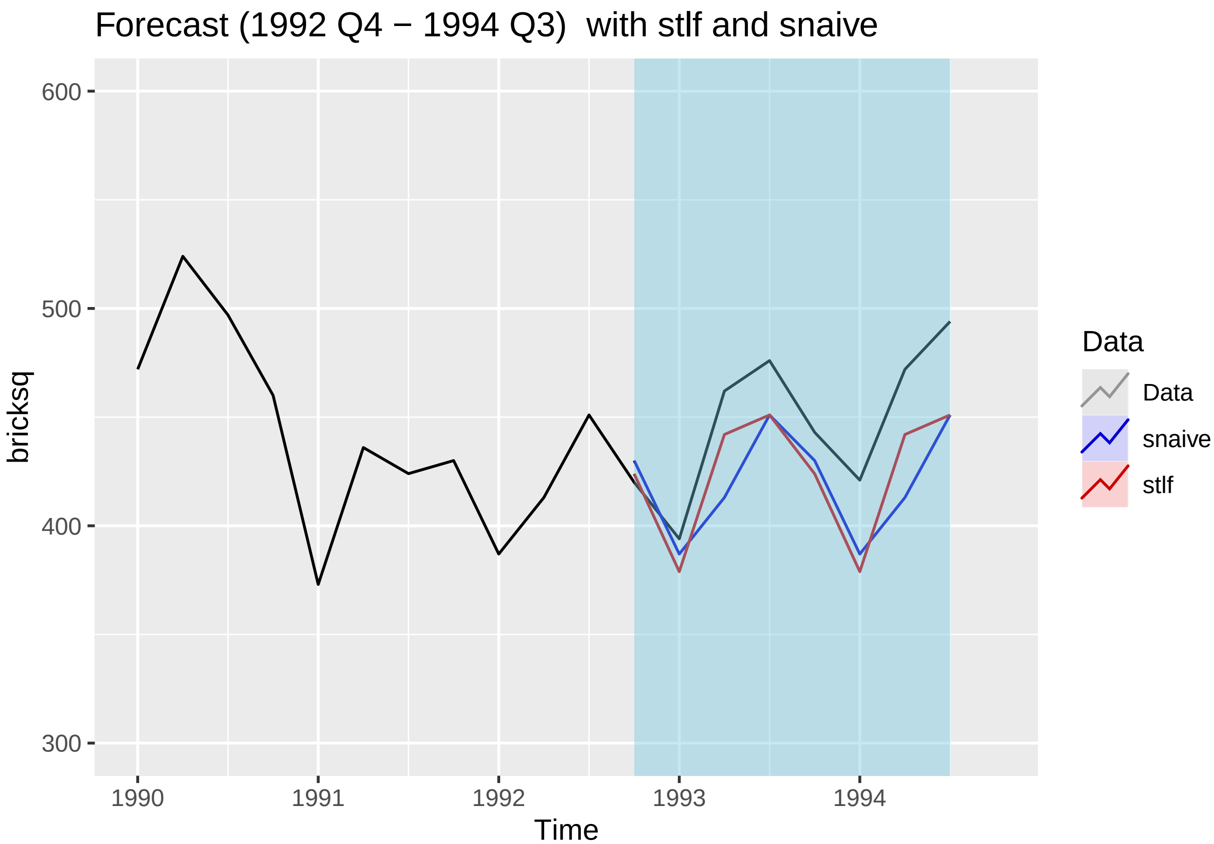
the plots are the same, but the correlations are smaller and less.

g. Compare forecasts from stlf() with those from snaive(), using a test set comprising the last 2 years of data. Which is better?

```
train_brick <- window(bricksq, end= c(1992,3))

test_brick <- window(bricksq, start = c(1992,4))

stlf_train <- stlf(train_brick)

snaive_train <- snaive(train_brick)

autoplot(bricksq, series = "Data") +
  autolayer(snaive_train, PI = FALSE, series = "snaive") + #PI = prediction interval ?
  autolayer(stlf_train, PI = FALSE, series = "stlf") +
  scale_color_manual(values = c("black", "blue", "red"), breaks = c("Data", "snaive", "stlf")) +
  xlim(1990, 1994.75) + ylim(300,600) +
  guides(colour = guide_legend(title = "Data")) +
```

```
  ggtitle("Forecast (1992 Q4 - 1994 Q3)  with stlf and snaive") +
  annotate(
    "rect",
    xmin=1992.75,xmax=1994.5,ymin=-Inf,ymax=Inf,
    fill="#72C6E2",alpha = 0.4)
```

```
## Scale for 'x' is already present. Adding another scale for 'x', which will
## replace the existing scale.
```

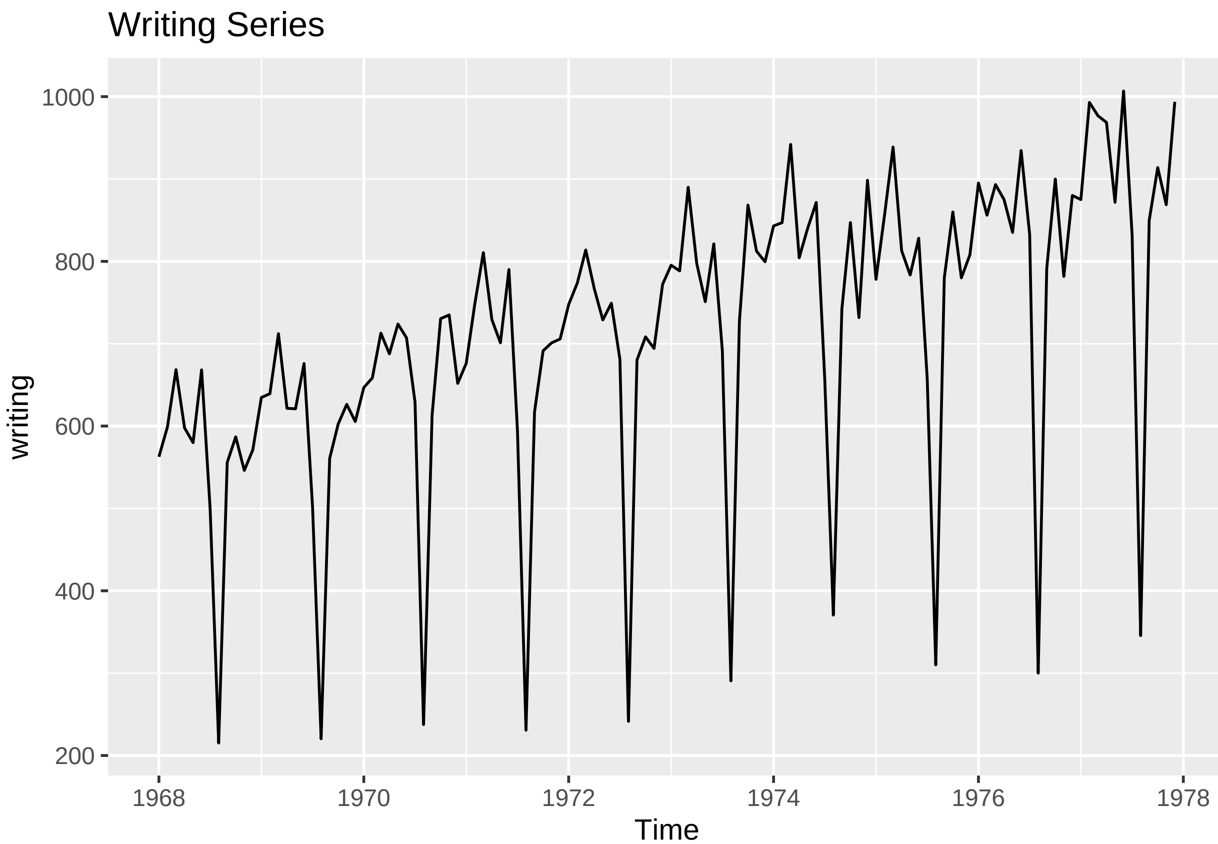


It looks like the `stlf()` was better at predicting the forecast. It follows more closely with the original data. stlf predicts trend and seasonality better than snaive.

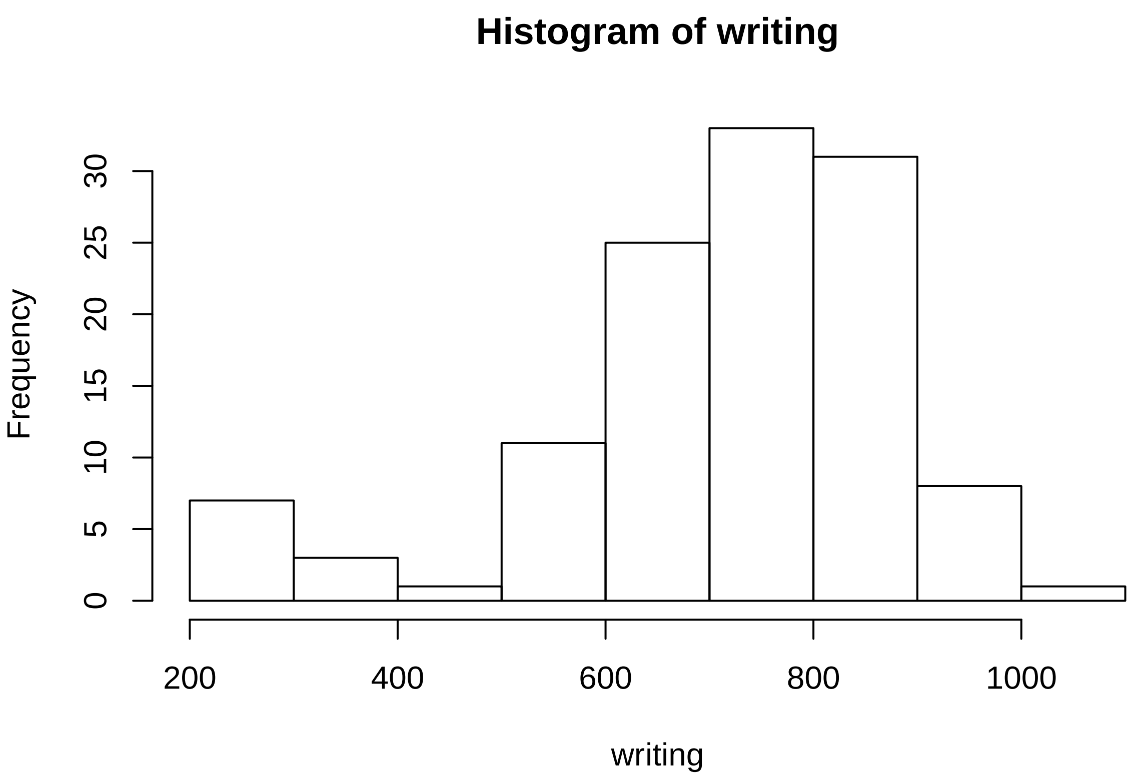
Problem 6.7 (Textbook C):

Use `stlf()` to produce forecasts of the `writing` series with either `method="naive"` or `method="rwdrift"`, whichever is most appropriate. Use the lambda argument if you think a Box-Cox transformation is required.

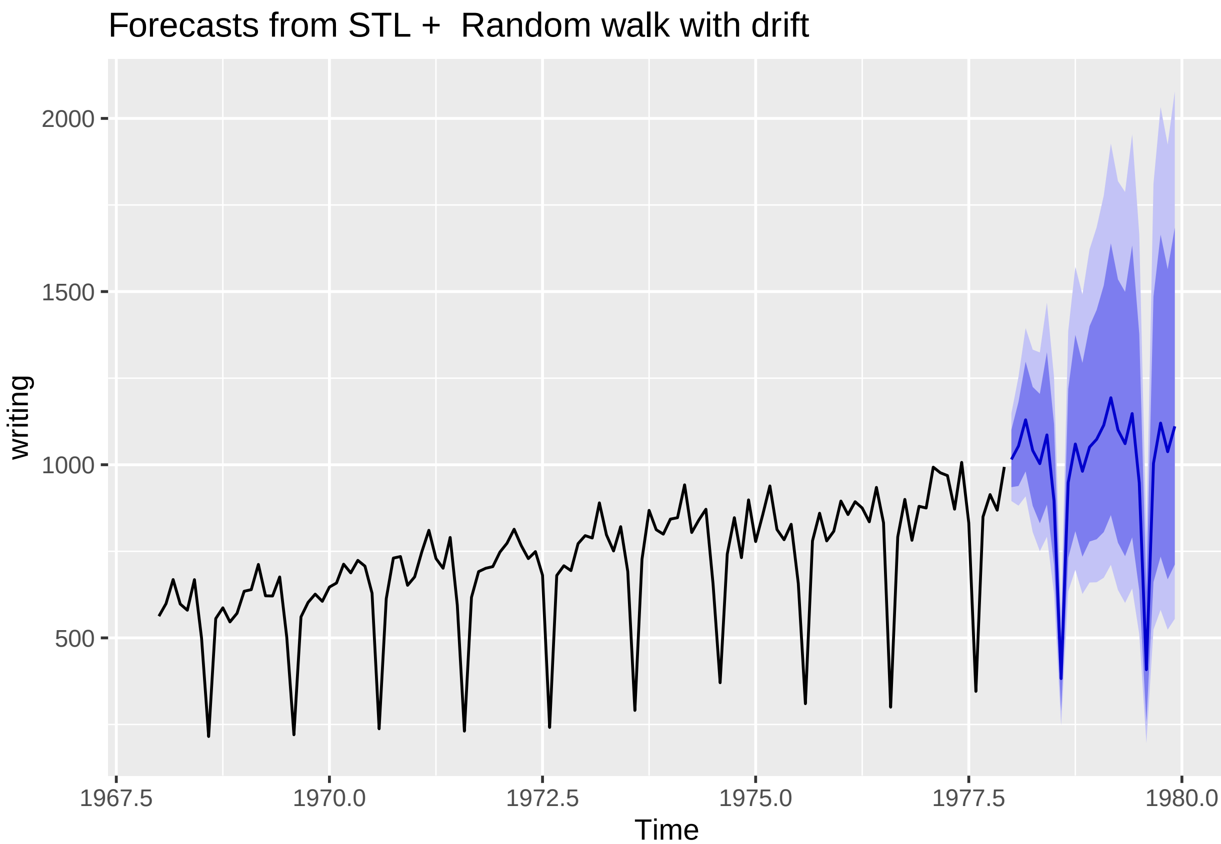
```
autoplot(writing, main= "Writing Series")
```




```r
hist(writing)
```



```r
stlf(writing, robust = TRUE, method = "rwdrift", lambda = BoxCox.lambda(writing)) %>% autoplot()
```

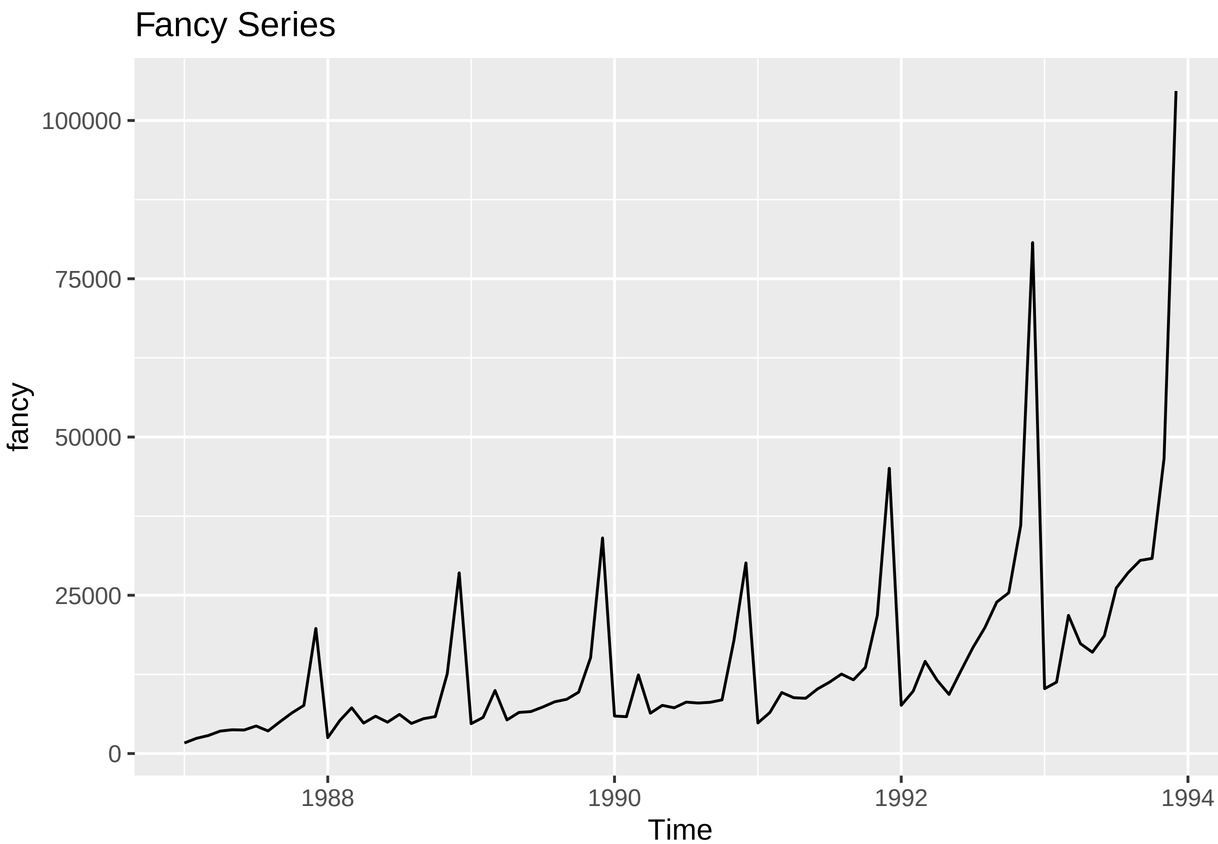


Used a Box-Cox Transformation because the distribution was slightly skewed to the right. I also chose the method `rwdrift` because it performs better than naive for a series that has an overall trend.

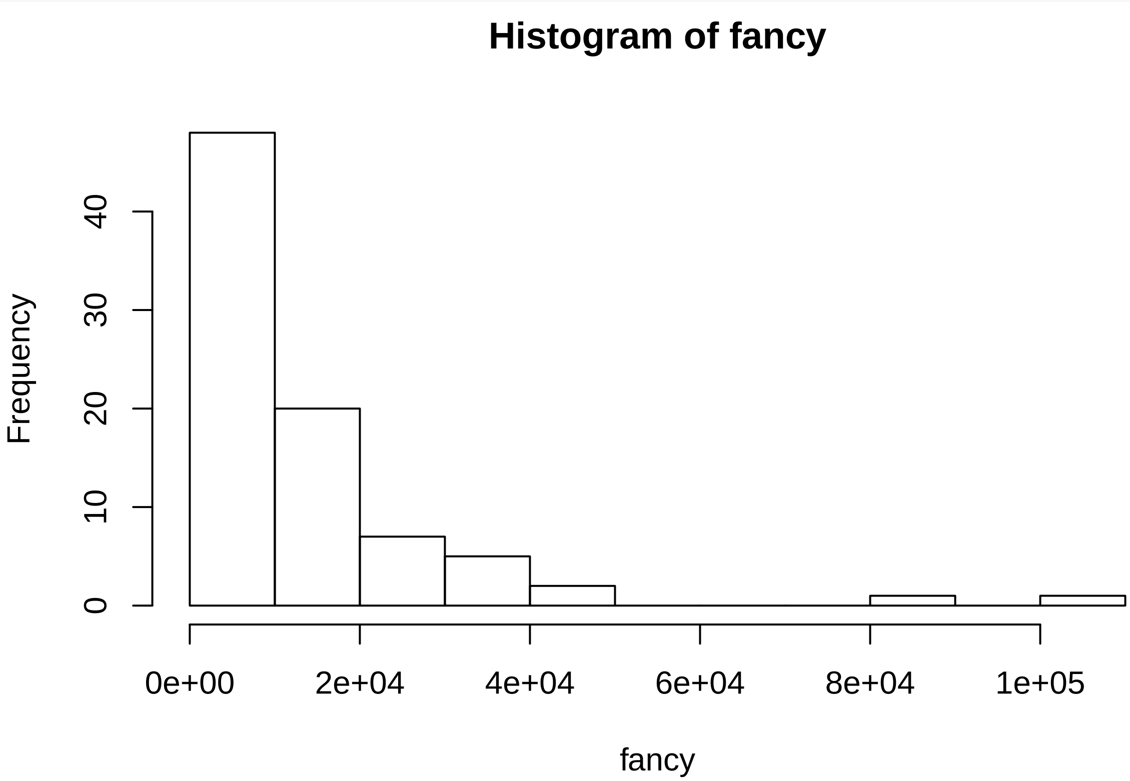
Problem 6.8 (Textbook C):

Use `stlf()` to produce forecasts of the `fancy` series with either `method="naive"` or `method="rwdrift"`, whichever is most appropriate. Use the lambda argument if you think a Box-Cox transformation is required.

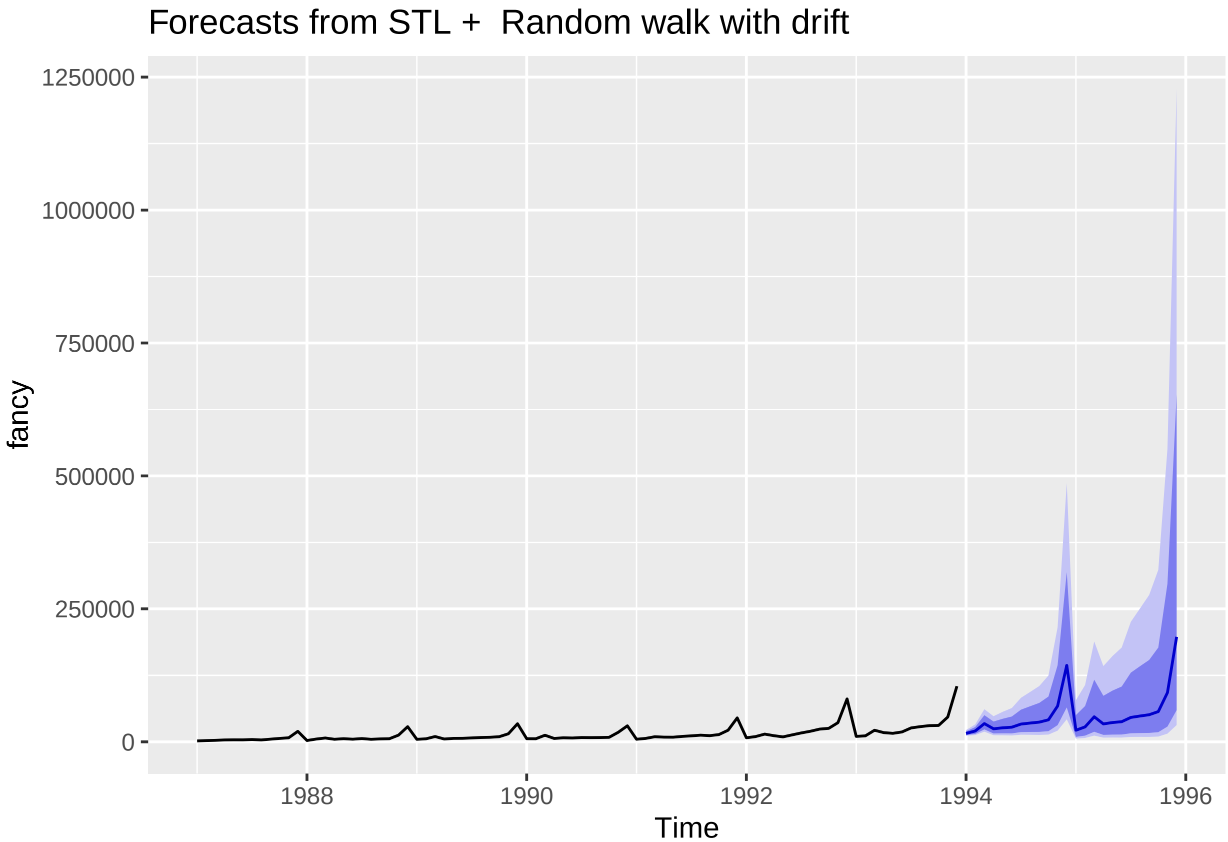
```r
autoplot(fancy, main = "Fancy Series")
```


Fancy Series
100000
75000
50000
25000
0
fancy
1988
1990
1992
1994
Time

```r
hist(fancy)
```


Histogram of fancy
Frequency
0
10
20
30
40
0e+00
2e+04
4e+04
6e+04
8e+04
1e+05
fancy

```r
stlf(fancy, robust = TRUE, method = "rwdrift", lambda = BoxCox.lambda(fancy)) %>% autoplot()
```



I used a Box-Cox transformation to make the distribution more evenly distributed and it would perform better forecasts. I chose to use `rwdrift` to perform the forecasts because the original data had an upward trend.