tharunte_Homework_4

Question 1: 1. Open link 1 of Links for Time Series. Run R commands pertinent to Ch.4-5. These commands are listed on pp.429-438.

Chapter 4

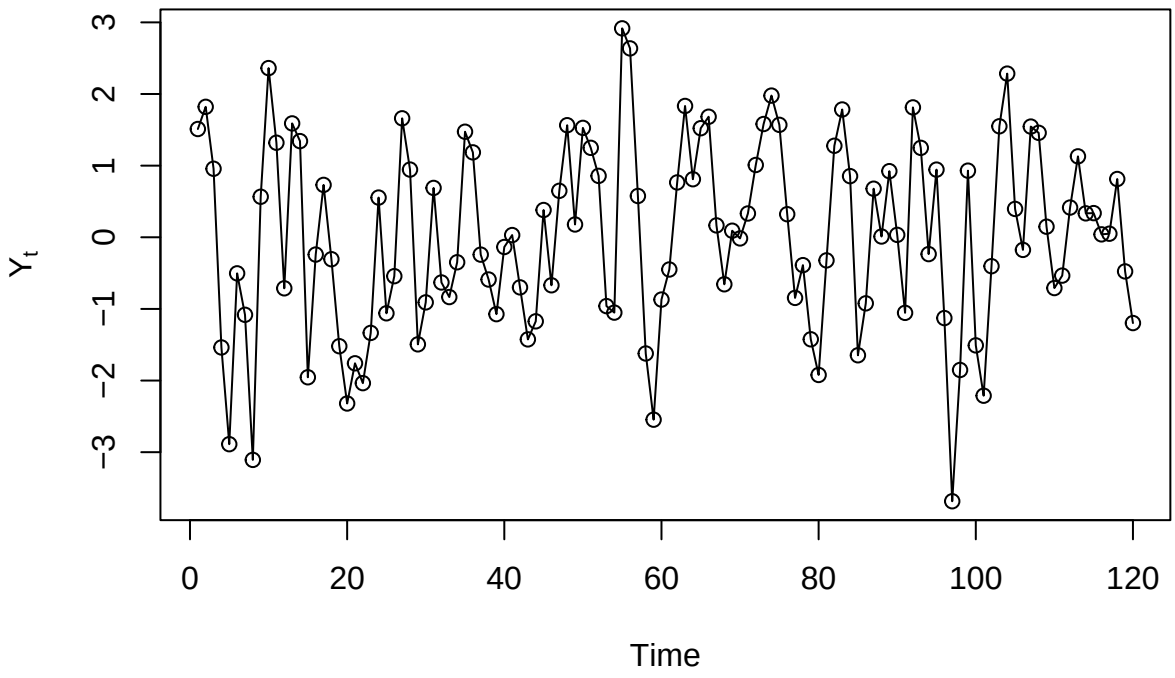
```
library(TSA)
```

```
##
## Attaching package: 'TSA'

## The following objects are masked from 'package:stats':
##
##     acf, arima

## The following object is masked from 'package:utils':
##
##     tar
```

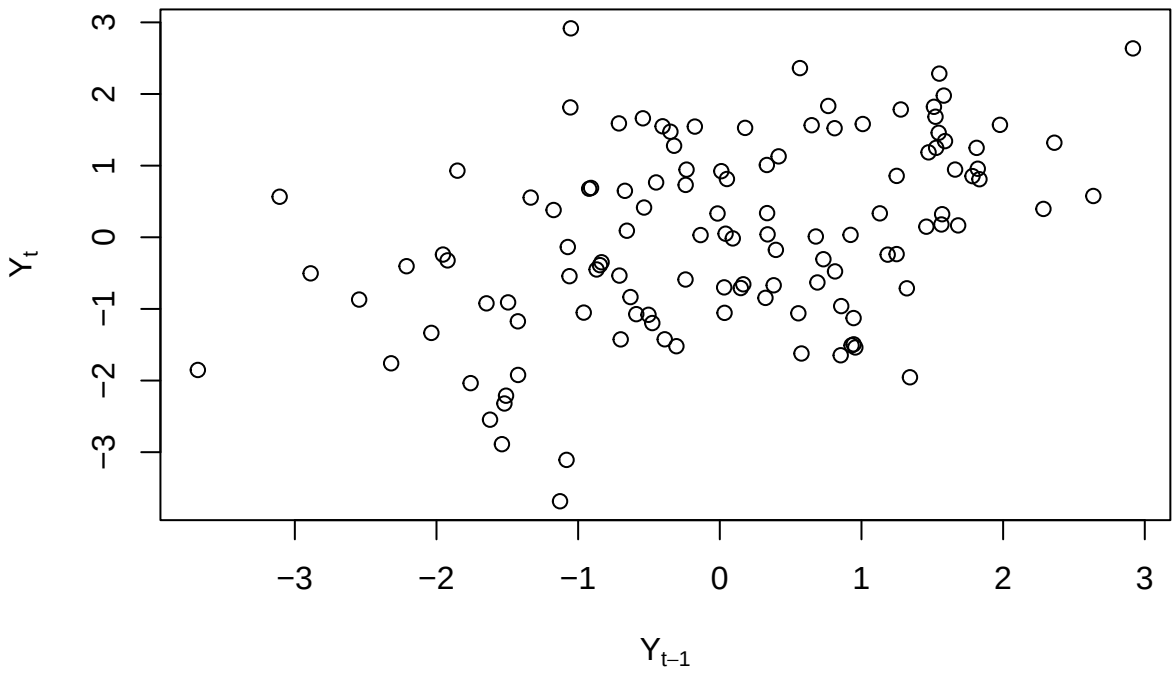
Moving Average

MA1

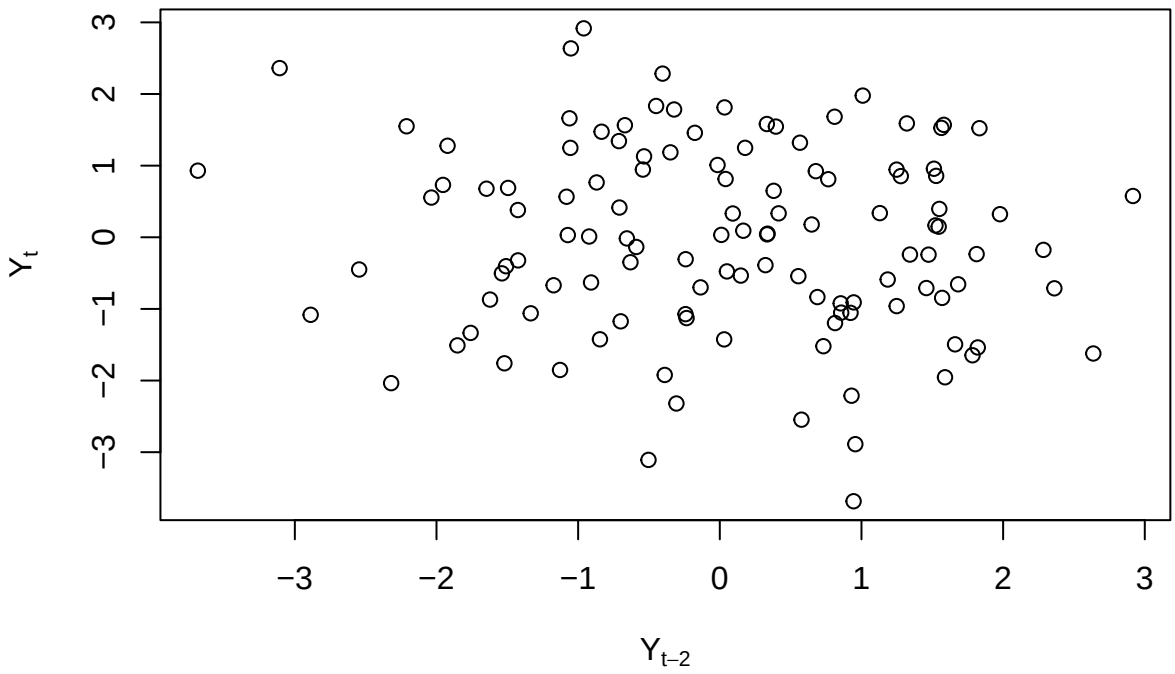
```
# Exhibit 4.2 on page 59.
data(ma1.2.s)
plot(ma1.2.s,ylab=expression(Y[t]),type='o')
```




```
plot(y=ma1.2.s,x=zlag(ma1.2.s),ylab=expression(Y[t]),xlab=expression(Y[t-1]),type='p')
```

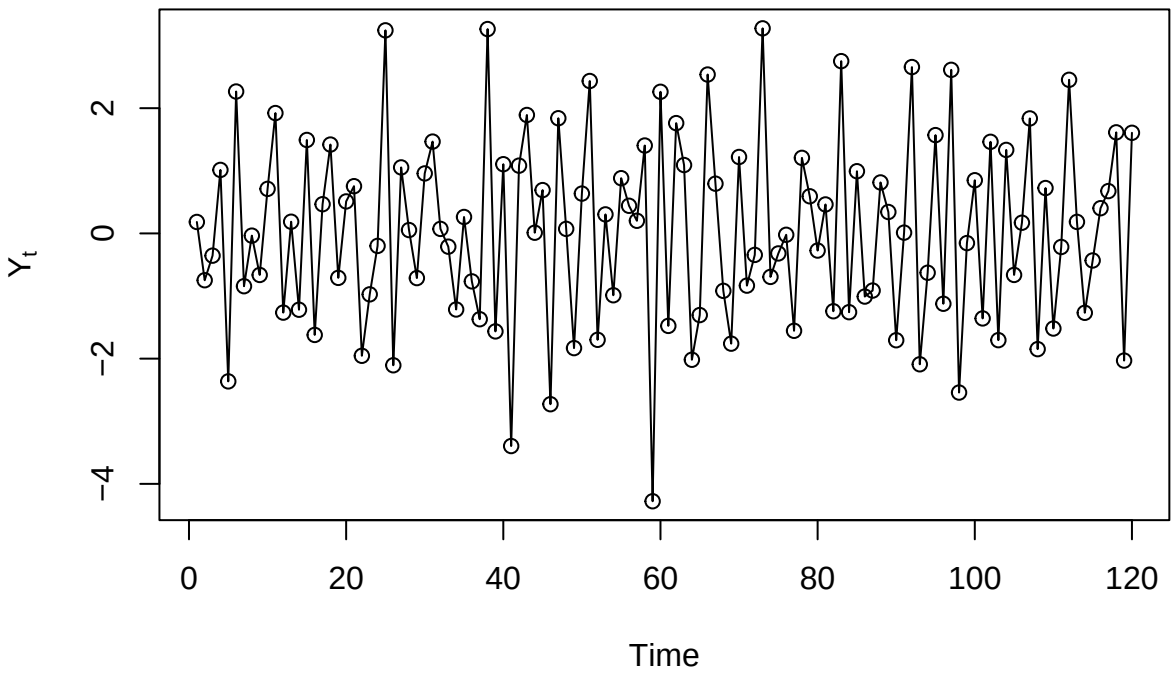



```
plot(y=ma1.2.s,x=zlag(ma1.2.s,2),ylab=expression(Y[t]),xlab=expression(Y[t-2]),type='p')
```

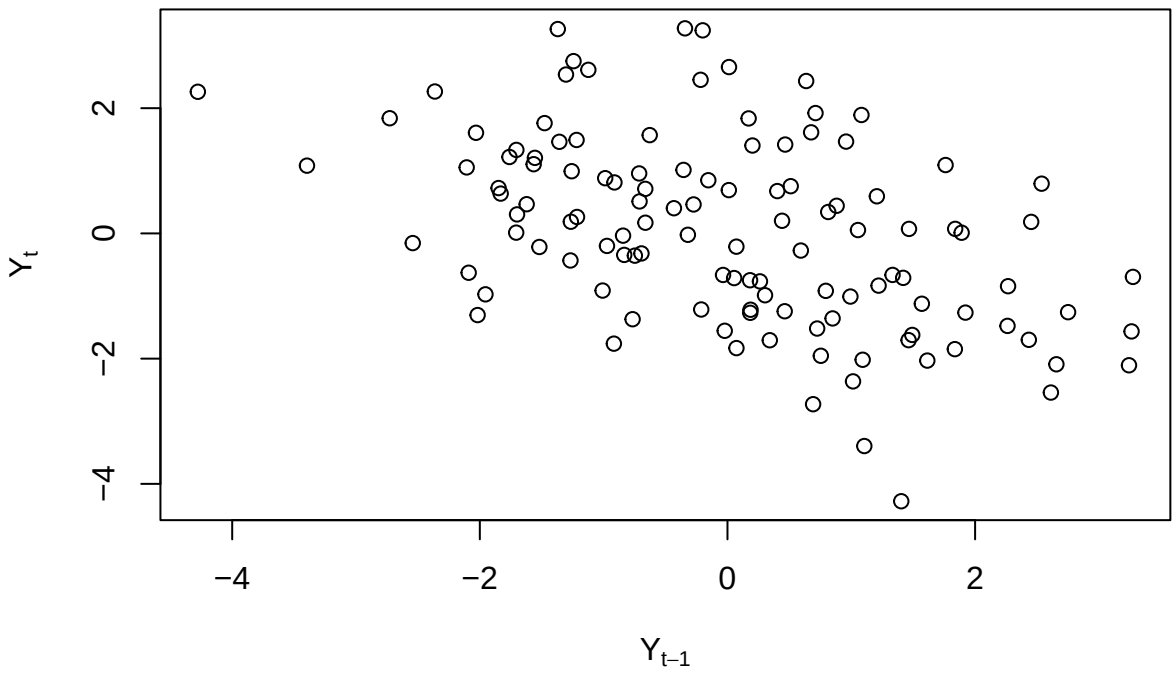


MA1

```
data(ma1.1.s)
plot(ma1.1.s,ylab=expression(Y[t]),type='o')
```

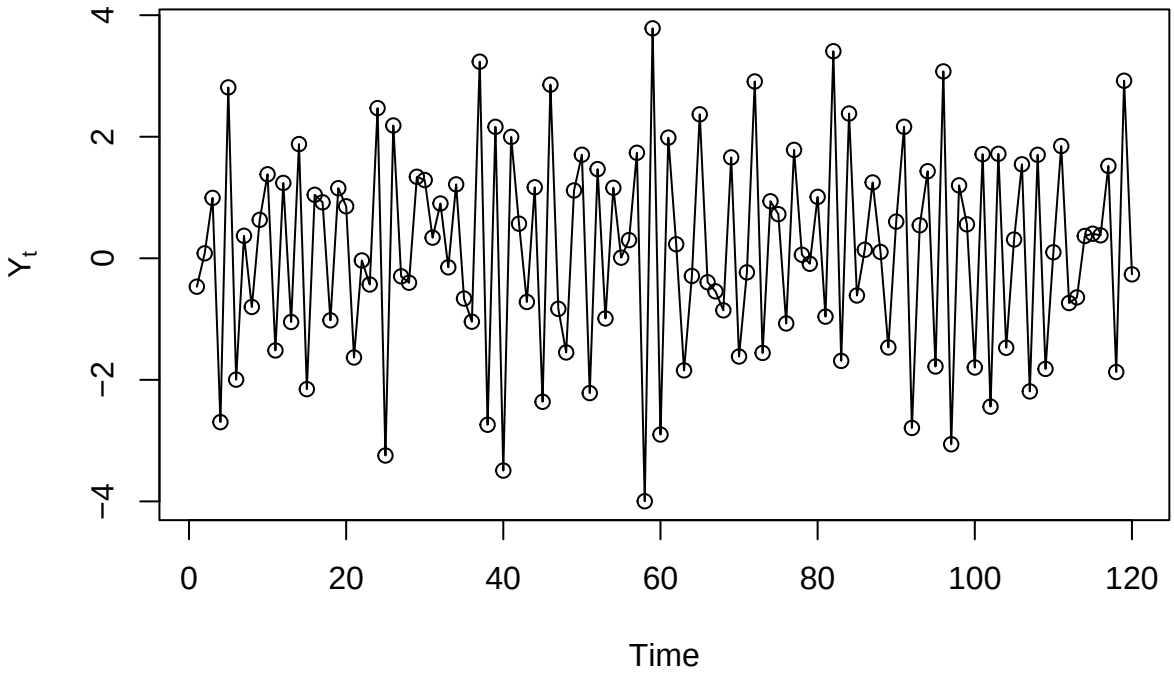



```
plot(y=ma1.1.s,x=zlag(ma1.1.s),ylab=expression(Y[t]),xlab=expression(Y[t-1]),type='p')
```

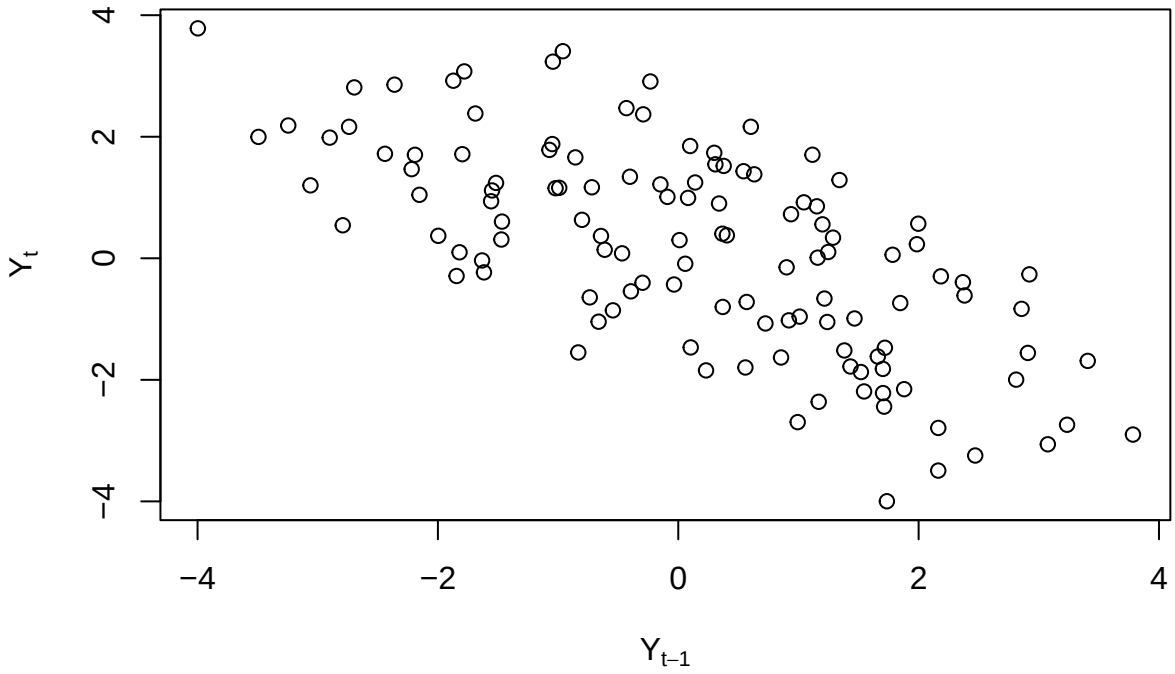


MA2 Additional code from listed in ch 4 60 - 70

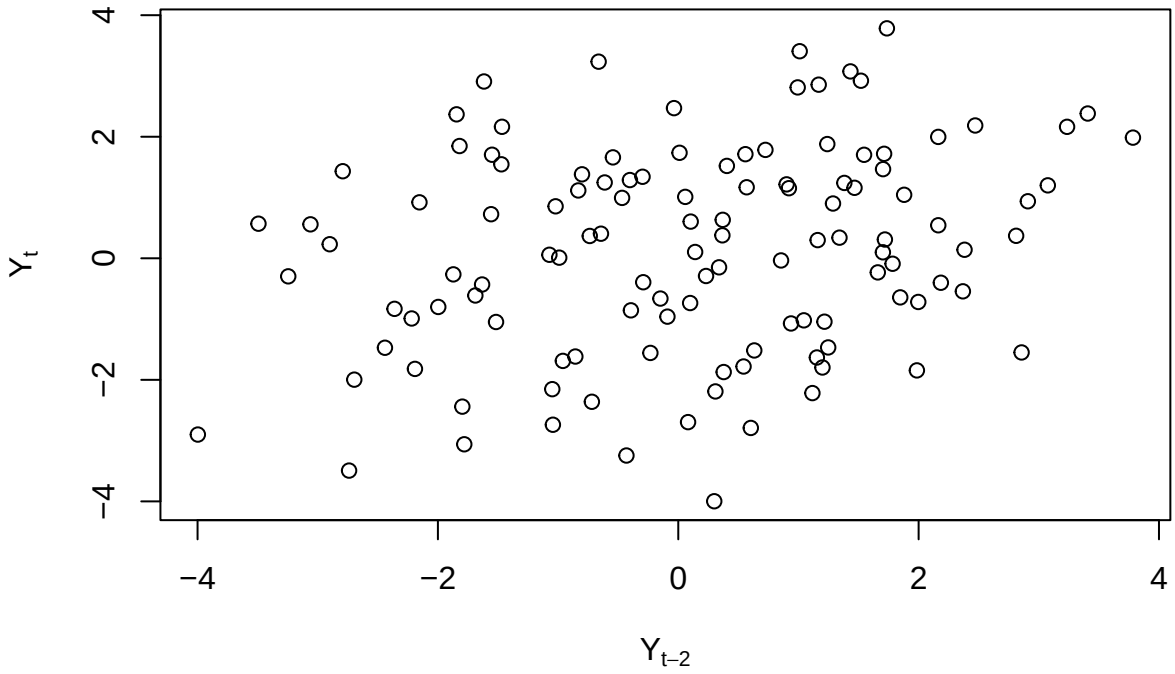
```
data(ma2.s)
plot(ma2.s,ylab=expression(Y[t]),type='o')
```



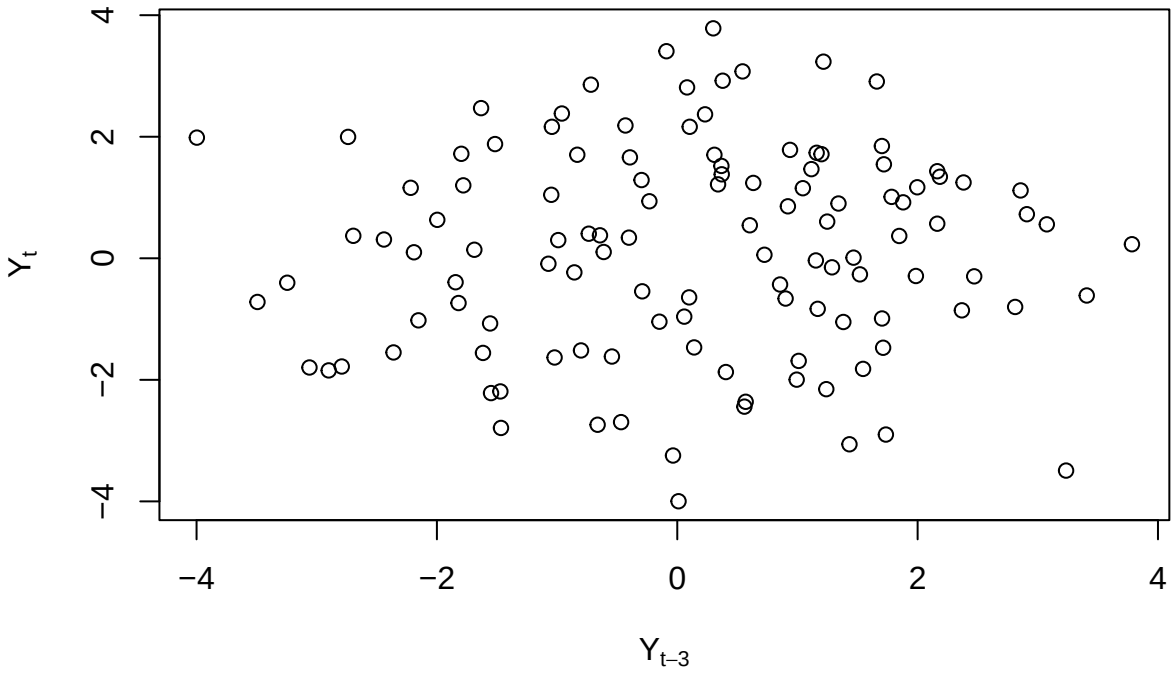

```
plot(y=ma2.s,x=zlag(ma2.s),ylab=expression(Y[t]),xlab=expression(Y[t-1]),type='p')
```




```
plot(y=ma2.s,x=zlag(ma2.s,2),ylab=expression(Y[t]),xlab=expression(Y[t-2]),type='p')
```



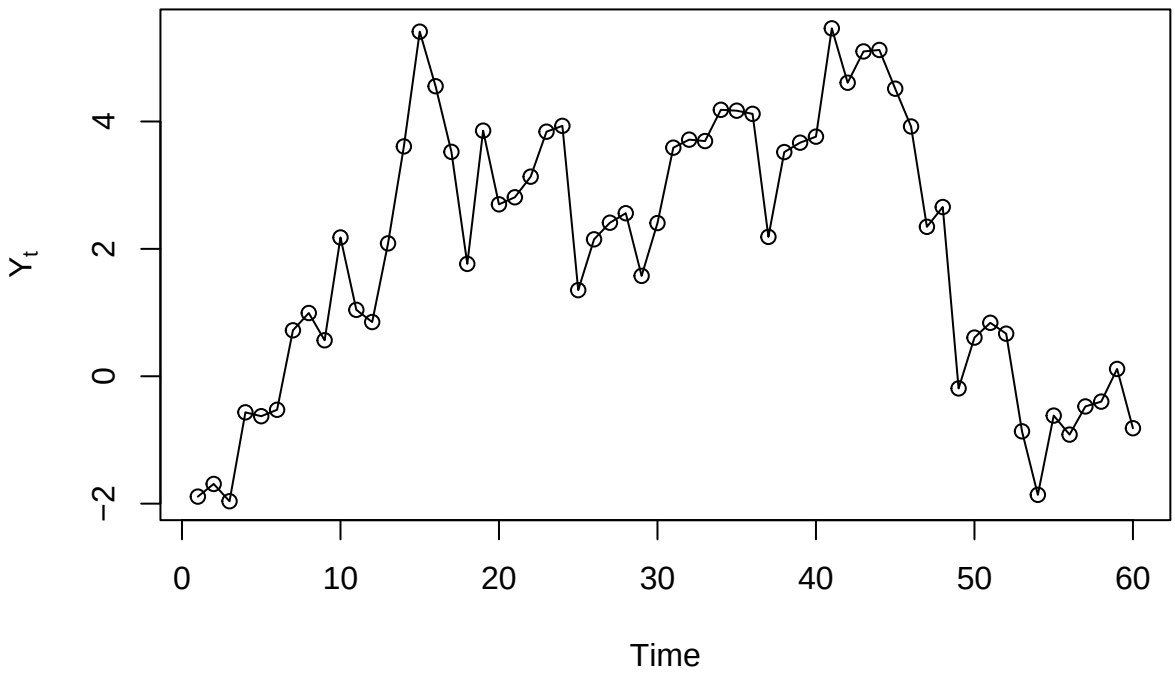

```
plot(y=ma2.s,x=zlag(ma2.s,3),ylab=expression(Y[t]),xlab=expression(Y[t-3]),type='p')
```



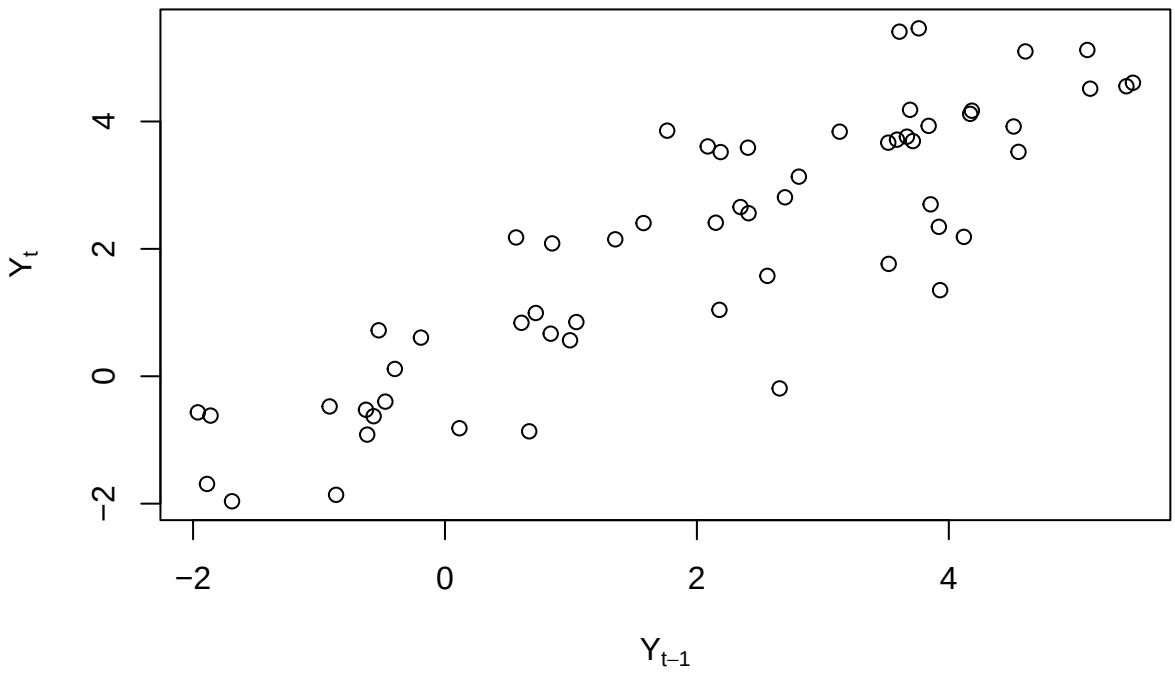
Autoregressive Processes

AR1

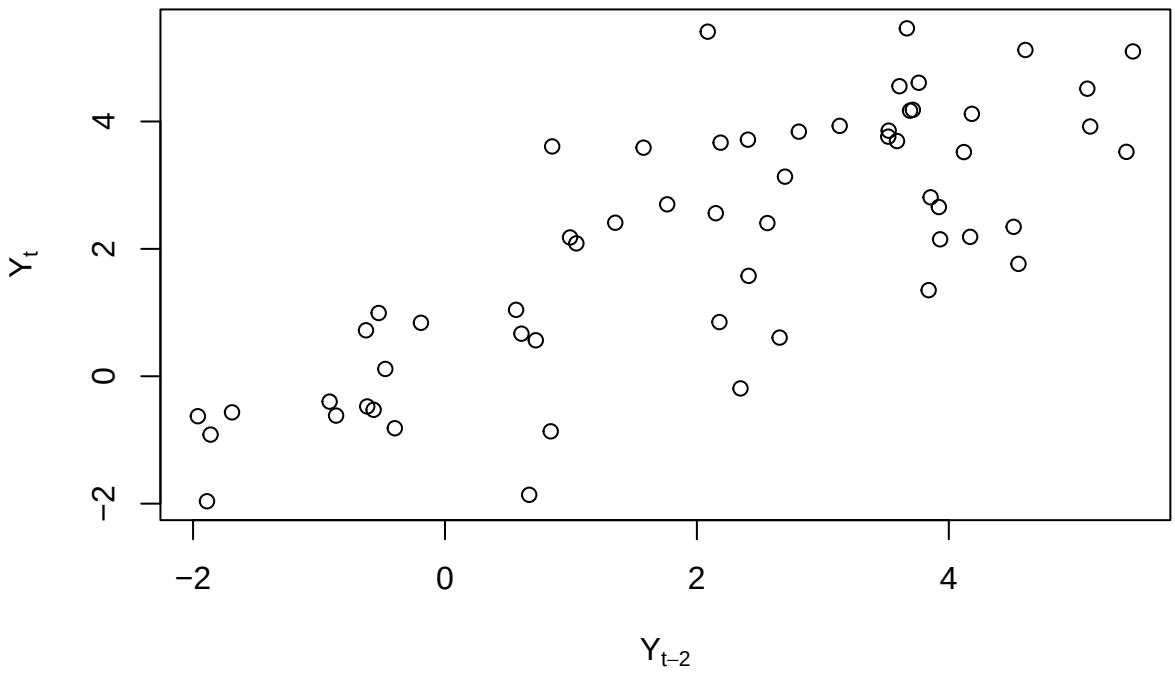
```
data(ar1.s)
plot(ar1.s,ylab=expression(Y[t]),type='o')
```



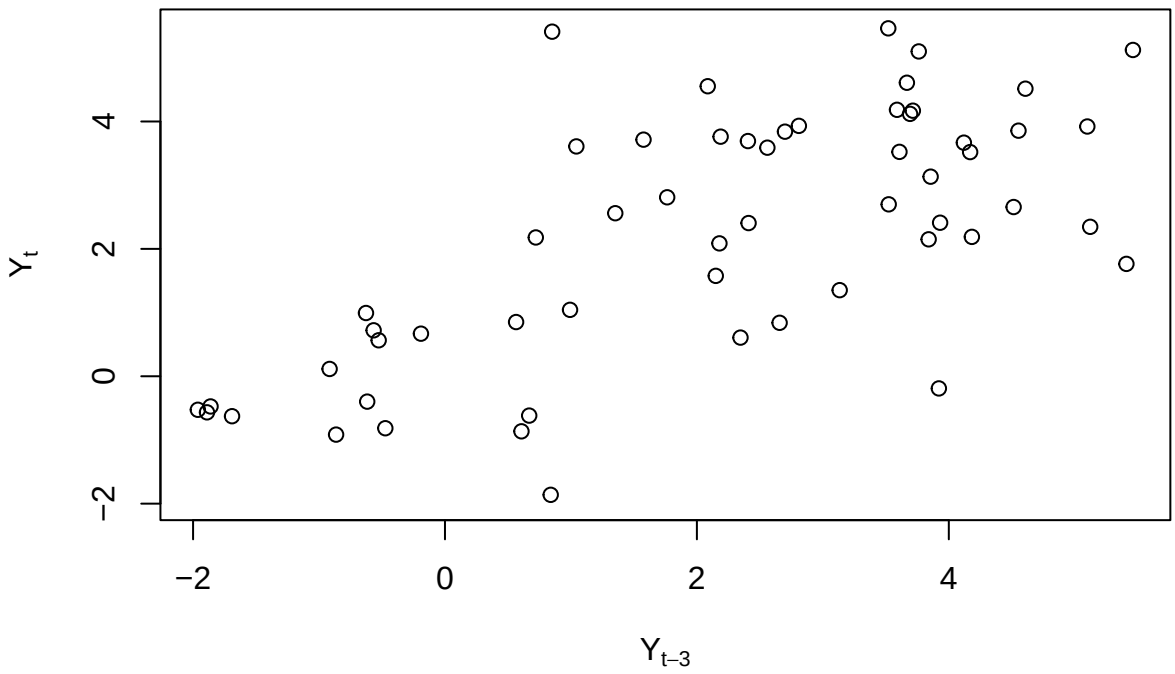

```
plot(y=ar1.s,x=zlag(ar1.s),ylab=expression(Y[t]),xlab=expression(Y[t-1]),type='p')
```




```
plot(y=ar1.s,x=zlag(ar1.s,2),ylab=expression(Y[t]),xlab=expression(Y[t-2]),type='p')
```

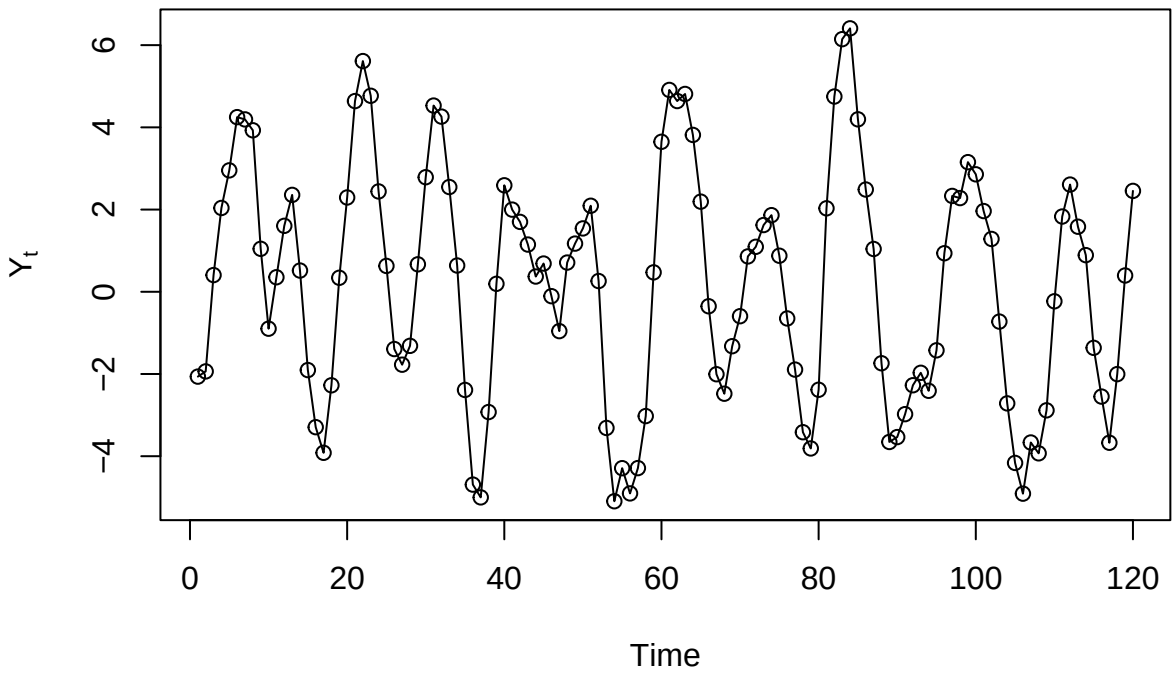



```
plot(y=ar1.s,x=zlag(ar1.s,3),ylab=expression(Y[t]),xlab=expression(Y[t-3]),type='p')
```



AR2

```
data(ar2.s)
plot(ar2.s,ylab=expression(Y[t]),type='o')
```



Continuation

```
set.seed(12345)
y=arima.sim(model=list(ma=-c(-0.9)),n=100)
```

```
list1=list(a=c(1,2,3),b=4,c=ts(c(5,6,7,8),start=c(2006,2),frequency=4))
list1
```

```
## $a
## [1] 1 2 3
##
## $b
## [1] 4
##
## $c
##      Qtr1 Qtr2 Qtr3 Qtr4
## 2006         5    6    7
## 2007    8
```

To retrieve an element of a list

```
list1$c
```

```
##      Qtr1 Qtr2 Qtr3 Qtr4
## 2006         5    6    7
## 2007    8
```

```
str(list1)
```

```
## List of 3
##  $ a: num [1:3] 1 2 3
##  $ b: num 4
##  $ c: Time-Series [1:4] from 2006 to 2007: 5 6 7 8
```

Chapter 5

Exhibit 5.1 Monthly Price of Oil: January 1986–January 2006

```
data(oil.price)
plot(oil.price, ylab='Price per Barrel',type='l')
```

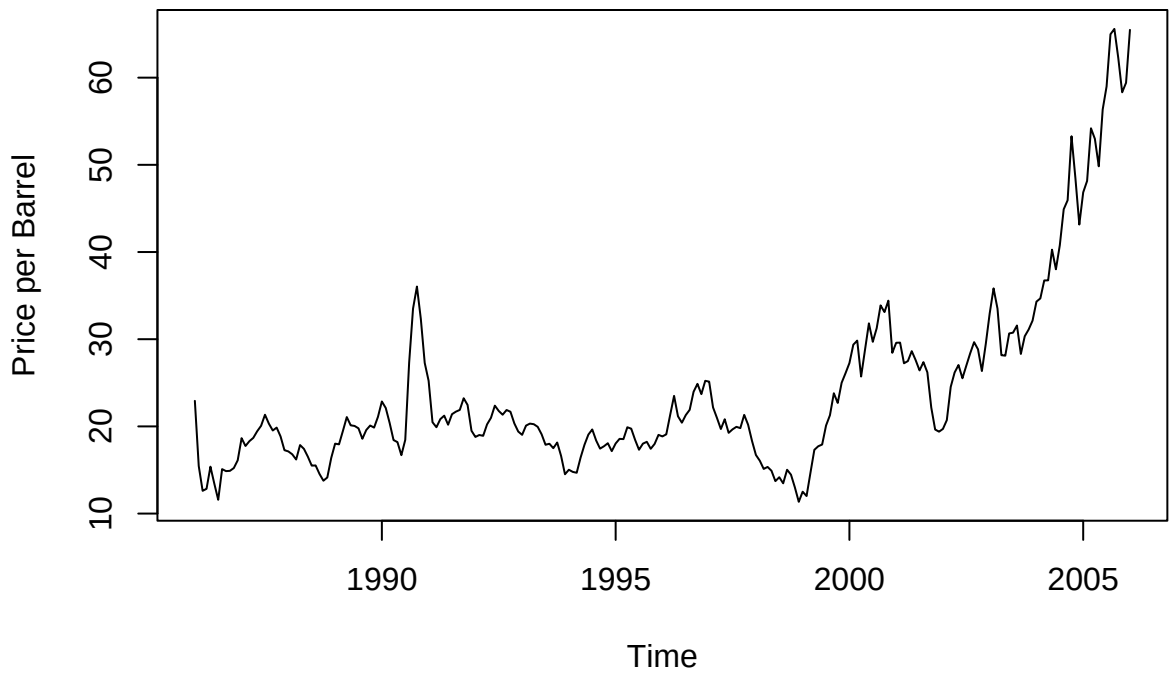


Exhibit 5.3 An Explosive "AR(1)" Series

```
data(explode.s)
plot(explode.s,ylab=expression(Y[t]),type='o')
```

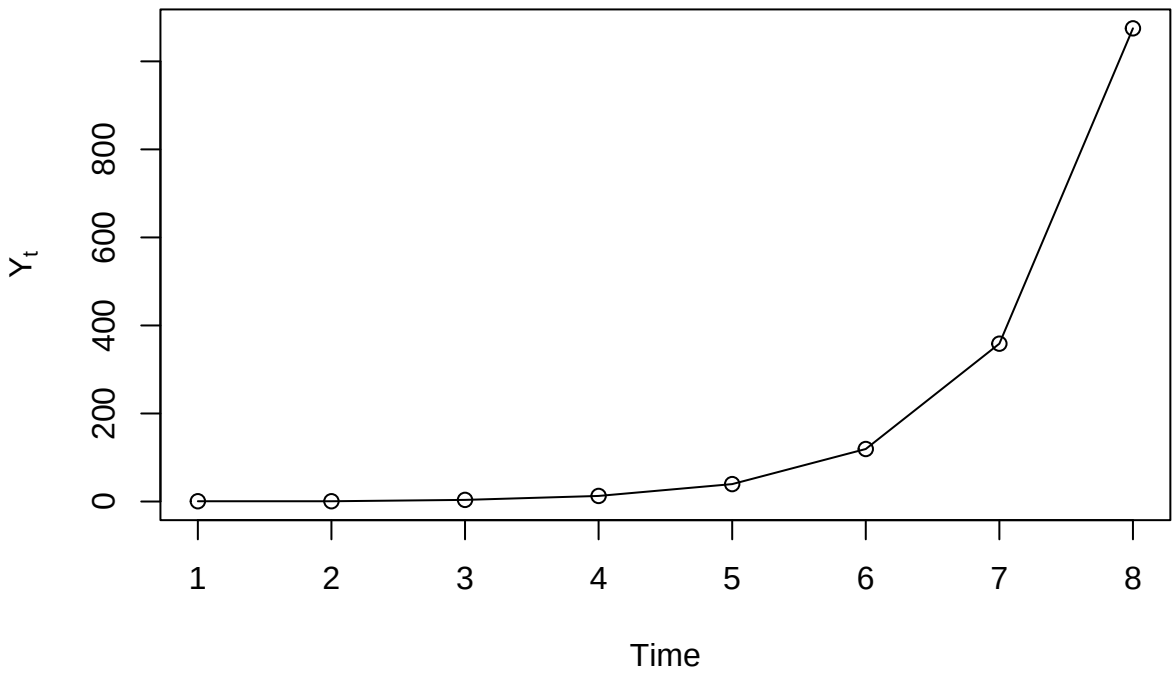


Exhibit 5.4 The Difference Series of the Logs of the Oil Price Time

```
plot(diff(log(oil.price)),ylab='Change in Log(Price)',type='l')
```

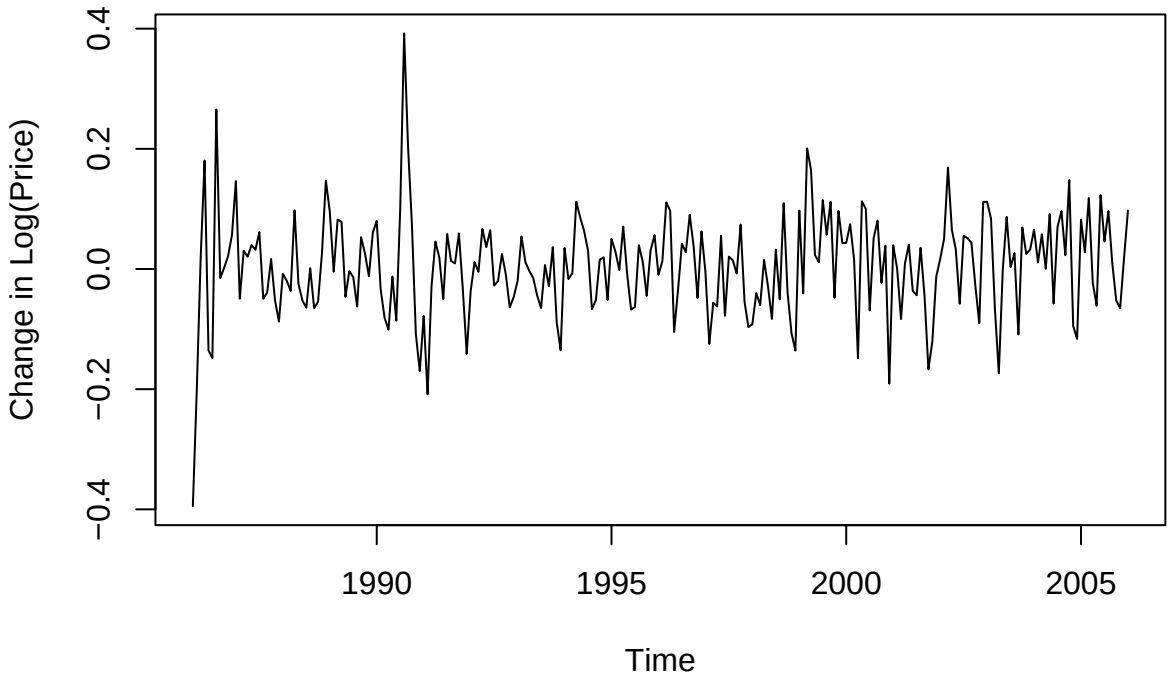


Exhibit 5.5 Simulation of an IMA(2,2) Series with theta_1 = 1 and theta_2 = −0.6

```
data(ima22.s)
plot(ima22.s,ylab='IMA(2,2) Simulation',type='o')
```

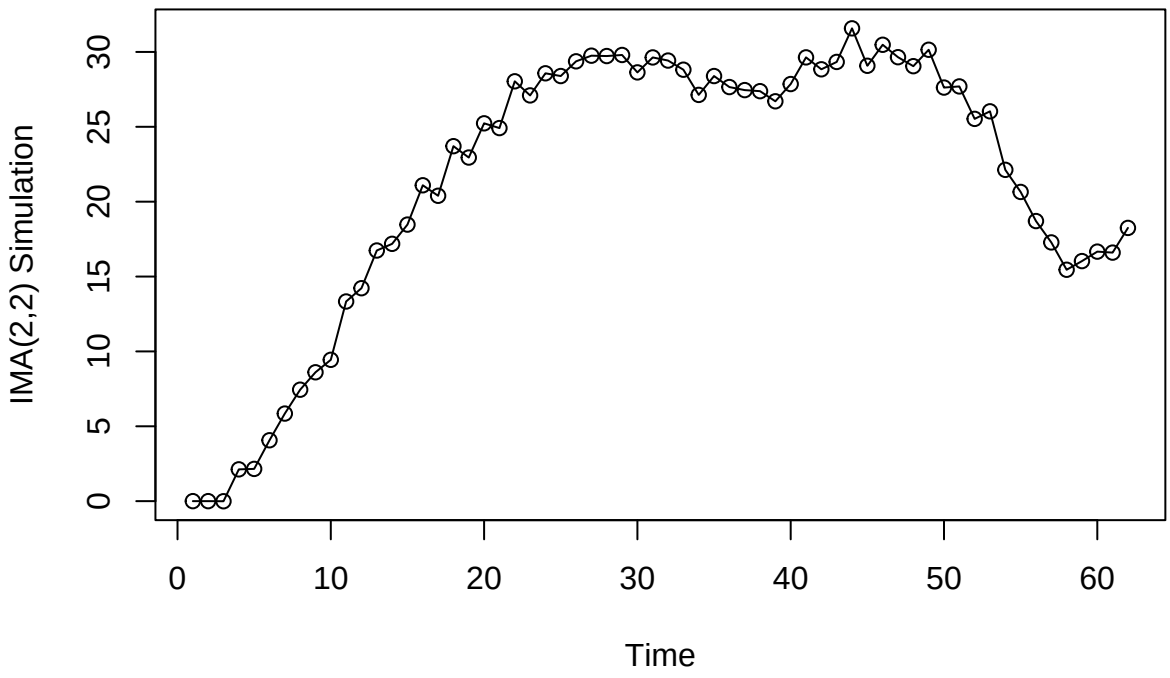


Exhibit 5.6 First Difference of the Simulated IMA(2,2) Series

```
plot(diff(ima22.s),ylab='First Difference',type='o')
```

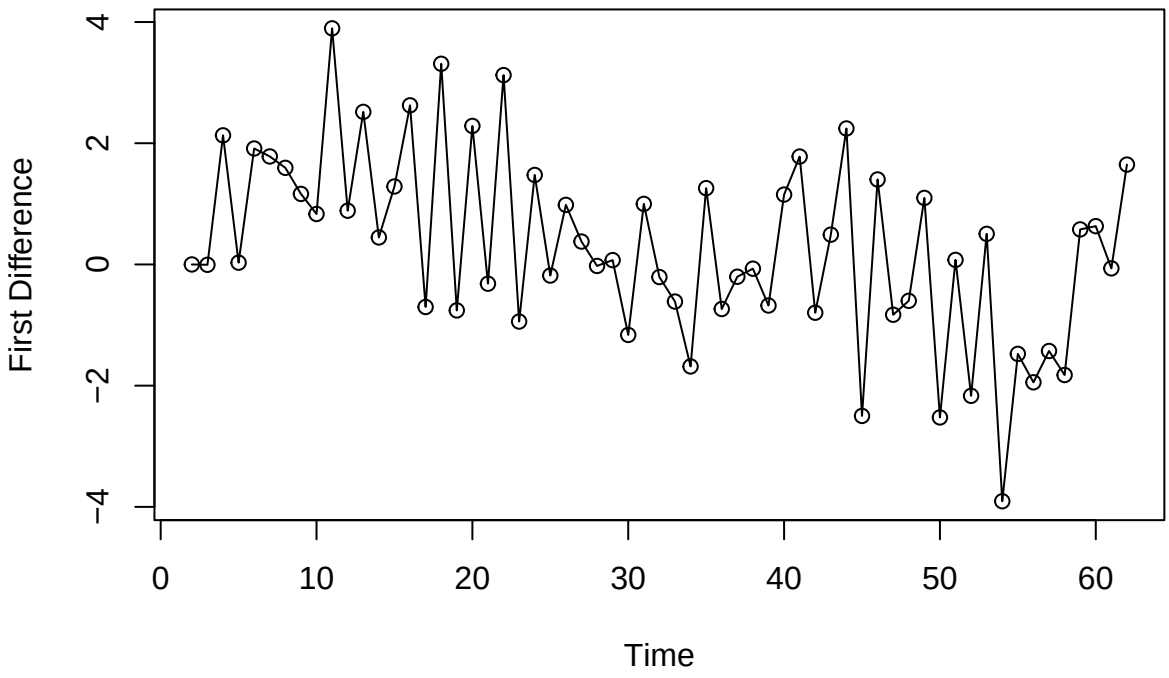


Exhibit 5.7 Second Difference of the Simulated IMA(2,2) Series

```
plot(diff(ima22.s,difference=2),ylab='DifferencedTwice',type='o')
```

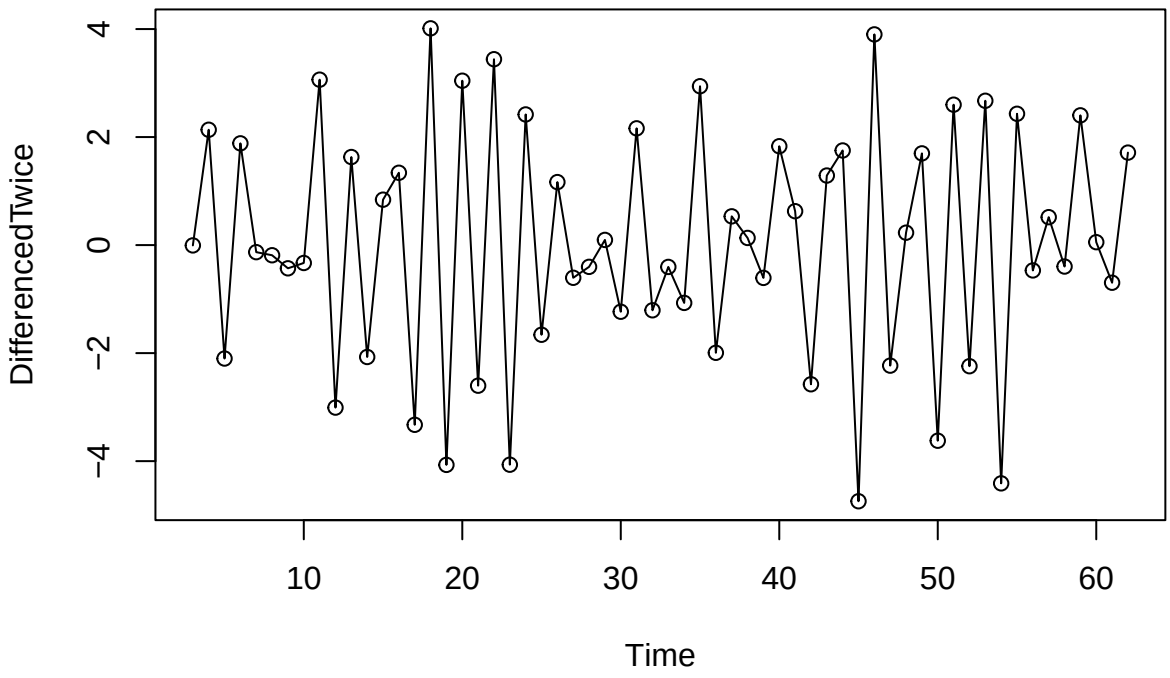


Exhibit 5.8 U.S. Electricity Generated by Month

```
data(electricity); plot(electricity)
```

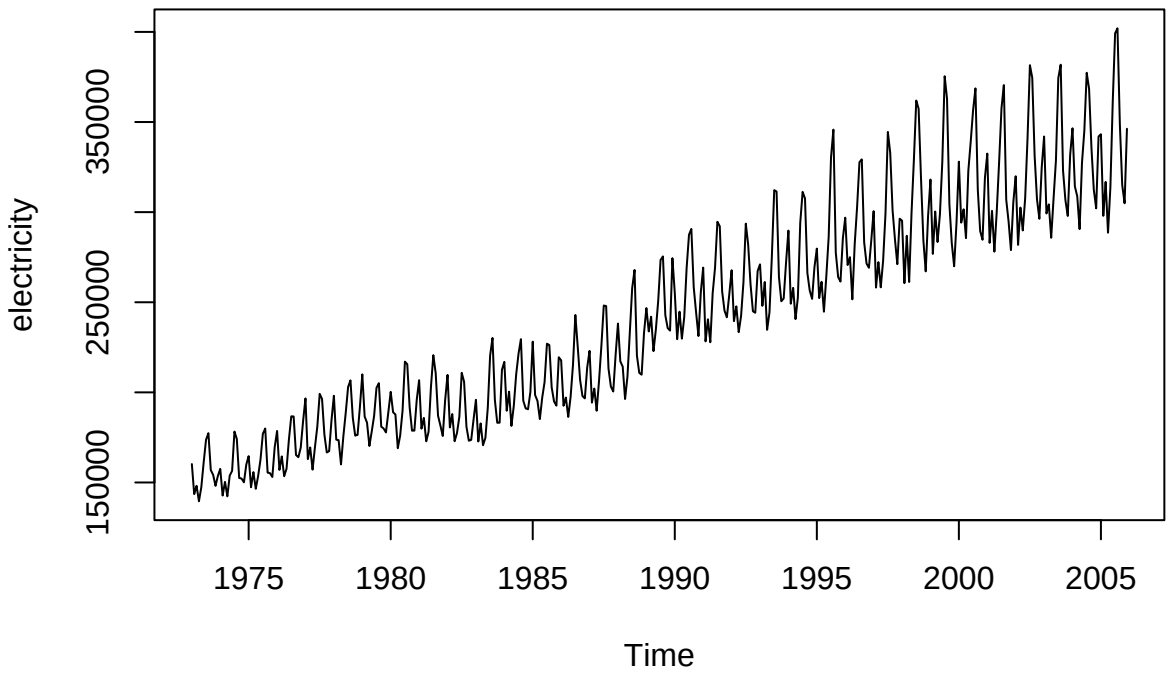


Exhibit 5.9 Time Series Plot of Logarithms of Electricity Values

```
plot(log(electricity),ylab='Log(electricity)')
```

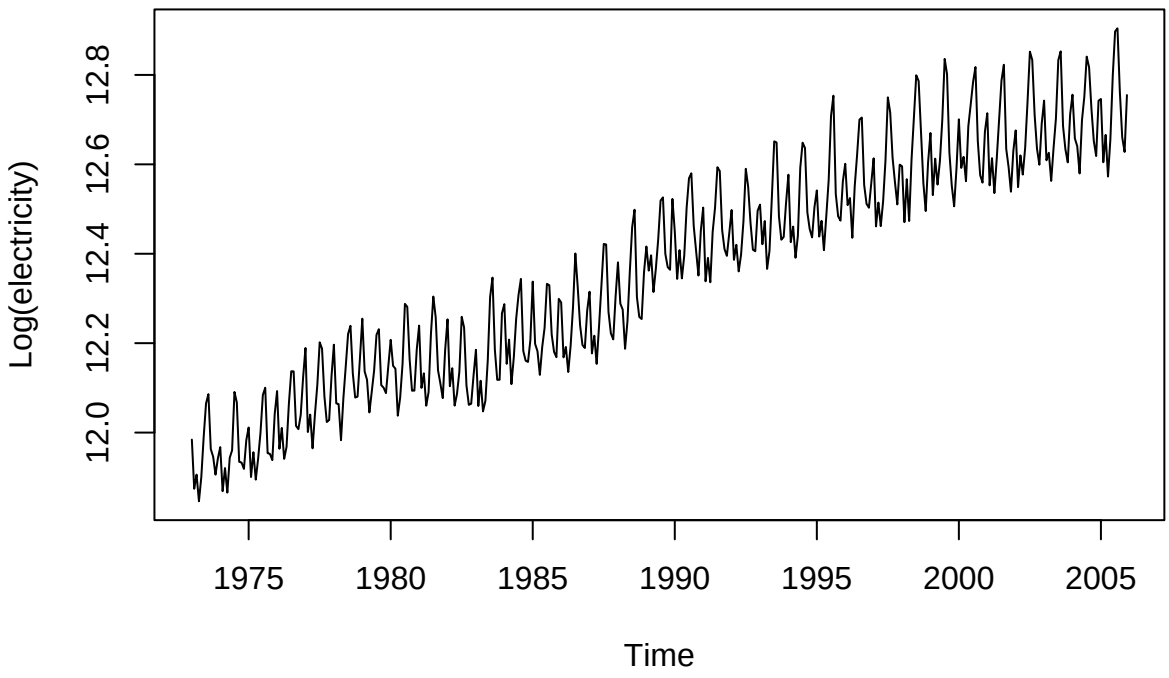


Exhibit 5.10 Difference of Logarithms for Electricity Time Series

```
plot(diff(log(electricity)),ylab='Difference of Log(electricity)')
```

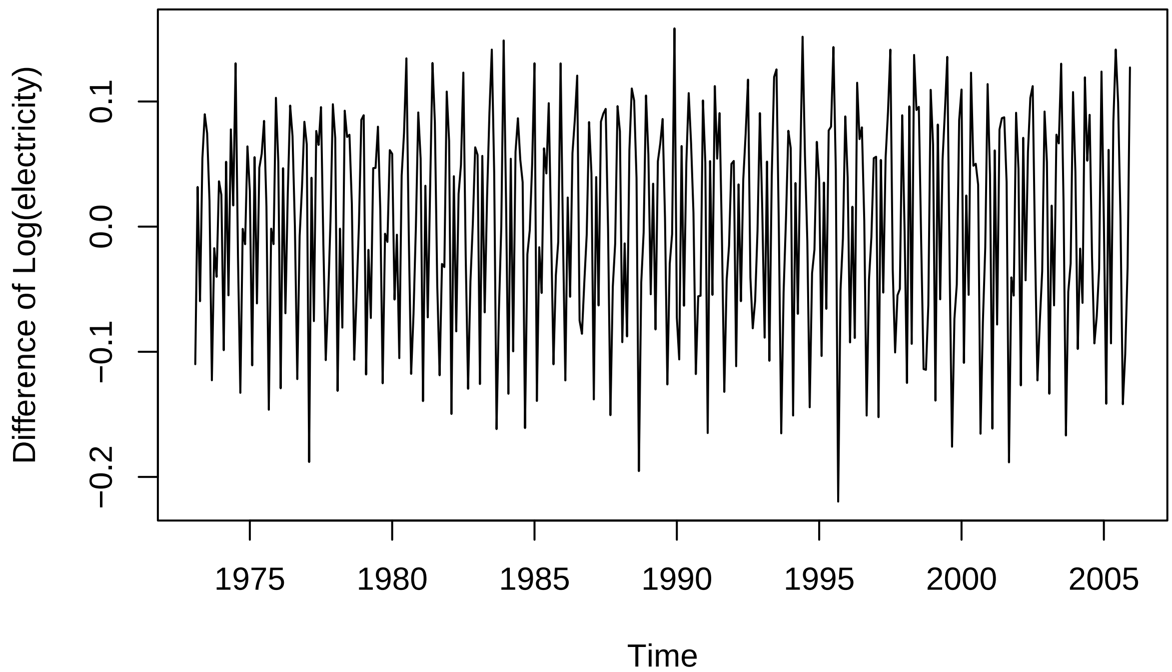
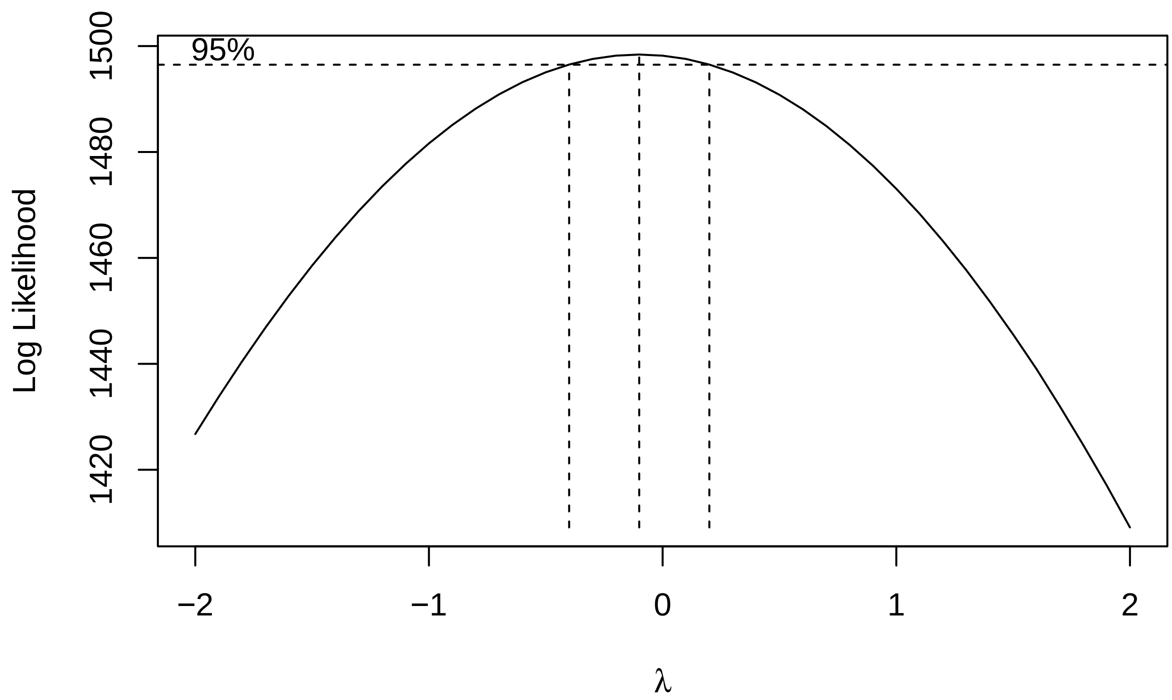


Exhibit 5.11 Log-likelihood versus Lambda

```
BoxCox.ar(electricity)
```

```
## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean): possible
## convergence problem: optim gave code = 1
## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean): possible
## convergence problem: optim gave code = 1
## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean): possible
## convergence problem: optim gave code = 1
## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean): possible
## convergence problem: optim gave code = 1
## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean): possible
## convergence problem: optim gave code = 1
## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean): possible
## convergence problem: optim gave code = 1
## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean): possible
## convergence problem: optim gave code = 1
## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean): possible
## convergence problem: optim gave code = 1
## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean): possible
## convergence problem: optim gave code = 1
## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean): possible
## convergence problem: optim gave code = 1
## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean): possible
## convergence problem: optim gave code = 1
```

```
## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean): possible
## convergence problem: optim gave code = 1
## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean): possible
## convergence problem: optim gave code = 1
## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean): possible
## convergence problem: optim gave code = 1
## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean): possible
## convergence problem: optim gave code = 1
```

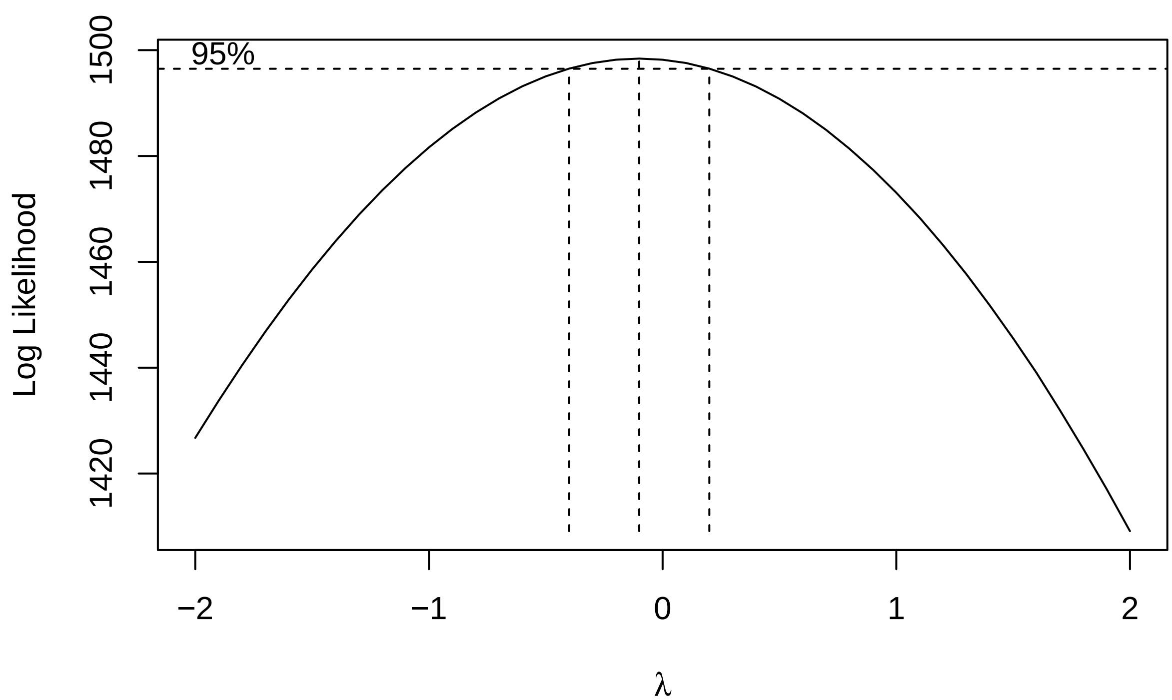


This plots the log-likelihood function of the power parameter for the model that accounts for autocorrelation in the data.

```
BoxCox.ar(electricity)
```

```
## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean): possible
## convergence problem: optim gave code = 1
## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean): possible
## convergence problem: optim gave code = 1
## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean): possible
## convergence problem: optim gave code = 1
## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean): possible
## convergence problem: optim gave code = 1
## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean): possible
## convergence problem: optim gave code = 1
## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean): possible
## convergence problem: optim gave code = 1
```

```
## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean): possible
## convergence problem: optim gave code = 1
## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean): possible
## convergence problem: optim gave code = 1
## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean): possible
## convergence problem: optim gave code = 1
## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean): possible
## convergence problem: optim gave code = 1
## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean): possible
## convergence problem: optim gave code = 1
## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean): possible
## convergence problem: optim gave code = 1
## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean): possible
## convergence problem: optim gave code = 1
## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean): possible
## convergence problem: optim gave code = 1
## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean): possible
## convergence problem: optim gave code = 1
```



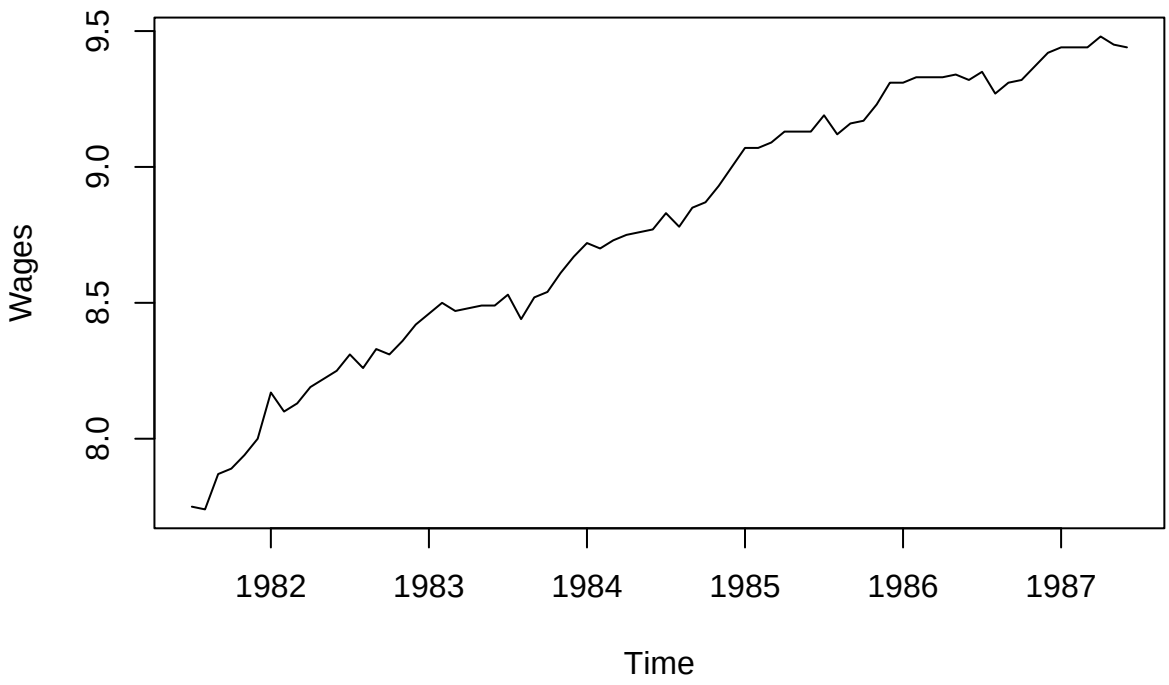
Question 2

3.5:

The data file wages contains monthly values of the average hourly wages (in dollars) for workers in the U.S. apparel and textile products industry for July 1981 through June 1987

(a) Display and interpret the time series plot for these data.

```
data(wages)
plot(wages)
```



Interpretation: will see the trend of wages over time from 1981 to 1987. As the wages goes on increasing with time with some randomness.

(b) Use least squares to fit a linear time trend to this time series. Interpret the regression output. Save the standardized residuals from the fit for further analysis.

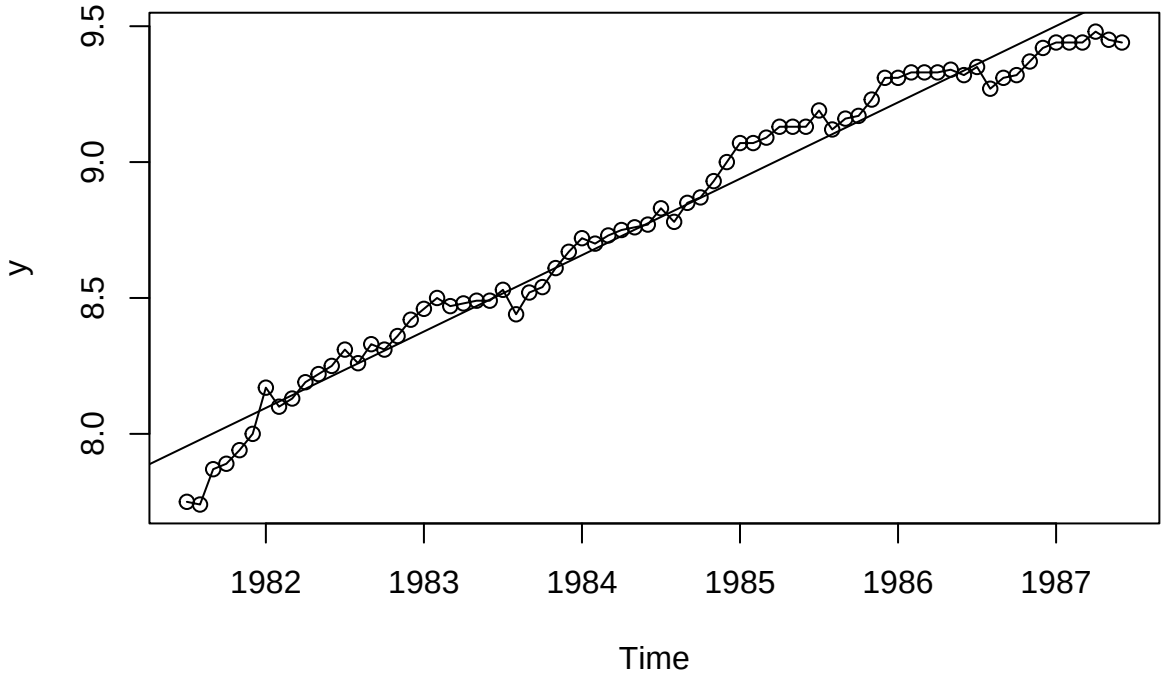
```
model1 = lm(wages~time(wages))
summary(model1)
```

```
##
## Call:
## lm(formula = wages ~ time(wages))
```

```
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.23828 -0.04981  0.01942  0.05845  0.13136
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -5.490e+02  1.115e+01  -49.24   <2e-16 ***
## time(wages)  2.811e-01  5.618e-03   50.03   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.08257 on 70 degrees of freedom
## Multiple R-squared:  0.9728, Adjusted R-squared:  0.9724
## F-statistic:  2503 on 1 and 70 DF,  p-value: < 2.2e-16
```

Interpretation: The linear model fits a strong upward trend in average wages over time, with a significant positive slope of 0.2811, indicating wages increased by 28 cents each month. The high R-squared value (0.9728) shows the model explains 97% of the variation in wages, confirming an excellent fit. Both the intercept and slope are statistically significant with p-values near zero. Residuals are small, suggesting the model captures the trend effectively.

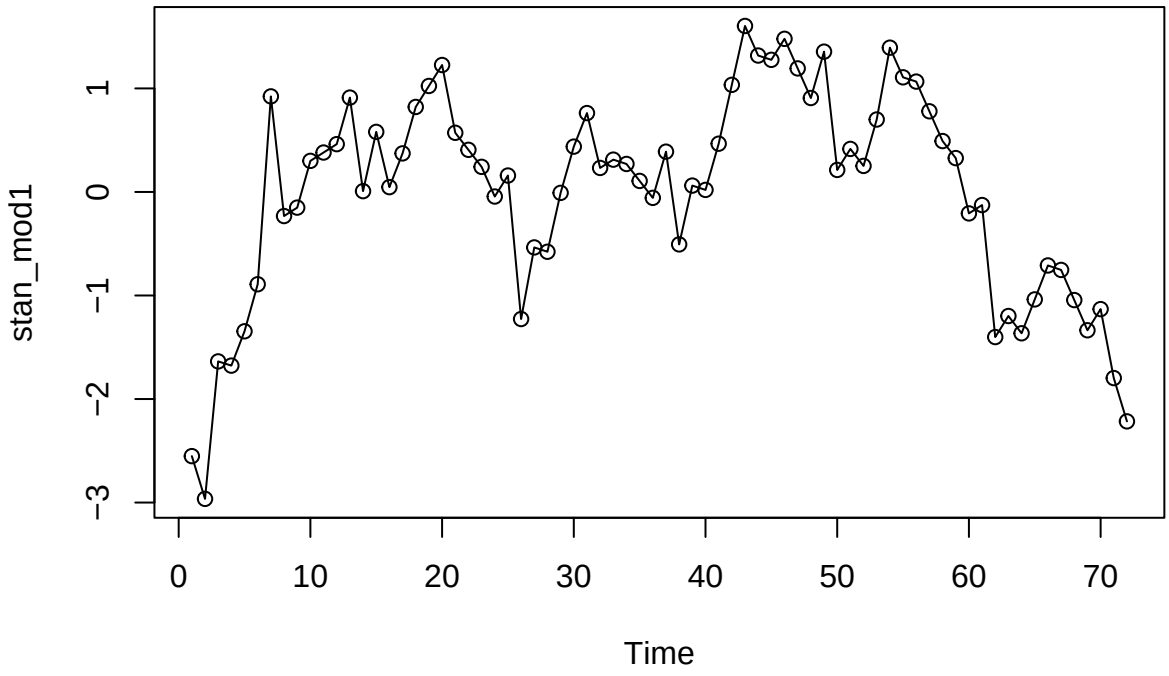
```
plot(wages,type='o',ylab='y')
abline(model1)
```



```
stan_mod1 <- rstandard(model1)
```

(c) Construct and interpret the time series plot of the standardized residuals from part (b)

```
plot(stan_mod1, type = "o", xlab="Time")
```



Interpretation: The time series plot of standardized residuals shows no clear patterns or trends, indicating a good fit for the linear model. The random scattering around zero confirms that the model effectively captures the data's trend.

(d) Use least squares to fit a quadratic time trend to the wages time series. Interpret the regression output. Save the standardized residuals from the fit for further analysis.

```
quad_model <- lm(wages ~ time(wages) + time(wages)^2)
summary(quad_model)
```

```
##
## Call:
## lm(formula = wages ~ time(wages) + time(wages)^2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.23828 -0.04981  0.01942  0.05845  0.13136
##
```

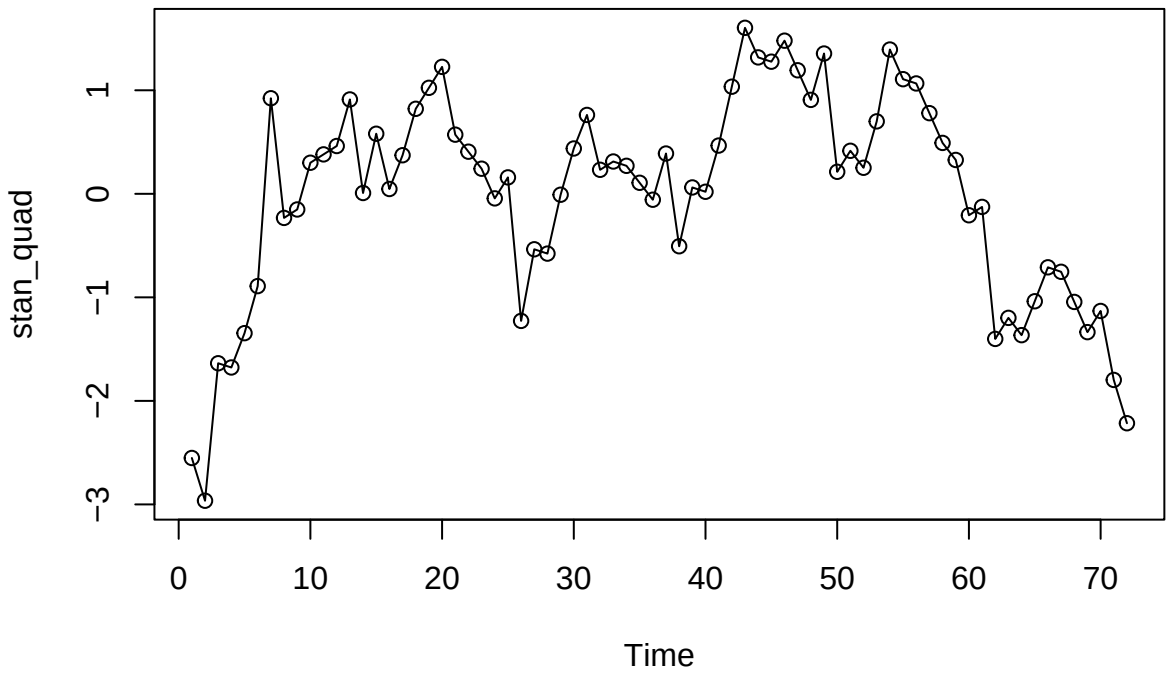
```
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -5.490e+02  1.115e+01  -49.24   <2e-16 ***
## time(wages)  2.811e-01  5.618e-03   50.03   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.08257 on 70 degrees of freedom
## Multiple R-squared:  0.9728, Adjusted R-squared:  0.9724
## F-statistic:  2503 on 1 and 70 DF,  p-value: < 2.2e-16
```

Interpretation: The quadratic model shows a strong upward trend in wages, with a significant positive coefficient for time, indicating wages increase by about 28 cents per month. The high R-squared value (0.9728) suggests the model explains 97% of the variance in the data. Both coefficients are statistically significant, confirming the model's effectiveness in capturing wage trends.

```
stan_quad = rstandard(quad_model)
```

(e) Construct and interpret the time series plot of the standardized residuals from part (d)

```
plot(stan_quad, type="o", xlab = 'Time')
```

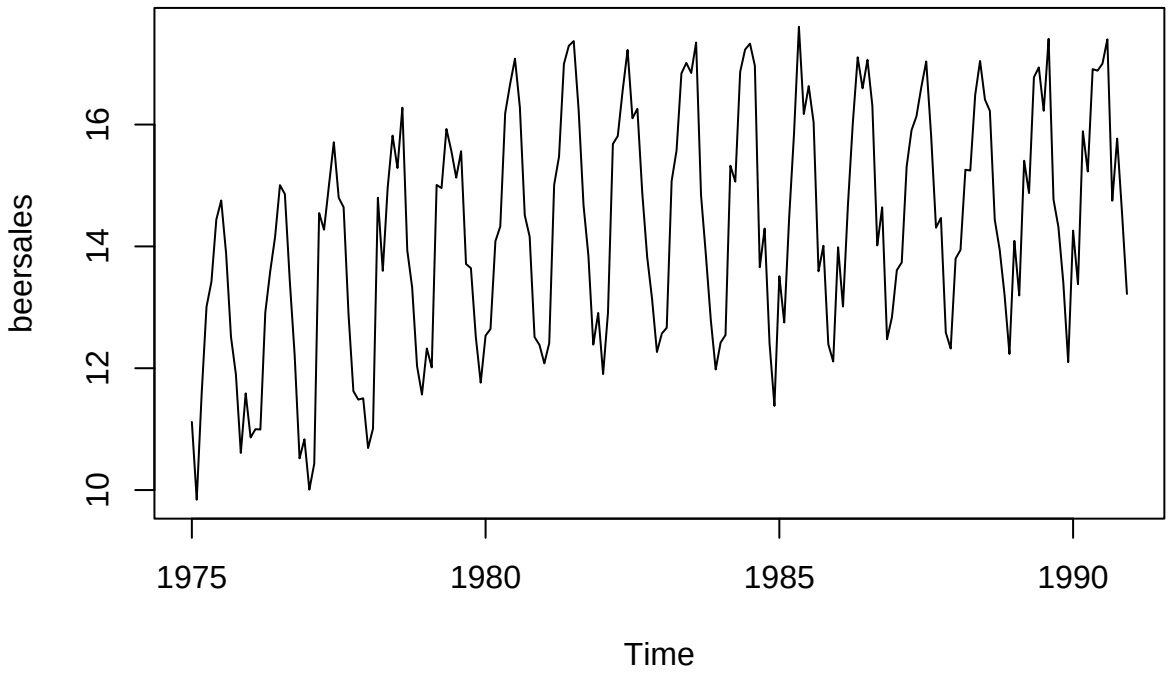


Interpretation: The time series plot shows the average hourly wages in the U.S. apparel and textile industry from 1981 to 1987. Wages initially rise, peak around the middle, and then decline towards the end. This suggests variability with an overall downward trend later in the period. The fluctuations indicate both growth and decline phases.

Question 3.6

The data file beersales contains monthly U.S. beer sales (in millions of barrels) for the period January 1975 through December 1990. (a) Display and interpret the plot the time series plot for these data

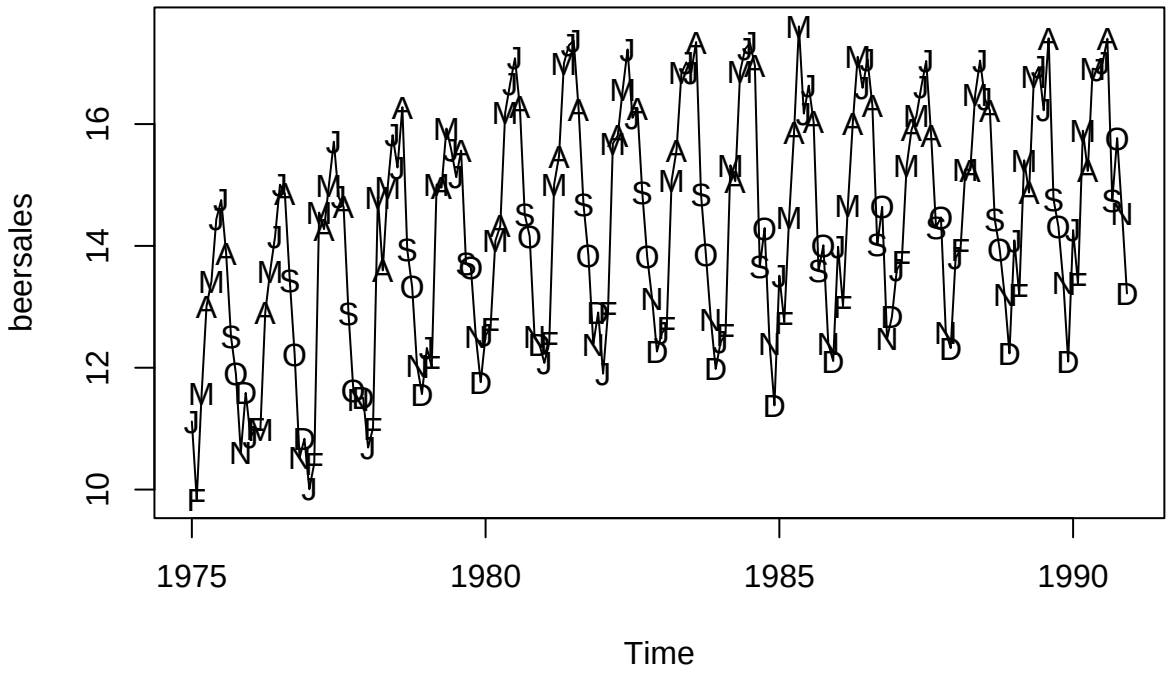
```
data(beersales)
plot(beersales)
```



Clear seasonal trends. There is an initial positive trend from 1975 to around 1981 that then levels out.

(b) Now construct a time series plot that uses separate plotting symbols for the various months. Does your interpretation change from that in part (a)?

```
plot(beersales, type = "l", )
points(y=beersales,x=as.vector(time(beersales)),pch = as.vector(season(beersales)))
```



The time series plot with separate symbols for each month reveals a clear seasonal pattern in U.S. beer sales from 1975 to 1990. The sales peak consistently in the summer months, indicating higher demand during this period. This detailed view confirms the initial positive trend until 1981, followed by a leveling off, and highlights the regular seasonal fluctuations more clearly than the initial plot.

It is now evident that the peaks are in the warm months and the slump in the winter and fall months. December is a particular low point, while May, June, and July seem to be the high points.

(c) Use least squares to fit a seasonal-means trend to this time series. Interpret the regression output. Save the standardized residuals from the fit for further analysis.

```
beer_model1 <- lm(beersales ~ season(beersales))
summary(beer_model1)
```

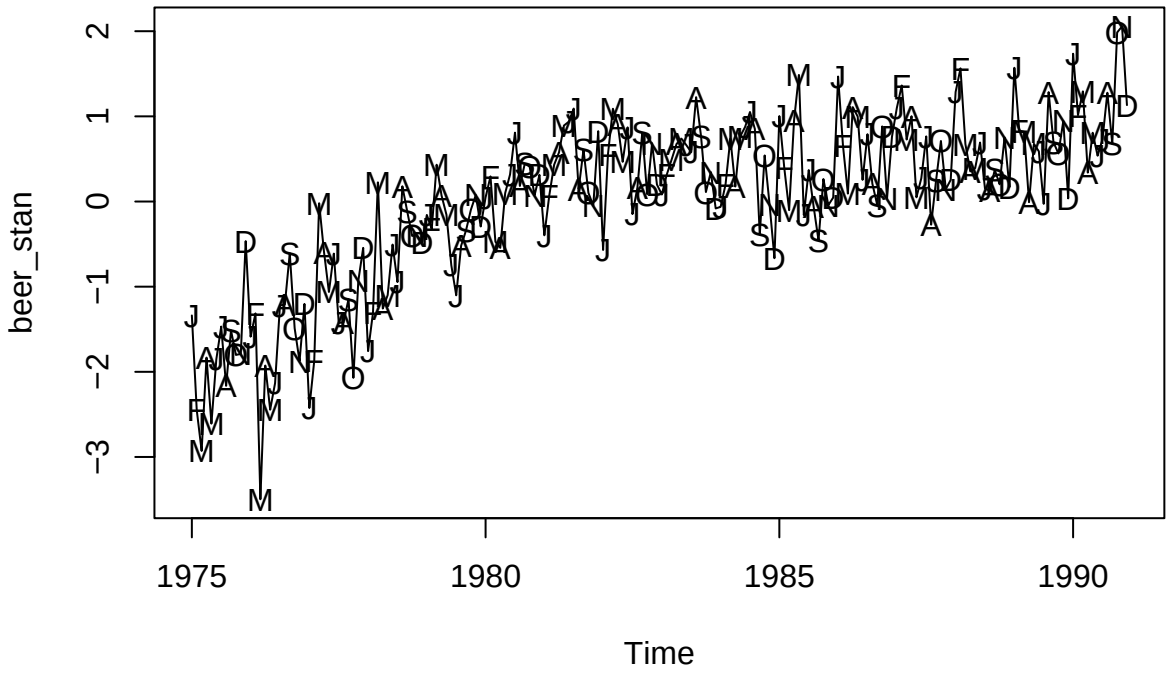
```
##
## Call:
## lm(formula = beersales ~ season(beersales))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.5745 -0.4772  0.1759  0.7312  2.1023
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  12.48568    0.26392  47.309  < 2e-16 ***
## season(beersales)February    -0.14259    0.37324  -0.382 0.702879
```

```
## season(beersales)March      2.08219    0.37324   5.579 8.77e-08 ***
## season(beersales)April      2.39760    0.37324   6.424 1.15e-09 ***
## season(beersales)May        3.59896    0.37324   9.643  < 2e-16 ***
## season(beersales)June       3.84976    0.37324  10.314  < 2e-16 ***
## season(beersales)July       3.76866    0.37324  10.097  < 2e-16 ***
## season(beersales)August     3.60877    0.37324   9.669  < 2e-16 ***
## season(beersales)September  1.57282    0.37324   4.214 3.96e-05 ***
## season(beersales)October    1.25444    0.37324   3.361 0.000948 ***
## season(beersales)November  -0.04797    0.37324  -0.129 0.897881
## season(beersales)December  -0.42309    0.37324  -1.134 0.258487
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.056 on 180 degrees of freedom
## Multiple R-squared:  0.7103, Adjusted R-squared:  0.6926
## F-statistic: 40.12 on 11 and 180 DF,  p-value: < 2.2e-16
```

```
beer_stan <- rstandard(beer_model1)
```

(d) Construct and interpret the time series plot of the standardized residuals from part (c). Be sure to use proper plotting symbols to check on seasonality in the standardized residuals

```
plot(beer_stan ~ time(beersales), type="l", xlab = 'Time')
points(y = beer_stan, x=as.vector(time(beersales)),pch = as.vector(season(beersales)), col = 1)
```



The plot of standardized residuals shows a clear upward trend, indicating that the model did not fully capture the increasing trend in beer sales. The residuals are not randomly scattered around zero, suggesting potential issues with the model fit. The presence of patterns implies that further adjustments might be needed to address these trends effectively.

(e) Use least squares to fit a seasonal-means plus quadratic time trend to the beer sales time series. Interpret the regression output. Save the standardized residuals from the fit for further analysis

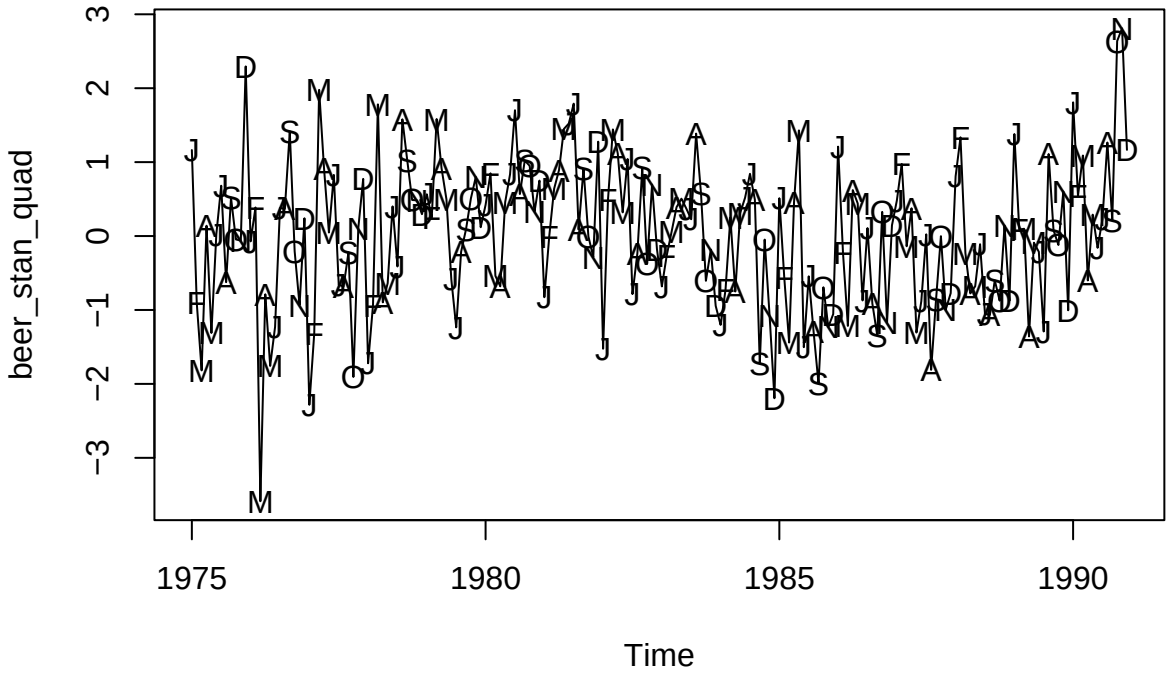
```
beer_model2 <- lm(beersales ~ season(beersales) + time(beersales) +
                  I(time(beersales) ^ 2))
summary(beer_model2)
```

```
##
## Call:
## lm(formula = beersales ~ season(beersales) + time(beersales) +
##     I(time(beersales)^2))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.03203 -0.43118  0.04977  0.34509  1.57572
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                -7.150e+04  8.791e+03  -8.133 6.93e-14 ***
## season(beersales)February  -1.579e-01  2.090e-01  -0.755  0.45099
## season(beersales)March      2.052e+00  2.090e-01   9.818  < 2e-16 ***
## season(beersales)April      2.353e+00  2.090e-01  11.256  < 2e-16 ***
## season(beersales)May        3.539e+00  2.090e-01  16.934  < 2e-16 ***
## season(beersales)June       3.776e+00  2.090e-01  18.065  < 2e-16 ***
## season(beersales)July       3.681e+00  2.090e-01  17.608  < 2e-16 ***
## season(beersales)August     3.507e+00  2.091e-01  16.776  < 2e-16 ***
## season(beersales)September  1.458e+00  2.091e-01   6.972 5.89e-11 ***
## season(beersales)October    1.126e+00  2.091e-01   5.385 2.27e-07 ***
## season(beersales)November  -1.894e-01  2.091e-01  -0.906  0.36622
## season(beersales)December  -5.773e-01  2.092e-01  -2.760  0.00638 **
## time(beersales)             7.196e+01  8.867e+00   8.115 7.70e-14 ***
## I(time(beersales)^2)       -1.810e-02  2.236e-03  -8.096 8.63e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5911 on 178 degrees of freedom
## Multiple R-squared:  0.9102, Adjusted R-squared:  0.9036
## F-statistic: 138.8 on 13 and 178 DF,  p-value: < 2.2e-16
```

```
beer_stan_quad = rstandard(beer_model2)
```

(f) Construct and interpret the time series plot of the standardized residuals from part (e). Again use proper plotting symbols to check for any remaining seasonality in the residuals

```
plot(beer_stan_quad  ~ time(beersales), type="l", xlab = 'Time')
points(y = beer_stan_quad, x=as.vector(time(beersales)),pch = as.vector(season(beersales)), col = 1)
```



Many of the values are still not being predicted successfully but at least we're able to model the long term trend better.

3.12

(Continuation of Exercise 3.6) Consider the time series in the data file beersales. (a) Obtain the residuals from the least squares fit of the seasonal-means plus quadratic time trend model

```
beer_quad_seasonal <- lm(beersales ~ time(beersales) + I(time(beersales)^2) + season(beersales))
beer_resid <- rstandard(beer_quad_seasonal)
```

(b) Perform a runs test on the standardized residuals and interpret the results.

```
runs(beer_resid)
```

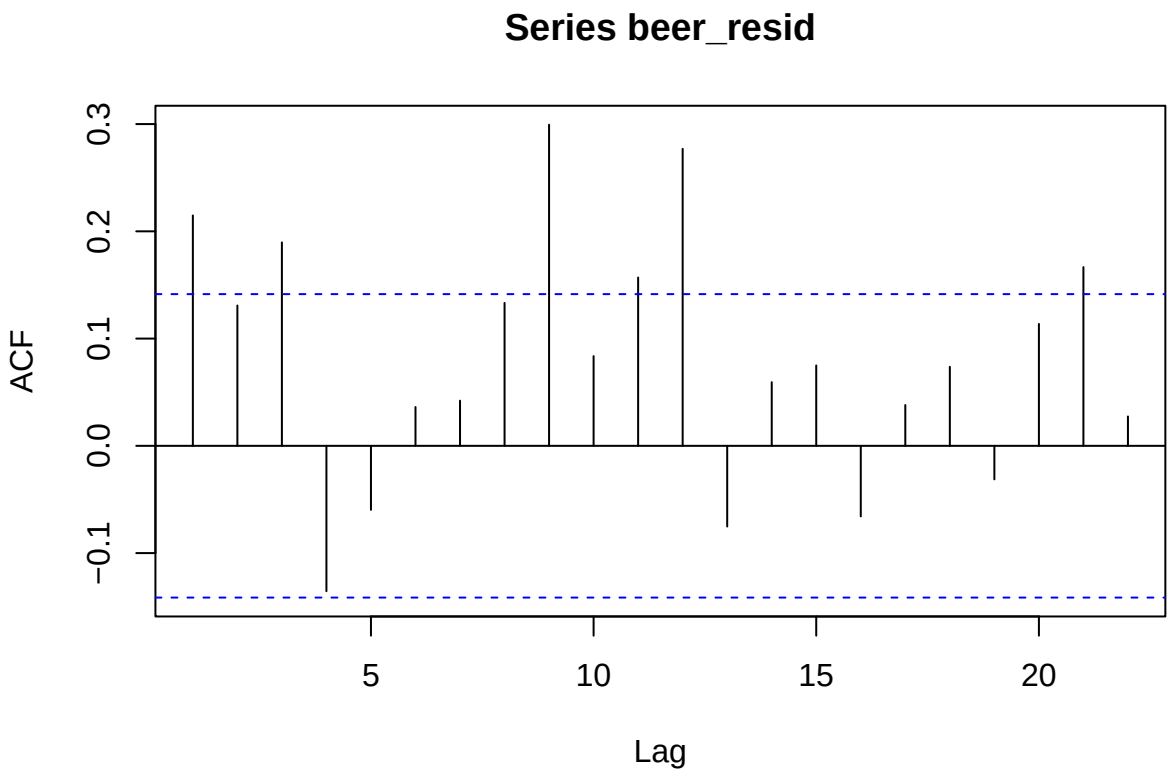
```
## $pvalue
## [1] 0.0127
##
## $observed.runs
## [1] 79
##
## $expected.runs
## [1] 96.625
##
```

```
## $n1
## [1] 90
##
## $n2
## [1] 102
##
## $k
## [1] 0
```

The runs test on the standardized residuals yields a p-value of 0.0127, indicating significant evidence against the null hypothesis of randomness. With only 79 observed runs compared to an expected 96.625, this suggests a non-random pattern in the residuals. The results imply that there may be underlying structures or seasonality not fully captured by the model, indicating potential issues with model fit.

(c) Calculate and interpret the sample autocorrelations for the standardized residuals.

```
acf(beer_resid)
```



The sample autocorrelation function (ACF) plot for the standardized residuals shows several spikes outside the blue dashed lines, which represent the 95% confidence interval. This indicates significant autocorrelation at various lags. The presence of these spikes suggests that the residuals are not purely random and that there may be patterns or dependencies not captured by the model. This implies potential inadequacies in the model's ability to account for all underlying structures in the data.