Lab 7

Pstat 174/274

Analysis of Monthly Sales of Austrailian Wine

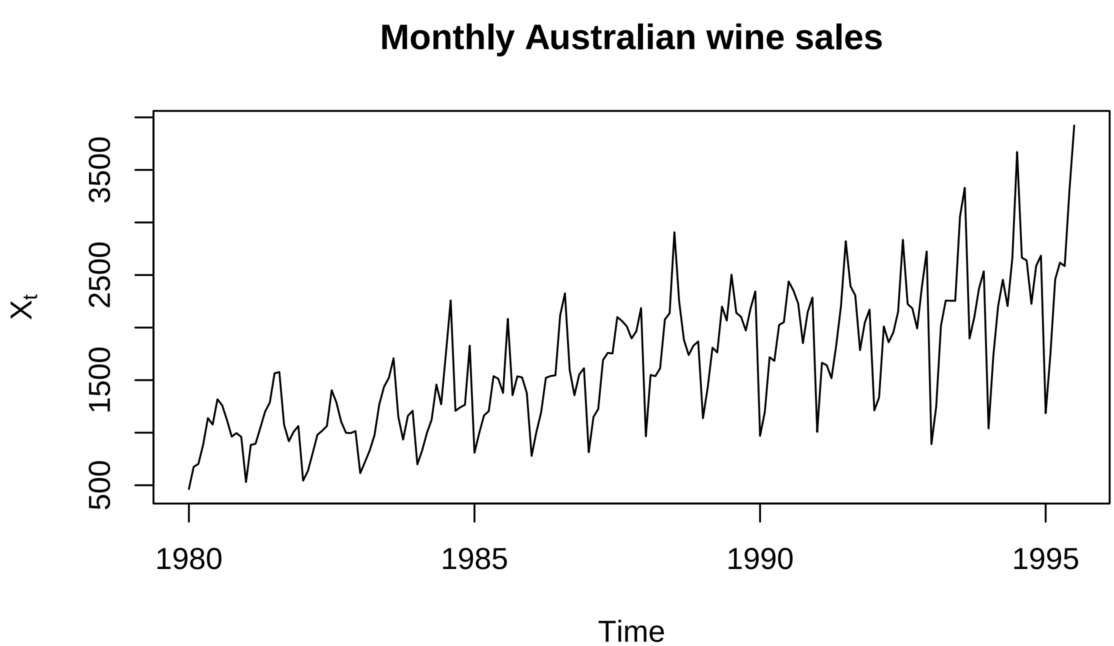
Exploratory Data Analysis

1. Analyze the monthly sales of wine by downloading `monthly-australian-wine-sales-th.csv` from Gauchospace as Lab 4. Let X_t denote the original monthly sales of wine.

```
# Load data
wine.csv <- read.table("monthly-australian-wine-sales-th.csv",
                       sep=",", header=FALSE, skip=1, nrows=187)
wine <- ts(wine.csv[, 2], start = c(1980, 1), frequency=12)
```

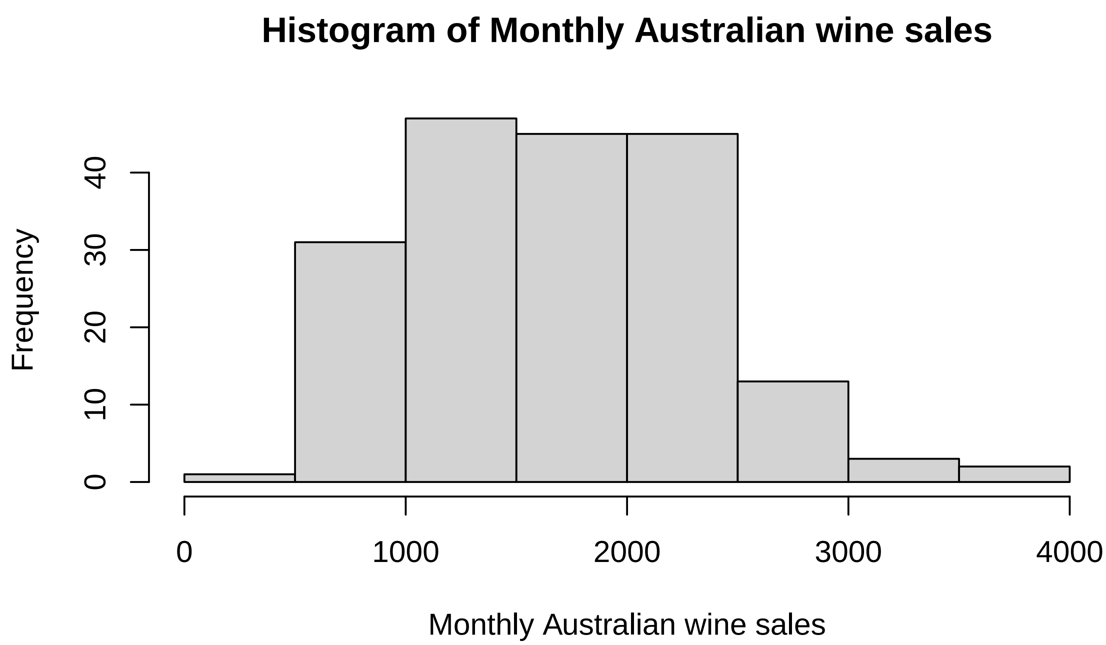
2. Plot the time series. What do you notice?

```
# Plot data
ts.plot(wine, main="Monthly Australian wine sales", ylab=expression(X[t]))
```



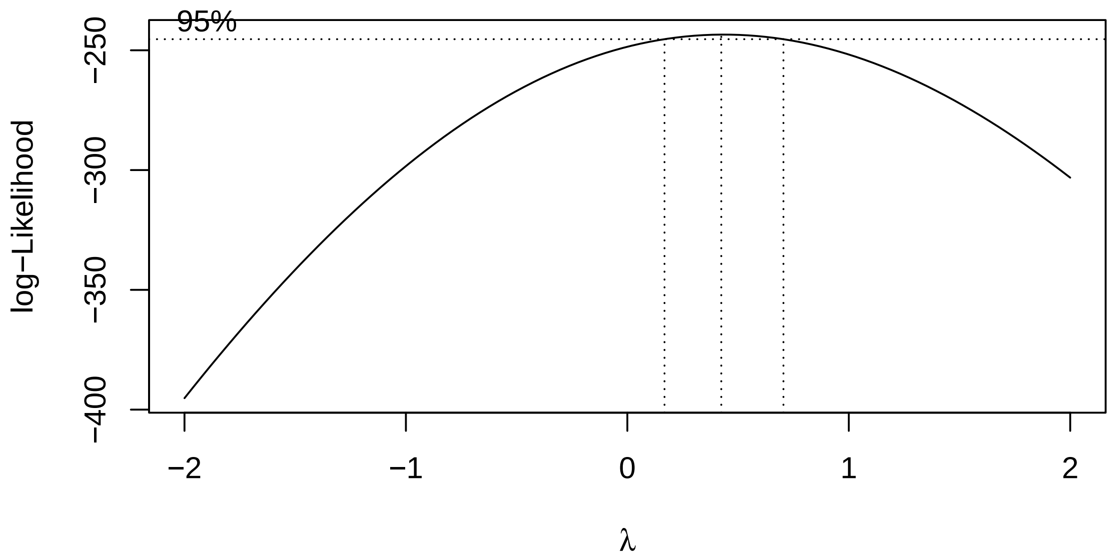
3. Make the data stationary (variance/seasonality/trend). What procedures were used? First, we deal with the variance. Is transformation required? I would like to see a histogram of the data and check that is somewhat symmetric. If not, perhaps a transformation is in order.

```r
hist(wine, main="Histogram of Monthly Australian wine sales",
     xlab="Monthly Australian wine sales")
```

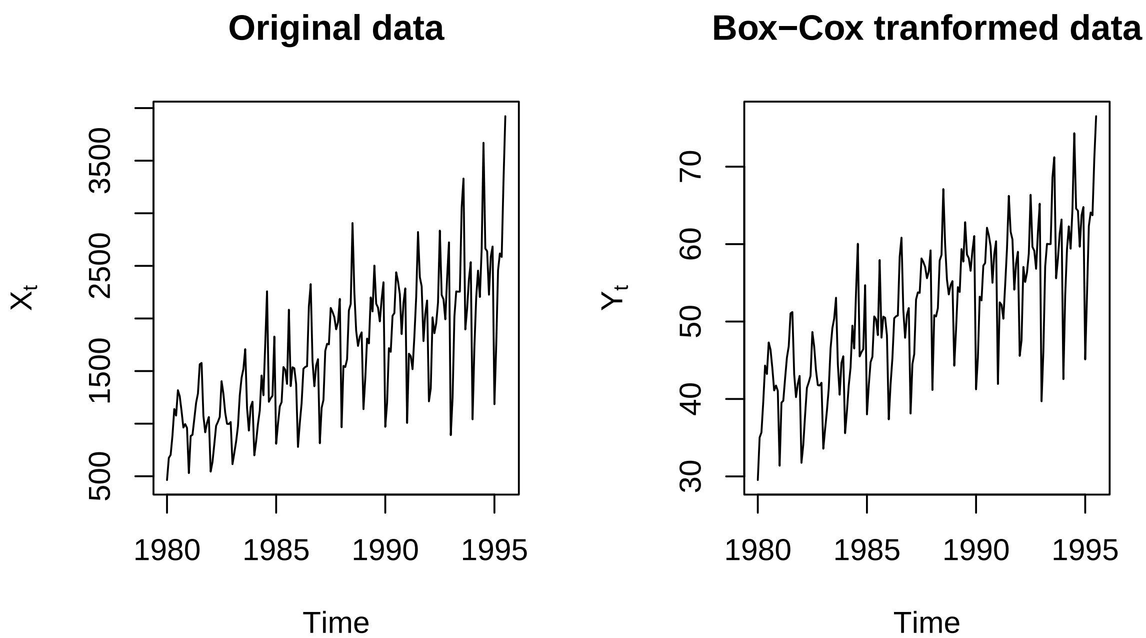


According to the above plot, we decide to transform the data.

```r
# Box-Cox transformation:
t <- 1:length(wine)
bcTransform <- boxcox(wine ~ t, plotit=TRUE)
```




```
lambda <- bcTransform$x[which.max(bcTransform$y)]
wine.bc = (1/lambda)*(wine^lambda-1)

# Plot and compare the two:
par(mfrow=c(1, 2))
ts.plot(wine, main = "Original data",ylab = expression(X[t]))
ts.plot(wine.bc, main = "Box-Cox tranformed data", ylab = expression(Y[t]))
```



Since the Box-Cox plot does not contain neither has 0 nor 1 in the confidence interval for λ, we choose $\lambda \approx 0.424$ that maximizes the log-likelihood. You can also use some other λ's for easier interpretability, and compare the difference between these two transformations.

Next, we should remove the trend and seasonlity. Is second de-trending necessary? One tries and checks variance and acfs to decide on a number of differences. Here, $Y_t \triangleq \frac{1}{\lambda}(X_t^\lambda - 1)$.

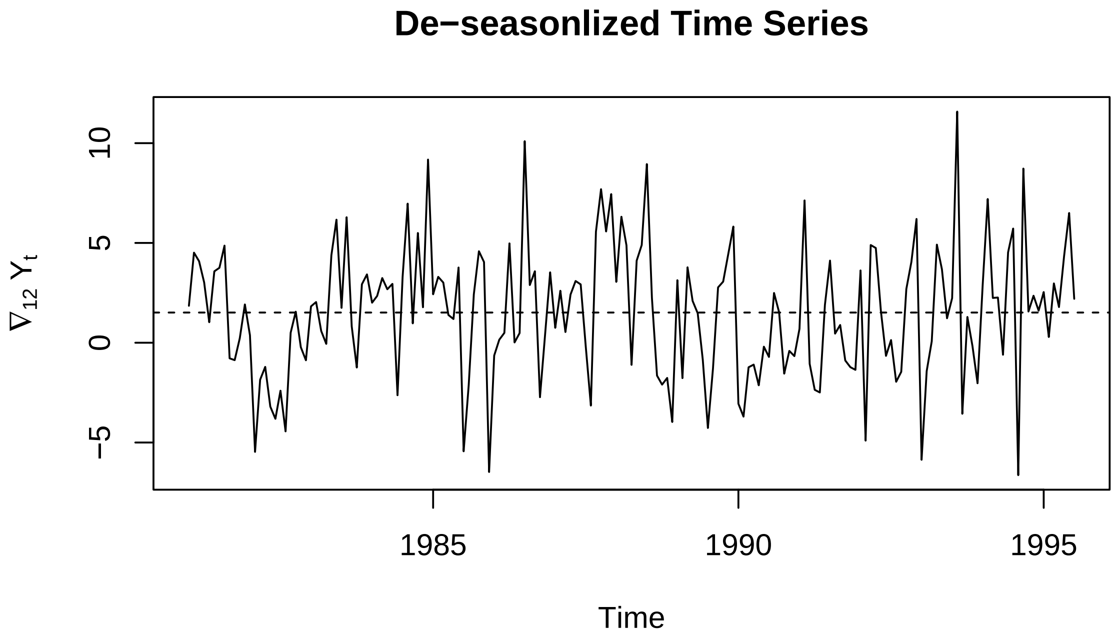
```
# Deseasonlize:
y.12 <- diff(wine.bc, 12)
y.12.2 <- diff(y.12, 12)
var(y.12); var(y.12.2)
```

```
## [1] 10.95448

## [1] 31.85612
```

```
# Detrend:
y.1.12 <- diff(y.12, 1)
var(y.12); var(y.1.12)
```

```
## [1] 10.95448

## [1] 18.18436
```

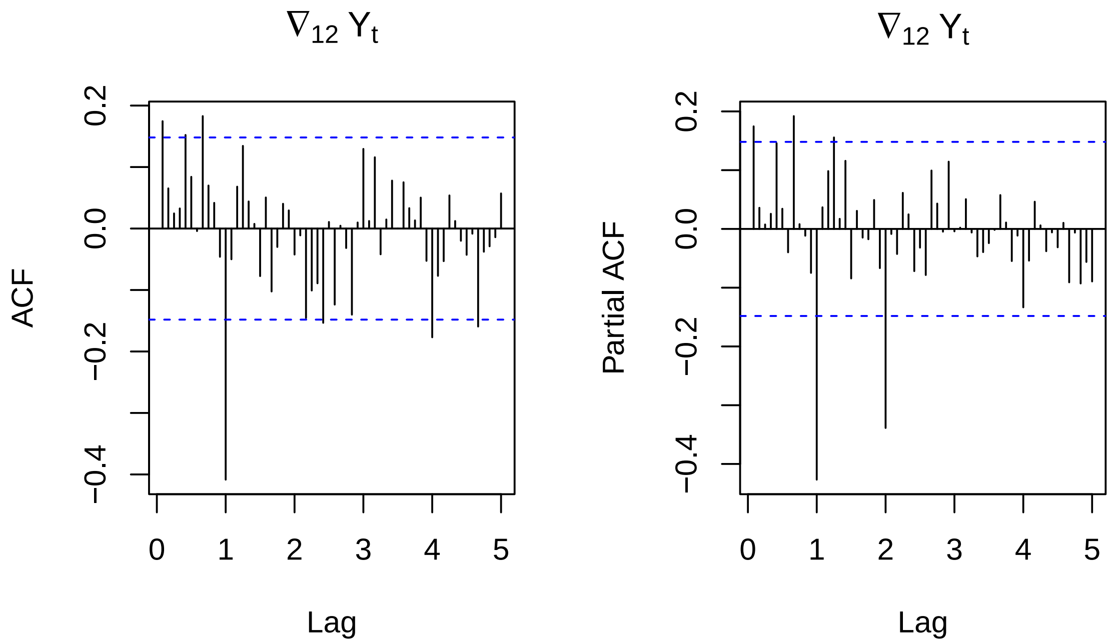
```
# Plot it:
par(mfrow=c(1, 1))
ts.plot(y.12, main="De-seasonlized Time Series",
        ylab=expression(nabla[12]~Y[t]))
abline(h=mean(y.12), lty=2)
```



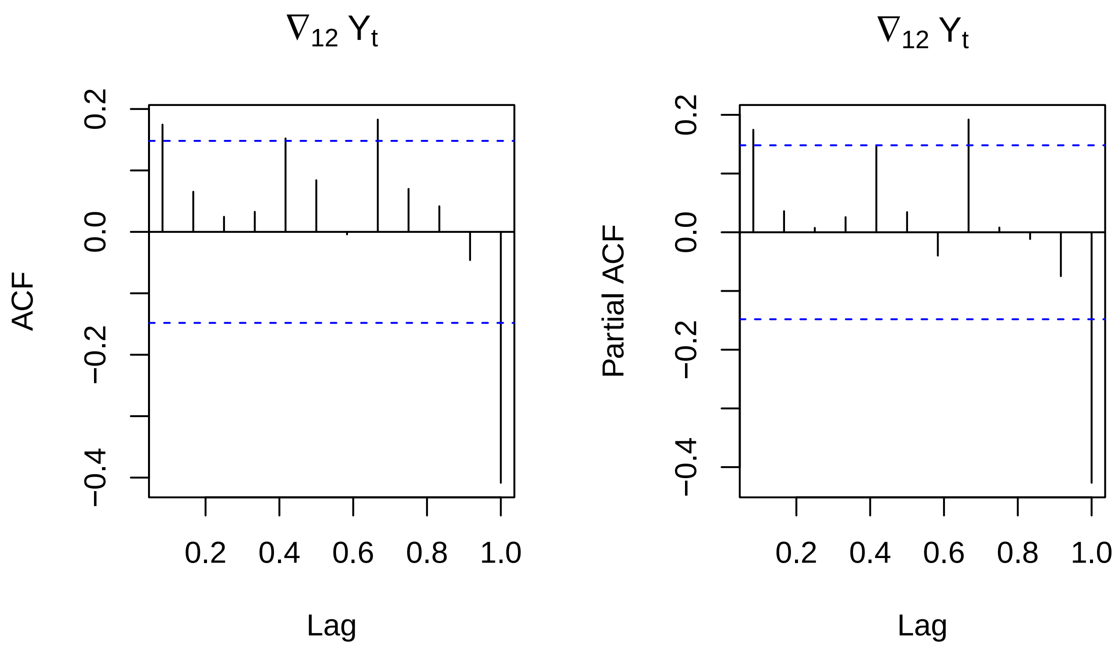
Model Identification

4. Plot the ACF and PACF. What models do they suggest?

```
par(mfrow=c(1, 2))
acf(y.12, lag.max=60, main=expression(nabla[12]~Y[t]))
pacf(y.12, lag.max=60, main=expression(nabla[12]~Y[t]))
```




```r
acf(y.12, lag.max=12, main=expression(nabla[12]~Y[t]))
pacf(y.12, lag.max=12, main=expression(nabla[12]~Y[t]))
```



Model Estimation

5. Fit different ARMA models using maximum likelihood estimation and compare the model fits using AICC (Hint: use `arima()` for estimation and `AICc()` in `library(qpcR)` for model comparison - you will need to install this package into `R` first). Which model is preferred?

```
# Candidate models:
df <- expand.grid(p=0:1, q=0:1, P=0:2, Q=0:1)
df <- cbind(df, AICc=NA)
# Compute AICc:
for (i in 1:nrow(df)) {
  sarima.obj <- NULL
  try(arima.obj <- arima(wine.bc, order=c(df$p[i], 0, df$q[i]),
                         seasonal=list(order=c(df$P[i], 1, df$Q[i]), period=12),
                         method="ML"))
  if (!is.null(arima.obj)) { df$AICc[i] <- AICc(arima.obj) }
  # print(df[i, ])
}
df[which.min(df$AICc), ]
```

```
##    p q P Q     AICc
## 16 1 1 0 1 857.3999
```

```
# Final model:
ind <- which.min(df$AICc)
fit <- arima(wine.bc, order=c(df$p[ind], 0, df$q[ind]),
             seasonal=list(order=c(df$P[ind], 1, df$Q[ind]), period=12),
             method="ML")
```

Model Diagnostics

6. Perform diagnostics on the chosen model fit. Do the residuals appear to be white noise? Are they normally distributed?

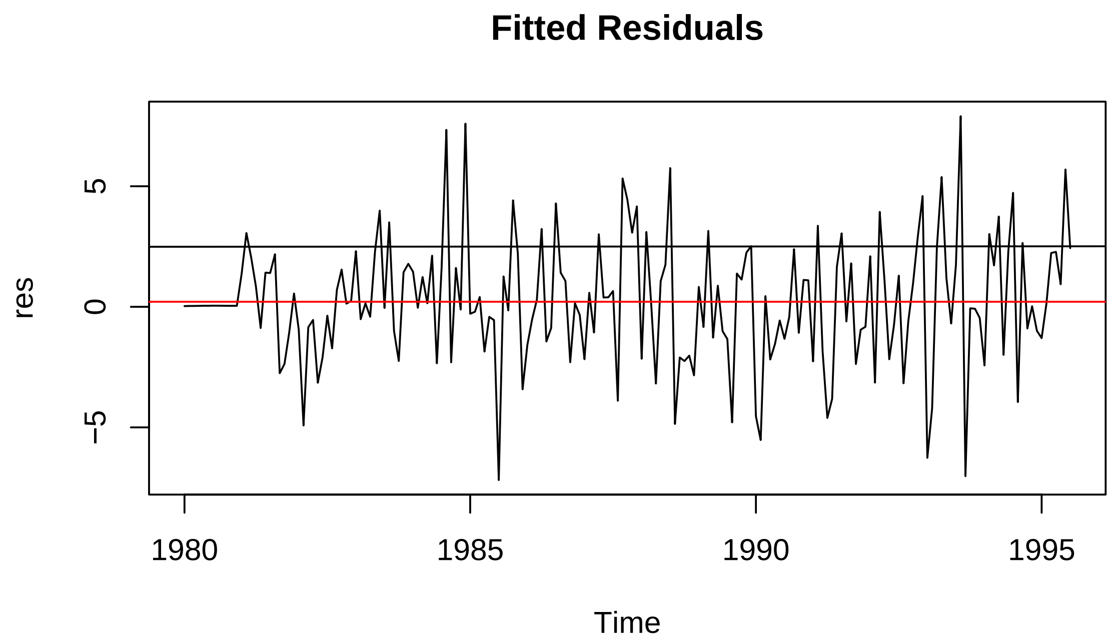
   First, we check the white noise assumption with some plots.

```
# Residual plots:
res <- residuals(fit)
mean(res); var(res)
```

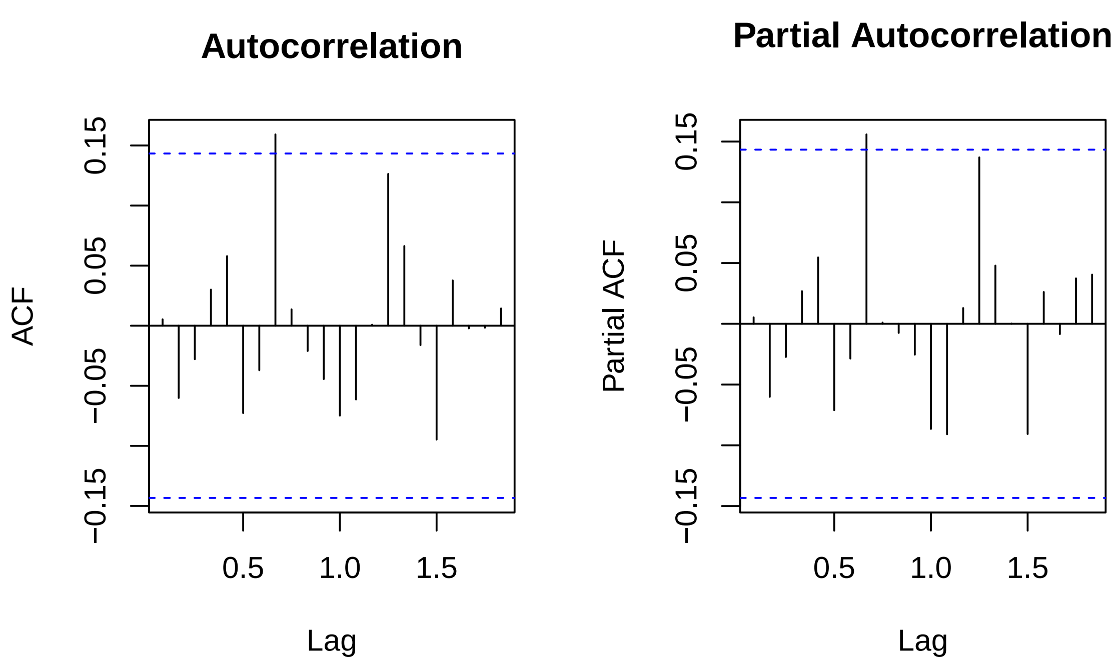
```
## [1] 0.2104241
```

```
## [1] 6.688304
```

```
# layout(matrix(c(1, 1, 2, 3), 2, 2, byrow=T))
par(mfrow=c(1, 1))
ts.plot(res, main="Fitted Residuals")
t <- 1:length(res)
fit.res = lm(res~ t)
abline(fit.res)
abline(h = mean(res), col = "red")
```




```r
# ACF and PACF:
par(mfrow=c(1, 2))
acf(res, main="Autocorrelation")
pacf(res, main="Partial Autocorrelation")
```



We can also use Box-Pierce, Box-Ljung, McLeod-Li tests.

```r
# Test for independence of residuals
Box.test(res, lag = 9, type = c("Box-Pierce"), fitdf = 1)
```

```
##
##  Box-Pierce test
##
## data:  res
## X-squared = 7.6436, df = 8, p-value = 0.469
```

```r
Box.test(res, lag = 9, type = c("Ljung-Box"), fitdf = 1)
```

```
##
##  Box-Ljung test
##
## data:  res
## X-squared = 8.0155, df = 8, p-value = 0.432
```

```r
Box.test(res^2, lag = 9, type = c("Ljung-Box"), fitdf = 0)
```

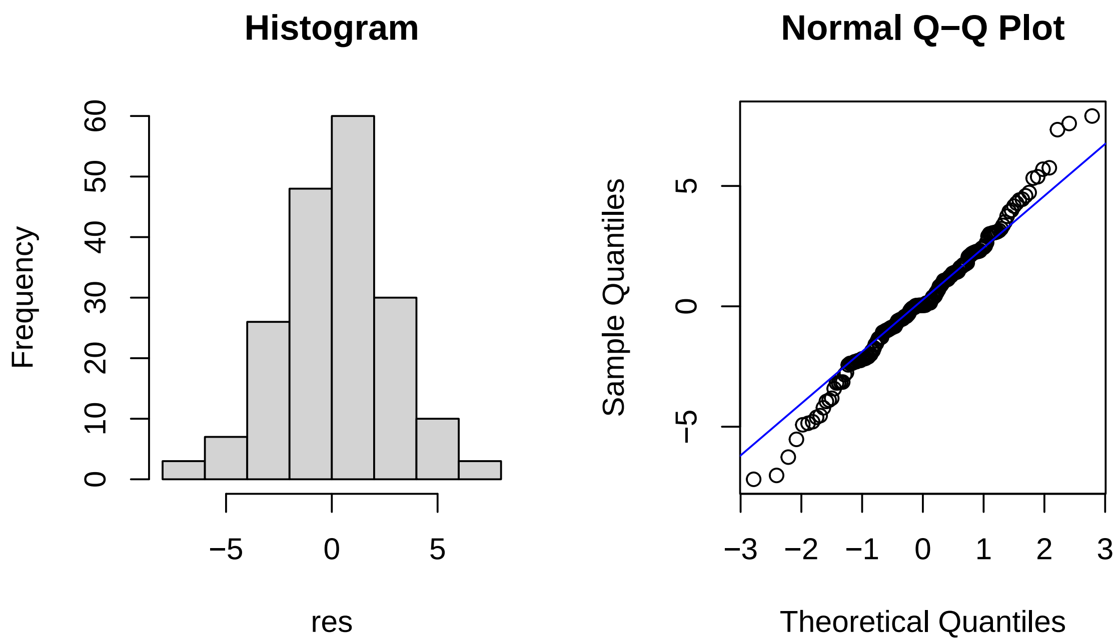
```
##
##  Box-Ljung test
##
## data:  res^2
## X-squared = 12.043, df = 9, p-value = 0.2109
```

Then, We check normality assumption.

```r
# Test for normality of residuals
shapiro.test(res)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  res
## W = 0.98874, p-value = 0.1456
```

```r
# Histogram and QQ-plot:
par(mfrow=c(1,2))
hist(res,main = "Histogram")
qqnorm(res)
qqline(res,col ="blue")
```



Hence, the final model will be

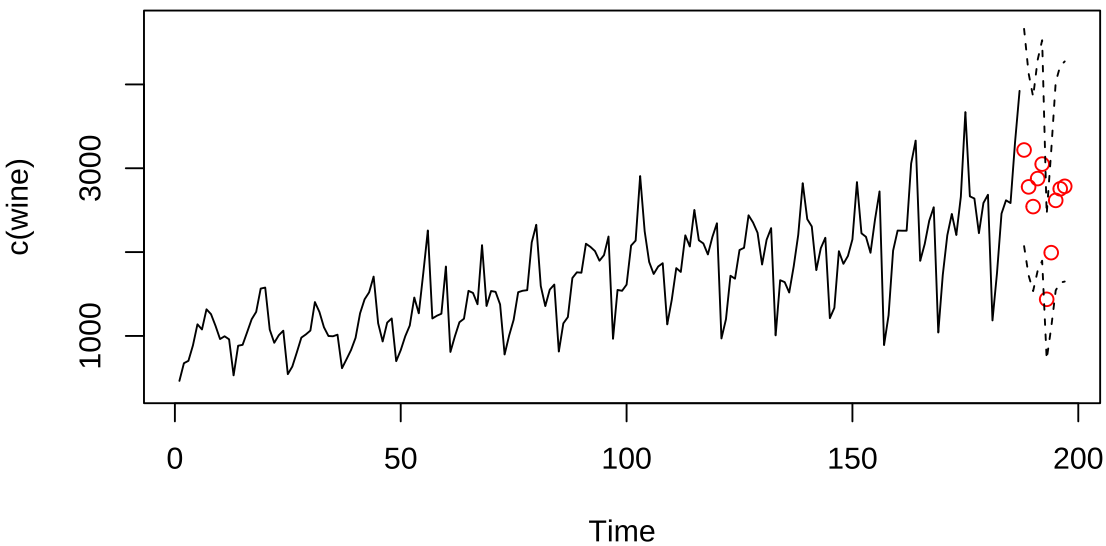
$$(1-0.9968B)(1-B^{12})Y_t = (1-0.8421B)(1-0.6846B)Z_t,$$

where $Z_t \sim N(0, 7.156)$.

Data Forecasting

7. Forecast the next 10 observations using your model.

```
# Predict 10 future observations and plot
par(mfrow=c(1, 1))
mypred <- predict(fit, n.ahead=10)
ts.plot(c(wine), xlim=c(1, length(wine) + 10), ylim=c(min(wine)*0.8, max(wine)*1.2))
points((length(wine) + 1):(length(wine) + 10), col="red",
      (lambda*mypred$pred + 1)^(1/lambda))
lines((length(wine) + 1):(length(wine) + 10),
      (lambda*mypred$pred + 1.96*mypred$se + 1)^(1/lambda), lty=2)
lines((length(wine) + 1):(length(wine) + 10),
      (lambda*mypred$pred - 1.96*mypred$se + 1)^(1/lambda), lty=2)
```


c(wine)
1000
3000
0
50
100
150
200
Time