MATH545_260677676_A4

##260677676 Dan Seol

```r
library(knitr)
```

##5.9 \ Let $\{X_1, ..., X_n\}$ be of AR(p) process $(n > p)$, i.e.

$$X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + ... + \phi_p X_{t-p} + Z_t$$

where $\{Z_t\} \sim WN(0, \sigma^2)$ \ Use equation 5.2.9 to show that likelihood is

$$L(\phi, \sigma^2) = (2\pi\sigma^2)^{-n/2}(detG_p)^{-1/2}*$$

$$exp\{\frac{-1}{\sigma^2}[\mathbb{X}_p^T G_p^{-1}\mathbb{X}_\shortmid + \sum_{t=p+1}^{n}(X_t - \phi_1 X_{t-1} - ... - \phi_p X_{t-p})^2]\}$$

where $\mathbb{X}_p = (X_1, ..., X_p)^T$ and $G_p = \frac{1}{\sigma^2}\Gamma_p$ Solution. \ We state the equation 5.2.9.

$$L(\phi, \theta, \sigma^2) = \frac{(2\pi\sigma)^{-n/2}}{\sqrt{\prod_{i=0}^{n} r_{i-1}}} exp\{-\frac{1}{2\sigma^2}\sum_{j=1}^{n}\frac{(X_j - \hat{X}_j)^2}{r_{j-1}}\}$$

From class we know $(D_n = [v_i]_{i=1}^n)$ will be a diagonal matrix

$$\mathbb{X}_n = C_n(\mathbb{X}_n - \hat{\mathbb{X}}_n)$$

and

$$\Gamma_n = C_n D_n C_n^T$$

finally

$$det(\Gamma_n) = \prod_{i=1}^{n} v_{i-1} = (\sigma^2)^{n-1}\prod_{i=1}^{n} r_{i-1}$$

implying:

$$\mathbb{X}_n^T \Gamma_n^{-1}\mathbb{X}_n = (\mathbb{X}_n - \widehat{\mathbb{X}}_n)C_n^T(C_n^T)^{-1}D_n^{-1}C_n^{-1}C_n(\mathbb{X}_n - \widehat{\mathbb{X}}_n) = (\mathbb{X}_n - \widehat{\mathbb{X}}_n)D_n^{-1}(\mathbb{X}_n - \widehat{\mathbb{X}}_n) = \sum_{i=1}^{n}\frac{\left(X_i - \hat{X}_i\right)^2}{v_{i-1}}$$

$$= \frac{1}{\sigma^2}\sum_{i=1}^{n}\frac{\left(X_i - \hat{X}_i\right)^2}{r_{i-1}}$$

From them we can deduce two following facts:

$$\mathbb{X}_p^T G_p^{-1}\mathbb{X}_p = \sigma^2\mathbb{X}_p^T\Gamma_p^{-1}\mathbb{X}_p = \sum_{i=1}^{p}\frac{\left(X_n - \hat{X}_n\right)^2}{r_{i-1}}$$

$$det(G_p) = (\frac{1}{\sigma^2})^{n-1}det(\Gamma_n) = \prod_{i=1}^{n} r_{i-1}$$

And we notice $r_i = 1$ for $i > p$ since for $i > p$ $\hat{X}_i = \phi_1 X_{i-1} + ... + \phi_p X_{i-p}$ Now the equation 5.2.9. can be rewritten as

$$\frac{(2\pi\sigma)^{-n/2}}{\sqrt{\prod_{i=1}^{p} r_{i-1}}} exp\{-\frac{1}{2\sigma^2}\sum_{j=1}^{p}\frac{(X_j - \hat{X}_j)^2}{r_{j-1}} + \sum_{j=p+1}^{n}(X_j - \phi_1 X_{j-1} - ... - \phi_p X_{j-p})^2\}$$

$$= (2\pi\sigma^2)^{-n/2}(detG_p)^{-1/2} * exp\{\frac{-1}{\sigma^2}[\mathbb{X}_p^T G_p^{-1}\mathbb{X}_{\shortmid} + \sum_{t=p+1}^{n}(X_t - \phi_1 X_{t-1} - ... - \phi_p X_{t-p})^2]\}$$

as we needed. \ ##5.10 From Section 5.2 we know what $\sigma^2\Gamma_n$ look like.

thus we would have

$$det(G_2)^{-1} = (1-\phi_2^2)^2 - \phi_1^2(1+\phi_2)^2$$

$$\mathbb{X}_2^T G_2^{-1}\mathbb{X}_2 = (1-\phi_2^2)(X_1^2 + X_2^2) - 2\phi_1(1-\phi_2)X_1X_2$$

And from taking the log on the result 5.9 we get the log likelihood:

$$-\frac{n}{2}\ln(2\pi\sigma^2) - \frac{1}{2}\ln((1-\phi_2)^2 - \phi_1^2(1+\phi_2^2)) - \frac{1}{2\sigma^2}\{(1-\phi_2^2)(X_1^2+X_2^2) - 2\phi_1(1+\phi_2)X_1X_2 + \sum_{j=3}^{n}(X_j - \phi_1X_{j-1} - \phi_2X_{j-2})^2\}$$

Take our

$$S(\phi,\sigma^2) = -\frac{1}{2\sigma^2}\{(1-\phi_2^2)(X_1^2+X_2^2) - 2\phi_1(1+\phi_2)X_1X_2 + \sum_{j=3}^{n}(X_j - \phi_1X_{j-1} - \phi_2X_{j-2})^2\}$$

If you differentiate $S(\phi,\sigma^2)$ with respect to ϕ_1

$$\frac{\partial S}{\partial\phi_1} = \frac{-1}{2\sigma^2}\{-2(1+\phi_2)X_1X_2 - 2\sum_{j=3}^{n}X_{j-1}(X_j - \phi_1X_{j-1} - \phi_2X_{j-2})\} =$$

$$\sigma^{-2}\{(1+\phi_2)X_1X_2 + \sum_{j=3}^{n}X_{j-1}(X_j - \phi_1X_{j-1} - \phi_2X_{j-2})\} = 0$$

and

$$\frac{\partial S}{\partial\phi_2} = \frac{-1}{2\sigma^2}\{-2\phi_2(X_1^2+X_2^2) - 2\phi_1X_1X_2 + -2\sum_{j=3}^{n}X_{j-1}(X_j - \phi_1X_{j-2} - \phi_2X_{j-2})\} =$$

$$\frac{1}{\sigma^2}\{\phi_2(X_1^2+X_2^2) + \phi_1X_1X_2 + \sum_{j=3}^{n}X_{j-1}(X_j - \phi_1X_{j-2} - \phi_2X_{j-2})\} = 0$$

$$\frac{1}{(1-\phi_2^2)-\phi_1} + \frac{1}{\sigma^2}(1+\phi_2X_1X_2 - \sum_{j=3}^{n}X_{j-1}(X_j - \phi_1X_{j-1} - \phi_2X_{j-2})$$

but with respect to ϕ_2 we don't get a linear equation..

##5.12

$$G_1^{-1} = (1-\phi^2) = \det(G_1)^{-1}$$

We have the log likelihood

$$-\frac{n}{2}(\ln 2\pi\sigma^2) - \frac{n}{2}(ln(1-\phi^2)) - \frac{1}{2\sigma^2}((1-\phi^2)X_1^2 + \sum_{j=2}^{n}(X_j - \phi X_{j-1})^2)$$

If we differentiate this with respect to ϕ, and set it to zero, we have

$$-\frac{n}{2}\frac{-2\phi}{1-\phi^2} - \frac{1}{2\sigma^2}(-2\phi X_1^2 - 2\sum_{j=2}^{n}(X_j - \phi X_{j-1})) = 0$$

$$n\frac{\phi}{1-\phi^2} + \frac{1}{\sigma^2}(\phi X_1^2 + \sum_{j=1}^{n}X_{j-1}(X_j - X_{j-1})) = 0$$

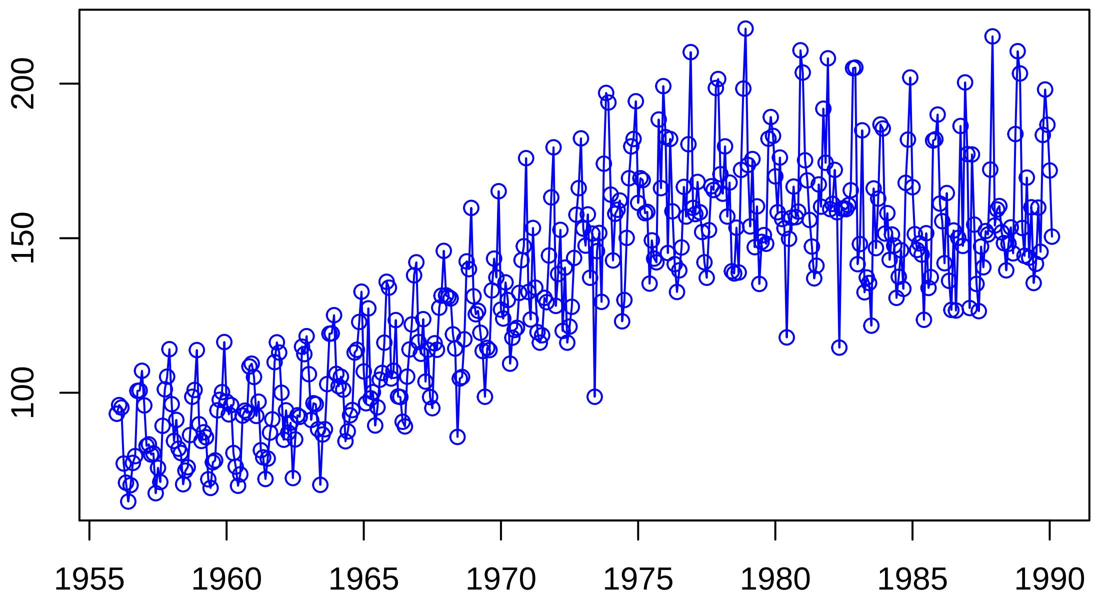
$$\implies n\phi + \frac{1}{\sigma^2}(\phi(1-\phi^2)X_1^2 + (1-\phi^2)\sum_{j=1}^{n} X_{j-1}(X_j - X_{j-1})) = 0$$

which is a cubic equation.

Question 6.9 a) First, we set up and plot our data:

```
library(tidyverse)
library(tidyquant)
library(tseries)
library(itsmr)
library(forecast)
library(tibbletime)
library(tsbox)
library(gridExtra)
library(here)
library(fpp2)

# Defining data
beer_data <- dget("beer.Rput")
beer_ts <- ts(beer_data[1:410], frequency=12, start=c(1956, 1))
plotc(beer_ts)
```



Remark that the data shows heteroscedasticity with increasing variance , and that there is an upward trend. The ADF and KPSS tests confirm that the series is not stationary:

```
adf.test(beer_ts)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  beer_ts
## Dickey-Fuller = -4.6428, Lag order = 7, p-value = 0.01
## alternative hypothesis: stationary
```

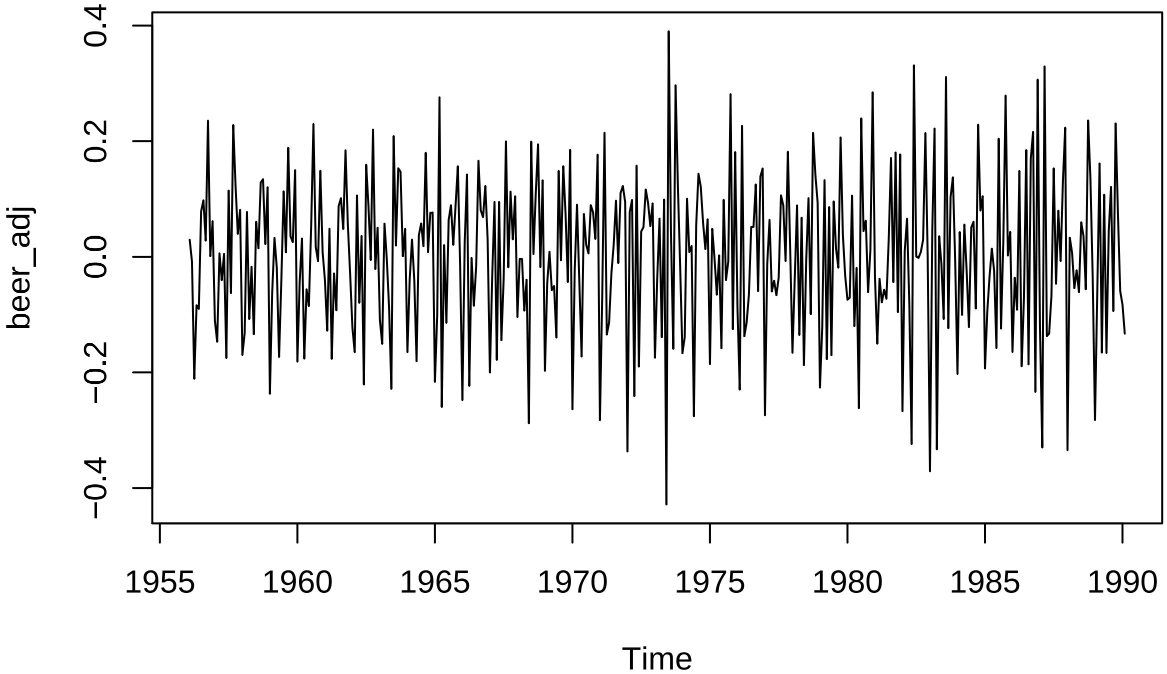
```
kpss.test(beer_ts)
```

```
##
##  KPSS Test for Level Stationarity
```

```
##
## data:  beer_ts
## KPSS Level = 5.7106, Truncation lag parameter = 5, p-value = 0.01
```

We take the log-difference of the series to adjust for variance and trend:

```
beer_log <- log(beer_ts)
beer_adj <- diff(beer_log)
plot(beer_adj)
```



Visual inspection suggests that the series might be stationary. Running the ADF and KPSS tests:

```
adf.test(beer_adj)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  beer_adj
## Dickey-Fuller = -17.162, Lag order = 7, p-value = 0.01
## alternative hypothesis: stationary
```

```
kpss.test(beer_adj)
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  beer_adj
## KPSS Level = 0.010296, Truncation lag parameter = 5, p-value = 0.1
```

The ADF test rejects the hypothesis that the series is non-stationary, and the KPSS test fails to reject the hypothesis that the series is stationary. At this stage, we can conclude that we have enough evidence to claim that the series is stationary after taking the log and differencing once. To keep things simple when back-transforming our model, we will only pass the log-series to the auto.arima function, and let it do the differencing:

```
beer_arima <- auto.arima(beer_log, seasonal=TRUE, stepwise=FALSE, approximation=FALSE)
beer_forecast <- forecast(beer_arima)
summary(beer_arima)
```

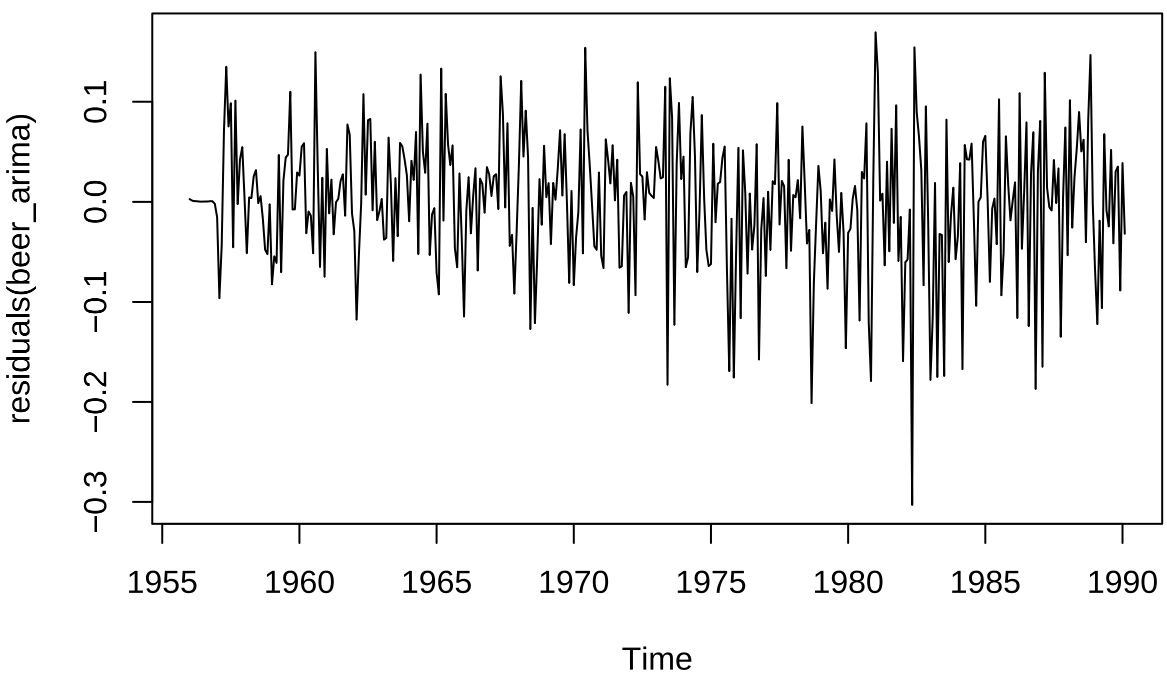
```
## Series: beer_log
## ARIMA(0,1,3)(0,1,2)[12]
##
## Coefficients:
##           ma1      ma2     ma3     sma1     sma2
##       -1.0663  -0.0187  0.1943  -0.7240  -0.1411
## s.e.   0.0534   0.0883  0.0627   0.0532   0.0520
##
## sigma^2 estimated as 0.004704:  log likelihood=494.31
## AIC=-976.61   AICc=-976.4   BIC=-952.71
##
## Training set error measures:
##                         ME       RMSE        MAE         MPE     MAPE
## Training set -0.0005984734 0.06706188 0.05045498 -0.01413405 1.036919
##                   MASE         ACF1
## Training set 0.7303307 -0.008357447
```

Hence the best model is an ARIMA(0,1,3)(0,1,2)[12].

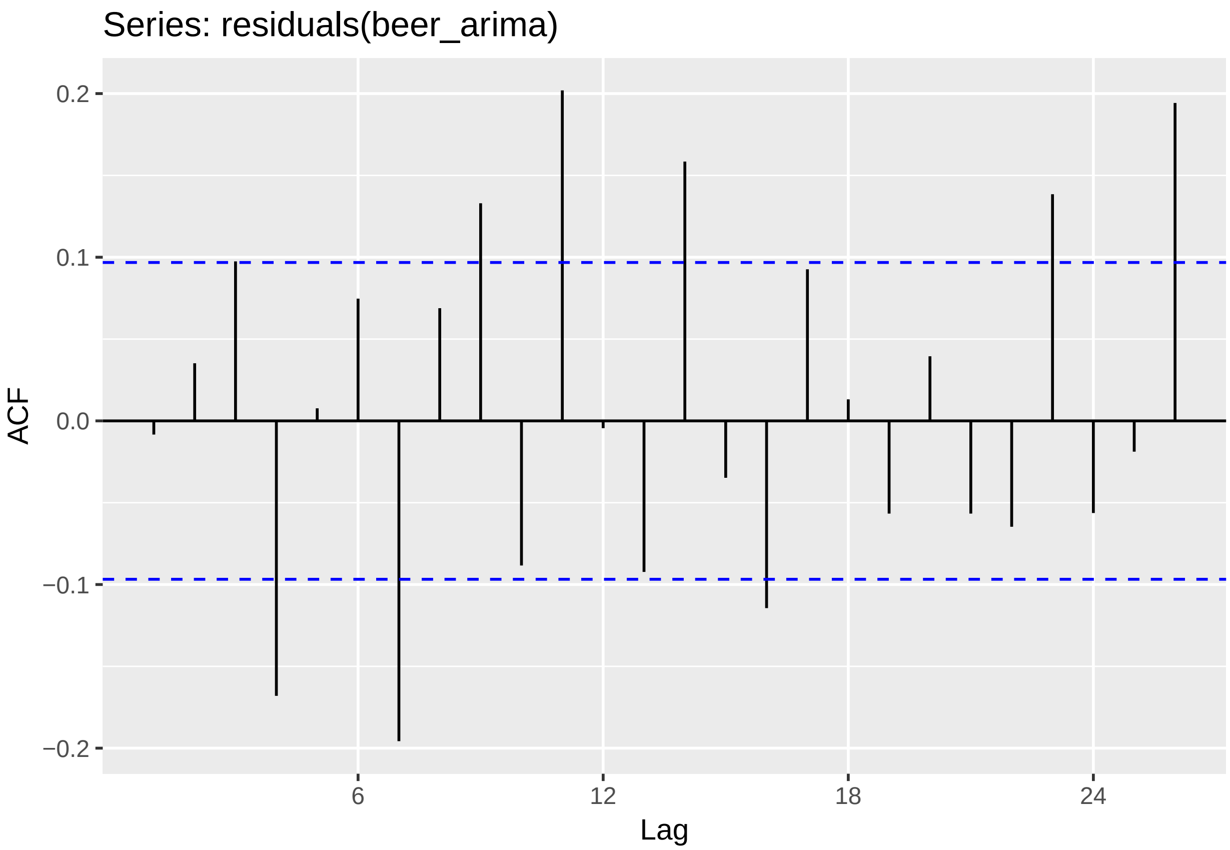
b) 95% Bounds

c) Checking the residuals Taking a look at the model residuals:

```
plot(residuals(beer_arima))
```



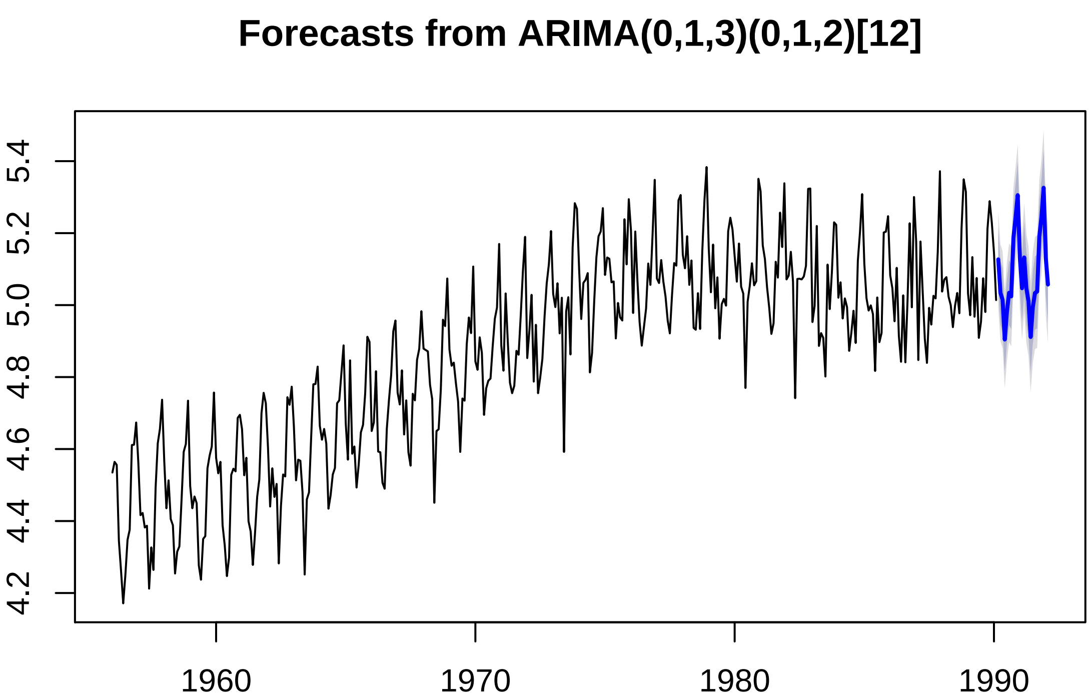

```
ggAcf(residuals(beer_arima))
```



The residual plot shows us that we can treat the residuals as approximate white noise while the ACF plot shows significant correlations at certain lags, this might suggest that the series is underdifferenced.

d) graphing the forecasts with 95% prediction bounds:

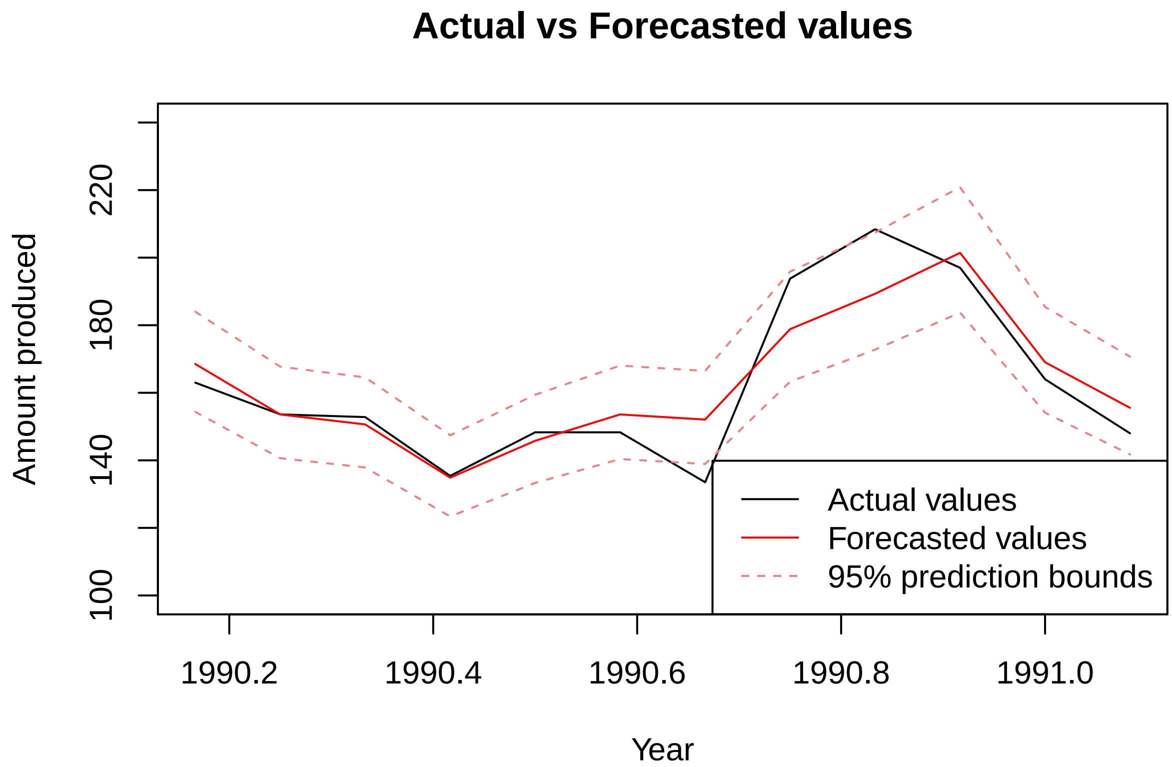
```
plot(forecast(beer_arima))
```



Zooming in

and comparing the actual last 12 values with the forecasted values & prediction bounds:

```
forecasts <- ts(exp(beer_forecast$mean[1:12]), start=c(1990, 3), frequency=12)
actual <- ts(beer_data[411:422], start=c(1990,3), frequency=12)
ts.plot(actual, col="black", ylim=c(100,240), main="Actual vs Forecasted values", ylab="Amount produced
lines(forecasts, col="red")
lines(ts(exp(beer_forecast$upper[1:12]), start=c(1990, 3), frequency=12), col="lightcoral", lty=2)
lines(ts(exp(beer_forecast$lower[1:12]), start=c(1990, 3), frequency=12), col="lightcoral", lty=2)
legend("bottomright", legend=c("Actual values", "Forecasted values", "95% prediction bounds"), col=c("b
```



e) The numerical values for the next-12 forecasts are:

```
forecasts[1:12]
```

```
##  [1] 168.5161 153.5746 150.6262 134.8654 145.7758 153.5844 152.0629
##  [8] 178.8348 189.3010 201.4252 169.0407 155.5360
```

And the 95% bound numerical values are:

```
# Lower bound
(exp(beer_forecast$lower[1:12]))
```

```
##  [1] 154.3364 140.6251 137.8816 123.3902 133.3033 140.3716 138.9099
##  [8] 163.2832 172.7519 183.7238 154.1082 141.7259
```

```
# Upper bound
(exp(beer_forecast$upper[1:12]))
```

```
##  [1] 183.9985 167.7167 164.5487 147.4078 159.4153 168.0409 166.4613
##  [8] 195.8677 207.4355 220.8322 185.4202 170.6917
```

We note here that the last value of the series is within the 95% prediction bound.

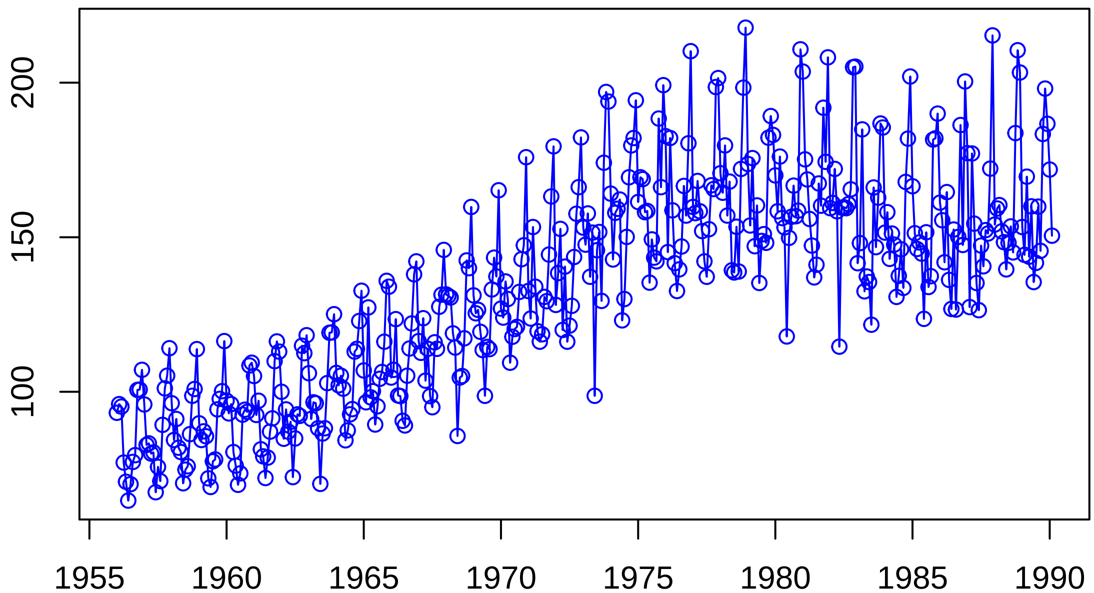
f) The actual forecast error is the difference between the actual values and the predicted values:

```r
actual - forecasts
```

```
##               Jan          Feb          Mar          Apr          May
## 1990                             -5.51608552   0.02537711   2.17383579
## 1991  -5.04074849  -7.53598898
##               Jun          Jul          Aug          Sep          Oct
## 1990   0.53458871   2.52420306  -5.28443498 -18.56290068  14.96518573
## 1991
##               Nov          Dec
## 1990  19.09897925  -4.42524417
## 1991
```

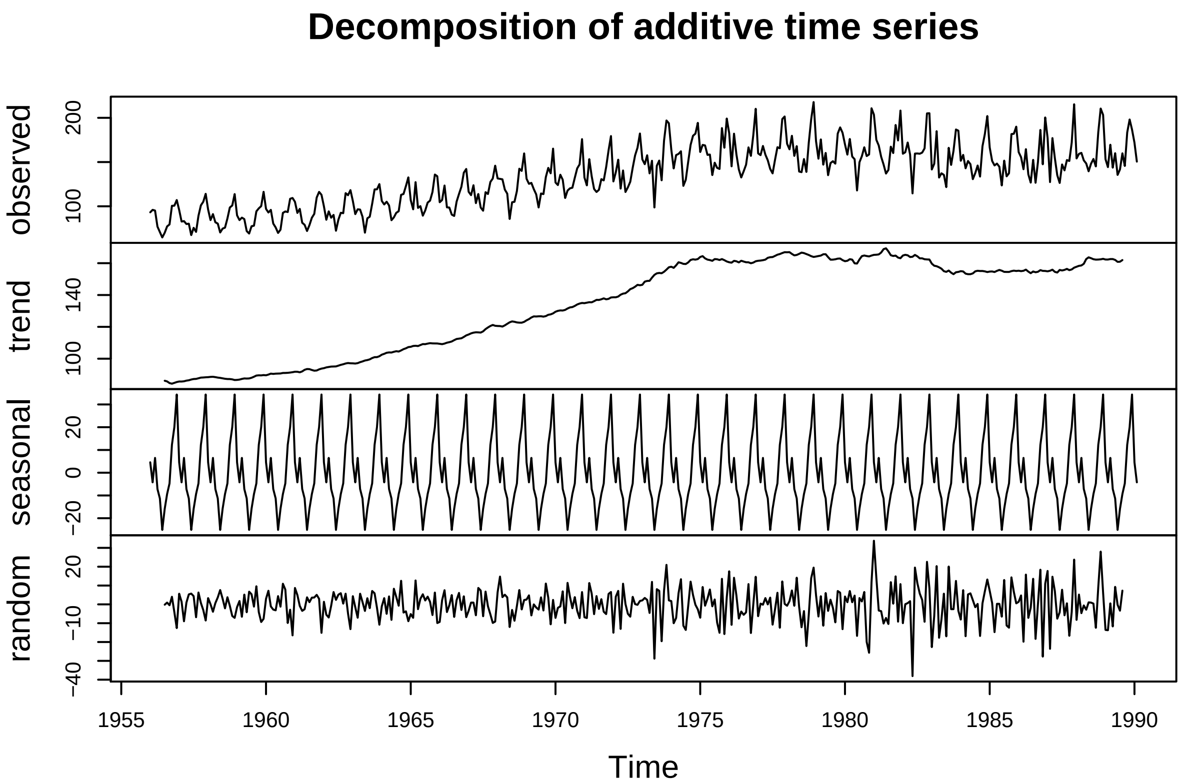
Question 6.10 - Repeating the above but with classical decomposition The classical decomposition process is the following: decompose the time series into a seasonal and trend component. Then, fit an ARIMA to each one of those components individually. The sum of the forecasts of the two ARIMA sums should then be a reasonable forecast of the original time series. **a)** Recall that our time series is:

```r
plotc(beer_ts)
```



We now decompose the time series and visualize the its different components:

```r
beer_decomp <- decompose(beer_ts)
plot(beer_decomp)
```



We then fit an ARIMA model to the seasonal and trend components, then define their forecast objects:

```
# Fitting ARIMA models to trend & seasonality
trend_arima <- auto.arima(beer_decomp$trend, seasonal=TRUE, stepwise=FALSE, approximation=FALSE)
season_arima <- auto.arima(beer_decomp$seasonal, seasonal=TRUE, stepwise=FALSE, approximation=FALSE)
summary(trend_arima)
```

```
## Series: beer_decomp$trend
## ARIMA(4,2,0)(0,0,1)[12]
##
## Coefficients:
##           ar1      ar2      ar3      ar4     sma1
##       -0.2127  -0.7989  -0.1112  -0.5326  -0.8330
## s.e.   0.0430   0.0438   0.0438   0.0431   0.0299
##
## sigma^2 estimated as 0.1906:  log likelihood=-248.93
## AIC=509.86   AICc=510.07   BIC=533.92
##
## Training set error measures:
##                       ME      RMSE       MAE         MPE      MAPE
## Training set 0.004123904 0.4393647 0.3255428 0.009047988 0.2366249
##                    MASE       ACF1
## Training set 0.08958375 0.08398002
```

```
summary(season_arima)
```

```
## Series: beer_decomp$seasonal
## ARIMA(0,0,0)(0,1,0)[12] with drift
##
## Coefficients:
## drift
```

```
##     0
##
## sigma^2 estimated as 7.528e-06:  log likelihood=13569.14
## AIC=-27136.28   AICc=-27136.27   BIC=-27132.29
##
## Training set error measures:
##                        ME        RMSE          MAE         MPE        MAPE
## Training set 1.949662e-17 0.002699831 0.0003794597 0.002926828 0.002926828
##              MASE      ACF1
## Training set  Inf 0.5829275
```

```r
# Fitting the overall model
model_fit <- trend_arima$fitted + season_arima$fitted

# Defining forecast objects
trend_forecast <- forecast(trend_arima)
season_forecast <- forecast(season_arima)

# Summing the forecasts
decomp_forecast <- ts((trend_forecast$mean + season_forecast$mean), frequency=12, start=c(1990, 3))
```