Computer exercises 4

Demo exercises

4.1

Examine the following time series.

File	Variable	Description	Interval	Length
`intel.txt`	`Intel_Close`	Intel stock price	Trading day	$n = 20$
`sunspot.txt`	`Spots`	Number of sunspots	1 year	$n = 215$
`sales.txt`	`Sales`	Sales volume of a wholesaler	1 month	$n = 144$

Fit SARIMA processes to the corresponding time series. Use the last fifth of the time series to verify the goodness of fit.

Solution

Package `forecast` includes R functions `Arima` and `auto.arima` that we will use during this session. Thus, install the package and attach it if you wish.

```
library(forecast)
```

Then we read the the data and create `ts` objects similarly as in Session 3.

```
# Read the data
intel <- read.table("data/intel.txt", header = TRUE)
sunspot <- read.table("data/sunspot.txt", header = TRUE)
sales <- read.table("data/sales.txt", header = TRUE, row.names = 4)

# Create ts objects
intel_ts <- ts(intel$Intel_Close)
sunspot_ts <- ts(sunspot$Spots, start = 1749)
sales_ts <- ts(sales$Sales, start = 1970, frequency = 12)
```

To ease our analysis we make a few functions for plotting and performing Ljung-Box test.

```
#' Plot autocorrelation and partial autocorrelation of time series
#'
#' @param data_ts Time series as a ts object.
#' @param lagmax Provides value for lag.max argument of acf and pacf.
plot_acf <- function(data_ts, lagmax = NULL) {
  par(mfrow = c(1, 2))
  acf(data_ts, main = "ACF", lag.max = lagmax)
  pacf(data_ts, main = "PACF", lag.max = lagmax)
  par(mfrow = c(1, 1))
}

#' Perform Ljung-Box test
```

```r
#'
#' Calculates p-values for Ljung-Box test for lags {k+1, ..., n}, where k
#' is the number of model parameters.
#'
#' @param model Object of class Arima, represents the fitted model.
#' @param k Number of model parameters.
#' @param n Sample size (length of the time series).
#'
#' @return Vector of p-values.
box_test <- function(model, k, n) {
  pvalues <- c(rep(NA, n - k - 1))
  for (i in 1:(n - k - 1)) {
    pvalues[i] <- Box.test(model$res, lag = i + k, fitdf = k,
                           type = "Ljung-Box")$p.value
  }
  pvalues
}

#' Plot p-values from Ljung-box test for different lags
#'
#' @param pvalues Vector of p-values.
#' @param alpha Significance level.
#' @param k Number of model parameters.
#' @param n Sample size (length of the time series).
plot_pvalues <- function(pvalues, alpha, k, n) {
  plot((k + 1):(n - 1), pvalues, pch = 16, col = "midnightblue",
       ylim = c(0, max(pvalues)), xlab = "lag", ylab = "p-value")
  abline(h = alpha, lty = 2, lwd = 2)
}

#' Plot original time series and fitted model.
#'
#' @param data_ts Time series as ts object.
#' @param model Fitted time series model as Arima object.
#'
plot_fit <- function(data_ts, model) {
  fit <- model$fitted
  plot(fit, type = "b", col = "blue", ylab = "Value", xlab = "Time", cex = 0.5,
       pch = 16, main = "")
  lines(data_ts, col = "red", type = "b", cex = 0.5, pch = 16)
  legend("topleft", legend = c("Time series", "Fit"), col = c("red", "blue"),
         lty = c(1, 1), cex = 0.5)
}

#' Plot original time series and s-step prediction.
#'
#' @param data_ts Time series as ts object.
#' @param prediction S-step prediction as ts object.
plot_pred <- function(data_ts, prediction, title = "", xlim = NULL,
                      ylim = NULL) {
  plot(data_ts, col = "red", type = "b", cex = 0.5, pch = 16, ylab = "Value",
       xlab = "Time", main = title, xlim = xlim, ylim = ylim)
  lines(prediction, col = "blue", type = "b", cex = 0.5, pch = 16)
```

```
  legend("topleft", legend = c("Time series", "Prediction"),
         col = c("red", "blue"), lty = c(1, 1), cex = 0.5)
}
```

Intel_Close

As a reminder, Figure 1 shows the time series `Intel_Close`.

```
plot(intel_ts)
```

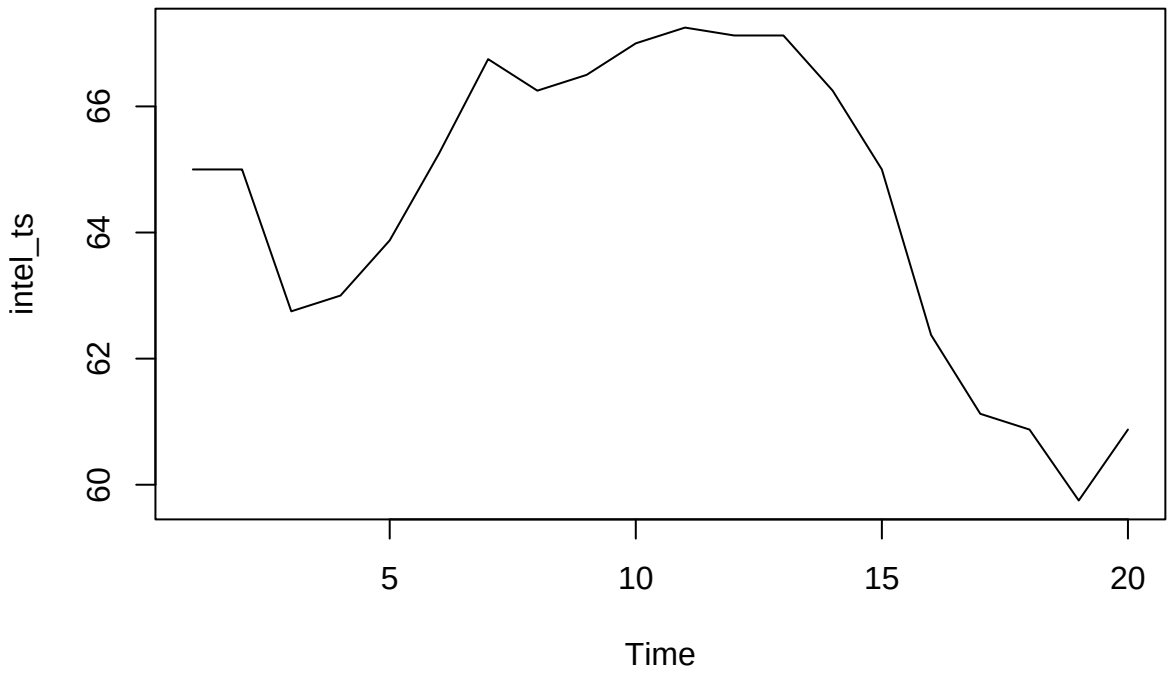

Figure 1: Time series `Intel_Close`.

Then we compute sample ACF and PACF, showed in Figure 2.

```
plot_acf(intel_ts)
```

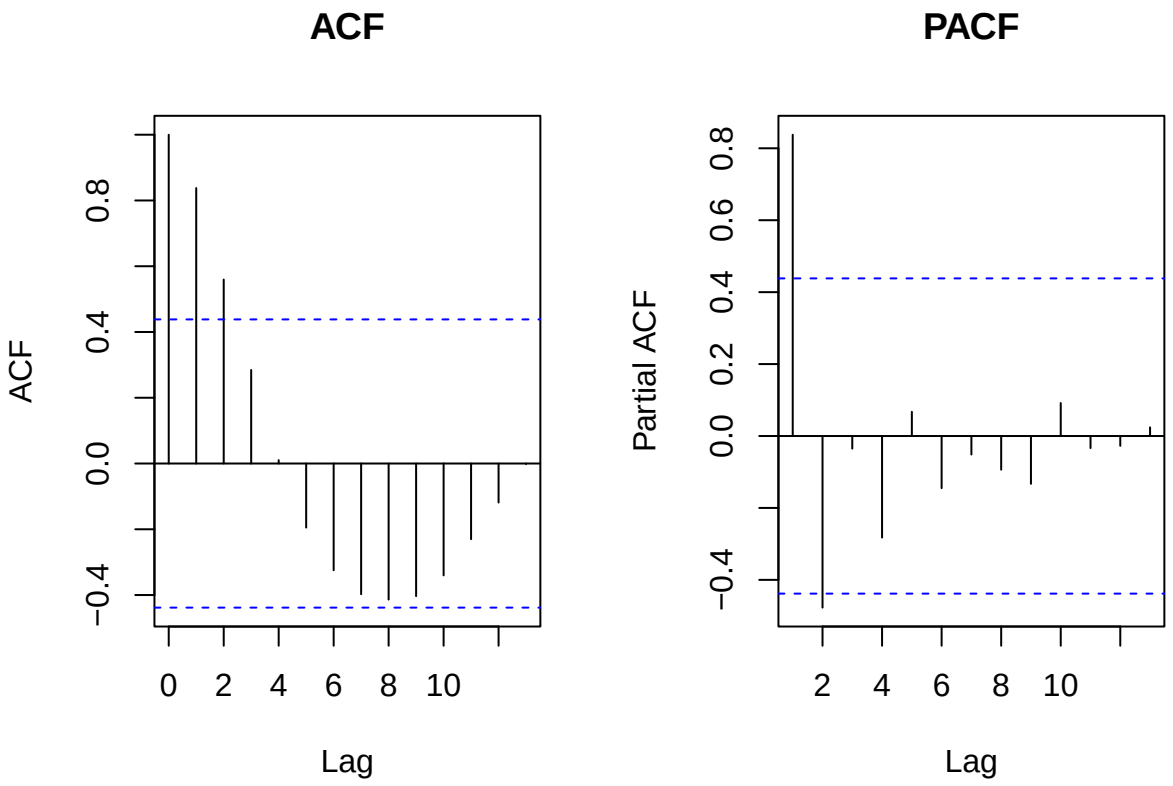

Figure 2: auto- and partial autocorrelation functions of `Intel_Close`.

Based on Figures 1 and 2 time series `Intel_Close` might be stationary. Additionally, ACF decays exponentially and PACF cuts off after lag 2. Thus we fit AR(2) model. Fitting can be done with the function `Arima` from package `forecast` which is largely a wrapper of the function `arima` from package `stats`. Model

```
Arima(y, order = c(p, h, q), seasonal = list(order = c(P, H, Q), period = s))
```

corresponds to

$$\text{SARIMA}(p,h,q)(P,H,Q)_s.$$

Thus AR(2) model is fitted in the following fashion.

```
model_intel <- Arima(intel_ts, order = c(2, 0, 0))
model_intel
```

```
## Series: intel_ts
## ARIMA(2,0,0) with non-zero mean
##
## Coefficients:
##          ar1      ar2     mean
##       1.3342  -0.5263  64.3965
## s.e.  0.1850   0.2038   1.0501
##
## sigma^2 = 1.04:  log likelihood = -28.2
## AIC=64.39   AICc=67.06   BIC=68.38
```

The estimated model involves a constant term (Intercept). Estimating the model without the constant term can be done by setting `include.mean = FALSE`. The function `Arima` estimates the model parameters with CSS-ML (conditional sum of squares - maximum likelihood) method as a default.

Next, we study the residuals of the fitted AR(2) model. By Figure 3 residuals seem to be uncorrelated.

```
plot_acf(model_intel$residuals)
```

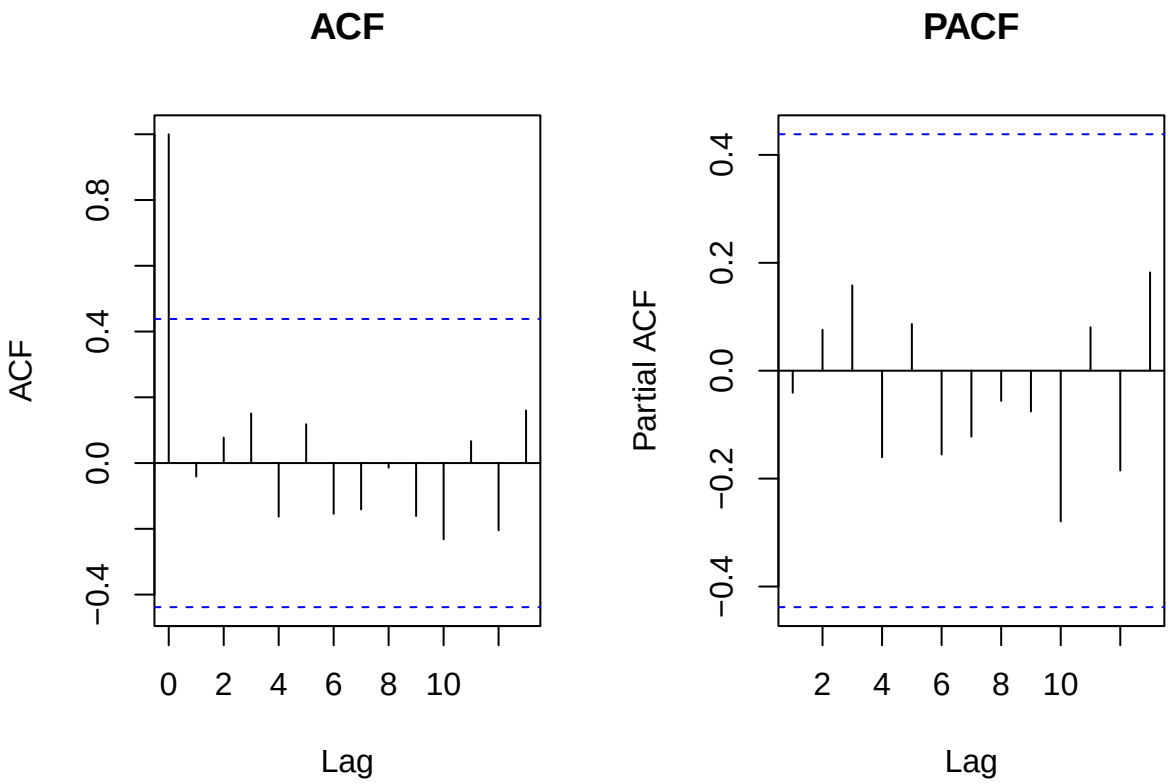

Figure 3: ACF and PACF of the residuals corresponding to the AR(2) model for `Intel_Close`.

In addition to visual inspection of ACF and PACF of residuals, we can use Ljung-Box test to test if residuals are white noise. Ljung-Box test tests the significance of h first autocorrelations, where $h \in \{k+1, \dots, n-1\}$ and k is the number of parameters of the ARIMA model. It is advisable to perform Ljung-Box test for several different values of h. R function `box_test` defined earlier, performs Ljung-Box test for all possible lags $h \in \{k+1, \dots, n-1\}$. In the case of AR(2) model we have $k = 2$.

```
k <- 2
n <- length(intel_ts)
pvalues <- box_test(model_intel, k, n)
pvalues
```

```
##  [1] 0.3786061 0.4713735 0.5906405 0.6158862 0.6485075 0.7648594 0.7350293
##  [8] 0.5646156 0.6414046 0.5078315 0.4545944 0.5403243 0.5671877 0.1945652
## [15] 0.1691062 0.2126523 0.2622378
```

```
which(pvalues > 0.05) + k
```

```
##  [1]  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19
```

```
plot_pvalues(pvalues, 0.05, k, n)
```

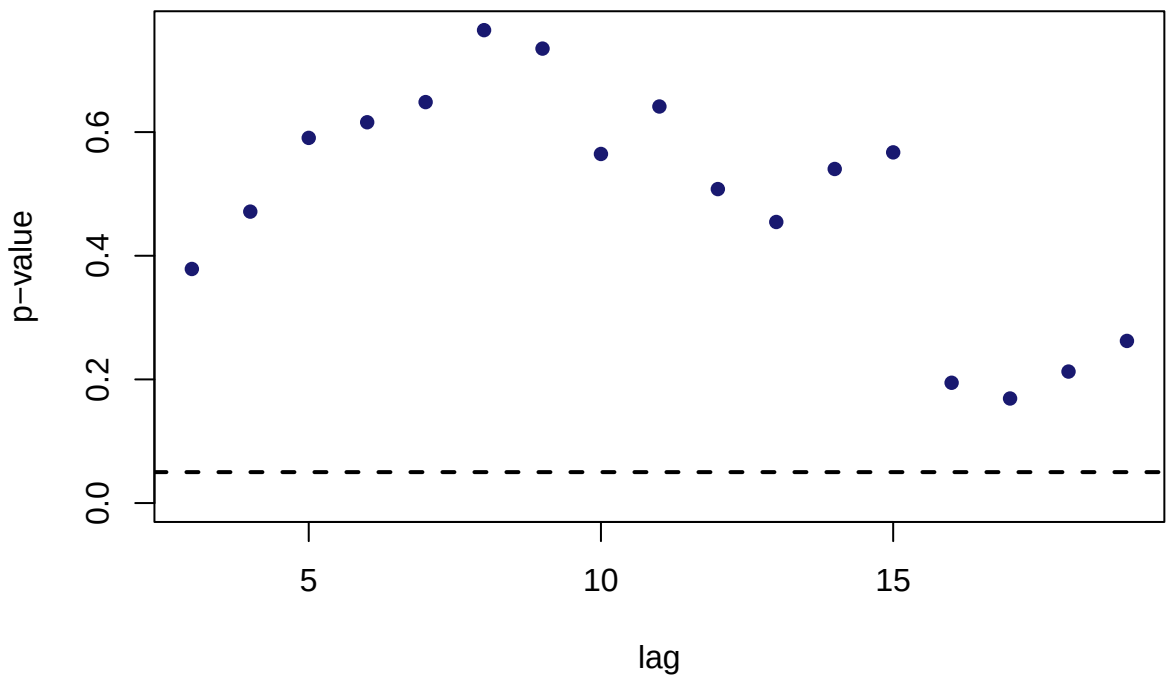

Figure 4: Ljung-Box test p-values for residuals of the AR(2) model for `Intel_Close`. Dashed line represents the significance level $\alpha = 0.05$.

According to Figure 4 the AR(2) model is satisfactory with significance level of 5%. The fitted model and the original time series are presented in Figure 5.

```
plot_fit(intel_ts, model_intel)
```

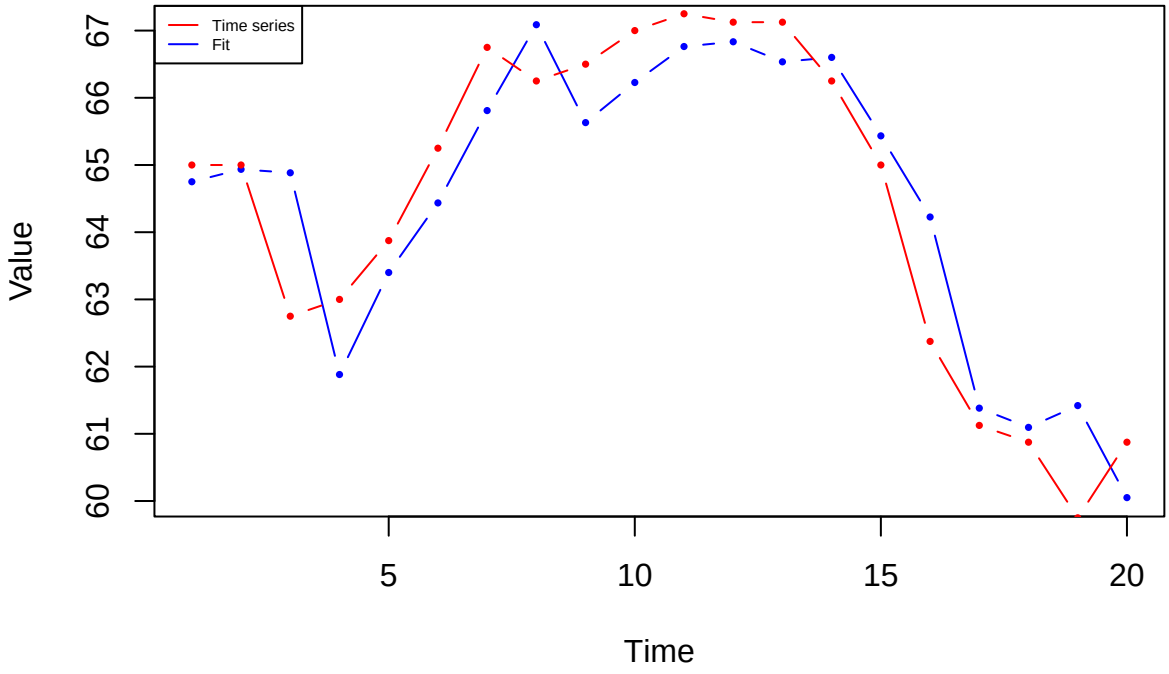

Figure 5: `Intel_Close` time series as red and the fitted AR(2) model as blue.

According to Figure 5, the fitted model and the original time series are pretty close to each other. Next, we estimate an AR(2) model by using the first 16 observations and study how well the model predicts the last four observations.

```
prediction <- forecast(Arima(intel_ts[1:16], order = c(2, 0, 0)), h = 4)$mean
plot_pred(intel_ts, prediction)
```

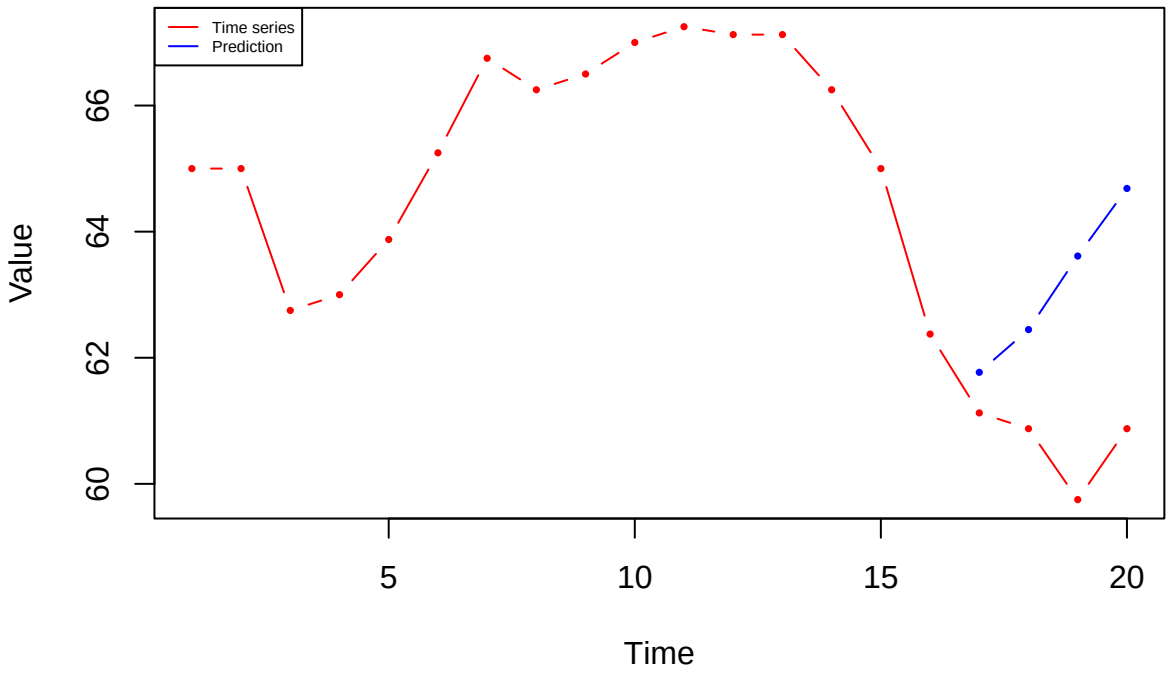

Figure 6: `Intel_Close` time series as red and the four step prediction given by AR(2) model as blue.

Even though the previous diagnostics imply that the AR(2) model is satisfactory, Figure 6 shows that the first 16 observations do not provide a good prediction for the last four observations. This is explained by the shortness of the original time series. When predicting the future behavior of time series, one has to be careful, especially when dealing with short time series.

Spots

As a reminder, Figure 7 shows the time series `Spots`.

```
plot(sunspot_ts)
```

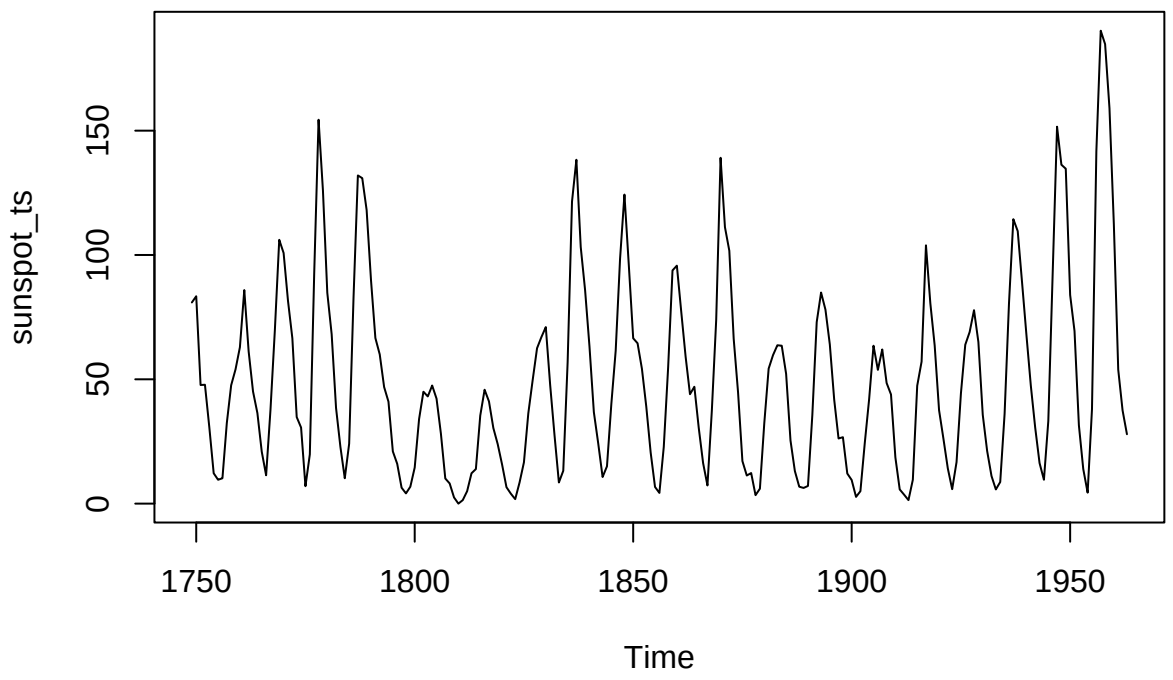

Figure 7: Time series `Spots`.

Then we compute sample ACF and PACF, showed in Figure 8.

```
plot_acf(sunspot_ts, lagmax = 50)
```

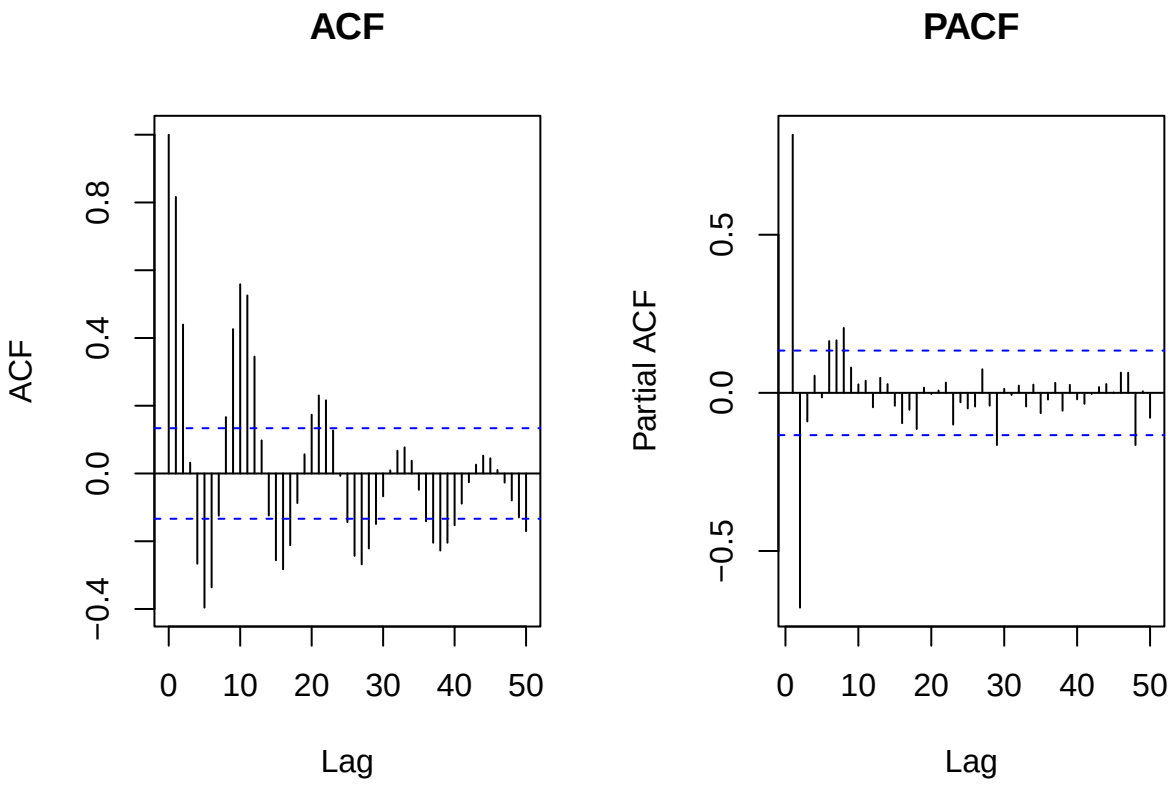

Figure 8: auto- and partial autocorrelation functions of `Spots`.

According to Figure 7 there seems to be a cyclical component of approximately 11 year. Thus time series `Spots` is definitely not stationary. However, according to Figure 8, PACF cuts off after lag 2, even though there are still some spikes outside the blue lines afterwards. Thus, regardless of the cyclical component, we try to fit an AR(2) model to the time series.

Next, fit the model and study the residuals.

```
model_spots1 <- Arima(sunspot_ts, order = c(2, 0, 0))
plot_acf(model_spots1$residuals)
```

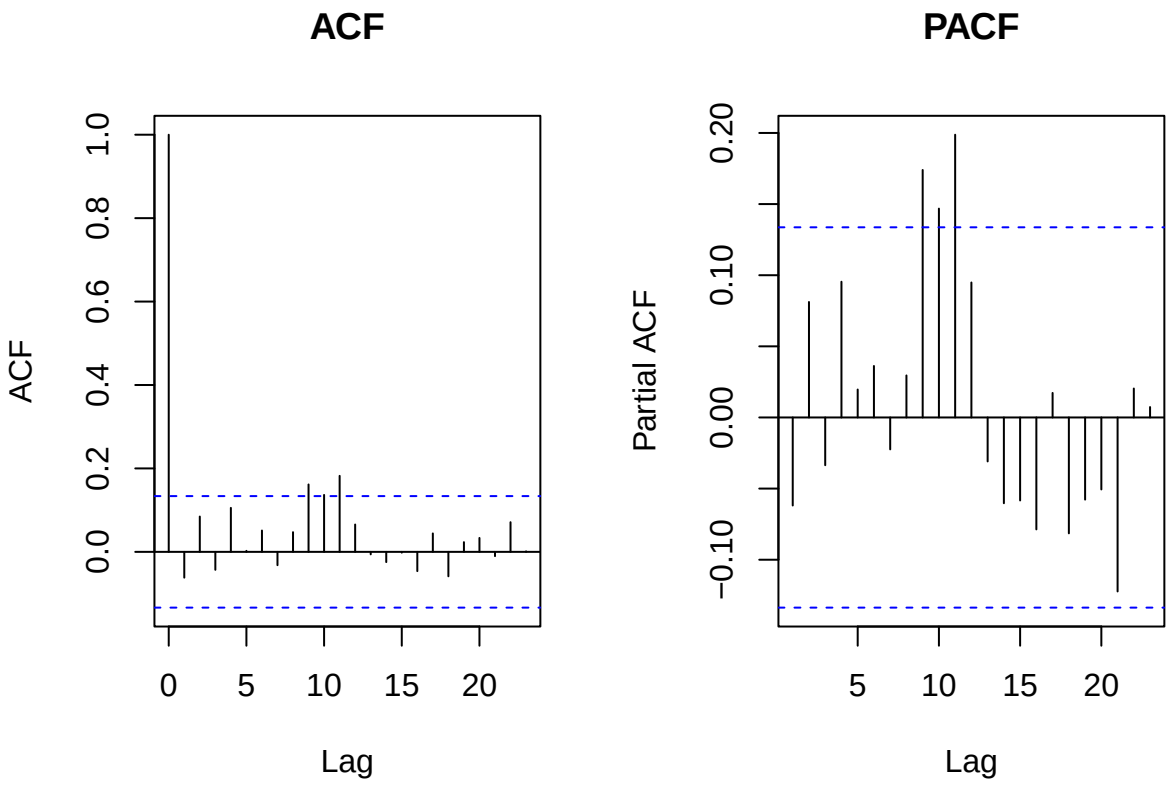

Figure 9: ACF and PACF of the residuals corresponding to the AR(2) model for `Sunspots`.

Figure 9 shows that the autocorrelations and partial autocorrelations corresponding to lags 9-11 are significant. Also, Figure 10 suggests that the AR(2) model is not sufficient for `Sunspots`, since for many lags null hypothesis of Ljung-Box test is rejected.

```
k <- 2
n <- length(sunspot_ts)
pvalues <- box_test(model_spots1, k, n)
plot_pvalues(pvalues, 0.05, k, n)
```

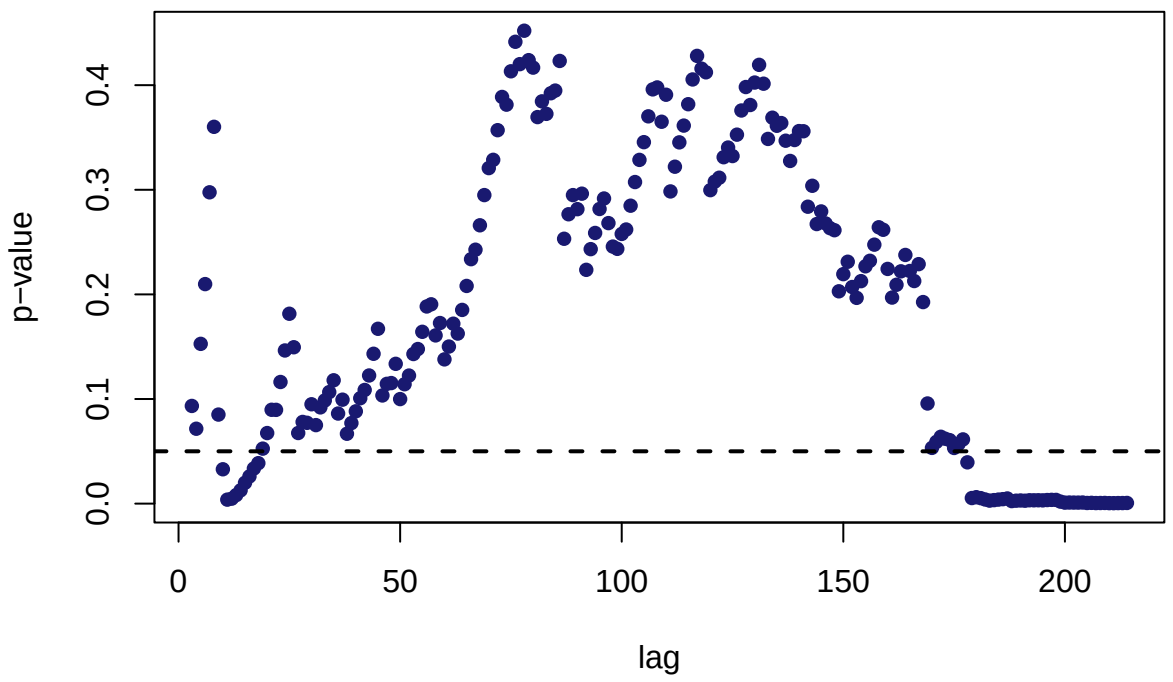

Figure 10: Ljung-Box test p-values for residuals of the AR(2) model for `Sunspots`. Dashed line represents the significance level $\alpha = 0.05$.

Next, we try the function `auto.arima`, which automatically fits a SARIMA model to a given time series.

```
model_spots2 <- auto.arima(sunspot_ts)
model_spots2
```

```
## Series: sunspot_ts
## ARIMA(3,0,1) with non-zero mean
##
## Coefficients:
##          ar1     ar2      ar3     ma1     mean
##       0.6574  0.3494  -0.5433  0.6580  49.1226
## s.e.  0.2000  0.2622   0.1308  0.2128   3.4416
##
## sigma^2 = 272.7:  log likelihood = -906.65
## AIC=1825.3   AICc=1825.7   BIC=1845.52
```

Function `auto.arima` arrives to ARMA$(3,1)$ model. The algorithm fits the best possible SARIMA model by using AIC, AICc or BIC as a minimizing criterion. Note that, the fitting of SARIMA models is usually not easy. Furthermore, there are no perfect algorithms that always find the best possible models. Therefore, don't trust the function `auto.arima` blindly. The function often chooses unnecessarily complicated models that are only marginally better than some simpler alternatives.

By Figure 11, autocorrelations and partial autocorrelations corresponding to lags 9,10 and 11 are significant also for the residuals of the ARMA$(3,1)$ fit. In addition, the null hypothesis of Ljung-Box is rejected for

multiple lags according to Figure 12. Thus, ARMA$(3,1)$ model is neither satisfactory.

```
plot_acf(model_spots2$residuals)
```

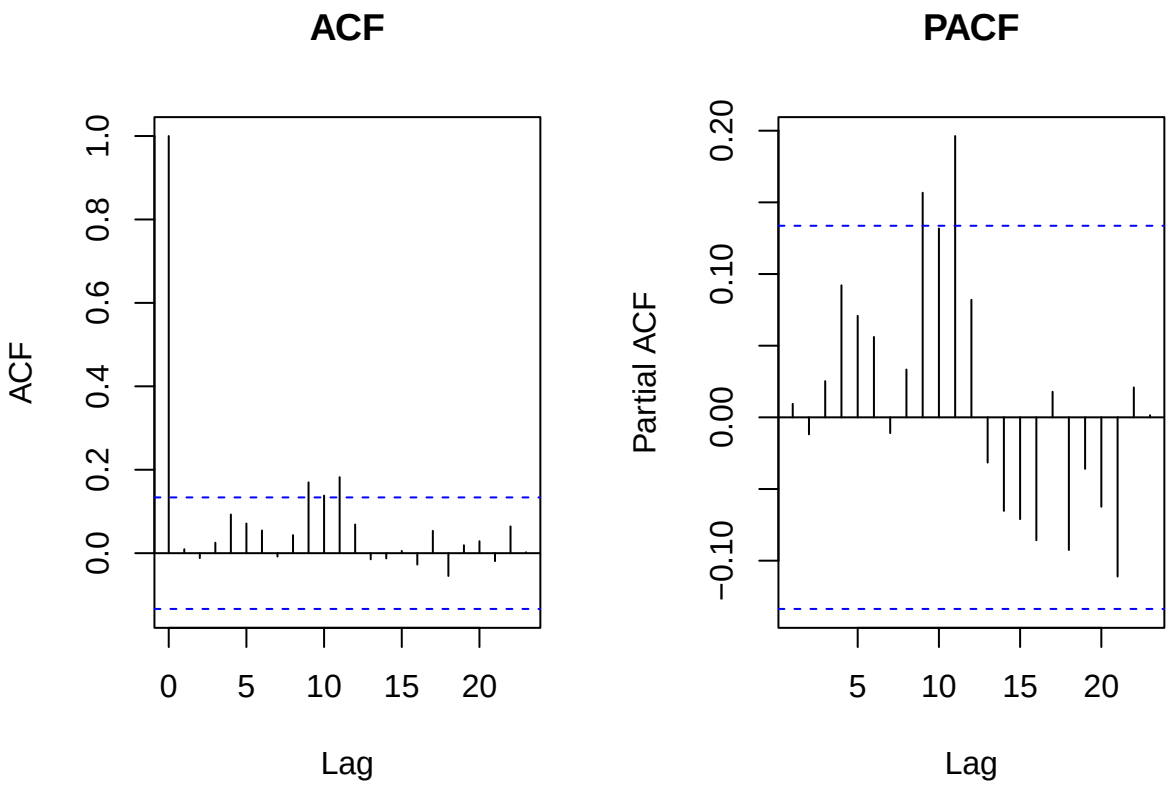

Figure 11: ACF and PACF of the residuals corresponding to the ARMA$(3,1)$ model for `Sunspots`.

```
k <- 4
pvalues <- box_test(model_spots2, k, n)
plot_pvalues(pvalues, 0.05, k, n)
```

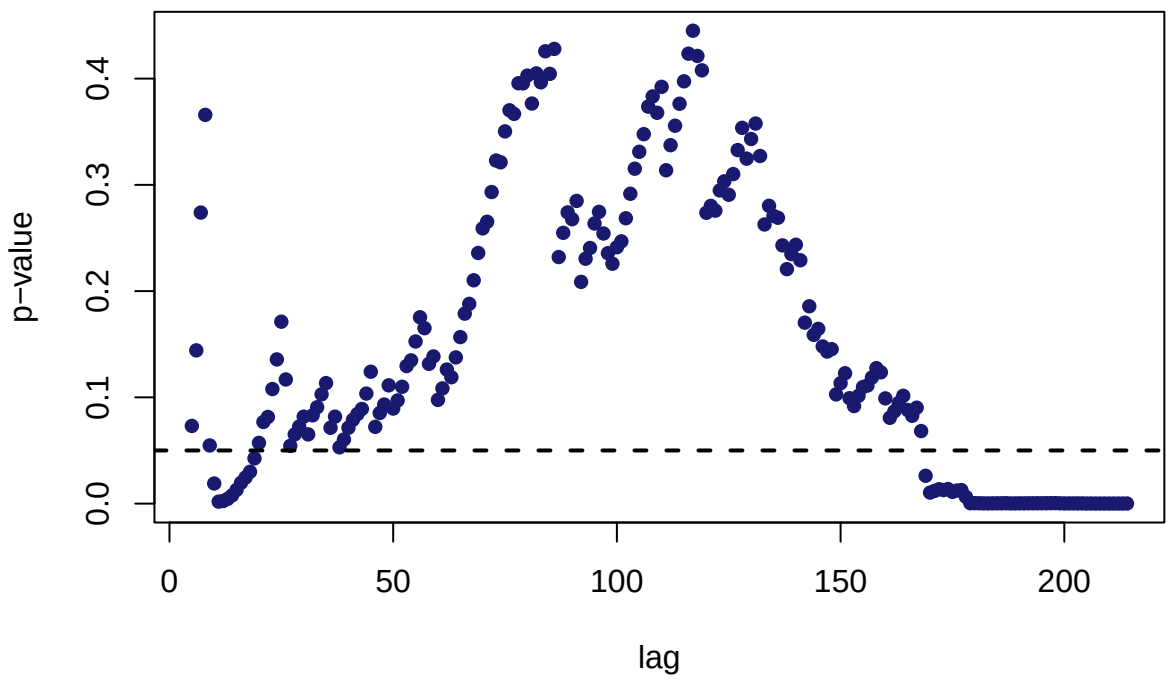

Figure 12: Ljung-Box test p-values for residuals of the ARMA$(3,1)$ model for `Sunspots`. Dashed line represents the significance level $\alpha = 0.05$.

Comment:

- Time series `Spots` turned out to be hard to model, since the period (around 11 years) of the cyclical component is not constant.

Figures 13 and 14 show time series `Sunspots` together with the fitted models AR(2) and ARMA$(3,1)$, respectively. Both fits are consistent with the original time series.

```
plot_fit(sunspot_ts, model_spots1)
```

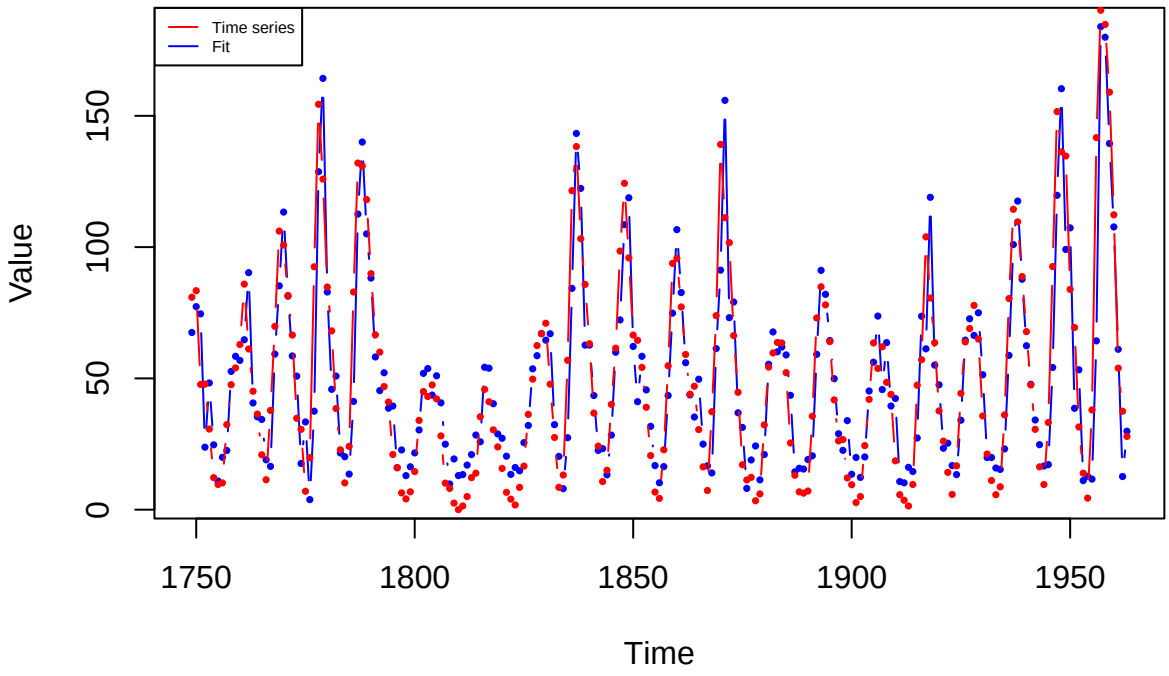

Figure 13: `Sunspots` time series as red and the fitted AR(2) model as blue.

```
plot_fit(sunspot_ts, model_spots2)
```

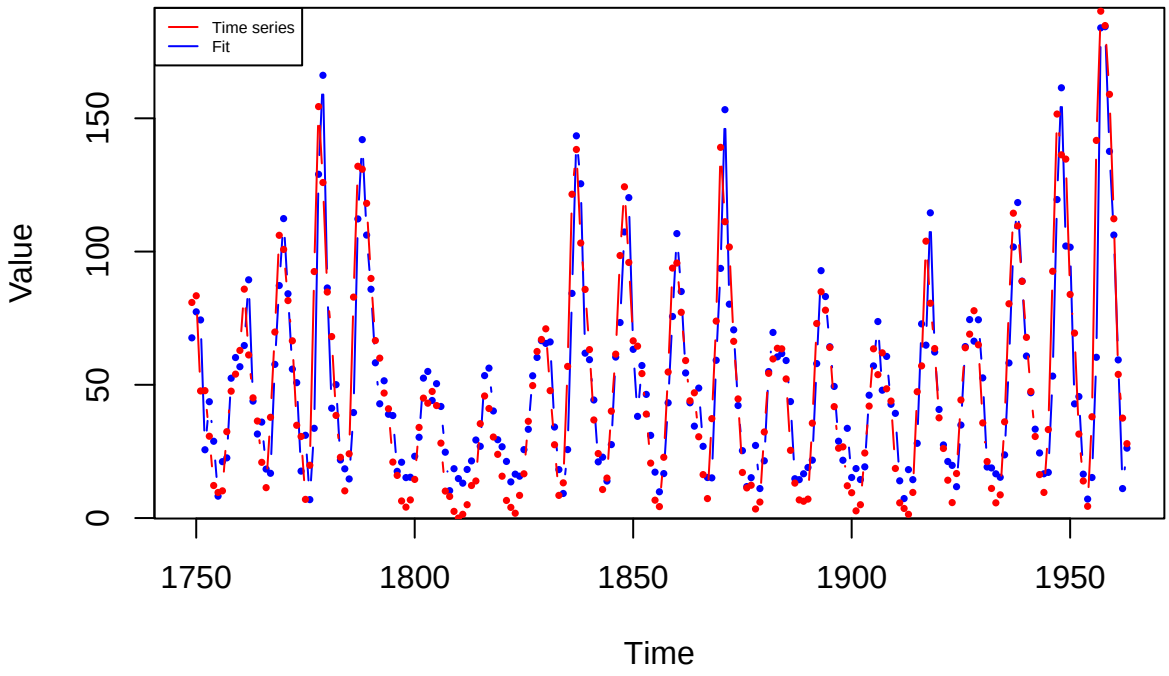

Figure 14: `Sunspots` time series as red and the fitted ARMA$(3,1)$ model as blue.

Next we give 43-step and 5-step predictions for both models.

```
prediction1 <- forecast(Arima(sunspot_ts[1:210], order = c(2, 0, 0)),
                        h = 5)$mean
prediction2 <- forecast(Arima(sunspot_ts[1:172], order = c(2, 0, 0)),
                        h = 43)$mean

prediction1 <- ts(prediction1, end = 1964)
prediction2 <- ts(prediction2, end = 1964)

par(mfrow = c(1, 2))
plot_pred(sunspot_ts, prediction1, title = "5-step prediction")
plot_pred(sunspot_ts, prediction2, title = "43-step prediction")
```

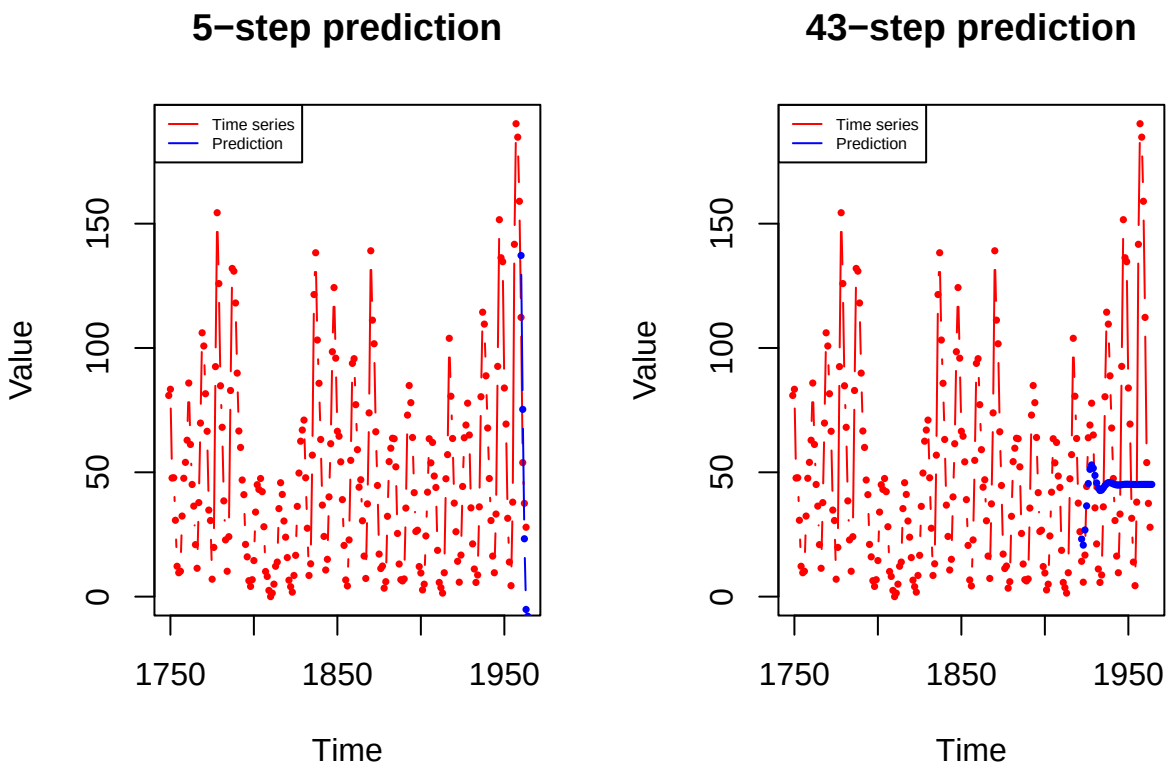

Figure 15: `Sunspots` time series as red. Left subplot shows 5-step prediction and right subplot shows 43-step prediction given by AR(2) model. Predictions are colored as blue.

```
par(mfrow = c(1, 1))
```

```
prediction1 <- forecast(Arima(sunspot_ts[1:210], order = c(3, 0, 1)),
                        h = 5)$mean
prediction2 <- forecast(Arima(sunspot_ts[1:172], order = c(3, 0, 1)),
                        h = 43)$mean

prediction1 <- ts(prediction1, end = 1964)
prediction2 <- ts(prediction2, end = 1964)

par(mfrow = c(1, 2))
plot_pred(sunspot_ts, prediction1, title = "5-step prediction")
plot_pred(sunspot_ts, prediction2, title = "43-step prediction")
```

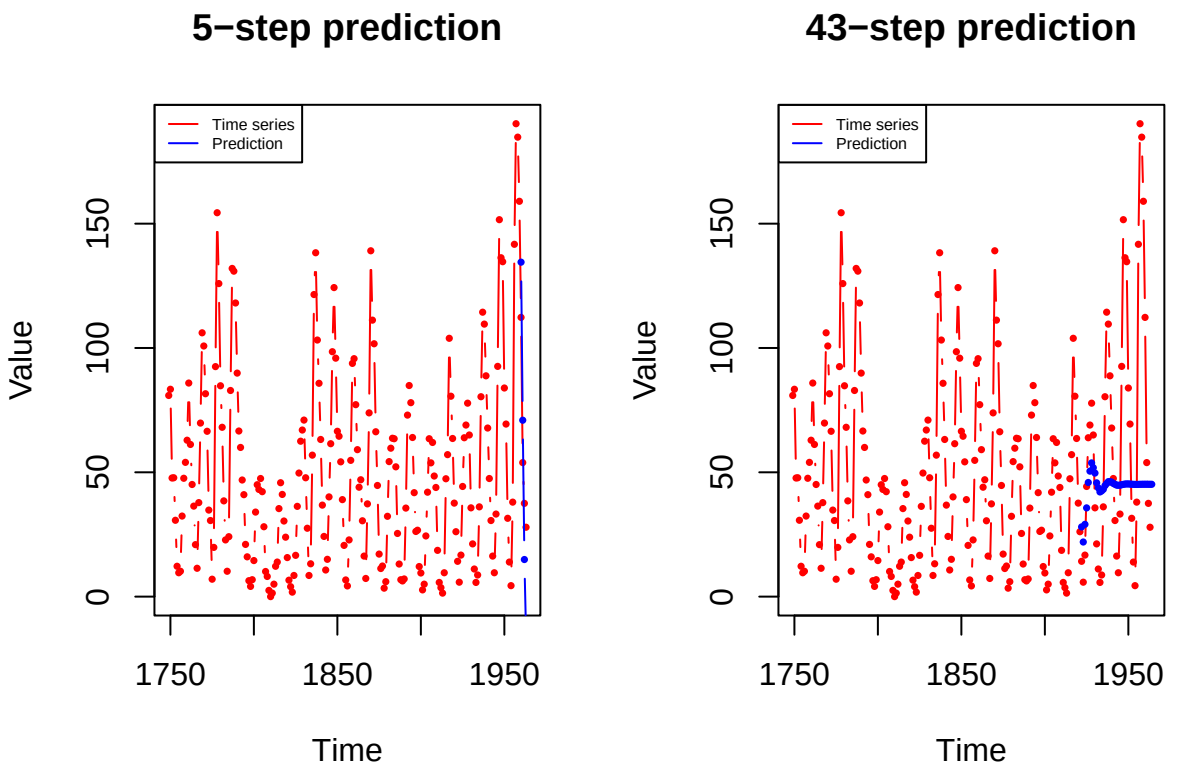

Figure 16: `Sunspots` time series as red. Left subplot shows 5-step prediction and right subplot shows 43-step prediction given by ARMA$(3, 1)$ model. Predictions are colored as blue.

```
par(mfrow = c(1, 1))
```

Figures 15 and 16 show that long term predictions can be unreliable, on the other hand, predicting just few next steps works out relatively well. It is no surprise that models are not suited for long term predictions, since they were labeled as unsatisfactory by the earlier diagnostics.

Sales

By the previous homework assignment and Figure 17, the time series **Sales** does not look stationary, but $D_{12}D \log \texttt{Sales}_t$ might be.

```
plot(sales_ts)
```

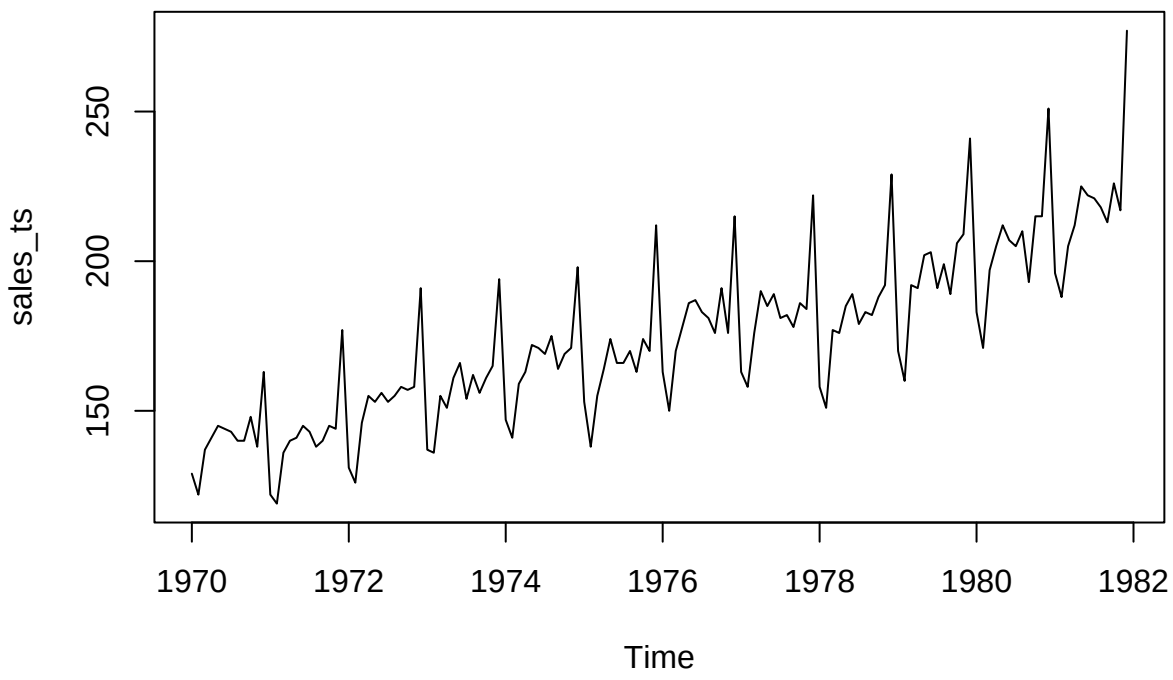

Figure 17: Time series `Sales`.

Below we show ACF and PACF of time series $D_{12}D\log(\texttt{Sales})_t$.

```
plot_acf(diff(diff(log(sales_ts), lag = 12)), lagmax = 50)
```

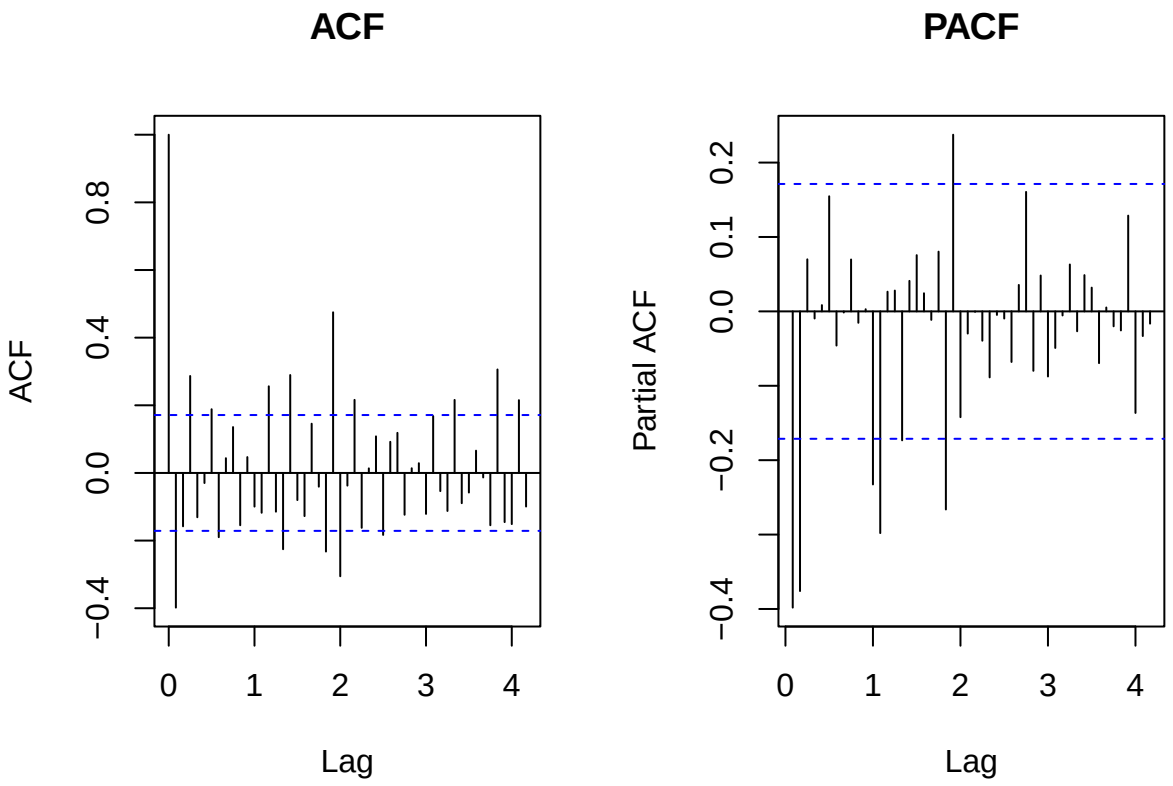

Figure 18: ACF and PACF of $D_{12}D\log(\texttt{Sales})_t$

Based on ACF and PACF one possible model could be SARIMA$(2,1,0)(1,1,0)_{12}$ after log transformation. Log transformation can be included in the model by setting $\lambda = 0$. Namely, argument `lambda` controls parameter λ in the Box-Cox transformation,

$$f_\lambda(\texttt{Sales}_t) = \begin{cases} \lambda^{-1}(\texttt{Sales}_t - 1), & \texttt{Sales}_t \geq 0, \lambda > 0 \\ \log(\texttt{Sales}_t), & \texttt{Sales}_t > 0, \lambda = 0. \end{cases}$$

Box-Cox transformation is applied before differencing to stabilize variance. When standard deviation increases linearly with the mean, $\lambda = 0$ is the appropriate choice.

```
model_sales <- Arima(sales_ts, order = c(2, 1, 0), seasonal = c(1, 1, 0),
                     lambda = 0)
model_sales
```

```
## Series: sales_ts
## ARIMA(2,1,0)(1,1,0)[12]
## Box Cox transformation: lambda= 0
##
## Coefficients:
##           ar1      ar2     sar1
##       -0.6984  -0.4677  -0.3496
## s.e.   0.0854   0.0819   0.0940
##
## sigma^2 = 0.0007994:  log likelihood = 281.82
```

```
## AIC=-555.64   AICc=-555.32   BIC=-544.13
```

By default argument `period` is equal to the `frequency(sales_ts)`. Thus we do not have to set any value for the argument `period` explicitly.

```
plot_acf(model_sales$residuals, lagmax = 50)
```

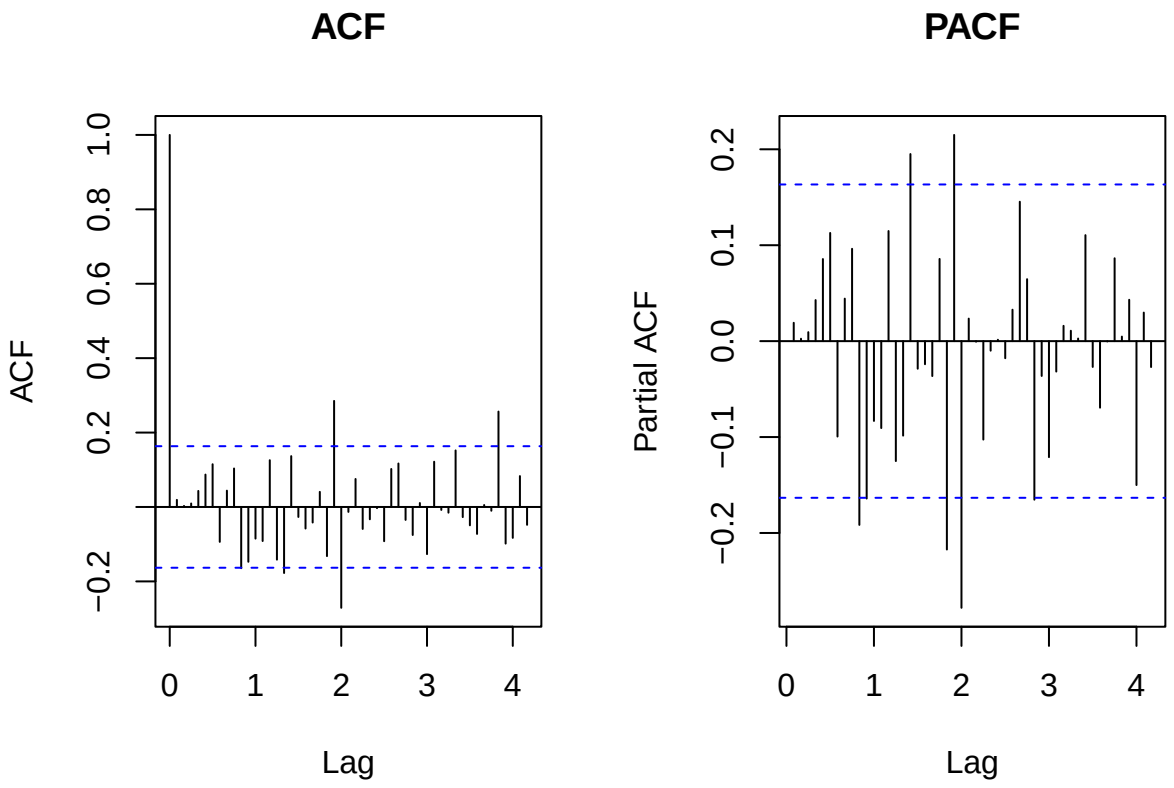

Figure 19: ACF and PACF of the residuals corresponding to the SARIMA$(2,1,0)(1,1,0)_{12}$ model with $\lambda = 0$ for `Sales` after log transformation.

```
n <- length(sales_ts)
k <- 3
pvalues <- box_test(model_sales, k, n)
plot_pvalues(pvalues, 0.05, k, n)
```

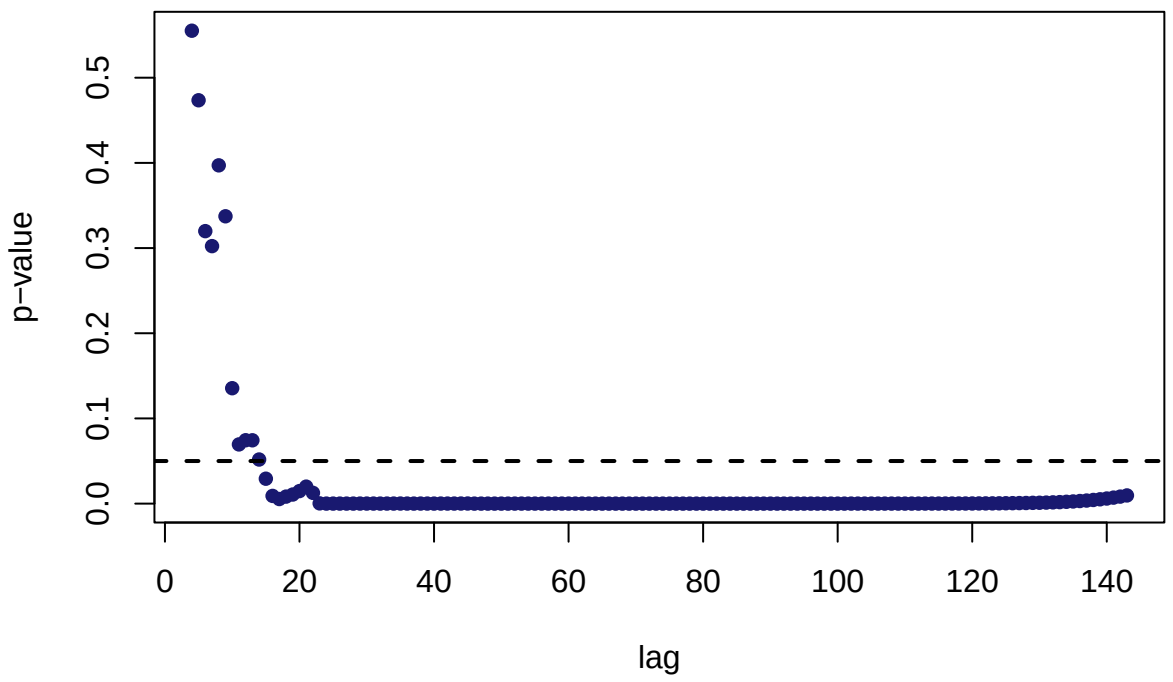

Figure 20: Ljung-Box test p-values for residuals of the SARIMA$(2,1,0)(1,1,0)_{12}$ model with $\lambda = 0$ for `Sales`. Dashed line represents the significance level $\alpha = 0.05$.

According to Figure 19 there are multiple significant spikes in ACF and PACF plots. Also, Figure 20 shows that the null hypothesis is rejected with multiple lags. Thus the fitted model is not particularly satisfactory. However, a better model is difficult to find, at least when restricted to SARIMA processes.

Nevertheless, we study how well the estimated SARIMA$(2,1,0)(1,1,0)_{12}$ model with $\lambda = 0$ behaves. Figure 21 shows that fit seems to be quite consistent with the original time series.

```
plot_fit(sales_ts, model_sales)
```

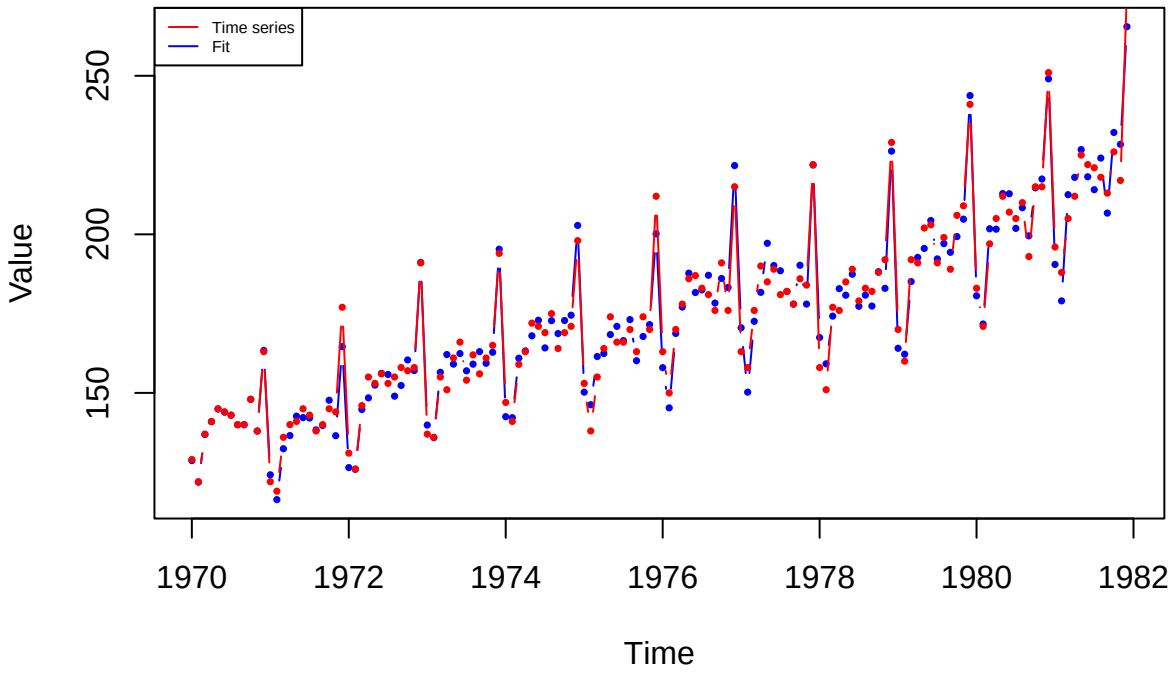

Figure 21: `Sales` time series as red and the fitted SARIMA$(2,1,0)(1,1,0)_{12}$ with $\lambda = 0$ model as blue.

Figure 22 shows that the 29-step prediction seems reasonable, although according to the Ljung-Box test the SARIMA$(2,1,0)(1,1,0)_{12}$ model with $\lambda = 0$ was not satisfactory.

```
sales_ts_pred <- ts(sales_ts[1:115], start = 1970, frequency = 12)
model_sales_pred <- Arima(sales_ts_pred, order = c(2, 1, 0),
                          seasonal = c(1, 1, 0), lambda = 0)
prediction <- forecast(model_sales_pred, h = 29)$mean
plot_pred(sales_ts, prediction)
```

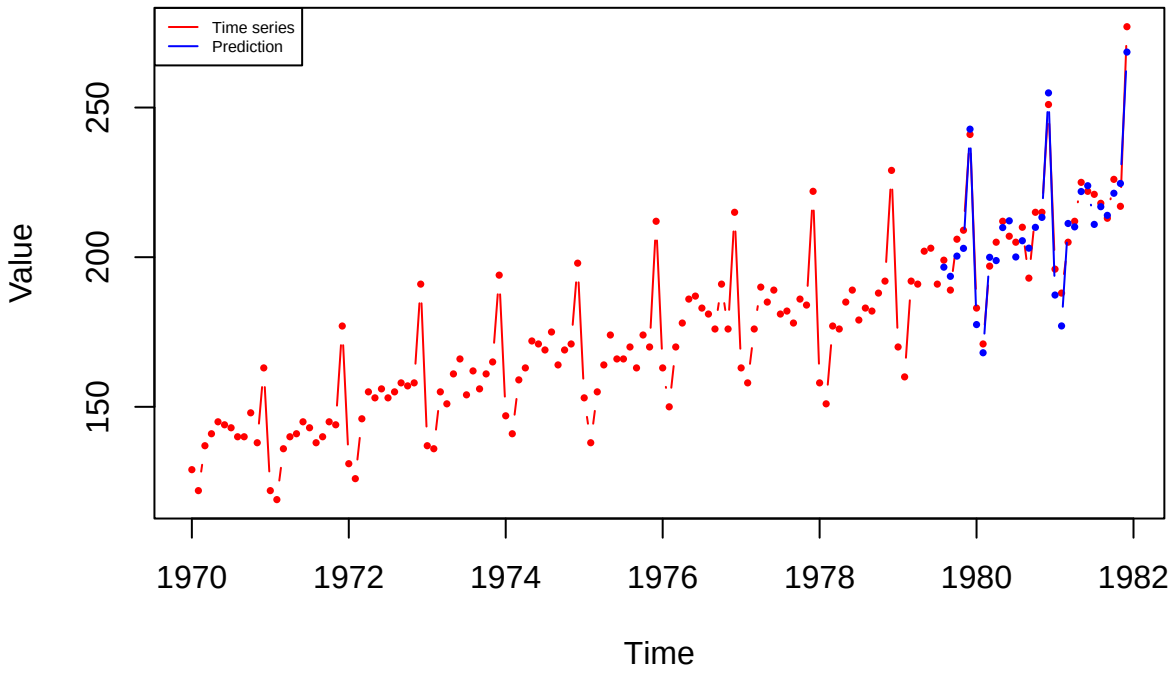

Figure 22: `Sales` time series as red and the 29-step prediction given by SARIMA$(2,1,0)(1,1,0)_{12}$ model with $\lambda = 0$ as blue.

Comments:

- Also time series `Sales` turned out to be hard to model. It is quite common that in practice, it is hard to find fitted models with all model assumptions satisfied. However, something is better than nothing. Even in this case, predictions seem visually good even though residual are not white noise.
- Remember, also regarding Exercise 4.2, that there is no single "correct" model. Many different models can be sufficient. However, you have to be able to justify your choices of model construction.

Homework

4.2

A time series of carbon dioxide measurements from the Mauna Loa volcano is given in the file `mlco2.txt`. The length of the time series is 216 months. Recall that, we studied this time series during the third computer exercises.

a) Using SARIMA processes, find the best possible model to describe the time series `MLCO2`.

b) Make 2 and 24 time step predictions by using the model chosen in a). Study the goodness of the predictions.