TSE Assignment 2

Maximilian J. Arrich

29 4 2020

1. ETF Data Analysis

(a) Use the R "quantmod" library to draw the backward-adjusted closing prices of the ETFs with ticker symbols EWA and EWC from 2012-02-02 to 2020-02-02. These two ETFs represent two equity baskets, the MSCI Australia and the MSCI Canada. Plot the two time series in one plot and comment.

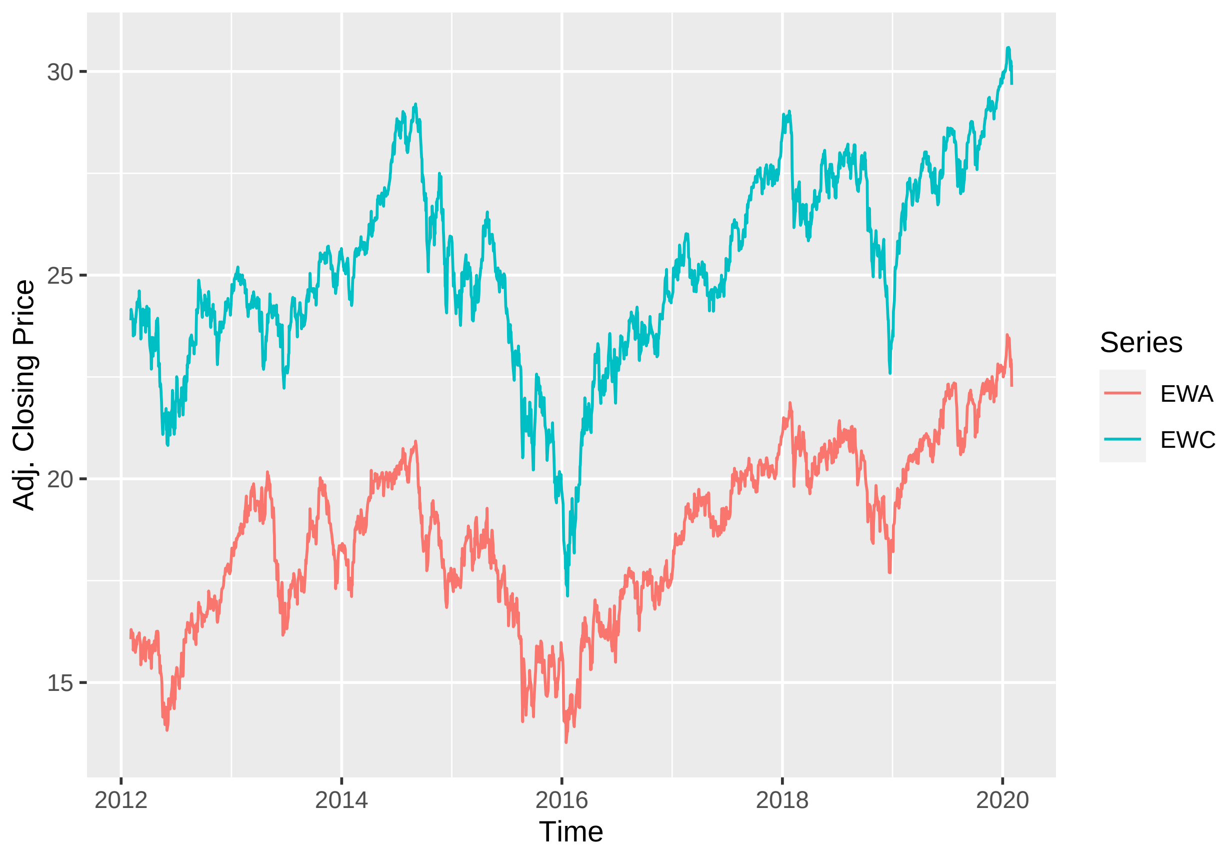
```
library(quantmod)
setwd(here::here())
# Load the data from quantmod
getSymbols("EWA",src="yahoo")
```

```
## [1] "EWA"
```

```
getSymbols("EWC",src="yahoo")
```

```
## [1] "EWC"
```

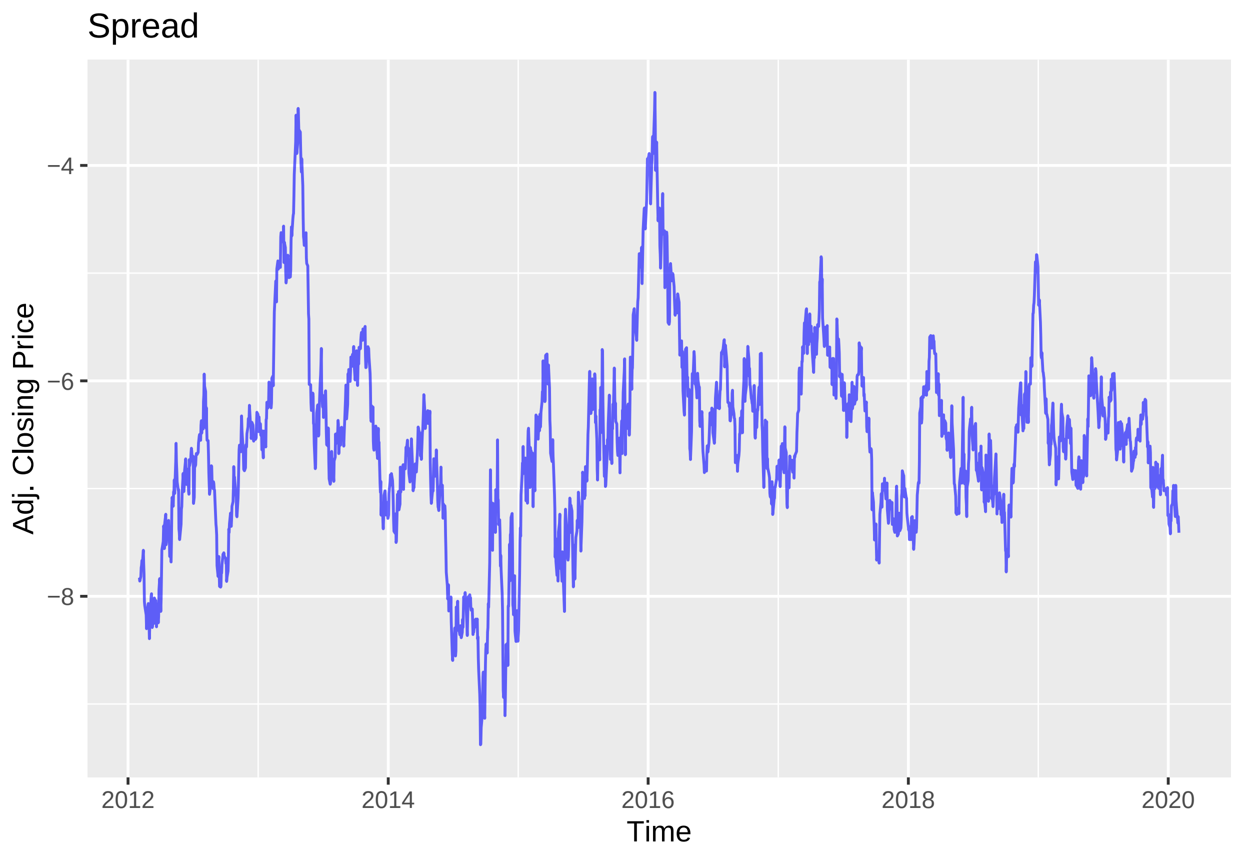
```
# clean up
ewa <- EWA %>% .[,6] %>% window(start = "2012-02-02", end = "2020-02-02") %>%
  set_colnames("EWA") %>% as_tibble(rownames = "time")
ewc <- EWC %>% .[,6] %>% window(start = "2012-02-02", end = "2020-02-02")  %>%
  set_colnames("EWC")%>% as_tibble(rownames = "time")
data <- full_join(ewa, ewc, by = "time") %>% mutate(time = as.Date(time))

# reshape and plot
data %>% pivot_longer(-time, "Series") %>%
  ggplot(aes(x = time, y = value, color = Series)) + geom_line() +
  ylab("Adj. Closing Price") + xlab("Time")
```



The two series seem to be heavily positively correlated.Furthermore, both series show a positive trend which leads to the impression of both series not being stationary. However the two series appear to be co-stationary.

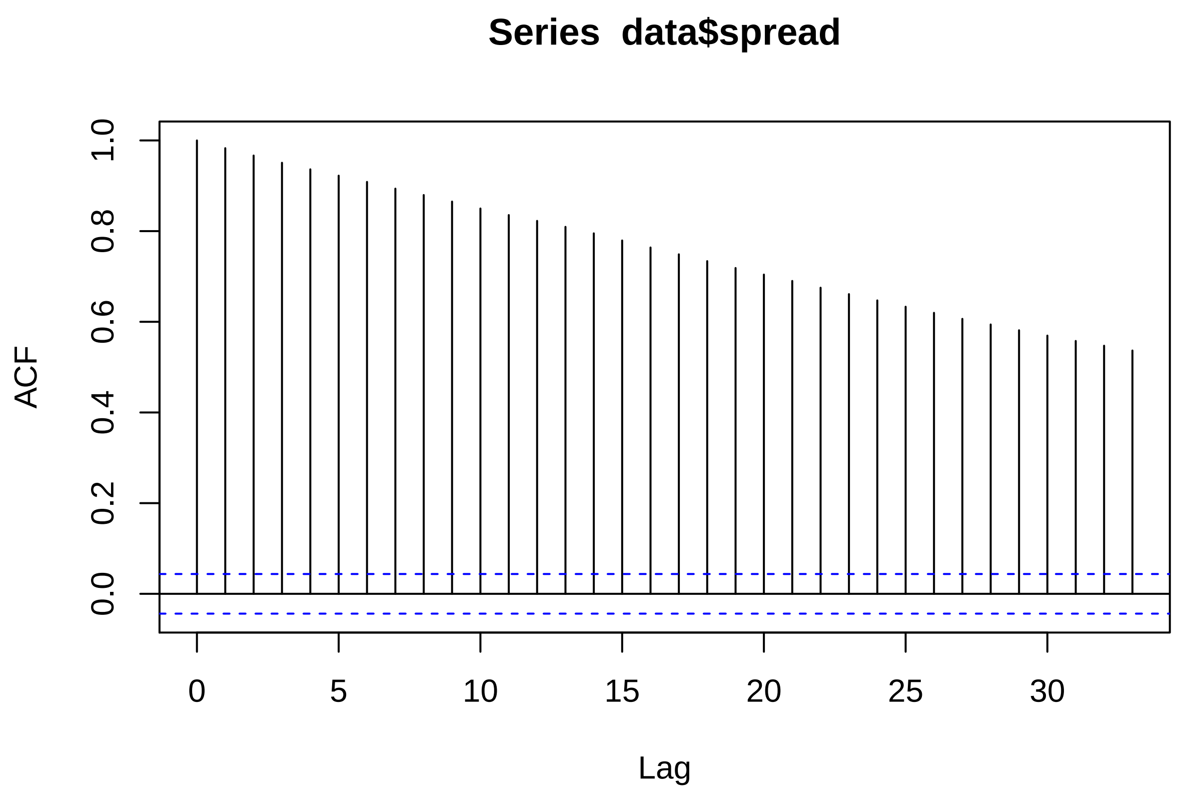
(b) Calculate the spread of the two times series, plot it and calculate ACF and PACF. Comment. Name two ARIMA(p,d,q) models that could suit the data according to the plots.

```
data <- data %>% mutate(spread = EWA - EWC)
data %>% ggplot(aes(x = time, y = spread)) + geom_line(color = "blue", alpha = 0.6) +
  ylab("Adj. Closing Price") + xlab("Time") + ggtitle("Spread")
```



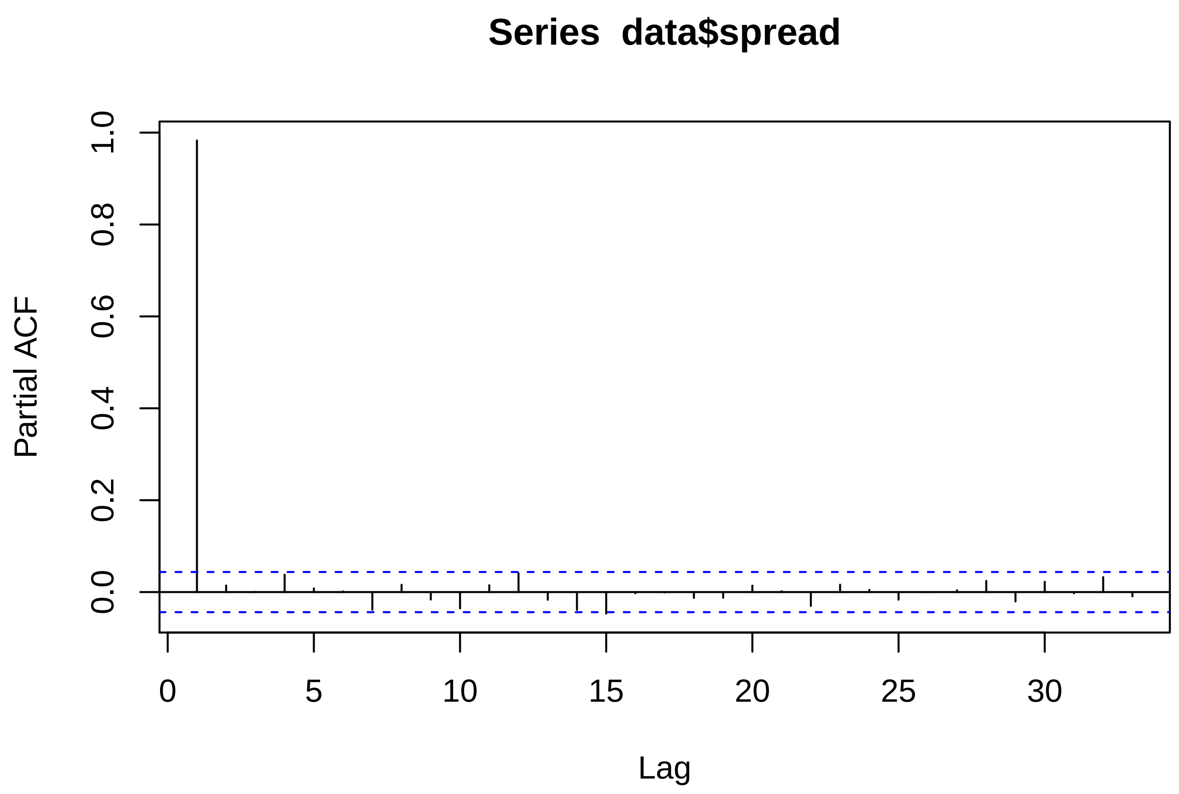
Contrary to the two series individually, the spread seems to be stationary. This is a furhter hint to costationarity of the two series.

```
acf(data$spread)
```



The spread shows strong autocorrelation, which is decaying slightly for further lags.

```r
pacf(data$spread)
```



The spread very significant partial autocorrelation on the first lag, and is insignificant for all further lags.

Models that could suit the data would thus be ARIMA(1,0,0) or a random walk in the form of ARIMA(0,1,0)

(c) Conduct an Augmented Dickey Fuller test on all three time series using the R library "urca". Explain which set-up you choose in the testing procedure (lag length and regression setting) using economic and econometric arguments. Comment on the results.

```
library(urca)
library(broom)


adf_res <- data %>% select(-time) %>%
  imap(~ur.df(.x, lag = 5, type = "none", selectlags = "AIC")@testreg %>% tidy() %>% filter(!grepl("dif
  do.call(what = bind_rows) %>% select(series, statistic, p.value)

pander(adf_res)
```

series	statistic	p.value
EWA	0.5895	0.5556
EWC	0.4029	0.6871
spread	-0.5501	0.5823

We can not reject a unit root for any of the series.

(d) What kind of econometric relationship do the EWA and EWC series adhere to? Explain in a few words the Johansen test. Conduct a Johansen test (command "ca.jo" in the "urca" library) and comment on your results.

```r
library(urca)
library(broom)

ca.jo(data %>% select(EWA, EWC))
```

```
##
## ######################################################
## # Johansen-Procedure Unit Root / Cointegration Test #
## ######################################################
##
## The value of the test statistic is: 3.4759 20.5372
```

Econometric Relationship: Both time series have an order of integration of one, or I(1), meaning that the minimum number of differences that are required to obtain a covariance-stationary series are equal to 1. As the two series move in a similar way, we might suspect that there is a linear combination of these series with integrated of order less than 1, or I, which would mean that the two series are co-integrated.

The Johansen test allows to test if cointegration is present in a set of k time series. There are two approaches to the Johansen's test. The trace and the the maximum eigenvalue approach. Compared to Engle-Granger 2-step approach to estimating cointegration relations, the Johansen test is more powerful as it allows to test for than one cointegration relationship.

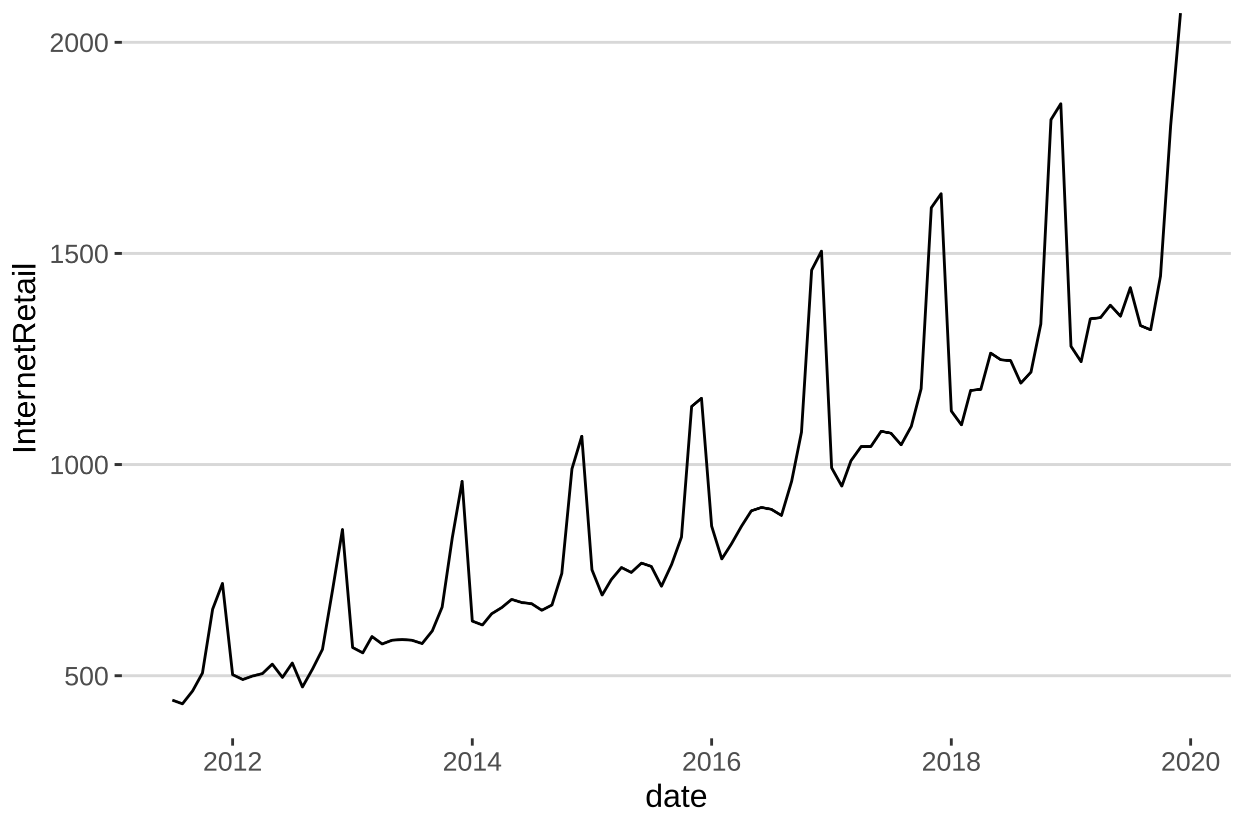
Trace: The null hypothesis is that the number of cointegration vectors is $r = r^* < k$, and the alternative hypothesis that $r = k$. Where testing is done sequentially for $r^* = 1,2,3,..$ and the first value of r that is not rejected is taken as the estimate of r.

Maximum Eigenvalue: The is identical to the trace test but the alternative is $r = r^* + 1$. Likewise, with sequential testing with $r^* = 1,2,\ldots$ the first value of r that is not rejected is taken as the estimate of r.

2. Seasonalities and Forecasting

(a) The title "InternetRetailSales.xlsx" contains information on monthly average in- ternet retail sales in the UK from July 2011 to December 2019 by the British Offce for National Statistics. Load the file into R and turn the data into a time series object using the "tseries" library. Plot the data and comment.

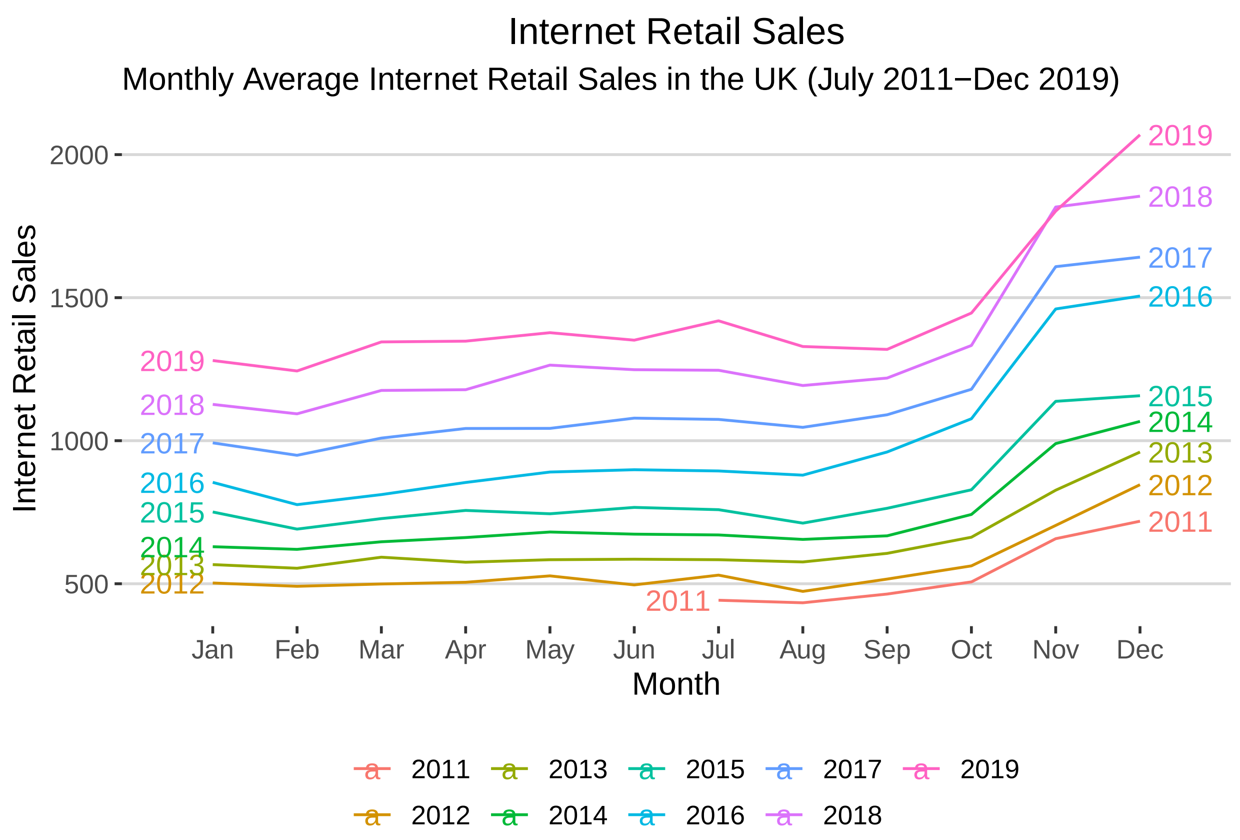
```r
# Read data and convert date column to comfortable format
data_raw <- read_csv2("./data/InternetRetailSales.csv") %>%
  unite(col = "date", Year:Month, sep = " ", remove = FALSE) %>%
  mutate_at("date", ~ymd(., truncated = 1))

# Plot the data
ggplot(data_raw, aes(x = date, y = InternetRetail)) +
  geom_line() +
  theme_hc()
```



From the plot we can observe that the sales are trending upwards. Additionally, we observe sales to be highly seasonal, with a sharp spike in the last quarter and an increase in magnitude over time.

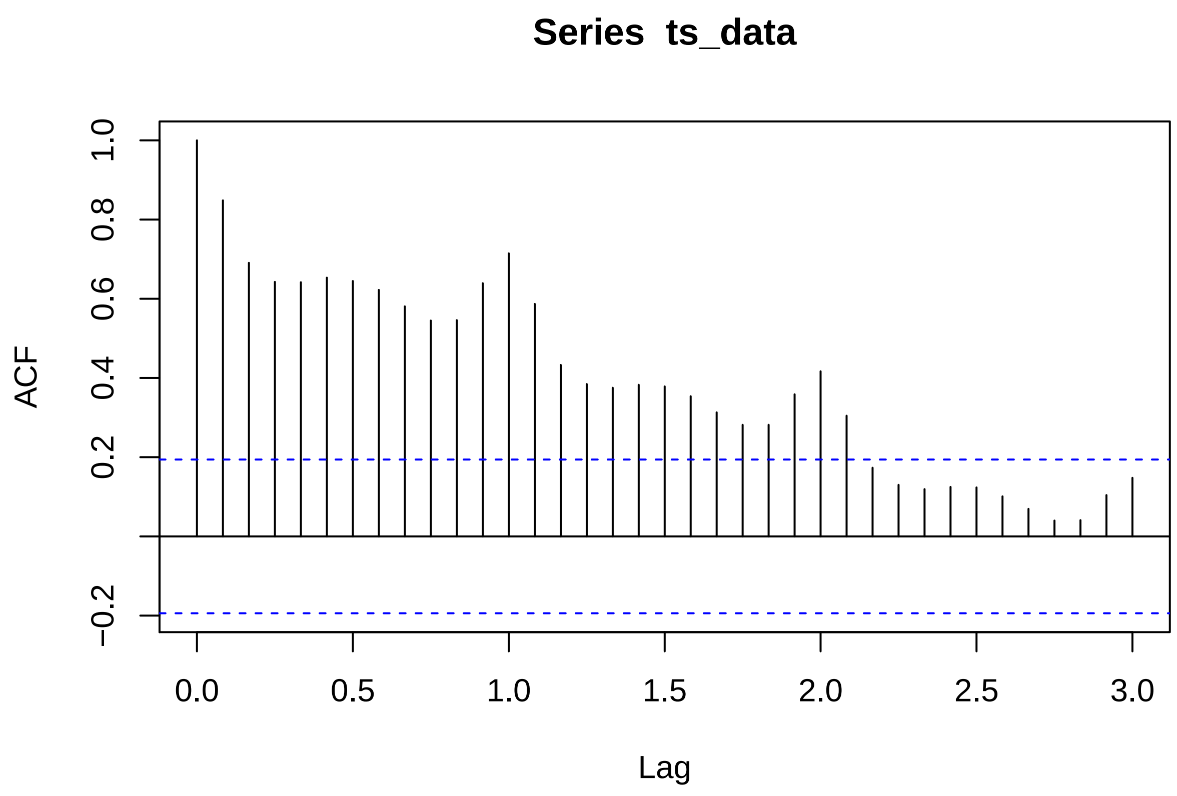
(b) Create a seasonal plot of the data (you can e.g. use the function “ggseasonplot”, choose the frequency appropriately) and plot the ACF of the time series for a number of h = 36 lags. Comment.

```
# Convert data to time series object
ts_data <- ts(data_raw$InternetRetail, frequency = 12, start = c(2011, 7))

# Create seasonal plot
ggseasonplot(ts_data,
             season.labels = TRUE,
             year.labels=TRUE,
             year.labels.left=TRUE) +
  theme(plot.title = element_text(hjust = 0.5)) +
  labs(title = "Internet Retail Sales",
       subtitle = "Monthly Average Internet Retail Sales in the UK (July 2011-Dec 2019)",
       x = "Month",
       y = "Internet Retail Sales")+
  theme(legend.position = "none")+
  guides(color=guide_legend(NULL))+
  theme_hc()
```



With this plot the time axis is fixed to one year and different years are represented by different lines. We can thus better compare sales across years and more clearly see how sales spike in november and december, due to christmas business. Black friday sales could also be considered as source for the increase.

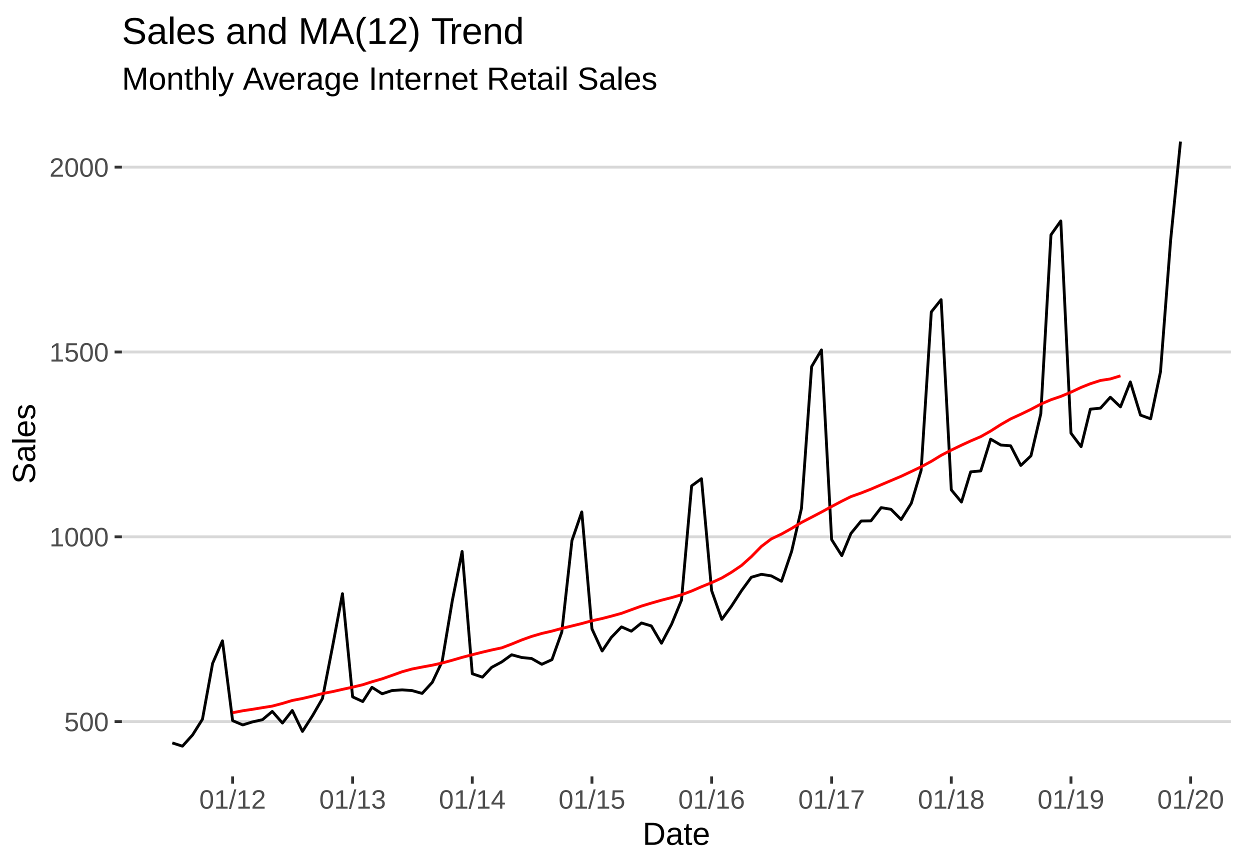
```r
# Create acf plot
acf(ts_data, lag.max = 36)
```



The acf shows how much the sales of one month correlate with the nth lag of monthly sales, up to 36 months. We can see that the correlation is especially high between the holiday sale seasons, which are 12 months apart. The acf is significant up to a lag of 24, resembling a 2 year time period.

(c) Calculate the trend using a moving average with a window of twelve observations. Add it to your plot and comment shortly.

```r
# Add simple moving average with window starting for t-6
data <- data_raw %>% mutate(ma = ma(ts_data, order = 12))

# Plot the data accordingly
ggplot(data, aes(x = date)) +
  geom_line(aes(y = InternetRetail)) +
  geom_line(aes(y = ma), color = "red") +
  scale_x_date(date_breaks = "12 months",
               date_labels = "%m/%y")+
  labs(title = "Sales and MA(12) Trend",
       subtitle = "Monthly Average Internet Retail Sales",
       x = "Date",
       y = "Sales")+
theme_hc()
```



The moving average slides a window of fixed size over the data and them computes the mean of them. It takes the following form

$$\hat{T}_t = \frac{1}{m}\sum_{j=-k}^{k} y_{t+j}$$

with $k = \frac{m-1}{2}$.

It can be seen that the MA can capture the upward trend of the time series and is a good indicator for the trend component of the time series.

(d) Explain the difference between an additive and a multiplicative time series de- composition into seasonal, trend and residual component. Justify which decom- position to choose for the data set.

The type of the time series determines how the trend, seasonal and residual comp onent interact. If the time series is multiplicative, the components multiply together. An increasing trend then results in an increased amplitude of the seasonal activity. It takes the following form:

$$y_t = m_t s_t u_t$$

where m_t is the trend component, s_t is the seasonal component and e_t the residual component. The additive time series is defined by an additive nature of the individual comp onents. An increasing trend then does not corresponds to increasing peaks and troughs of the time series. It stays relatively stable and no change in amplitude can be observed.

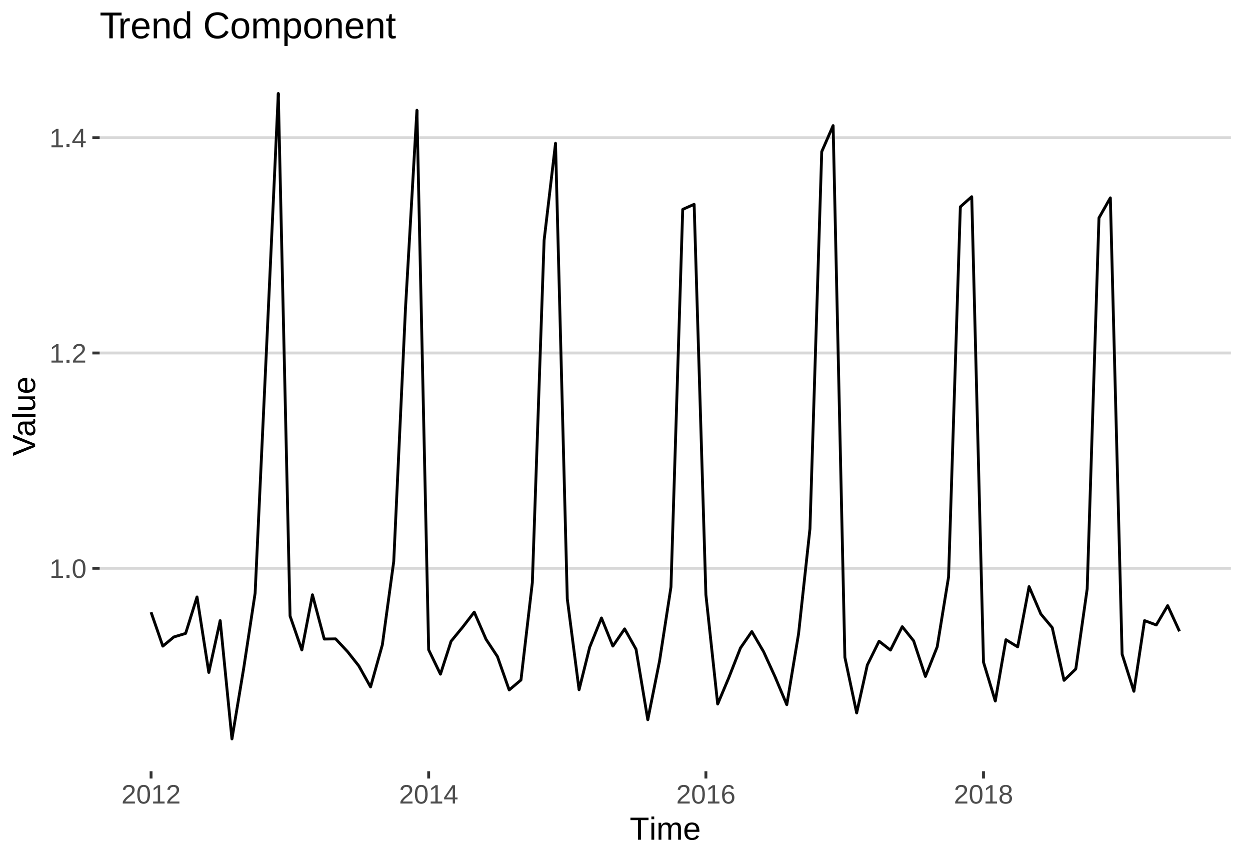
$$y_t = m_t + s_t + u_t$$

For this dataset a multiplicative decomposition should be chosen, since we clearly see an increase in the fluctuation with increasing trend.

The MA process is of order 12, and thus averages over one year. We can clearly see the smooth upwards trend of sales.

(e) Detrend the time series according to your decomposition choice (either multiplica- tive or additive). Would you deem the resulting series to be (weakly) stationary?

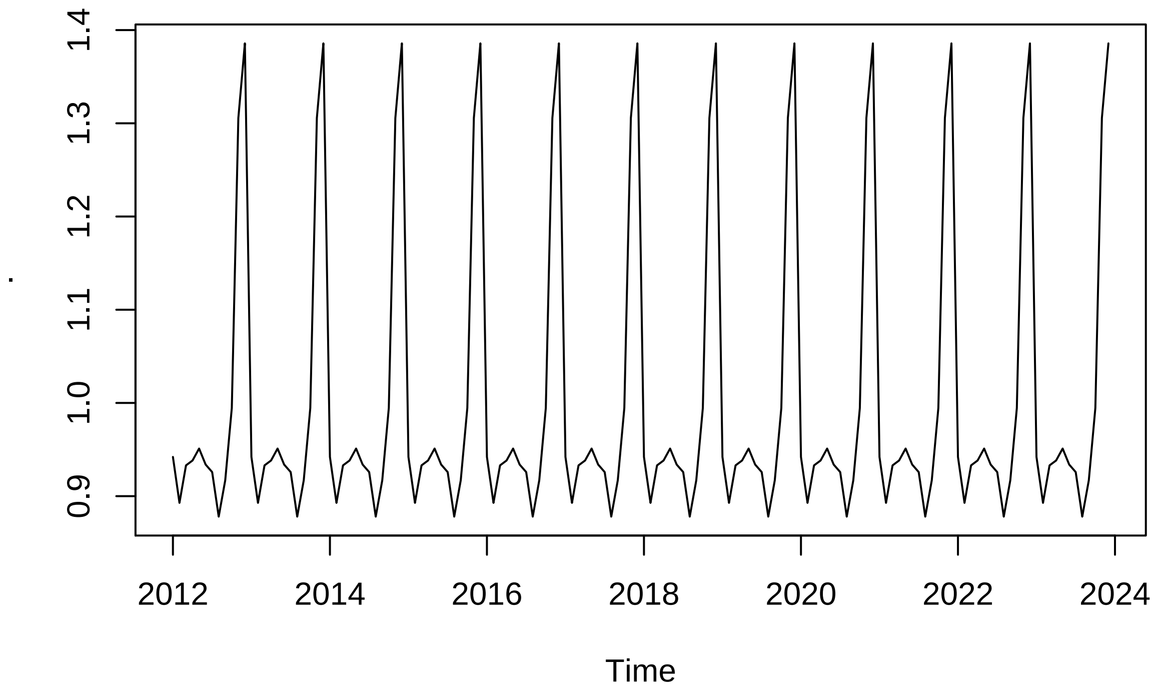
```r
data <- data %>% mutate(m_t = InternetRetail/ma) %>% na.omit
ggplot(data, aes(x = date)) +
  geom_line(aes(y = m_t)) +
    labs(title = "Trend Component",
       x = "Time",
       y = "Value") +
  theme_hc()
```



No, the detrended data is not weakly stationary since the variance seems to be a function of time and seasonal patterns pertain. Variance seems to decreaes over time.

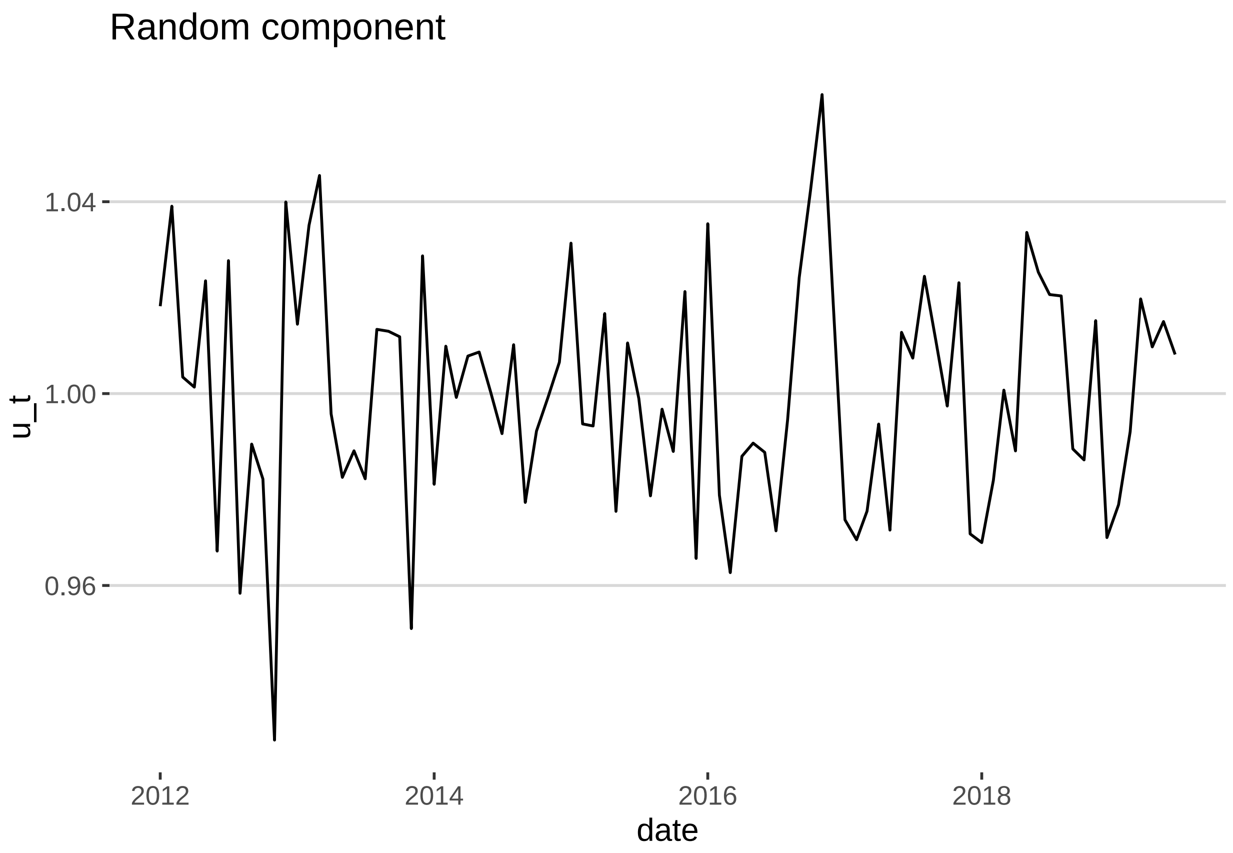
The average seasonality can then be captured by averaging over each month in every year.

```r
seasonal_component <- data %>%
  group_by(Month) %>%
  summarize(s_t = mean(m_t)) %>%
  mutate(Month = month(mdy(Month, truncated = 2), label = T, locale = "eng")) %>%
  arrange(Month)

# Plot seasonal component
ts(rep(seasonal_component$s_t, 12),  frequency = 12, start = c(2012, 1)) %>%
  plot()
```



From that the residual component can be extracted:

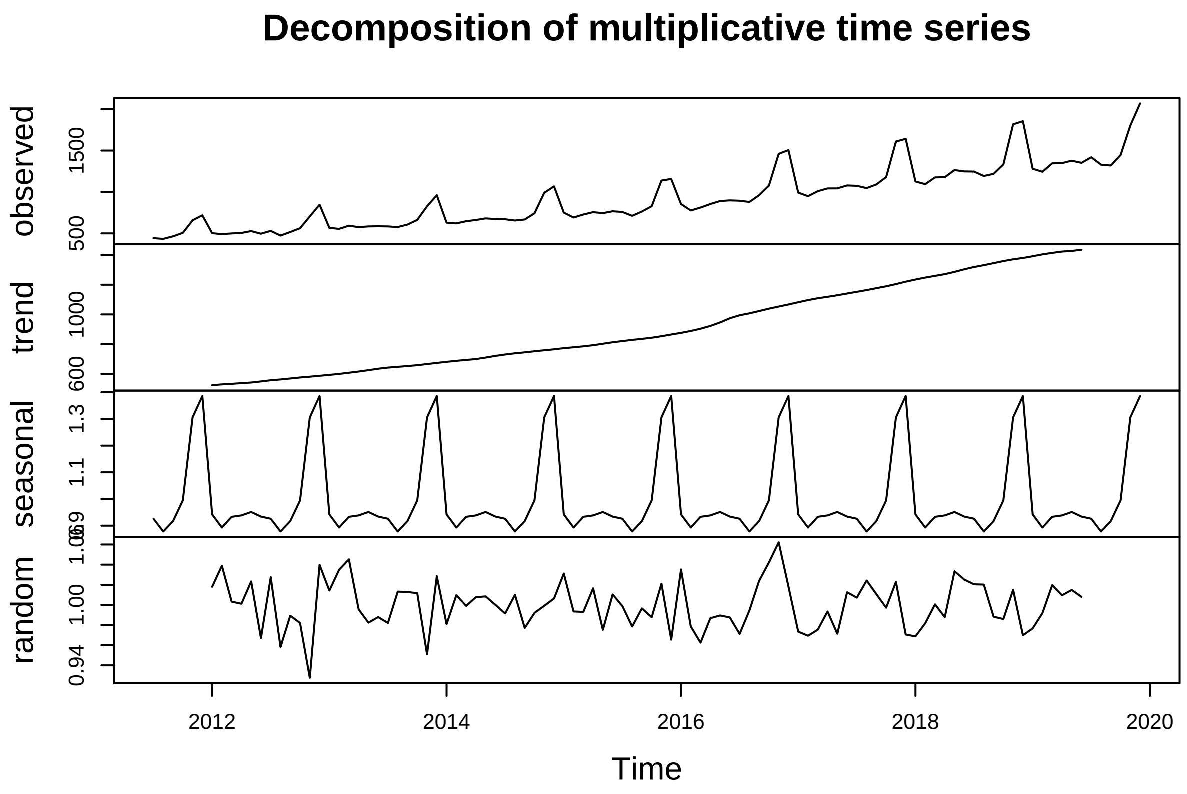
```
# Plot residual
data <- data %>% right_join(., seasonal_component, by = "Month") %>%
  mutate(u_t = InternetRetail / (ma * s_t))

data %>% ggplot(aes(x=date, y = u_t)) +
  geom_line() +
  labs(title = "Random component") +
  theme_hc()
```



Clearly, the error term is white noise, hence the process can be seen as weakly stationary.

(f) Decompose the time series using the "decompose" command and comment.

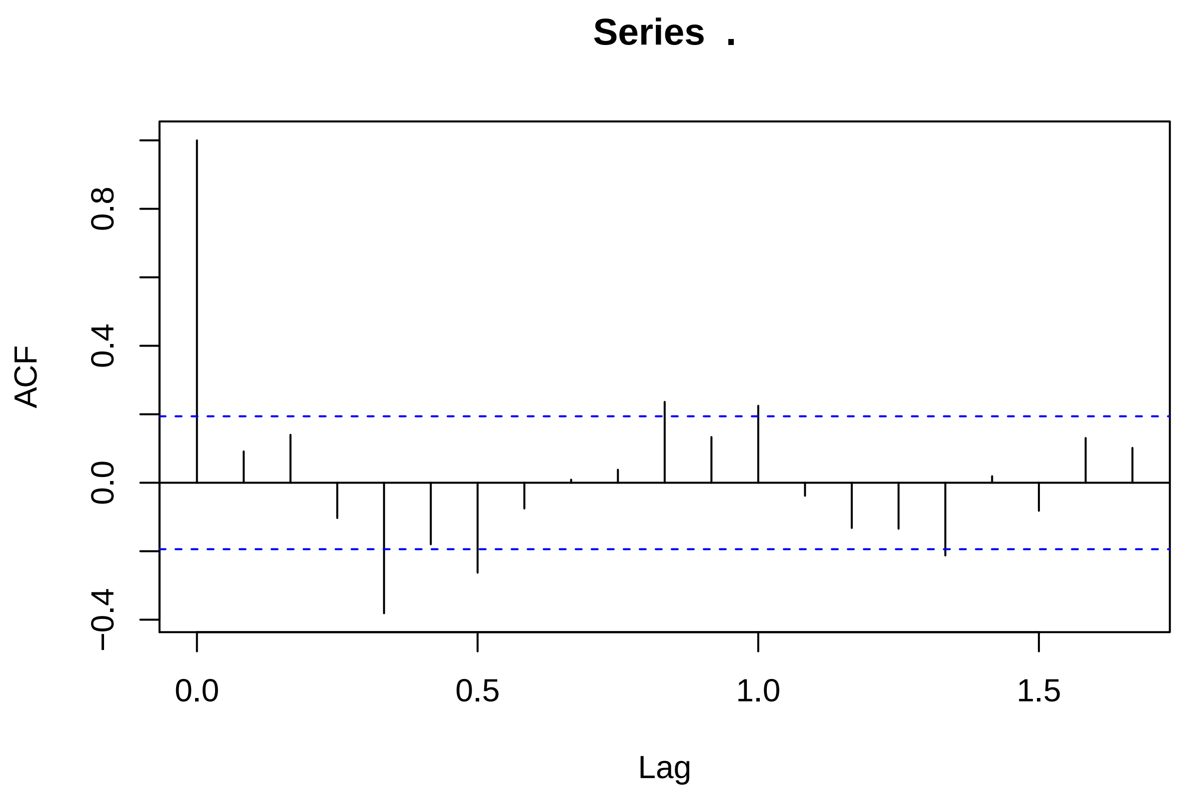
```r
dec_data <- decompose(ts_data, type = "multiplicative")
plot(dec_data)
```



The process is again decomposed into the trend, seasonal and random component. The trend is smooth and continuously rising. The random component seems to be a stationary process with constant mean and variance (white noise). The seasonal component also seems to be stationary since it fluctuates around the trend component.

(g) Investigate the ACF of the residual component of the decomposition. Are they white noise?

```
dec_data$random %>% acf(na.action = na.pass)
```



The residual component of the decomposition, seem to be white noise as the spikes of the acf are only rarely significant.

(h) Fit an ARIMA model using the "auto.arima" command of the "forecast" library. Check the model diagnostics.

```
arima <- auto.arima(ts_data)
summary(arima) %>% pander
```

```
## Series: ts_data
## ARIMA(0,1,1)(0,1,0)[12]
##
## Coefficients:
##           ma1
##       -0.6060
## s.e.   0.0997
##
## sigma^2 estimated as 1957:  log likelihood=-463.28
## AIC=930.56   AICc=930.7   BIC=935.54
##
## Training set error measures:
##                    ME     RMSE      MAE         MPE     MAPE      MASE
## Training set 1.208206 41.08904 26.14508 -0.07369533 2.616899 0.2116649
##                    ACF1
## Training set 0.05520268
```

	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1
Training set	1.208	41.09	26.15	-0.0737	2.617	0.2117	0.0552

A seasonal ARIMA (0,1,1)x(0,1,0) with period 12 has been chosen by the function. So for the non-seasonal part pf the model a (0,1,1) ARIMA has been selected and for the seasonal part a (0,1,0) applies.

The RMSE with 41. looks appealing, particularly when comparing to a ordinal ARIMA model:

```r
arima_ord <- auto.arima(data$InternetRetail)
map_dfr(list(arima, arima_ord), sw_glance) %>% pander
```

Table 3: Table continues below

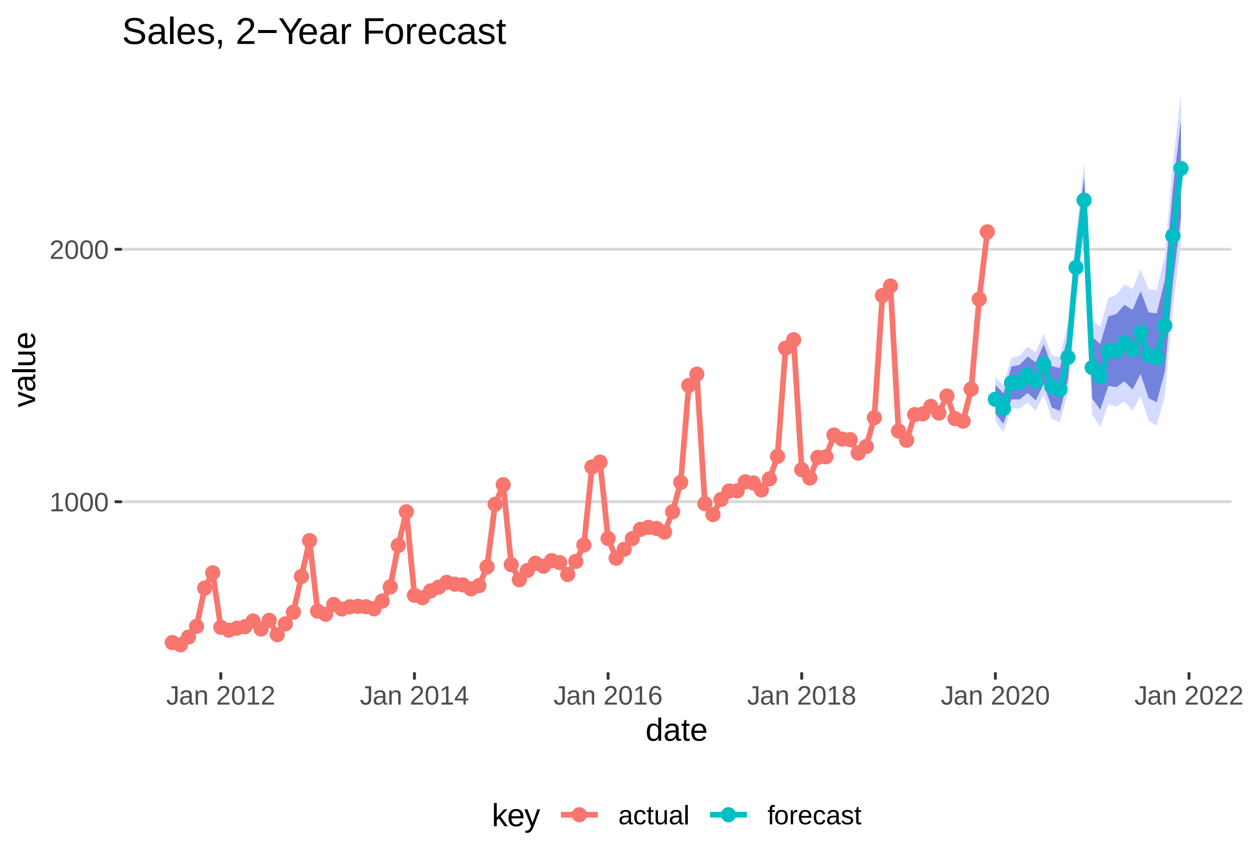
model.desc	sigma	logLik	AIC	BIC	ME	RMSE
ARIMA(0,1,1)(0,1,0)[12]	44.24	-463.3	930.6	935.5	1.208	41.09
ARIMA(4,1,2)	210.7	-602.8	1220	1237	32.77	202.4

MAE	MPE	MAPE	MASE	ACF1
26.15	-0.0737	2.617	0.2117	0.0552
152.4	-1.133	17.97	0.7308	-0.02428

which has a higher RMSE and performs worse in the other metrics as well.

(i) Calculate h = 24 step ahead forecasts for the time series using the function "fore- cast" of the library "forecast". Does your forecast live up to your expectations?

```r
preds <- forecast(arima, h = 24) %>% sweep::sw_sweep(timekit_idx = TRUE, rename_index = "date")

preds %>%
  ggplot(aes(x = date, y = value, color = key)) +
  # Prediction intervals
  geom_ribbon(aes(ymin = lo.95, ymax = hi.95),
              fill = "#D5DBFF", color = NA, size = 0) +
  geom_ribbon(aes(ymin = lo.80, ymax = hi.80, fill = key),
              fill = "#596DD5", color = NA, size = 0, alpha = 0.8) +
  # Actual & Forecast
  geom_line(size = 1) +
  geom_point(size = 2) +
  # Aesthetics
  theme_hc() +
  labs(title = "Sales, 2-Year Forecast", x = "date", y = "value")
```



```r
#' Comment:
```

The forecast follows our expectations as it continues the seasonal pattern closely, while accounting for the multiplicative nature of the series.