Time Series Econometrics

Assignment II

Matthias Spichiger (15-937-557), Louis Pfau (15-607-732)

April 25, 2020

Assignment 2

```r
rm(list = ls(all.names = TRUE))
set.seed(2020)
getwd()
# setwd("C:/your_directory")
```

```r
# Restart with fresh R session if necessary with: .rs.restartR()
# install.packages(c("readxl", "ggplot2", "quantmod", "urca", "tseries", "forecast"))
library(readxl)   # To import excel file
library(ggplot2)  # For nice plots in R
library(quantmod) # For financial trading strategies.
library(urca)     # Unit root and cointegration tests
library(tseries)  # For Time series analysis and computational finance.
library(forecast) # For time series forecasts, automatic ARIMA modelling
```

1. ETF Data Analysis

Exercise 1 (a) ETF Time Series

Use the R **`quatnmod`** 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.

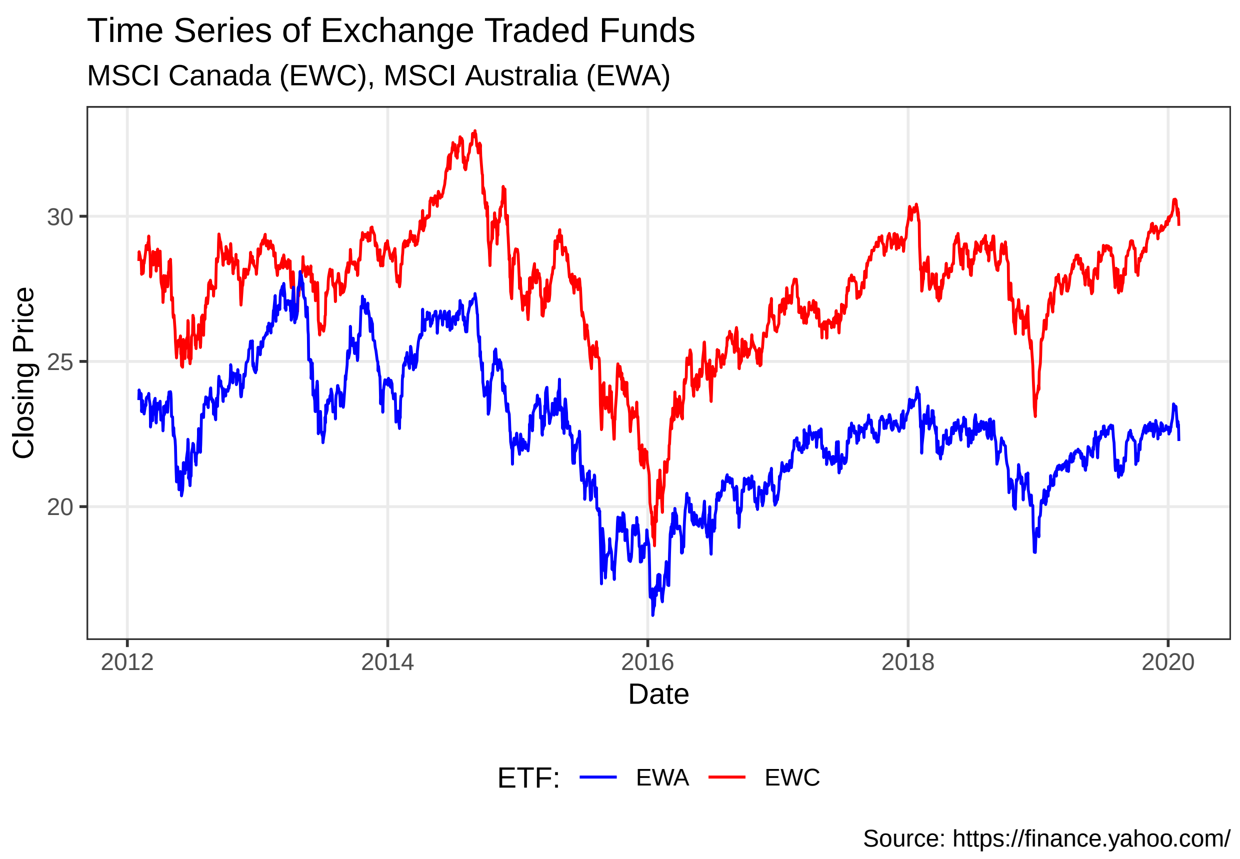
Plot the two time series in one plot and comment.

```r
getSymbols(Symbols = c("EWA", "EWC"),
           from = "2012-02-02",
           to = "2020-02-02",
           src = "yahoo")
```

```r
EWA = zoo(EWA$EWA.Close)

EWC = zoo(EWC$EWC.Close)
```

Next, we plot the two time series in one plot:

```r
a1 = autoplot(EWC, facet = NULL, color="red")+ scale_x_date()+
  labs(title = "Time Series of Exchange Traded Funds",
       subtitle = "MSCI Canada (EWC), MSCI Australia (EWA)",
       caption = "Source: https://finance.yahoo.com/",
       x = "Date",
       y = "Closing Price") +
  geom_line(mapping = aes(y = EWA,
                          color="blue"))+
  scale_color_manual(name = "ETF:",
                     labels = c("EWA", "EWC"),
                     values = c("blue", "red")) +
  theme_bw()+
  theme(panel.grid.minor = element_blank(),
        legend.position = "bottom")


a1
```



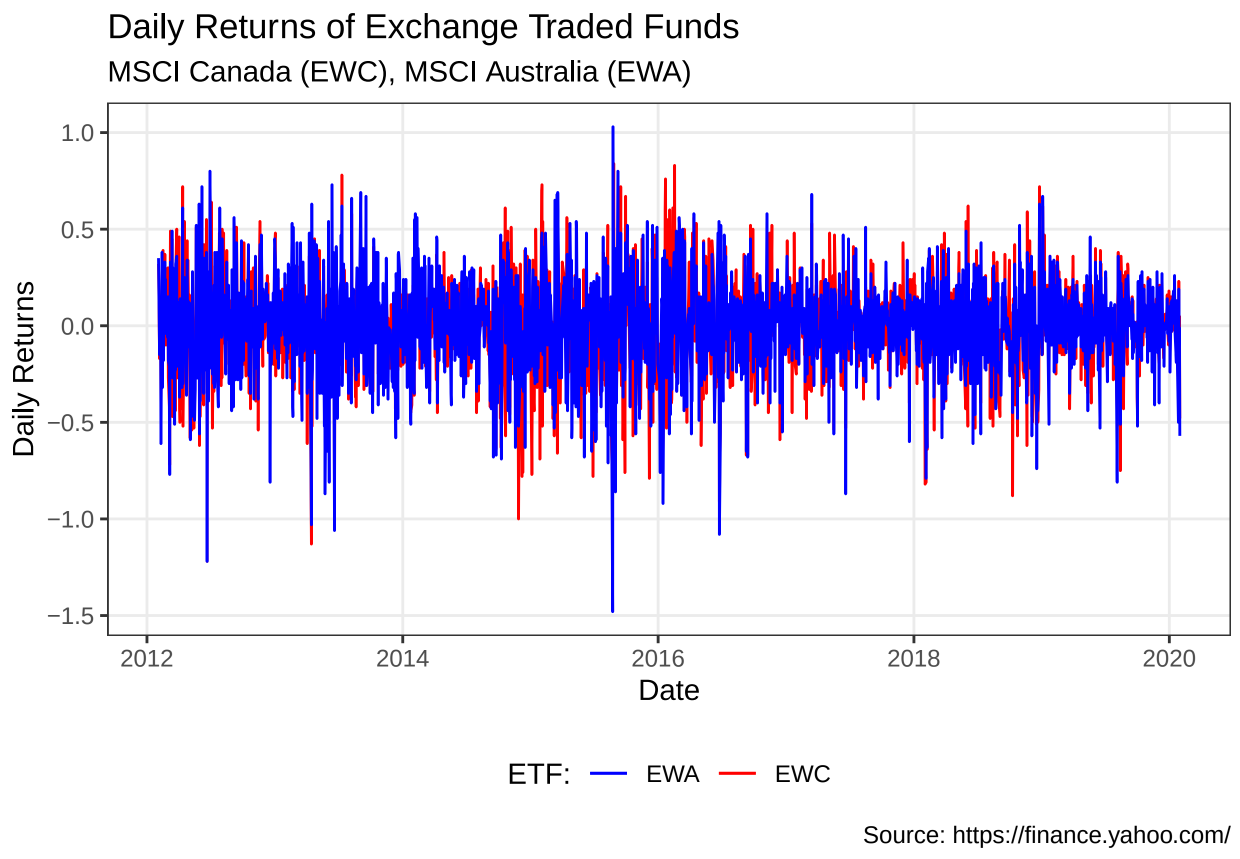
```r
# ggsave("ETFplots.png", plot = a1, width = 17, height = 12, units = "cm")
# dev.off()
```

We want to take the first difference of the two time series, that is, calculate the daily returns of the two ETFs and plot the resulting time series in one plot:

```r
EWAdiff = diff(x = EWA, lag = 1, differences = 1)
EWCdiff = diff(x = EWC, lag = 1, differences = 1)
```

Then we create the plot:

```r
a2 = autoplot(EWCdiff, facet = NULL, color="red")+
  scale_x_date()+
  labs(title = "Daily Returns of Exchange Traded Funds",
       subtitle = "MSCI Canada (EWC), MSCI Australia (EWA)",
       caption = "Source: https://finance.yahoo.com/",
       x = "Date",
       y = "Daily Returns") +
  geom_line(mapping = aes(y = EWAdiff, color="blue"))+
  scale_color_manual(name = "ETF:",
                     labels = c("EWA", "EWC"),
                     values = c("blue", "red")) +
  theme_bw()+
  theme(panel.grid.minor = element_blank(),
        legend.position = "bottom")

a2
```



```r
# ggsave("ETFreturns.png", plot = a2, width = 17, height = 12, units = "cm")
# dev.off()
```

Comment:

- Both ETF time series look like random walks, that is, nonstationary time series, possibly with drift.
- The daily returns of the two ETFs look like time series that are stationary around a mean of zero.
- The difference between the Canadian ETF and the Australian ETF, that is the spread, seems to be about 4 or five percentage points on average.

Exercise 1 (b) ACF & PACF

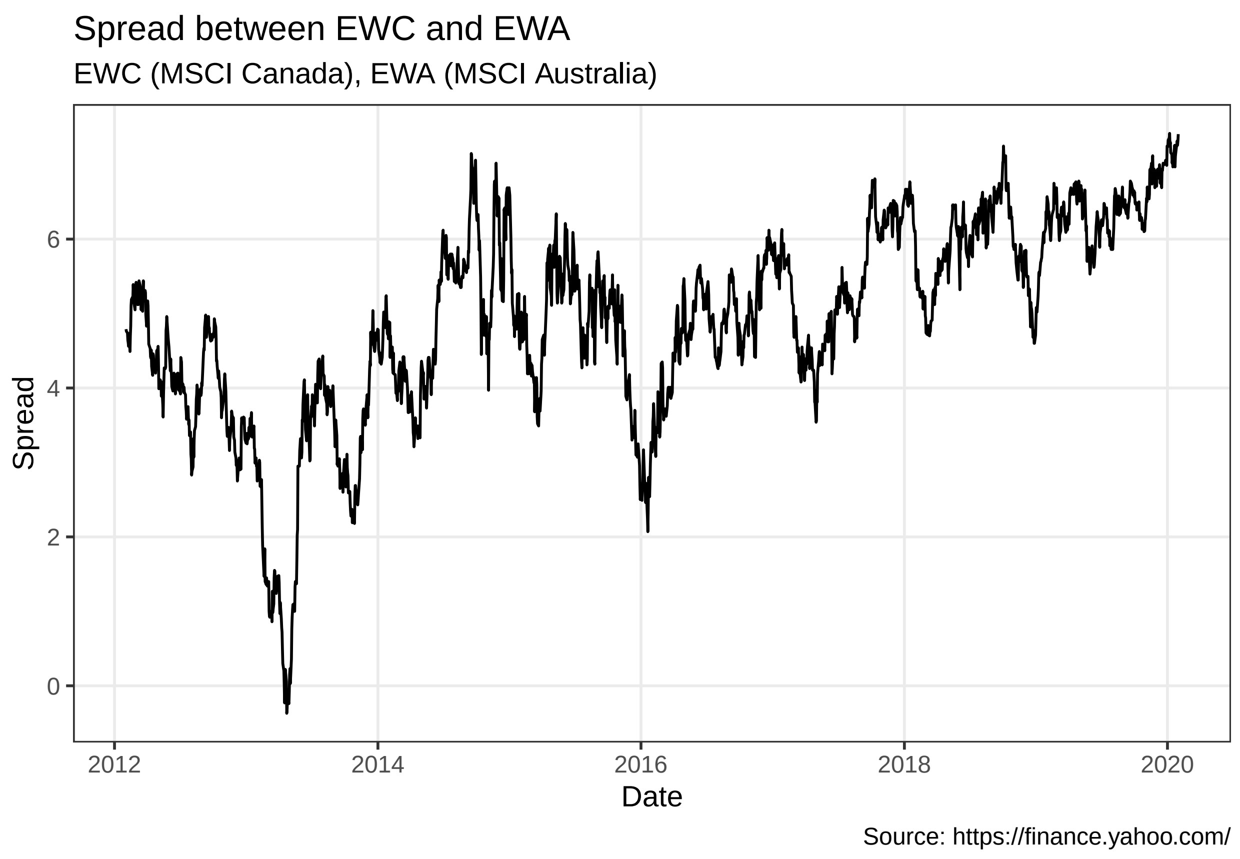
Calculate the spread of the two time series, plot it and calculate ACF and PACF.

First, we calculate the spread:

```r
SPREAD = EWC - EWA
```

Then we plot the spread of the two time series:

```r
a3 = autoplot(SPREAD, facet = NULL, color="black")+
  scale_x_date()+
  labs(title = "Spread between EWC and EWA",
       subtitle = "EWC (MSCI Canada), EWA (MSCI Australia)",
       caption = "Source: https://finance.yahoo.com/",
       x = "Date",
       y = "Spread") +
  scale_color_manual(labels = c("Spread"),
                     values = c("black")) +
  theme_bw()+
  theme(panel.grid.minor = element_blank(),
        legend.position = "none")+
  guides(color=guide_legend(NULL))

a3
```



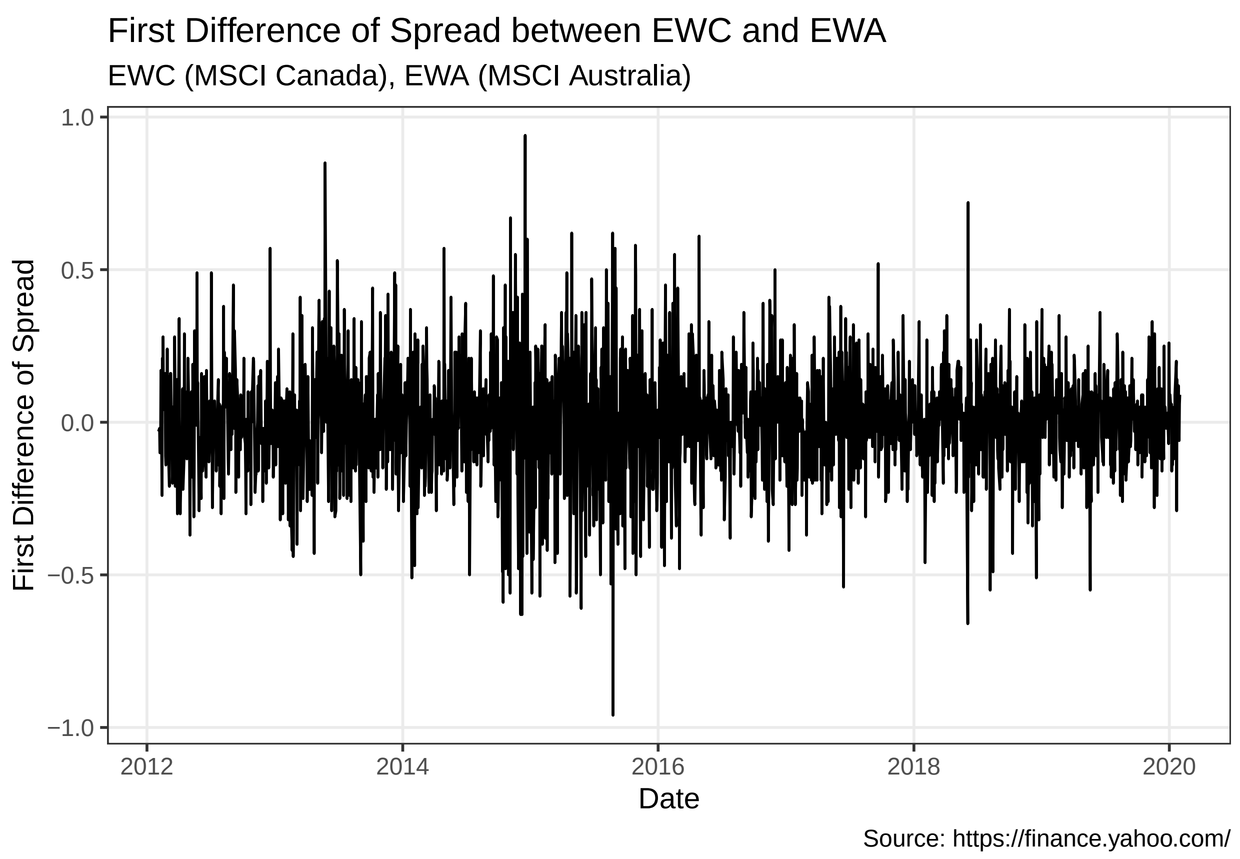
```
# ggsave("ETFspread.png", plot = a3, width = 17, height = 12, units = "cm")
# dev.off()
```

We may want to calculate the first difference of the spread time series and plot the resulting time series. First we calculate the difference:

```
SPREADdiff = diff(x = SPREAD, lag = 1, differences = 1)
```

Then we create the plot:

```r
a4 = autoplot(SPREADdiff, facet = NULL, color="black")+
  scale_x_date()+
  labs(title = "First Difference of Spread between EWC and EWA",
       subtitle = "EWC (MSCI Canada), EWA (MSCI Australia)",
       caption = "Source: https://finance.yahoo.com/",
       x = "Date",
       y = "First Difference of Spread") +
  scale_color_manual(labels = c("Diff. of Spread"),
                     values = c("black")) +
  theme_bw()+
  theme(panel.grid.minor = element_blank(),
        legend.position = "none")+
  guides(color=guide_legend(NULL))

a4
```



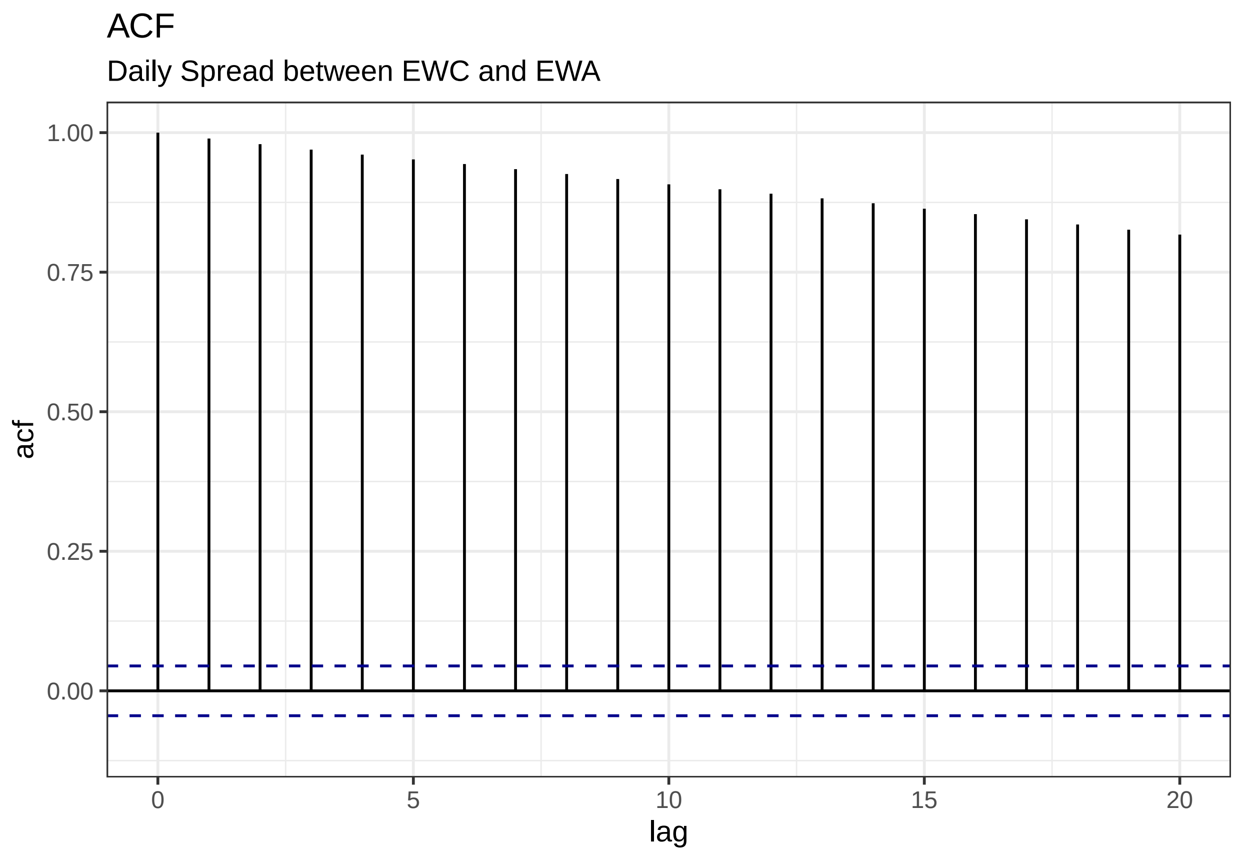
```
# ggsave("SpreadDiff.png", plot = a4, width = 17, height = 12, units = "cm")
# dev.off()
```

Comment:

- The spread between the Canadian and the Australian index looks like a time series that is stationary around a mean of about 5 percentage points.
- Because the two ETFs form an economic equilibrium relationship, it may be that the two time series are cointegrated, that is, both ETFs are integrated of order 1, that is, they are $I(1)$. It would follow that their linear combination is integrated of order zero, that is, the spread is $I(0)$.

Finally, we crate the ACF of the Spread:

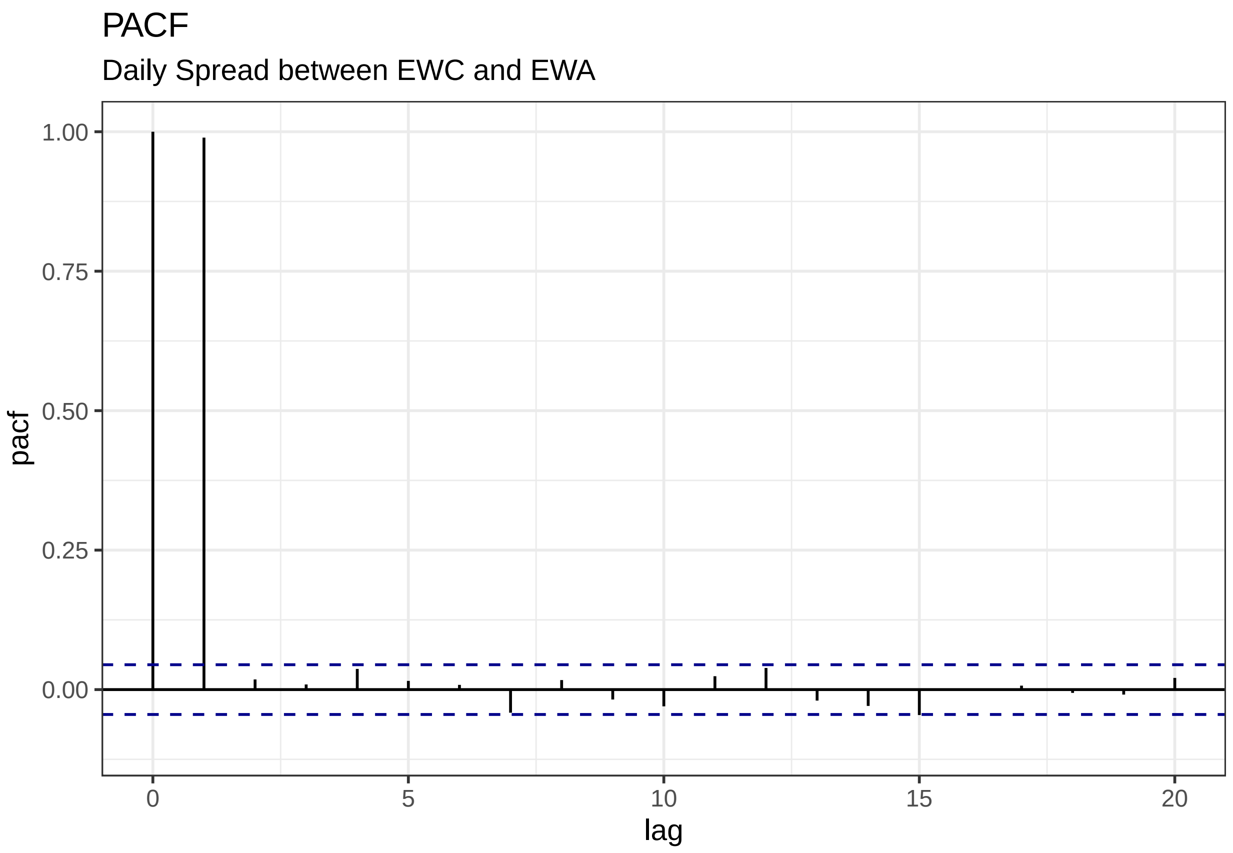
```r
maxlag = 20
SPREAD = as.numeric(SPREAD)
# Plot ACF using ggplot2 (or more simply: acf(SPREAD), pacf(SPREAD))
qacf      = acf(SPREAD, lag.max = maxlag, plot = FALSE)
qacfdf    = with(qacf, data.frame(lag, acf))
ciline    = 2/sqrt(length(SPREAD)) # iid confidence interval at 5% level
q1        = ggplot(data = qacfdf,
                   mapping = aes(x = lag, y = acf)) +
  geom_hline(aes(yintercept = 0)) +
  geom_segment(mapping = aes(xend = lag, yend = 0)) +
  geom_hline(aes(yintercept = ciline),
             linetype = 2, color = 'darkblue') +
  geom_hline(aes(yintercept = -ciline),
             linetype = 2, color = 'darkblue') +
  labs(title="ACF",
       subtitle="Daily Spread between EWC and EWA") +  # title
  theme(panel.grid.minor = element_blank()) +
  theme_bw()+
  coord_cartesian(xlim = c(0, maxlag), ylim = c(-0.1,1))

q1
```



```r
# ggsave("spreadACF.png", plot = q1, width = 12, height = 7, units = "cm")
# dev.off()
```

We also create a pot of the PACF:

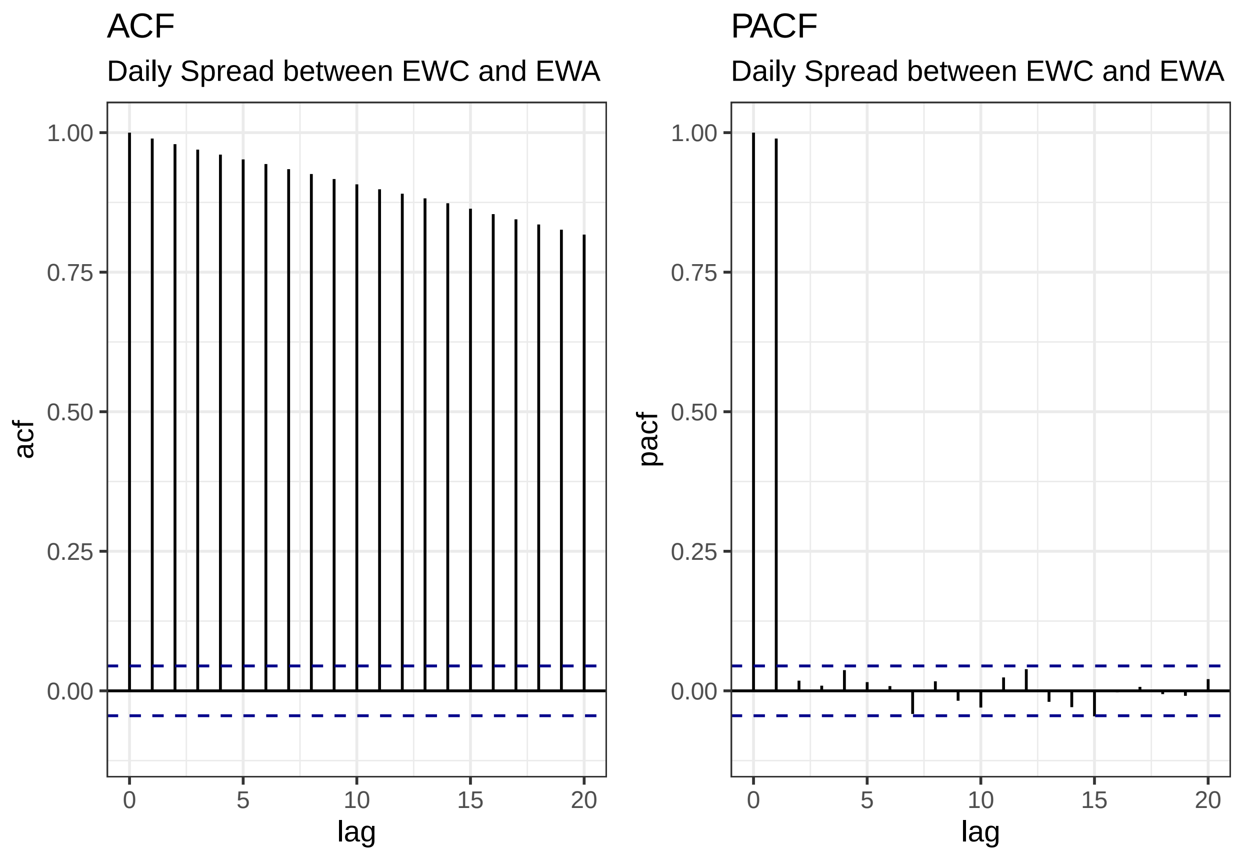
```r
maxlag = 20
qpacf     = pacf(SPREAD, lag.max = maxlag, plot = FALSE)
qpacfdf   = with(qpacf, data.frame(c(0,lag), c(1,acf)))
ciline    = 2/sqrt(length(SPREAD)) # iid confidence interval at 5% level
q2        = ggplot(data = qpacfdf,
                   mapping = aes(x = c.0..lag., y = c.1..acf.)) +
  geom_hline(aes(yintercept = 0)) +
  geom_segment(mapping = aes(xend = c.0..lag., yend = 0)) +
  geom_hline(aes(yintercept = ciline),
             linetype = 2, color = 'darkblue') +
  geom_hline(aes(yintercept = -ciline),
             linetype = 2, color = 'darkblue') +
  labs(title="PACF",
       y ="pacf",
       x ="lag",
       subtitle="Daily Spread between EWC and EWA") +  # title
  theme(panel.grid.minor = element_blank())  +
  theme_bw()+
  coord_cartesian(xlim = c(0, maxlag), ylim = c(-0.1,1))
q2
```



```
# ggsave("spreadPACF.png", plot = q2, width = 12, height = 7, units = "cm")
```

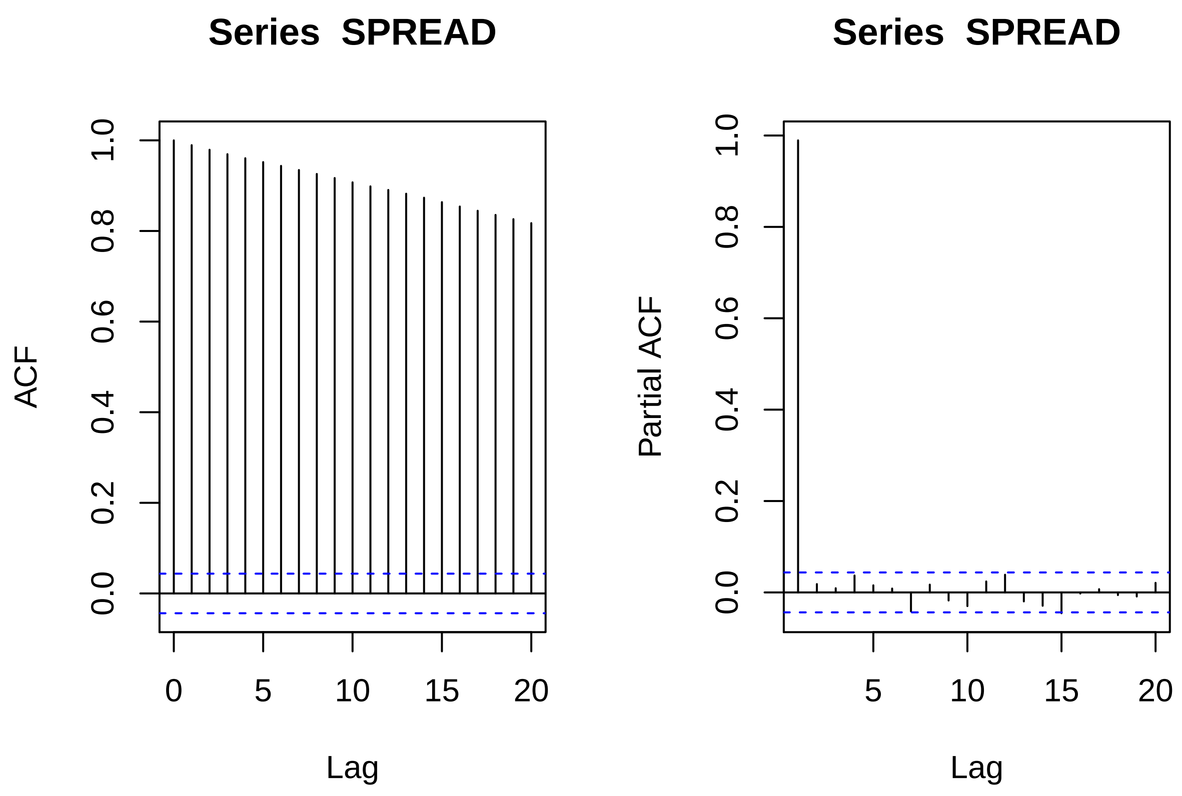
For convenience, we want to plot the two functions next to each other:

```r
# Combine plots via the multiplot function
multiplot <- function(..., file, cols=NULL) {
  library(grid)
  # Make a list from the ... arguments
  plots <- c(list(...))
  numPlots = length(plots)
  # Make the panel
  # ncol: Number of columns of plots
  # nrow: Number of rows needed, calculated from # of cols
  layout <- matrix(seq(1, cols * ceiling(numPlots/cols)),
                   ncol = cols, nrow = ceiling(numPlots/cols))
  # Set up the page
  grid.newpage()
  pushViewport(viewport(layout = grid.layout(nrow(layout), ncol(layout))))
  # Create each plot in correct location
  for (i in 1:numPlots) {
    # Get the i,j matrix positions of the regions that contain this subplot
    matchidx <- as.data.frame(which(layout == i, arr.ind = TRUE))
    print(plots[[i]], vp = viewport(layout.pos.row = matchidx$row,
                                    layout.pos.col = matchidx$col))
  }
}
multiplot(q1, q2, cols=2)
```



We can calculate the above plot more quickly:

```r
par(mfrow=c(1,2))
acf(x = SPREAD, lag.max = 20)
pacf(x = SPREAD, lag.max = 20)
```



```
# dev.off()
```

Comment:

- The ACF plot that the autocorrelations are slowly decaying with increasing lags. This suggests an autogregressive process in the data.
- The PACF is basically zero after the first lag.
- Together, the ACF and PACF plots suggest an AR(1) process.
- The ACF plot would suggest that the first coefficient of the AR(1) process is close to one, that is, barely smaller than one.

Becaause the autocorrelation function of an AR(1) process is given by

$$\rho_h = \phi_1^{|h|}$$

we know that the ACF plot would show a geometric decay if $|\phi_1| < 1$. Because the decay is so slow, we assume that ϕ_1 is very close to one.

If we had $\phi_1 = 1$, we would have a unit root process without drift. $\phi_1| < 1$, the process would be stationary.

If we had $|\phi_1| < 1$, the process would be stationary.

Name two ARIMA(p,d,q) models that could suit the data according to the plots.

1. AR(1) with positive first coefficient that is smaller than one. This would be an ARIMA(1,0,0) model.
2. Random Walk without drift. This would be a unit root process, that is, the first coefficient would be equal to one. This would be ARIMA(0, 1, 0), that is a model that is integrated of order 1, or I(1).

Exercise 1 (c) Augmented Dickey Fuller Test

Conduct an Augmented Dickey Fuller test on all three time series using the R libary `urca`.

Explain which set-up you choose in the testing procedure (lag length and regression setting) using economic and eocnometric arguments.

Explanation:

We assume that a model may look like the equation below:

$$y_t = \rho y_{t-1} + \alpha + \beta t + \epsilon_t$$

First difference leads to the following relationship:

$$\Delta y_t = (\rho - 1) y_{t-1} + \alpha + \beta t + \epsilon_t$$

where $\rho = 1$ would be a unit root, $\rho < 1$ could be interpreted as the adjustment speed, β is the time trend and ϵ_t is white noise $\epsilon_t \sim (0, \sigma_\epsilon^2)$. We differentiate between two possible cases:

This leads us with the following possile Null hypotheses to test:

1. Test for a unit root: $H_0 : \rho = 1$, `type = "none"`

$$\Delta y_t = (\rho - 1)y_{t-1} + \epsilon_t$$

$H_0 : \rho = 1$, Test Statistic τ, Critical Values for 95% and 99%: (-1.95, -2.60)

2. Test for a unit root with drift: $H_0 : \rho = 1, \quad \alpha = 0$, `type = "drift"`

$$\Delta y_t = \alpha + (\rho - 1)y_{t-1} + \epsilon_t$$

$H_0 : \rho = 1$, Test Statistic τ, Critical Values for 95% and 99%: (-2.89, -3.51)

$H_0 : \rho = 1, \quad \alpha = 0$, Test Statistic ϕ_1, Critical Values for 95% and 99%: (4.71, 6.70)

3. Test for a unit root with drift and deterministic time trend, `type = "trend"`

$$\Delta y_t = \alpha + (\rho - 1)y_{t-1} + \beta t + \epsilon_t$$

$H_0 : \rho = 1$, Test Statistic τ, Critical Values for 95% and 99%: (-3.45, -4.04)

$H_0 : \rho = 1, \quad \beta = 0$, Test Statistic ϕ_3, Critical Values for 95% and 99%: (-6.49, 8.73)

$H_0 : \rho = 1, \quad \alpha = \beta = 0$, Test Statistic ϕ_2, Critical Values for 95% and 99%: (4.88, 6.50)

Note the `ur.df()` always reports the Test statistics in the following order: $\tau, (\phi_1), \phi_2, \phi_3$.

Because we are not sure, whether EWA is growing or not, we start with the full model and 8 lags:

```r
adfEWA = ur.df(y = EWA, type = "trend", lags = 8)
summary(adfEWA)
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression trend
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 + 1 + tt + z.diff.lag)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.52413 -0.13557  0.01298  0.14960  0.96837
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   1.676e-01  6.863e-02   2.441  0.01471 *
## z.lag.1      -6.926e-03  2.777e-03  -2.494  0.01270 *
## tt           -1.140e-05  1.076e-05  -1.059  0.28951
## z.diff.lag1 -1.686e-02  2.245e-02  -0.751  0.45271
## z.diff.lag2  2.422e-03  2.244e-02   0.108  0.91406
## z.diff.lag3  4.553e-03  2.239e-02   0.203  0.83889
## z.diff.lag4 -6.166e-02  2.237e-02  -2.757  0.00589 **
## z.diff.lag5 -3.634e-02  2.238e-02  -1.624  0.10460
## z.diff.lag6  4.758e-02  2.240e-02   2.124  0.03378 *
## z.diff.lag7  7.519e-04  2.242e-02   0.034  0.97326
## z.diff.lag8 -1.192e-02  2.241e-02  -0.532  0.59498
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2483 on 1992 degrees of freedom
## Multiple R-squared:  0.01145,    Adjusted R-squared:  0.006485
## F-statistic: 2.307 on 10 and 1992 DF,  p-value: 0.01082
##
##
## Value of test-statistic is: -2.4942 2.0794 3.1136
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau3 -3.96 -3.41 -3.12
## phi2  6.09  4.68  4.03
## phi3  8.27  6.25  5.34
```

We choose a smaller model, because barely any lags were significant, and we failed to reject any Null hypotheses.

```r
adfEWA = ur.df(y = EWA, type = "trend", lags = 1)
summary(adfEWA)
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression trend
##
##
```

```
## Call:
## lm(formula = z.diff ~ z.lag.1 + 1 + tt + z.diff.lag)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.5171 -0.1364  0.0078  0.1524  0.9623
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.758e-01  6.790e-02   2.590  0.00968 **
## z.lag.1     -7.318e-03  2.749e-03  -2.662  0.00782 **
## tt          -1.113e-05  1.070e-05  -1.040  0.29847
## z.diff.lag  -1.939e-02  2.234e-02  -0.868  0.38557
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.249 on 2006 degrees of freedom
## Multiple R-squared:  0.004067,   Adjusted R-squared:  0.002578
## F-statistic: 2.731 on 3 and 2006 DF,  p-value: 0.04248
##
##
## Value of test-statistic is: -2.6624 2.3805 3.558
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau3 -3.96 -3.41 -3.12
## phi2  6.09  4.68  4.03
## phi3  8.27  6.25  5.34
```

The intercept and the first lag are significant. We fail to reject any of the Null hypothesis.

Therefore we have to accept the alternative hypothesis that there is a unit root ($\rho = 1$), that there is a zero intercept ($\alpha = 0$) and that there is no deterministic time trend ($\beta = 0$).

The model that fits the EWA ETF time series is therefore look like a random walk without drift.

$$y_t = y_{t-1} + \epsilon_t$$

This process is non-stationary, because repeated substitution shows that

$$y_t = y_{t-T} + \epsilon_{t-T} + \cdots + \epsilon_{t-2} + \epsilon_{t-1} + \epsilon_t$$

Changing the indices via $t - T = 0$ allows for the redefinition to

$$y_t = y_0 + \sum_{i=0}^{t} \epsilon_i$$

This process is non-stationary. Its mean is

$$E\left[y_0\right]$$

but the variance is

$$Var[y_0] = Var\left[y_0 + \sum_{i=0}^{t} \epsilon_i\right] = Var\left[\sum_{i=0}^{t} \epsilon_i\right] = \sum_{i=0}^{t} Var\left[\epsilon_i\right] = t \times \sigma_\epsilon^2$$

Therefore the variance is not time-independent, but growing over time. This is the defining feature of a random walk.

Note however that the first difference is a stationary process

$$\Delta y_t = \epsilon_t$$

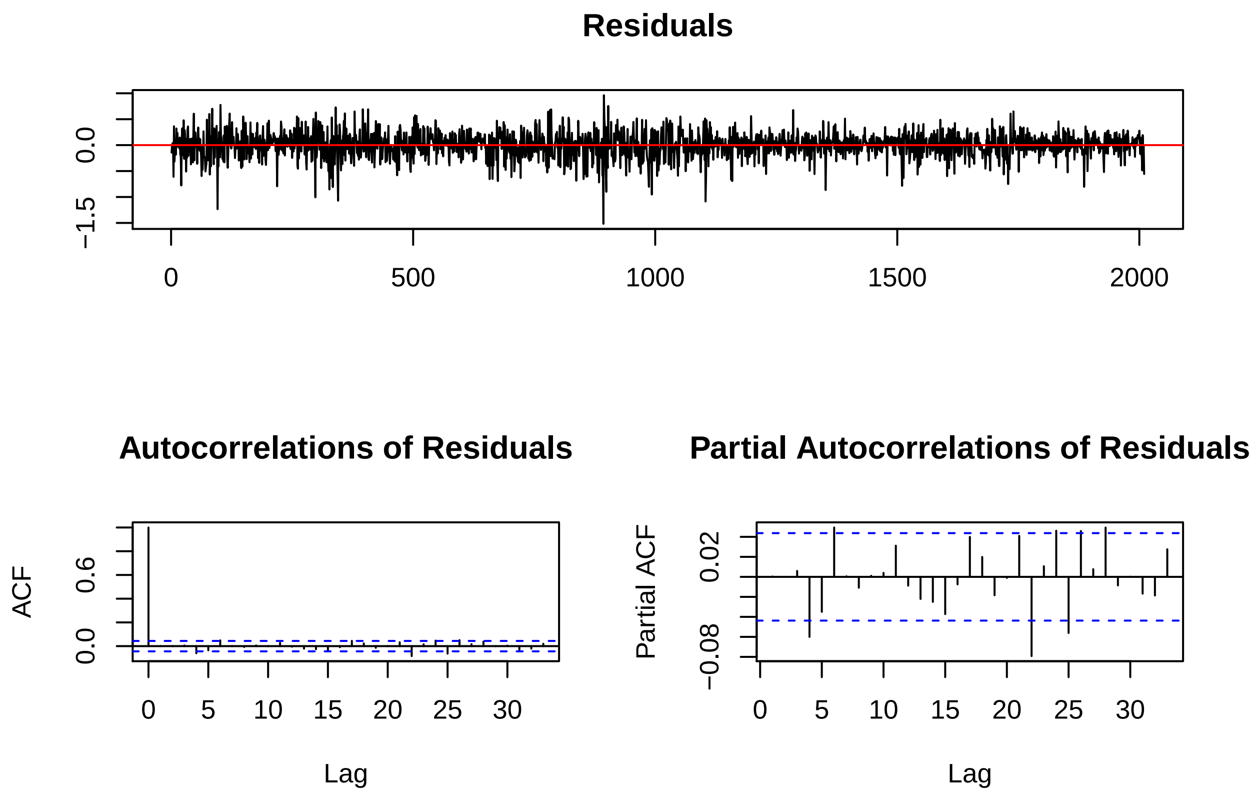
with mean $E[\Delta y_t] = 0$ and variance $Var[\Delta y_t] = \sigma_\epsilon^2$ and $\gamma_h = 0 \quad \forall \quad h > 0$.

This matches our plots for the daily returns of the ETFs, which were a time series that is stationary around mean zero, with a nearly constant variance.

Because applying the difference operator once to the ETF time series leads to a stationary process, we can state that the ETF time series is integrated of order one, or $I(1)$.

We can plot the acf and pacf of the residuals:

```r
plot(adfEWA)
```



The residuals seem to be white noise, which would support that the random walk model is a good approximation for the ETF time series.

We repeat the above analysis with the EWC ETF time series:

```r
adfEWC = ur.df(y = EWC, type = "trend", lags = 8)
summary(adfEWC)
```

```
##
```

```
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression trend
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 + 1 + tt + z.diff.lag)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.22861 -0.13212  0.01753  0.14791  0.78841
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.627e-01  7.019e-02   2.319   0.0205 *
## z.lag.1     -5.991e-03  2.503e-03  -2.394   0.0168 *
## tt           2.967e-06  9.524e-06   0.312   0.7555
## z.diff.lag1  1.694e-02  2.243e-02   0.755   0.4503
## z.diff.lag2  2.715e-02  2.244e-02   1.210   0.2264
## z.diff.lag3  1.342e-02  2.242e-02   0.599   0.5495
## z.diff.lag4 -3.571e-02  2.240e-02  -1.594   0.1110
## z.diff.lag5 -3.931e-02  2.241e-02  -1.755   0.0795 .
## z.diff.lag6  4.216e-02  2.242e-02   1.880   0.0603 .
## z.diff.lag7  4.899e-03  2.244e-02   0.218   0.8272
## z.diff.lag8 -4.873e-03  2.243e-02  -0.217   0.8280
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 0.2461 on 1992 degrees of freedom
## Multiple R-squared:  0.00846,    Adjusted R-squared:  0.003482
## F-statistic:   1.7 on 10 and 1992 DF,  p-value: 0.0753
##
##
## Value of test-statistic is: -2.3938 1.9748 2.9514
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau3 -3.96 -3.41 -3.12
## phi2  6.09  4.68  4.03
## phi3  8.27  6.25  5.34
```

We want to choose a smaller model because none of the lags were highly significant

```
adfEWC = ur.df(y = EWC, type = "trend", lags = 1)
summary(adfEWC)
```

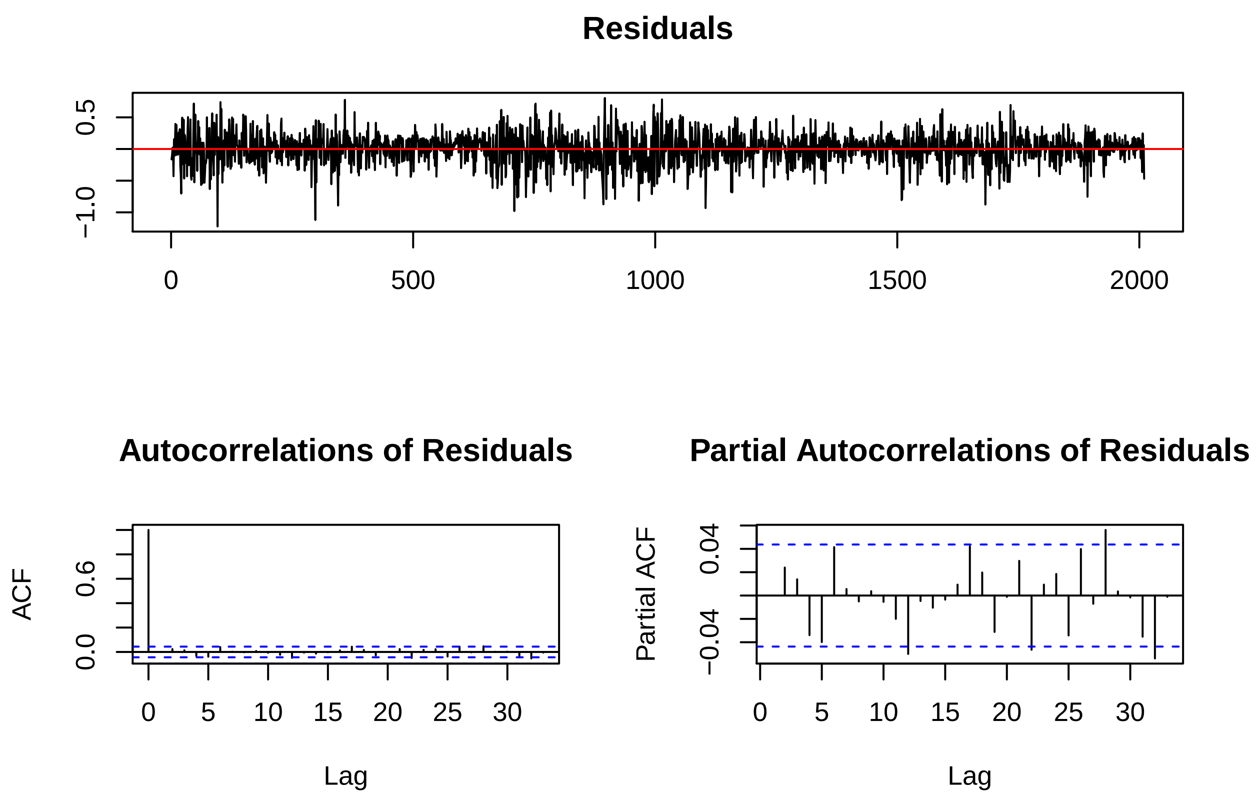
```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression trend
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 + 1 + tt + z.diff.lag)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.2228 -0.1336  0.0144  0.1462  0.8060
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.624e-01  6.930e-02   2.343   0.0192 *
## z.lag.1     -6.029e-03  2.471e-03  -2.440   0.0148 *
## tt           4.032e-06  9.469e-06   0.426   0.6703
## z.diff.lag   1.637e-02  2.234e-02   0.733   0.4638
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2461 on 2006 degrees of freedom
## Multiple R-squared:  0.00327,    Adjusted R-squared:  0.00178
## F-statistic: 2.194 on 3 and 2006 DF,  p-value: 0.08684
##
##
## Value of test-statistic is: -2.4396 2.077 3.1124
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau3 -3.96 -3.41 -3.12
## phi2  6.09  4.68  4.03
## phi3  8.27  6.25  5.34
```

Again, we cannot reject any of the Null hypotheses. Therefore we accept the alternative hypothesis that the model that best fits the EWC ETF time series is that of a random walk without drift.

Note that the result is the same if we test with the specificaton **`type = "drift"`** or **`type = "none"`**. The test aloways suggests a random walk without drift.

We can plot the acf and pacf of the residuals:

```r
plot(adfEWC)
```



Again, the residuals look like white noise, which supports our model selection.

We now turn towards the spread between the daily closing prices of the EWC ETF and the EWA ETF:

```r
adfSPR = ur.df(y = SPREAD, type = "trend", lags = 8)
summary(adfSPR)
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression trend
```

```
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 + 1 + tt + z.diff.lag)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.9388 -0.1067 -0.0042  0.1060  0.9293
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.903e-02  1.689e-02   3.496 0.000483 ***
## z.lag.1     -1.852e-02  4.433e-03  -4.177 3.09e-05 ***
## tt           3.311e-05  1.008e-05   3.286 0.001033 **
## z.diff.lag1 -1.216e-02  2.243e-02  -0.542 0.587797
## z.diff.lag2 -6.038e-03  2.241e-02  -0.269 0.787631
## z.diff.lag3 -3.682e-02  2.240e-02  -1.644 0.100315
## z.diff.lag4 -1.198e-02  2.241e-02  -0.535 0.593002
## z.diff.lag5  2.326e-04  2.240e-02   0.010 0.991715
## z.diff.lag6  5.000e-02  2.237e-02   2.236 0.025491 *
## z.diff.lag7 -1.050e-02  2.239e-02  -0.469 0.639153
## z.diff.lag8  1.302e-02  2.239e-02   0.581 0.561066
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1861 on 1992 degrees of freedom
## Multiple R-squared:  0.01435,    Adjusted R-squared:  0.009399
## F-statistic: 2.899 on 10 and 1992 DF,  p-value: 0.001321
##
##
```

```
## Value of test-statistic is: -4.1767 5.9425 8.8512
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau3 -3.96 -3.41 -3.12
## phi2  6.09  4.68  4.03
## phi3  8.27  6.25  5.34
```

The intercept and the first lag are highly significant, as is the time trend `tt`. We want to test again, using only one lag.

```
adfSPR = ur.df(y = SPREAD, type = "trend", lags = 1)
summary(adfSPR)
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression trend
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 + 1 + tt + z.diff.lag)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.9420 -0.1050 -0.0042  0.1068  0.9622
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.962e-02  1.651e-02   3.611 0.000312 ***
## z.lag.1     -1.862e-02  4.296e-03  -4.335 1.53e-05 ***
## tt           3.310e-05  9.884e-06   3.349 0.000827 ***
## z.diff.lag  -1.343e-02  2.232e-02  -0.602 0.547400
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1861 on 2006 degrees of freedom
## Multiple R-squared:  0.009897,   Adjusted R-squared:  0.008416
## F-statistic: 6.684 on 3 and 2006 DF,  p-value: 0.0001733
##
##
## Value of test-statistic is: -4.3345 6.3793 9.5163
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau3 -3.96 -3.41 -3.12
## phi2  6.09  4.68  4.03
## phi3  8.27  6.25  5.34
```

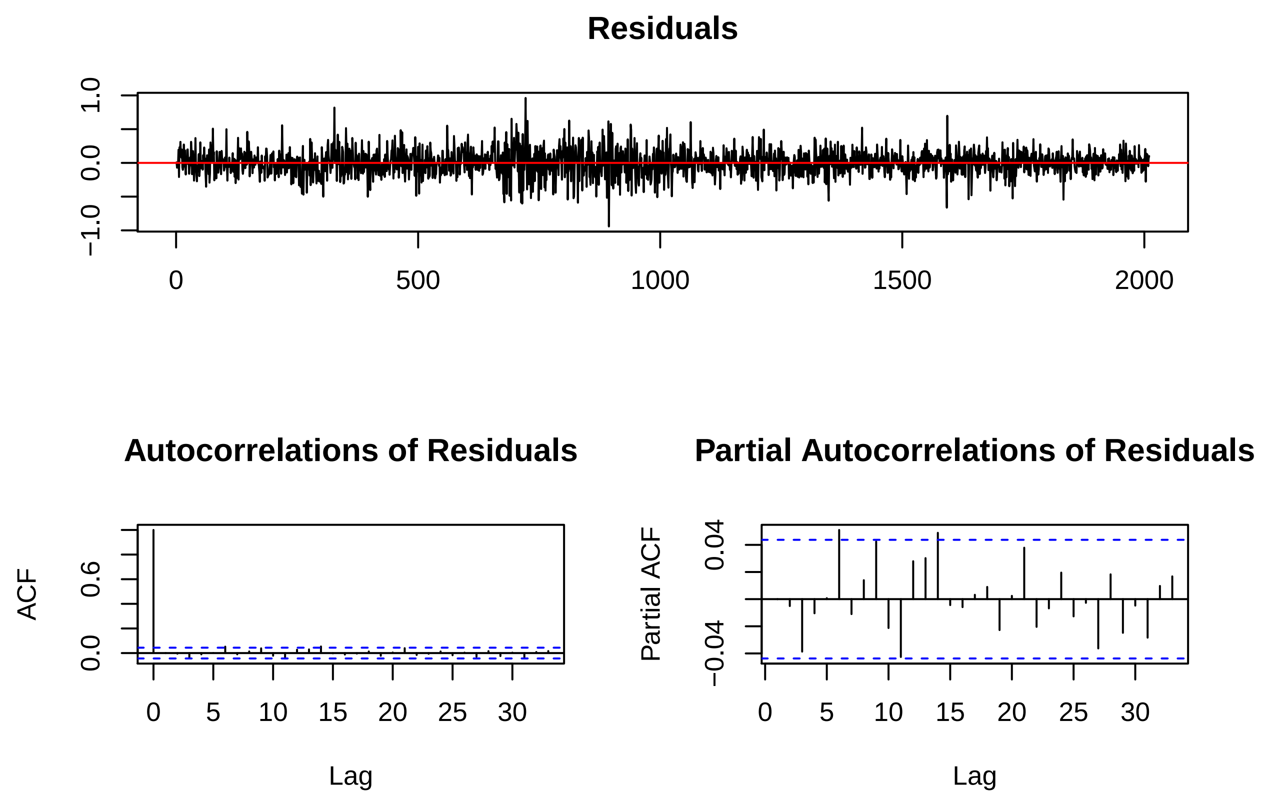
Again, the intercept, the first lag and the time trend **`tt`** are highly significant.

Furthermore, we have to reject all three Null hypothesis on a 95% confidence level. Therefore we have to assume that we deal with the full model.

$$y_t = \alpha + \rho y_{t-1} + \beta t + \epsilon_t \Delta y_t = \alpha + (\rho - 1) y_{t-1} + \beta t + \epsilon_t$$

That is, the spread between the two ETF time series has a drift term $\alpha \neq 0$ and a deterministic time trend $\beta \neq 0$. Furthermore it has no unit root, so $\rho \neq 1$, and instead we can interpret $\rho - 1$ as the speed of adjustment of the stochastic trend component in the time series.

```r
plot(adfSPR)
```



Again, the residuals seem to be white noise, because the ACF plot shows zero autocorrelations and the PACF plot shows approximately zero partial autocorrelations.

We can also use the Bayesian Information Criterion to select the optimal number of lags:

```r
adfBIC = ur.df(y = SPREAD,type = "trend", selectlags = "BIC")
summary(adfBIC)
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression trend
```

```
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 + 1 + tt + z.diff.lag)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.9420 -0.1050 -0.0042  0.1068  0.9622
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.962e-02  1.651e-02   3.611 0.000312 ***
## z.lag.1     -1.862e-02  4.296e-03  -4.335 1.53e-05 ***
## tt           3.310e-05  9.884e-06   3.349 0.000827 ***
## z.diff.lag  -1.343e-02  2.232e-02  -0.602 0.547400
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1861 on 2006 degrees of freedom
## Multiple R-squared:  0.009897,   Adjusted R-squared:  0.008416
## F-statistic: 6.684 on 3 and 2006 DF,  p-value: 0.0001733
##
##
## Value of test-statistic is: -4.3345 6.3793 9.5163
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau3 -3.96 -3.41 -3.12
## phi2  6.09  4.68  4.03
## phi3  8.27  6.25  5.34
```

This yields the same result as before. We can also employ the Akaike information Criterion to select the optimal number of lags:

```r
adfAIC = ur.df(y = SPREAD,type = "trend", selectlags = "AIC")
summary(adfAIC)
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression trend
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 + 1 + tt + z.diff.lag)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.9420 -0.1050 -0.0042  0.1068  0.9622
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.962e-02  1.651e-02   3.611 0.000312 ***
## z.lag.1     -1.862e-02  4.296e-03  -4.335 1.53e-05 ***
## tt           3.310e-05  9.884e-06   3.349 0.000827 ***
## z.diff.lag  -1.343e-02  2.232e-02  -0.602 0.547400
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 0.1861 on 2006 degrees of freedom
## Multiple R-squared:  0.009897,   Adjusted R-squared:  0.008416
## F-statistic: 6.684 on 3 and 2006 DF,  p-value: 0.0001733
##
##
## Value of test-statistic is: -4.3345 6.3793 9.5163
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau3 -3.96 -3.41 -3.12
## phi2  6.09  4.68  4.03
## phi3  8.27  6.25  5.34
```

We get the same results with the AIC test, the BIC test and the ADF test.

We can also calculate the test statistics by hand:

```
lags     = 1
DY       = diff(SPREAD) # Dependent variable in regression = Differences of log RGDP
n        = length(DY)  # = num - 1
tt       = (lags+1):n  # time variable for estimation of deterministic
DYa      = embed(DY, lags+1)[,1] # all ts need to have same length, start with second v
Ylag     = SPREAD[(lags+1):n] # all ts need to have same length
k        = lags+1
DYlag    = embed(DY, lags+1)[, 2:k]
s1       = summary(lm(formula = DYa ~ Ylag + 1 + tt + DYlag))
s1 # same result!
```

```
##
## Call:
## lm(formula = DYa ~ Ylag + 1 + tt + DYlag)
##
```

```
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.9420 -0.1050 -0.0042  0.1068  0.9622
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.962e-02  1.651e-02   3.611 0.000312 ***
## Ylag        -1.862e-02  4.296e-03  -4.335 1.53e-05 ***
## tt           3.310e-05  9.884e-06   3.349 0.000827 ***
## DYlag       -1.343e-02  2.232e-02  -0.602 0.547400
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1861 on 2006 degrees of freedom
## Multiple R-squared:  0.009897,   Adjusted R-squared:  0.008416
## F-statistic: 6.684 on 3 and 2006 DF,  p-value: 0.0001733
```

Again, we get the same result.

Comment on the results:

The result for the EWC and EWA ETF time series were conclusive. The model that best fits the data is a unit root process without drift, that is, a random walk.

The result for the Spread between the EWC and EWA ETF time series showed that a model with a deterministic time trend, a stochastic component and drift fits the data optimally.

Exercise 1 (d) Johansen Test

What kind of economic relationship do the EWA and EWC series adhere to?

Both time series are integrated of order one, or $I(1)$. Their first differences are stationary processes.

Because they move together, and the direction of their movement cannot be predicted, we may conclude that they for an economic equilibrium relationship, and that any deviations from this equilibrium can only be of transitory nature. In that case, both time series would be cointegrated of order 1 or $CI(1,1)$, and there would exist a linear combination which is integrated of order zero or $I(0)$, that is, it would be a stationary process.

Furthermore, we know that a regression of both these random walks would result in spurious regressions.

Explain in a few words the Johansen test

The Johansen test is a procedure to thest the cointegration of several time series that are integreated of odrder 1, that is, that are I(1).

This test permits more than one cointegrating relationship so is more generally applicable than the Engle–Granger test which is based on the Dickey–Fuller (or the augmented) test for unit roots in the residuals from a single (estimated) cointegrating relationship.

Just like a unit root test, there can be a constant term, a trend term, both, or neither in the model. For a general vector-valued autoregressive process of order p, or VAR(p) model given by

$$y_t = \mu + A_1 y_{t-1} + \cdots + A_p y_{t-p} + \epsilon_t$$

where y_t is an $nx1$ vecto of variables that are integrated of order one, or $I(1)$ and ϵ_t is an $nx1$ vector of innovations.

This VAR(p) can be re-written as

$$\Delta y_t = \mu + \Pi y_{t-1} + \sum_{i=1}^{p-1} \Gamma_i \Delta y_{t-i} + \epsilon_t$$

where

$$\Pi = \sum_{i=1}^{p} A_i, \quad \text{and} \quad \Gamma_i = -\sum_{j=i+1}^{p} A_j.$$

If thte coefficient matrix Π has reduced rank $r < n$, then there exist nxr matrices α and β each with rank r such that

$$\Pi = \alpha\beta'$$

and

$$\beta' y_t$$

is stationary.

r is the number of cointegrating relationships, the elements of α are known as the adjustment parameters in the vector error correction model or VECM and each column of β is a cointegrating vector.

It can be shown that for a given r, the maximum likelihood estimator fo β dfines the combination fo y_{t-1} that yields the r largest canonical correlations of Δy_t with y_{t-1} after correcting for lagged differences and deterministic variables when present.

There are two different likelihood ratio tests of tthe significance of these canonical correlations and thereby the reduced rank of the Π matrix:

The trace test

$$J_{\text{trace}} = -T \sum_{i=r+1}^{n} \log\left(1 - \hat{\lambda}_i\right)$$

and the maximum eigenvalue test

$$J_{\max} = -T \log\left(1 - \hat{\lambda}_{r+1}\right)$$

Here, T is the sample size and $\hat{\lambda}_i$ is the ith largest canonical correlation.

The trace test tests the null hypothesis of r cointegrating vectors against the alternative of n cointegrating vectors.

Teh maximum eigenvalue test tests the null hypothesis of r cointegrating vectors against the alternative hypothesis of $r + 1$ cointegrating vectors.

Asymptotic critical values do not follow a Chi square distribution but are given by econometric software packages.

Since the ccritical values used for the maximum eigenvalue and trace tes astatistics are based on a pure unit-root assumption, they will no longer be correct when the variables in the system are near-unit-root-processes.

Source: IMF

Conduct a Johansen Test

(Command **`ca.jo`** in the **`urca`** library)

Note the model specification of this test:

Given a general $VAR(p)$ model of the form:

$$X_t = \Pi_1 X_{t-1} + \cdots + \Pi_k X_{t-k} + \mu + \Phi D_t + \epsilon_t, \quad (t = 1, \ldots, T),$$

the following two specificaitons of a VECM exist:

$$\Delta X_t = \Gamma_1 \Delta X_{t-1} + \cdots + \Gamma_{k-1} \Delta X_{t-k+1} + \Pi X_{t-k} + \mu + \Phi D_t + \epsilon_t$$

where

$$\Gamma_i = -(I - \Pi_1 - \cdots - \Pi_i), \quad (i = 1, \ldots, k-1),$$

and

$$\Pi = -(I - \Pi_1 - \cdots - \Pi_k)$$

The Γ_i matrices contain the cumulative long-run impacts, hence if **`spec="longrun"`** is chosen, the above VECM is estimated.

The other VECM specification is of the form:

$$\Delta X_t = \Gamma_1 \Delta_{t-1} + \cdots + \Gamma_{k-1} \Delta X_{t-k+1} + \Pi X_{t-1} + \mu + \Phi D_t + \epsilon_t$$

where

$$\Gamma_i = -(\Pi_{i+1} + \cdots + \Pi_k), \quad (i = 1, \ldots, k-1),$$

and

$$\Pi = -(I - \Pi_1 - \cdots - \Pi_k).$$

The Π matrix is the same as in the first specification. The Π_i matrices differ, however, in the sense that they measure transitory effects. Hence, by setting **`spec="transitory"`**, the second VECM form is estimated.

Note that inferences drawn on Π will be the same, regardless which specification is choosen and that the explanatory power is the same, too.

Critical values are only reported for systems with less than 11 variables an are taken from Osterwald-Lenum.

Source CRAN

We now conduct a Johansen test for the EWC and EWA ETF time series:

```r
data = cbind(EWA,EWC)
ca.jo1 = ca.jo(x = data,
      type = c("eigen"),  # "eigen" or "trace"
      ecdet = c("trend"), # "none", "const", "trend"
      K = 2,              # Lag order of series (levels) in VAR, min. K=2
      spec=c("longrun"),  # Error Correction Model (VECM)
      season = NULL,      # Seasonal dummies, ('4' for quarterly data)
      dumvar = NULL)      # Dummy variables, matrix with row dim. = x
ca.jo1
```

```
##
## ######################################################
## # Johansen-Procedure Unit Root / Cointegration Test #
## ######################################################
##
```

```
## The value of the test statistic is: 6.3428 21.9627
```

The values of the test statistic are 6.3428 and 21.9627

```r
summary(ca.jo1)
```

```
##
## ######################
## # Johansen-Procedure #
## ######################
##
## Test type: maximal eigenvalue statistic (lambda max) , with linear trend in cointegra
##
## Eigenvalues (lambda):
## [1] 1.086723e-02 3.150628e-03 3.469447e-18
##
## Values of teststatistic and critical values of test:
##
##           test 10pct  5pct  1pct
## r <= 1 |  6.34 10.49 12.25 16.26
## r = 0  | 21.96 16.85 18.96 23.65
##
## Eigenvectors, normalised to first column:
## (These are the cointegration relations)
##
##               EWA.Close.l2 EWC.Close.l2    trend.l2
## EWA.Close.l2  1.000000000 1.0000000000  1.00000000
## EWC.Close.l2 -0.869444123 1.9844629709 -1.36307596
## trend.l2      0.001781346 0.0007476123 -0.01397097
##
## Weights W:
```

```
## (This is the loading matrix)
##
##              EWA.Close.l2 EWC.Close.l2      trend.l2
## EWA.Close.d -0.006369467 -0.002155280 -5.375792e-19
## EWC.Close.d  0.015081739 -0.001858612 -3.203308e-19
```

- We cannot reject the $H_0 : r \leq 1$ at any significance level.
- However, we can reject $H_0 : r = 0$ at a 5% significance level.
- Therefore we conclude that $r = 1$.
- This means that the number of cointegration vectors is one. The number of cointegrating vectors is also called the cointegration rank of the process.
- Because the cointegration rank $r = 1$, and the number of nonstationary components in the $k-\textit{dimensional}$ process is $k = 2$, the number of common stochastic trends (unit-roots) that rmeain is $k - r = 2 - 1 = 1$.

Comment on your results:

We have found that the EWA and EWC ETF time series follow an economic equilibrium relationship. They are both integraded of order 1, i.e., $I(1)$, but their linear rcombinations are $I(0)$.

2. Seasonalities and Forecasting

Exercise 2 (a) Time Series Object

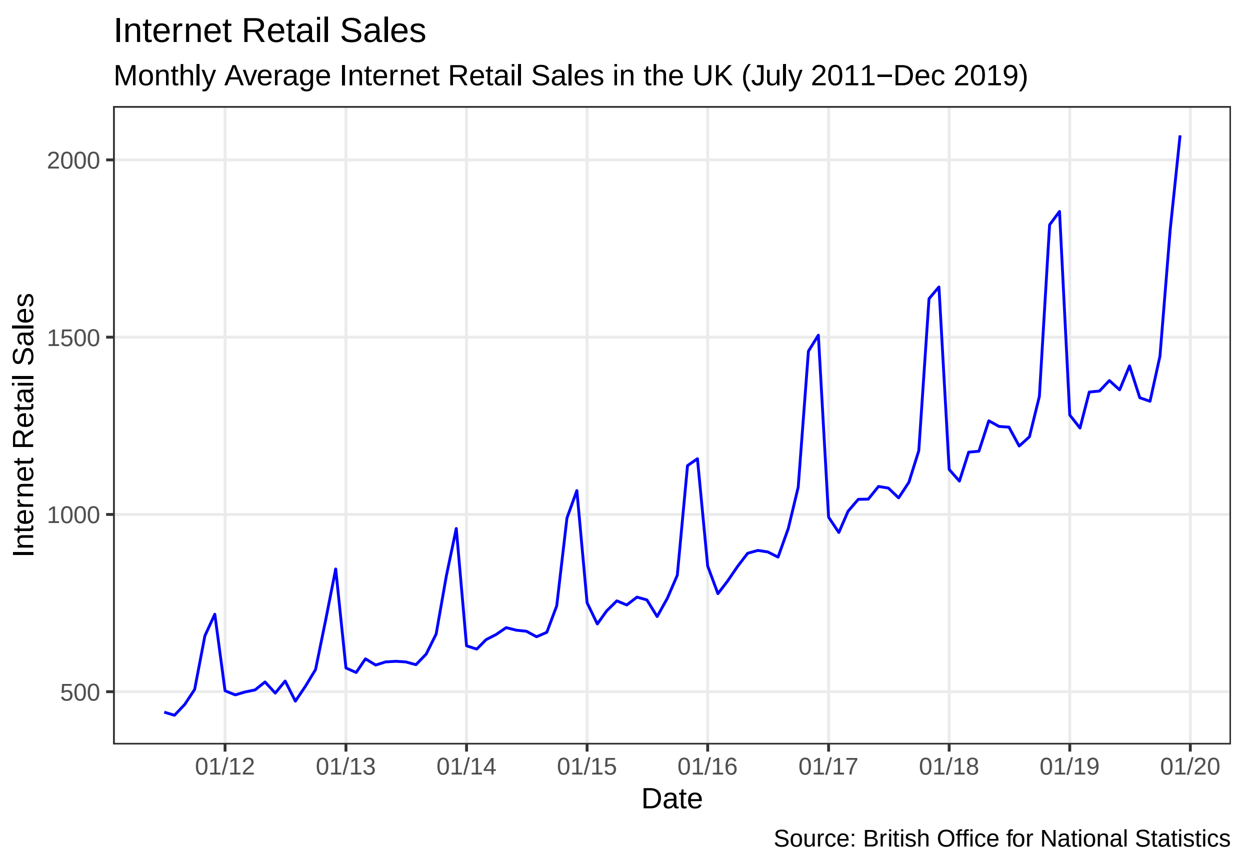
The file **`InternetRetailSales.xlsx`** contains information on monthly average internet retail sales in the UK from July 2011 to December 2019 by the british Office for National statistics.

Load the file into R and turn the data into a time series object using the **`tseries`** library.

```
# Load data file (adjust file path if necessary)
data      = read_xls(path = "InternetRetailSales.xls",
                     sheet = 1, col_names = TRUE)
num       = dim(data)[1] # length of time series
vnum      = length(data) # number of variables

data$Month = match(data$Month, month.abb)

data$Date = as.yearmon(paste(data$Year, data$Month), format = "%Y %m")

z = zoo(x = as.matrix(data[, 3]), order.by = as.Date(data$Date))
```

Plot the data:

```
a5 = autoplot(z, facet = NULL, color="blue")+
  scale_x_date(date_breaks = "12 months",
               date_labels = "%m/%y")+
  labs(title = "Internet Retail Sales",
       subtitle = "Monthly Average Internet Retail Sales in the UK (July 2011-Dec 2019
       caption = "Source: British Office for National Statistics",
       x = "Date",
       y = "Internet Retail Sales") +
  scale_color_manual(labels = c("Sales"),
                     values = c("blue")) +
  theme_bw()+
  theme(panel.grid.minor = element_blank(),
        legend.position = "none")+
  guides(color=guide_legend(NULL))

a5
```



```r
# ggsave("PlotData.png", plot = a5, width = 17, height = 12, units = "cm")
# dev.off()
```

Comment:

Clearly, the monthly average internet retail sales in the UK have been increasing over time. There is a clearly discernible pattern of seasonality, with sales peaking in the weeks before the holidays and rapidly decreasing right after.

Exercise 2 (b) Seasonal Plot

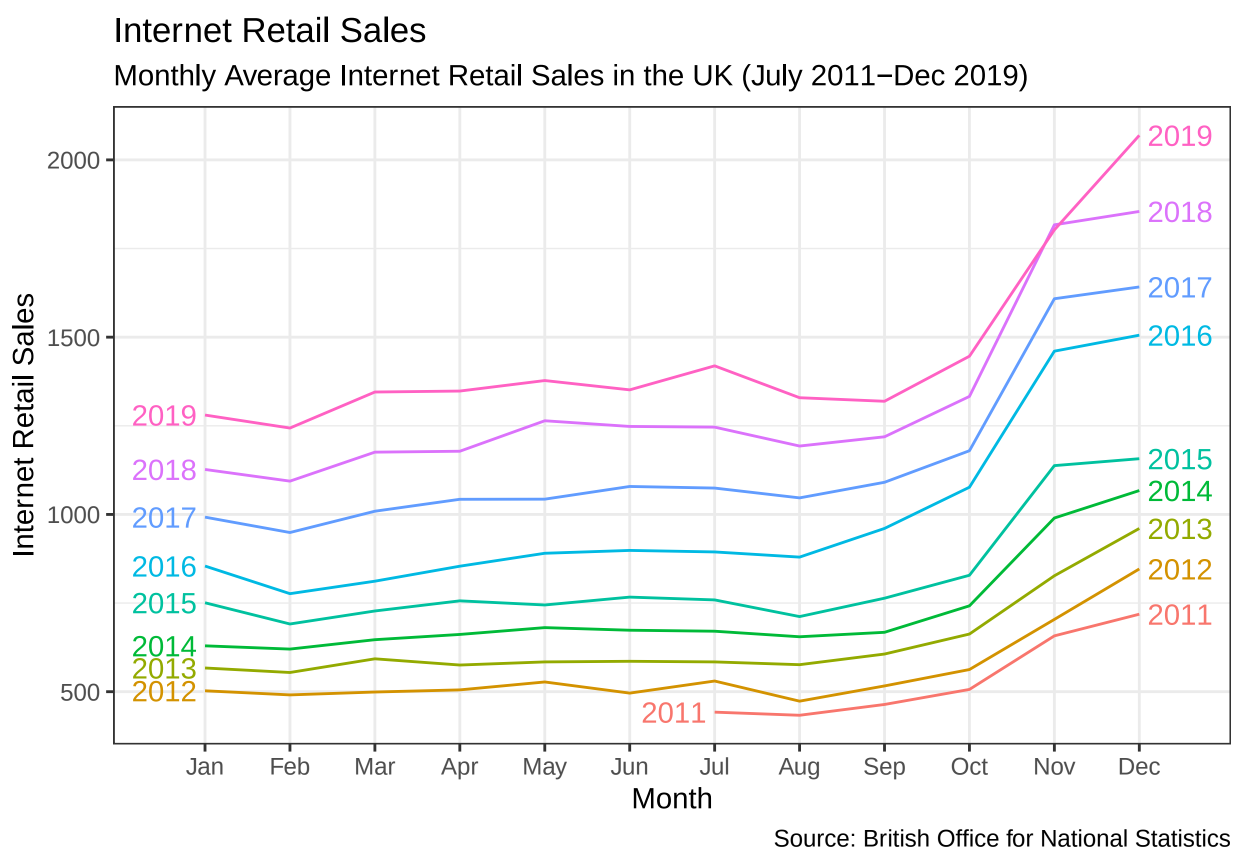
Create a seasonal plot of the data (you can e.g. use the function **`ggseasonplot`**, choose the frequency appropriately) and plot the ACF of the time series for a number of $h = 36$ lags.

For the seasonal plot, we have to make the data seasonal:

```r
tseries <- ts(data$InternetRetail,
              start = c(2011,  7),
              end   = c(2019, 12),
              frequency = 12)
```

Then, we make the seasonal plot:

```r
a6 = ggseasonplot(x = tseries,
               season.labels = TRUE,
               year.labels=TRUE,
               year.labels.left=TRUE) +
    # Center the plot title:
    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)",
    caption = "Source: British Office for National Statistics",
    x = "Month",
    y = "Internet Retail Sales") +
    theme_bw()+
    theme(legend.position = "none")+
    guides(color=guide_legend(NULL))
a6
```



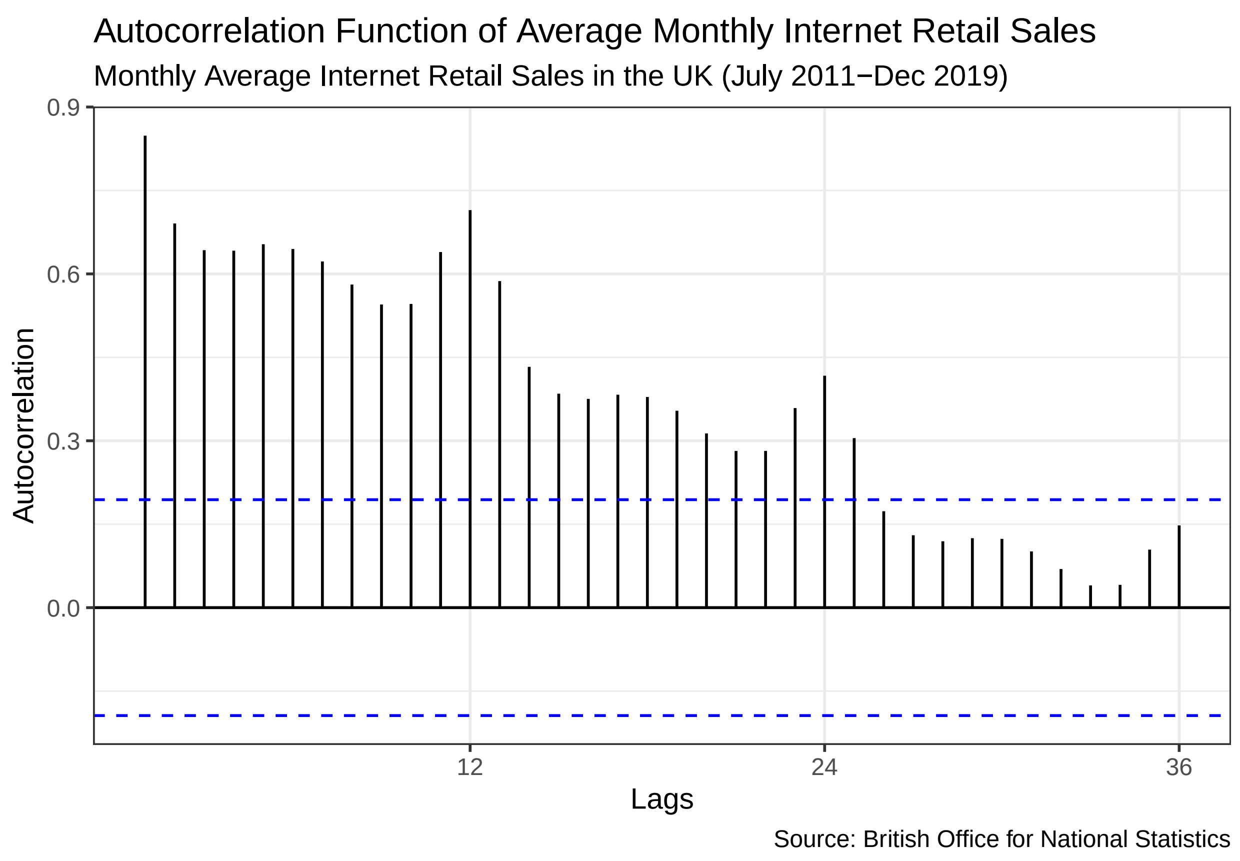
```r
# ggsave("Season.png", plot = a6, width = 17, height = 12, units = "cm")
# dev.off()
```

We clearly see in the seasonal plot that the monthly average internet sales in the UK started increasing every year in October, that is, abouth two months before the holidays.

There may e a decrease after "Black Friday", which is the November 27.

Next, we plot the ACF of the time series for a number of $h = 36$ lags:

```r
a7 = ggAcf(x = tseries, lag.max = 36)+
  # Center the plot title
    theme(plot.title = element_text(hjust = 0.5)) +
    labs(title = "Autocorrelation Function of Average Monthly Internet Retail Sales",
    subtitle = "Monthly Average Internet Retail Sales in the UK (July 2011-Dec 2019)",
    caption = "Source: British Office for National Statistics",
    x = "Lags",
    y = "Autocorrelation") +
    theme_bw()+
    theme(legend.position = "none")
a7
```



```
# ggsave("ACFts.png", plot = a7, width = 17, height = 12, units = "cm")
# dev.off()
```

- We have used a number of $h = 36$ lags. Because we use monthly data, the autocorrelation function in essence measures the correlation of the average internet sales in a given month with those in another month.
- 36 lags starting from December 2019 would mean that the autocorrelation function plots this reletionship for the three years before 2019.

- We can see that the autocorrelation increases after 12 months, because every holiday season, the monthly average internet sales move in the same direction.

Exercise 2 (c) Moving Average Trend

Calculate the trend using a moving average with a window of twelve observations.

The moving average smoother averages the nearest **`order`** periods of each observation. As neighbouring observations of a time series are likely to be similar in value, averaging eliminates some ofthe randomness in the data, leaving a smooth trend-cycle component.

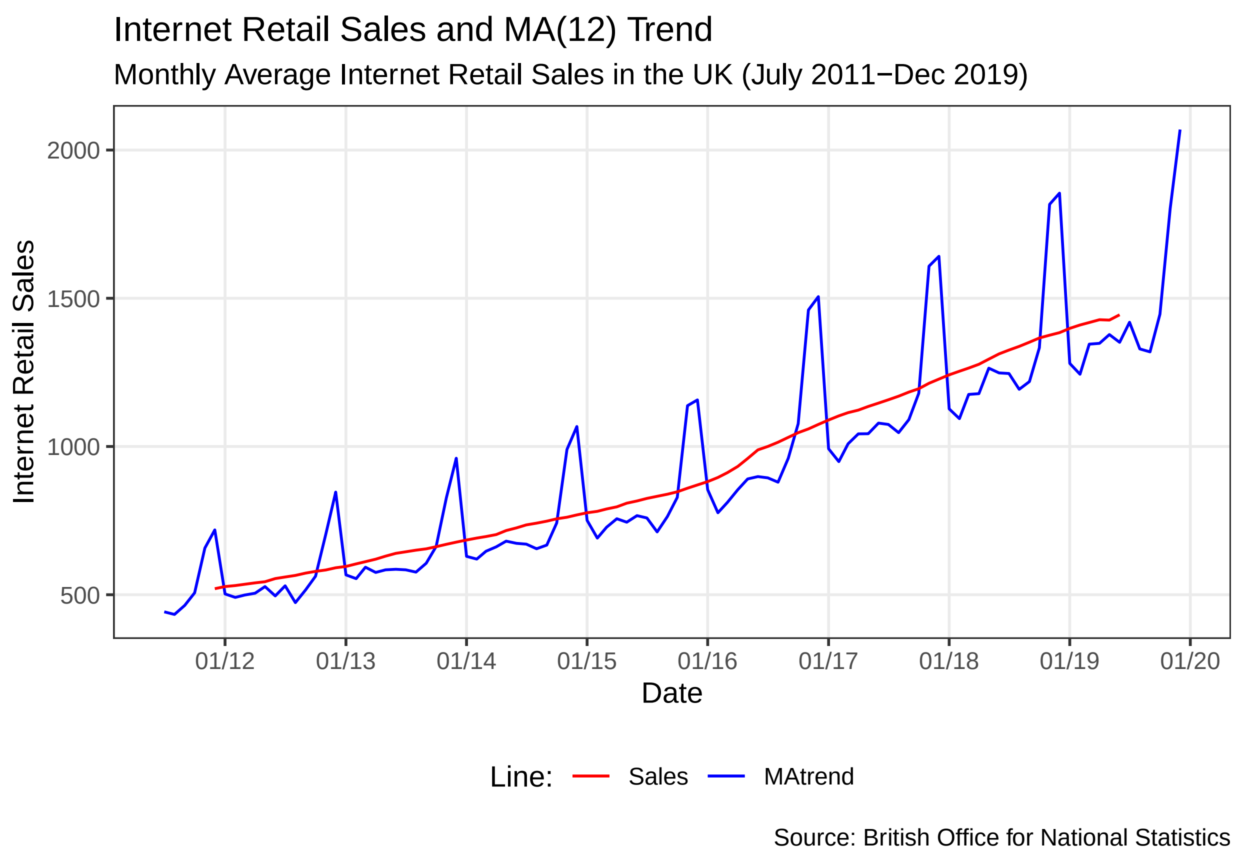
$$\hat{T}_t = \frac{1}{m} \sum_{j=-k}^{k} y_{t+j}$$

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

When an **`order`** is specified, the observations averaged will include one more observation from the future than the past (k is rounded up). If centre is **`TRUE`**, the value from two moving averages(where k is rounded up and down respectively) are averaged, centering the moving average.

Source: forecast

```r
MAtrend = ma(x = data$InternetRetail, order = 12, centre = FALSE)


a8 = autoplot(z, facet = NULL, color="red")+
  scale_x_date(date_breaks = "12 months",
               date_labels = "%m/%y")+
  labs(title = "Internet Retail Sales and MA(12) Trend",
  subtitle = "Monthly Average Internet Retail Sales in the UK (July 2011-Dec 2019)",
  caption = "Source: British Office for National Statistics",
  x = "Date",
  y = "Internet Retail Sales") +
    geom_line(mapping = aes(y = MAtrend, color="blue"))+
    scale_color_manual(name = "Line:",
                       labels = c("Sales", "MAtrend"),
                       values = c("red", "blue")) +
    theme_bw()+
    theme(panel.grid.minor = element_blank(), legend.position = "bottom")

a8
```



```
# ggsave("Trend.png", plot = a8, width = 17, height = 12, units = "cm")
# dev.off()
```

- If we calculate a Moving Average Trend of order 12, we average over the whole year.
- Clearly, average internet retail sales have been increasing over the past years.

Exercise 2 (d) TS Decomposition

Explain the difference between and additive and a multiplicative time series decomposition into seasonal, trend and residual component.

Plots of time series often suggest a decomposition of the series into

$$x_t = m_T + s_t + u_t$$

where m_t is a (deterministic) trend component, s_t is a (deterministic) seasonality component, u_t is an irregular component with $E\left[u_T\right] = 0$ and $Var[u_t] = \sigma_u^2 < \infty$ (White Noise, covariance stationary).

Sometimes a multiplicative model

$$X_t = M_t S_t U_t$$

seems more appropriate. Aplying the logarithm to all the elements redces the multiplicative model to the additive model. All elements need to be positive.

Justify which decomosition to choose for the data set.

- Because the monthly average internet sales increase in a nonlinear way, the multiplicative model seems more appropriate for this time series.
- A more prominent reason to choose the multiplicative model is that the magnitude of the seasonality is increasing over time (the peak, or the amplitude is increasing over time).

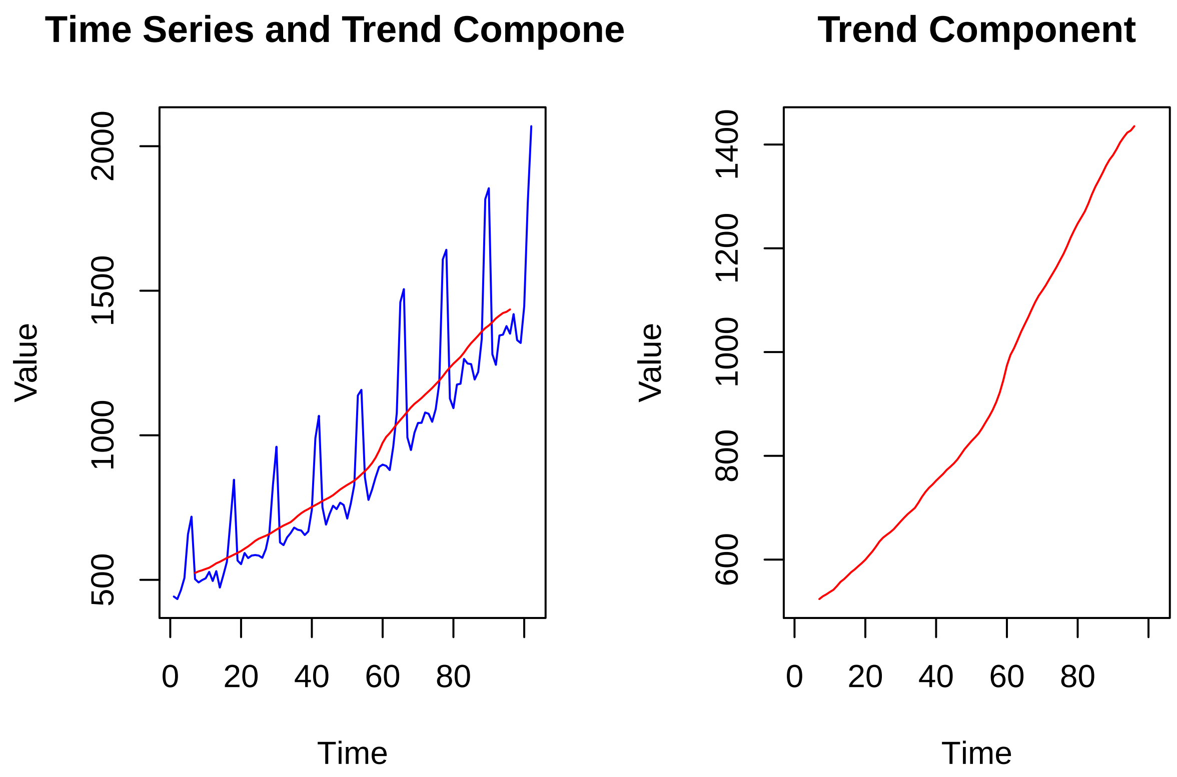
Exercise 2 (e) Detrending

Detrend the time series according to your decomposition choice (either multiplicative or additive).

We want to detrend the time series according to the multiplicative decomposition:

We have already detected a trend using the Moving Average filter of order 12 before.

```
par(mfrow=c(1,2))
trend_sales = ma(data$InternetRetail, order = 12, centre = TRUE)

plot(as.ts(data$InternetRetail),
     col="blue",
     main="Time Series and Trend Component",
     xlab="Time", ylab="Value")
lines(trend_sales, col="red")
plot(as.ts(trend_sales), col="red",
     main="Trend Component",
     xlab="Time", ylab="Value")
```



Removing the previously calculated trend from the time series will result into a new time series that clearly exposes seasonality.

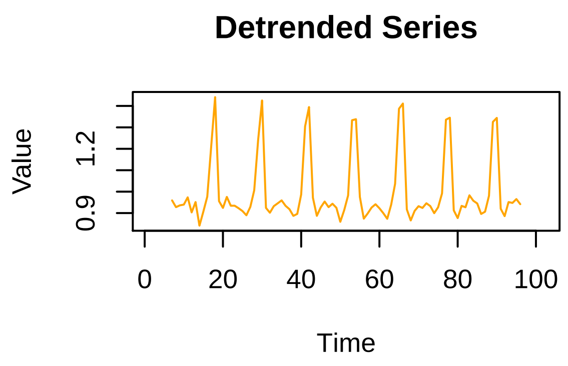
From the detrended time series, we can compute the average seasonality. We add the seasonality together and divide by the seasonality period. Technically speaking, to average together the time series we feed the time series into a matrix. Then, we transform the matrix so each column contains elements of the same period (same day, same month, same quarter, etc...). Finally, we compute the mean of each column. Here is how to do it in R:

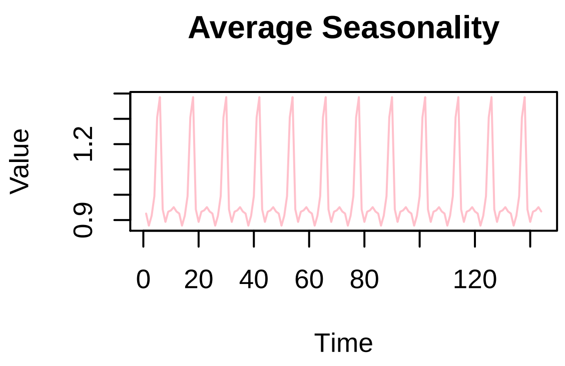
The previous steps have already extracted most of the data from the original time series, leaving behind only "random" noise.

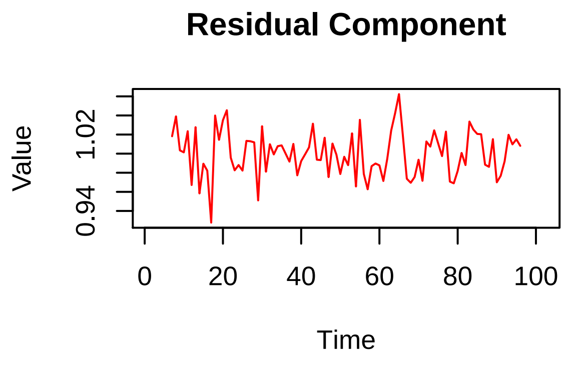
The decomposed time series can logically be recomposed using the model formula to reproduce the original signal. Some data points will be missing at the beginning and the end of the reconstructed time series, due to the moving average windows which must consume some data before producing average data points.

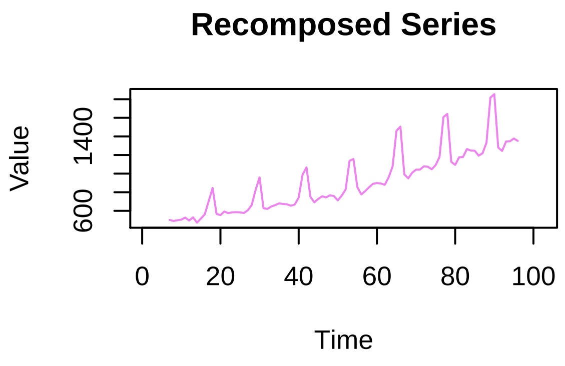
```r
par(mfrow=c(2,2))
# Create the Detrended Time Series:
detrend_sales = data$InternetRetail / trend_sales
plot(as.ts(detrend_sales),
     col="orange",
     main="Detrended Series",
     xlab="Time", ylab="Value")

# Create the Average Seasonality:
m_sales = t(matrix(data = detrend_sales, nrow = 12))
seasonal_sales = colMeans(m_sales, na.rm = TRUE)
plot(as.ts(rep(seasonal_sales, 12)),
     col="pink",
     main="Average Seasonality",
     xlab="Time", ylab="Value")
```

```r
# Create the Residual Component:
random_sales = data$InternetRetail / (trend_sales * seasonal_sales)
plot(as.ts(random_sales),
     col="red",
     main="Residual Component",
     xlab="Time", ylab="Value")

# Create the Recomposed Series:
recomposed_sales = trend_sales*seasonal_sales*random_sales
plot(as.ts(recomposed_sales),
     col="violet",
     main="Recomposed Series",
     xlab="Time", ylab="Value")
```







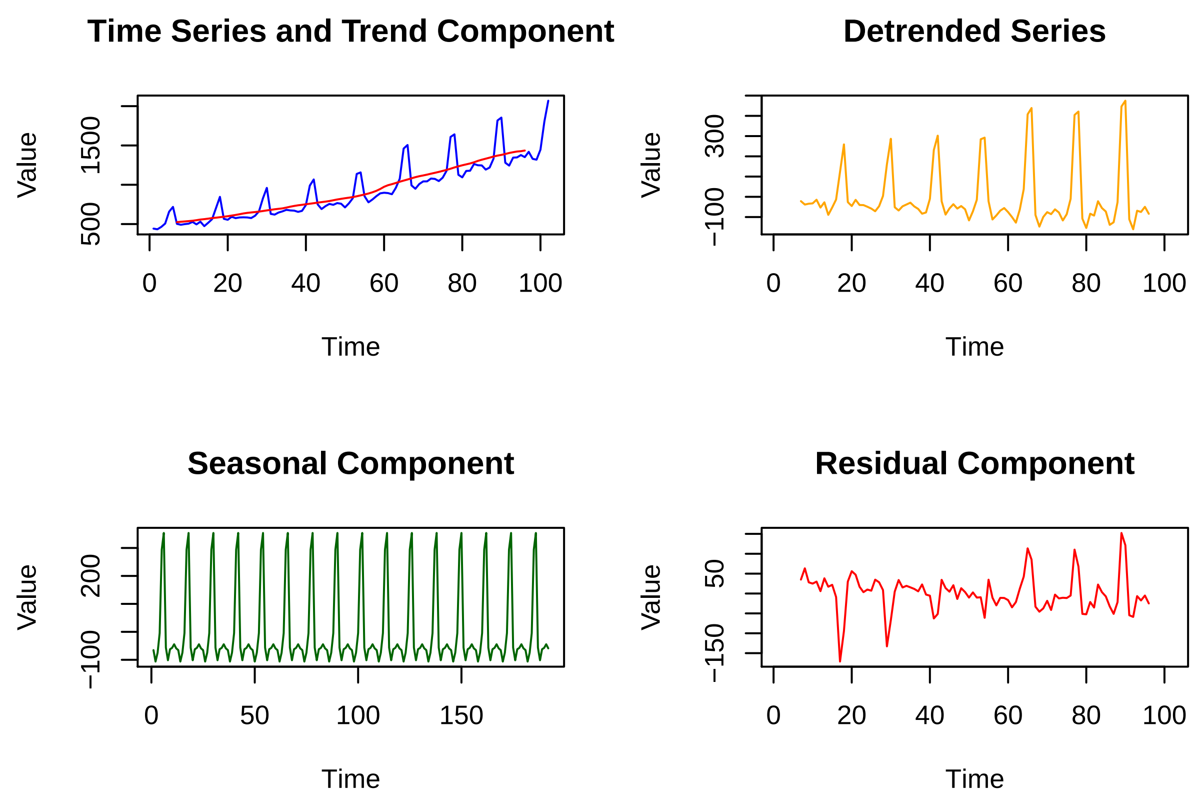


Inspiration: Anomaly

Would you deem the resulting series to be (weakly) stationary?

- Clearly, the error term is a white noise process with mean zero and finite variance, so it is covariance or weakly stationary (white noise is even strictily stationary).
- The seasonality term is most likely also a stationary process, as it represent a seasonal fluctuation around the moving average trend term.
- We can also perform an additive detranding of the time series to compare the results:

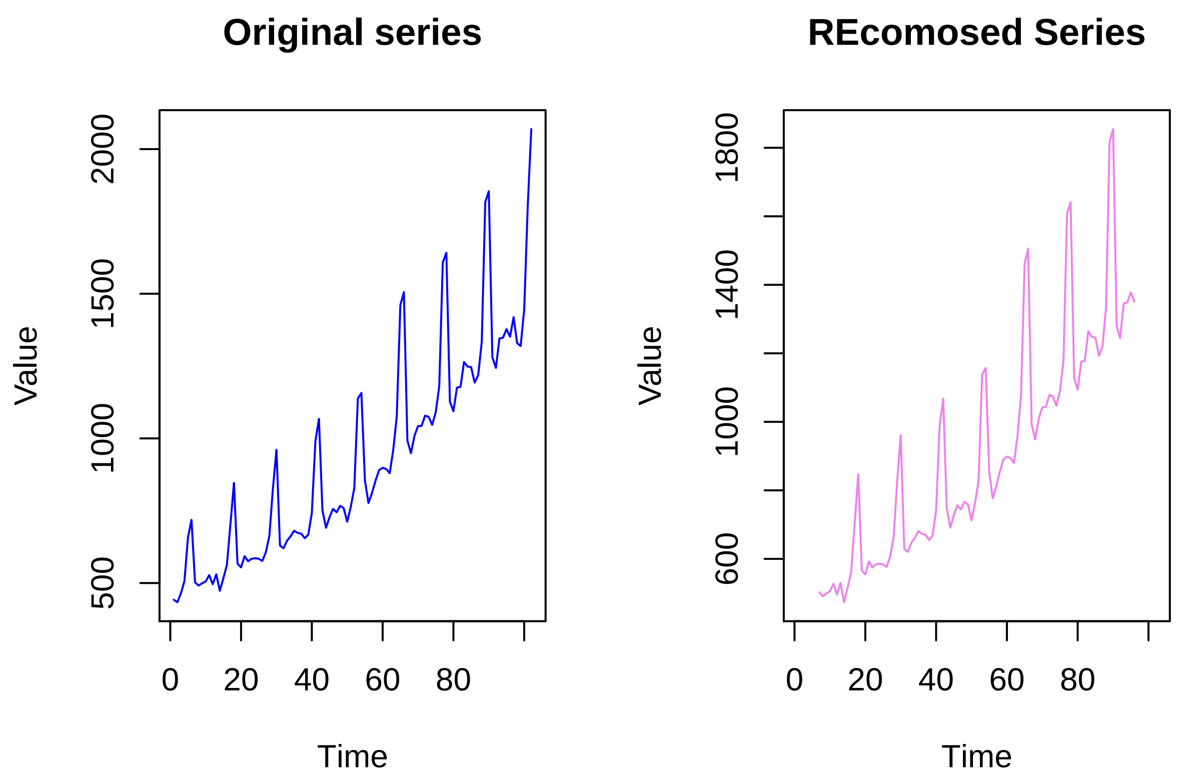
```r
par(mfrow=c(2,2))
# Create the Trend Component:
trend_sales = ma(data$InternetRetail, order = 12, centre = TRUE)
plot(as.ts(data$InternetRetail),
     col="blue",
     main="Time Series and Trend Component",
     xlab="Time", ylab="Value")
lines(trend_sales, col="red")

# Create the Detrended Series:
detrend_sales = data$InternetRetail - trend_sales
plot(as.ts(detrend_sales),
     col="orange",
     main="Detrended Series",
     xlab="Time", ylab="Value")

# Create the Seasonal Comonent:
m_sales = t(matrix(data = detrend_sales, nrow = 12))
seasonal_sales = colMeans(m_sales, na.rm = TRUE)
plot(as.ts(rep(seasonal_sales,16)),
     col="darkgreen",
     main="Seasonal Component",
     xlab="Time", ylab="Value")


# Create the Residual Component:
random_sales = data$InternetRetail - trend_sales - seasonal_sales
plot(as.ts(random_sales),
     col="red", main="Residual Component",
     xlab="Time", ylab="Value")
```



Let’s now compare the additive model to the original time series:

```
par(mfrow=c(1,2))
recomposed_sales = trend_sales+seasonal_sales+random_sales
plot(as.ts(data$InternetRetail),
     col="blue", main="Original series", xlab="Time", ylab="Value")
plot(as.ts(recomposed_sales),
```

```r
    col="violet", main="REcomosed Series", xlab="Time", ylab="Value")
```

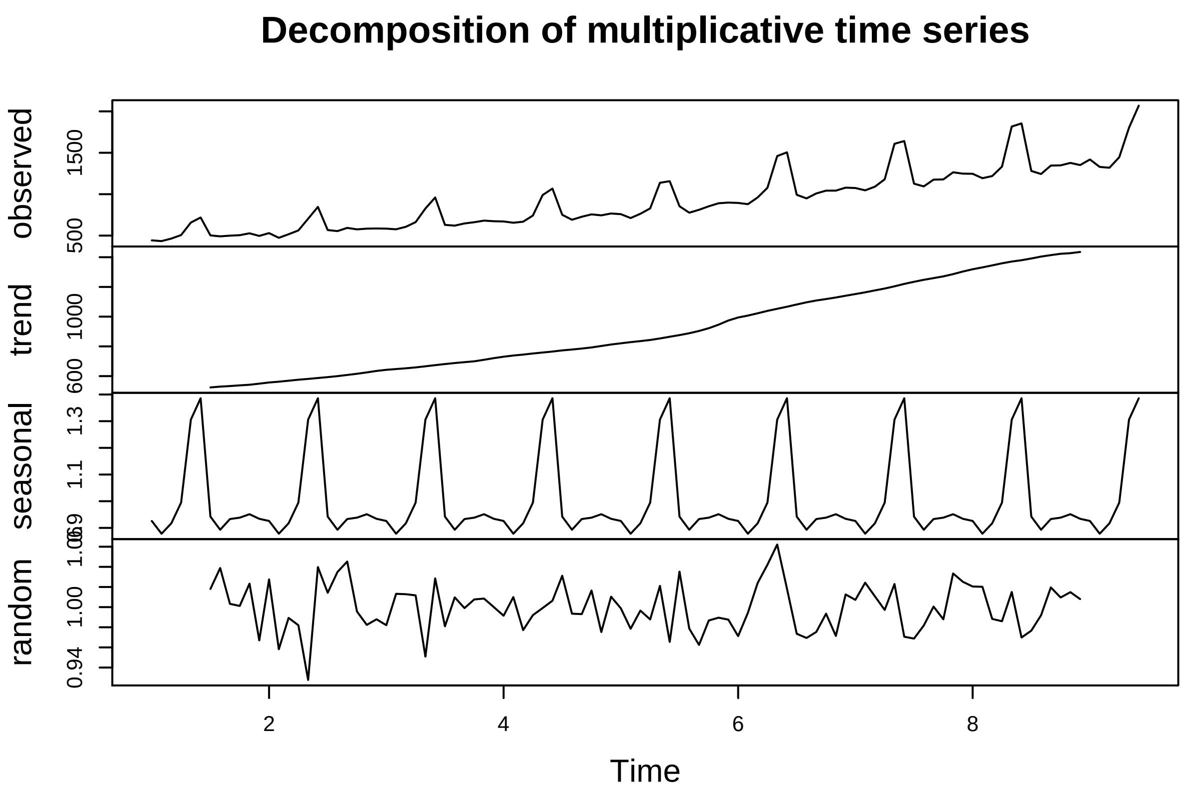


Exercise 2 (f) Decomposition

Decompose the time series using the `decompose` comand and comment.

To make life easier, some R packages provides decomposition with a single line of code. As expected, our step-by-step decomposition provides the same results as the DECOMPOSE( ) and STL( ) functions (see the graphs).

```r
par(mfrow=c(4,1))
ts_sales = ts(data$InternetRetail, frequency = 12)
decompose_sales = decompose(ts_sales, "multiplicative")
plot(decompose_sales)
```

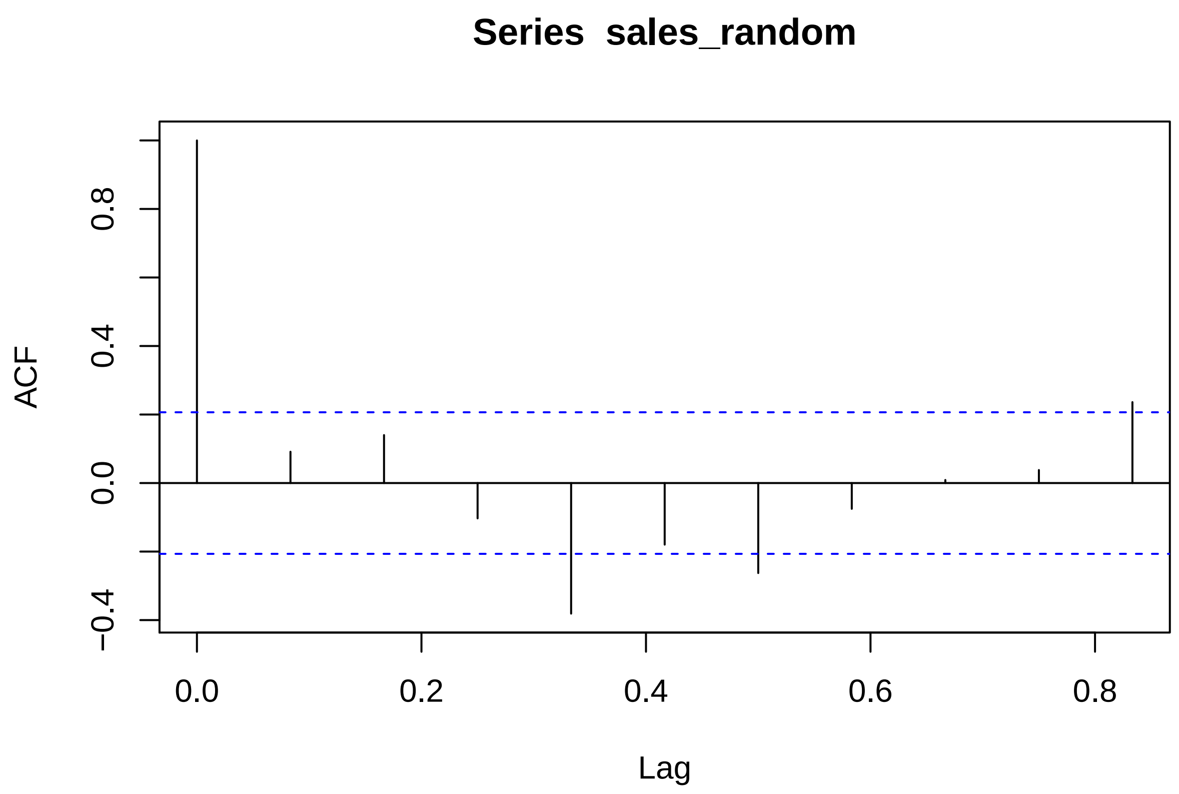


- The result looks exactly like the manaual multiplicative decomposition we did before.

Exercise 2 (g) Residual Component

Investigate the ACF of the residual component of the decomposition.

```r
sales_random = na.remove(as.ts(decompose_sales$random))

acf(x = sales_random, lag.max = 10)
```



Are they white noise?

- Clearly, we the residuals are white noise.

Exercise 2 (h) ARIMA

Fit an ARIMA model using the `auto.arima` commandof the `forecast` library.

The **`auto.arima`** function returns the best ARIMA model according to either AIC, AICc or BIC value. The function conducts asearch over possible model within the order constraints provided.

```r
arimaEst = auto.arima(y = data$InternetRetail)
summary(arimaEst)
```

```
## Series: data$InternetRetail
## ARIMA(2,1,3) with drift
##
## Coefficients:
##          ar1      ar2      ma1      ma2     ma3    drift
##       1.0615  -0.4095  -1.1277  -0.5128  0.7156  10.5268
## s.e.  0.1535   0.1295   0.1238   0.1709  0.1045   2.4103
##
## sigma^2 estimated as 13456:  log likelihood=-622.55
## AIC=1259.09   AICc=1260.3   BIC=1277.4
##
## Training set error measures:
##                    ME     RMSE      MAE       MPE     MAPE      MASE
## Training set 1.980348 111.9509 80.80768 -1.341132 9.065502 0.8850595
##                     ACF1
## Training set -0.03735354
```

- An ARIMA(p=2, d=1, q=3) model was chosen, with a nonzero intercept that acts as a drift term.

Check the model diagnostics.

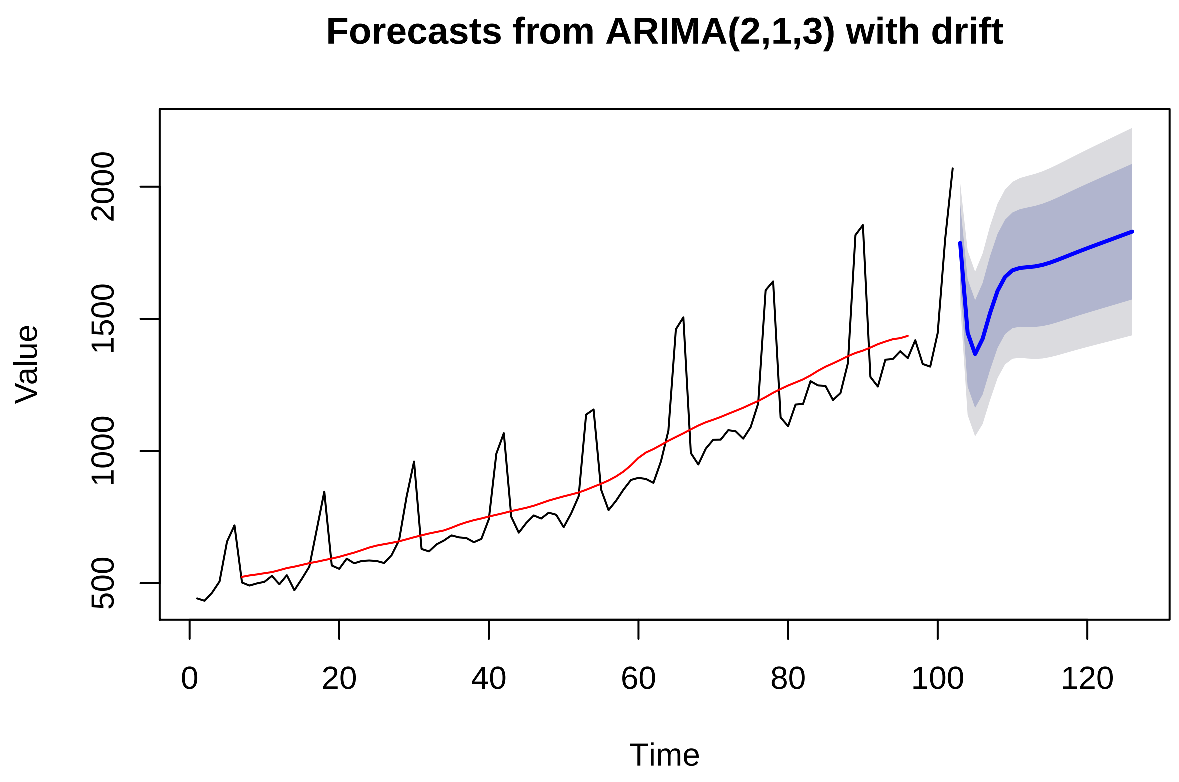
- The modle diagnostics suggest that the fit of the model is very good, because the RMSE is very low in the training set.

Exercise 2 (i) Forecast

Calculate $h = 24$ step ahead forecasts for the time series using the function **`forecast`** of the library **`forecast`**.

```r
plot(forecast(arimaEst, h = 24),
     xlab="Time", ylab="Value")
```

```r
trend_sales = ma(data$InternetRetail, order = 12, centre = TRUE)
lines(trend_sales, col="red")
```



Does your forecast live up to your expectations?

- The forecast seems to follow the trend line estimated by the 12-months moving average filter.