Zeitreihenanalyse Take Home Exam

Felix Reichel, k12008176

30.01.2022

0. Pre-Setup

Working Directory setzen, Variable mit Geburtsdatum *YYYYMMDD* anlegen, Warnmeldungen unterdrücken und Requirements (still) laden:

```
# setwd("./GitHub/learning-econometrics/Time_Series_Analysis_Statistics_JKU")
gebdate <- 20000117
options(warn = -1)
require(astsa, quietly = T, warn.conflicts = F)
require(tseries, quietly = T, warn.conflicts = F)
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
require(forecast, quietly = T, warn.conflicts = F, attach.required = T)
```

1. Setup:

In diesem Teil erzeugen wir uns das Zeitreihenobjekt mit unseren individuellen Zeitreihenausschnitt für das Exam. In meinem Fall ist das von **Januar 1975** bis **Dezember 2016**.

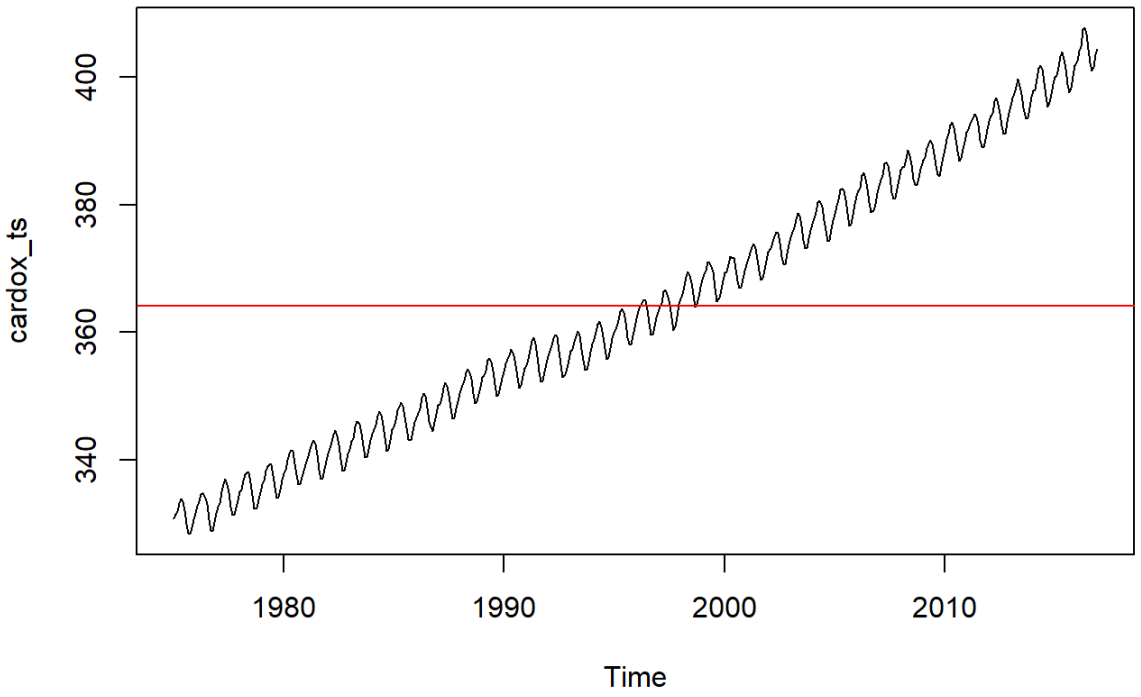
Zusätzlich erzeuge ich noch ein Objekt mit der gesamten Zeitreihe (Länge: **729**), welches ich später brauchen werde.

```
# Seed auf Geburtsdatum (YYYYMMDD) setzen
set.seed(gebdate)

# Ganzahlige Zufallszahl 1 <= x <= 20 generieren (mit aktuellen Seed randomInt = 5)
randomInt <- floor(runif(1, min = 1, max = 20))

# Startjahr bestimmen
J <- 1980 - randomInt # J=1975

# Zeitreihe ansehen # ?cardox
freq <- frequency(cardox)
J_end <- 2016+1-(1/freq) # 2017-(1/12) entspricht Dec 2016

# Zeitreihenobjekte anlegen, Zeitreihenausschnitt für das Exam
full_cardox_ts <- ts(data = cardox, start=1958+(2/12), end = 2019-(2/12), frequency = freq)
cardox_ts <- window(x = full_cardox_ts, start = J, end = J_end)
```

2. Analyse der Zeitreihenkomponenten:

Zunächst plotte ich die **gesamte Zeitreihe** und zeichne mir den **Mittelwert** (als rote Gerade) ein.

```
plot(cardox_ts)
abline(h = mean(cardox_ts), col="red")
```



Als nächstes habe ich mir noch einige Zeitreihenausschnitte mit Werten aus der näheren Vergangenheit geplottet. Aus Platzgrunden lasse ich diese Zeilen jedoch auskommentiert.

```
# Ein paar neuere Zeitreihenausschnitte plotten
# plot(window(cardox_ts, start=2010, end=J_end))
# plot(window(cardox_ts, start=2014, end=J_end))
```

Um die **Komponenten** der Zeitreihe zu **analysieren** und ein passendes Modell zu finden verwende ich die Funktion **decompose**.

```
decomposed <- decompose(cardox_ts, type = "additive")
# plot(decomposed)
```

Ein klassischen **additives** Dekompositionsmodell **scheint** hier **ausreichend** zu sein, da weder saisonale noch irreguläre Schwankungen (“sichtbar”) so ausehen als würden diese proportional zum Level der Zeitreihe ansteigen. Später wird sich jedoch herausstellen, dass die **saisonale Komponente multiplikativ** deutlich **besser** modelliert werden kann.

Als nächstes möchte ich wissen, ob unsere **Irreguläre Komponente normalverteilt** ist:

```
# qqnorm(decomposed$random)
# qqline(decomposed$random)
# hist(decomposed$random)
jarque.bera.test(na.omit(decomposed$random))
```

```
##
##  Jarque Bera Test
##
## data:  na.omit(decomposed$random)
## X-squared = 17.451, df = 2, p-value = 0.0001624
```

Der **QQ-Plot** zeigt leider einige **Ausreißer** (“dicke Enden”).
Das Histogramm ist leicht **linksschief** und weist einen **isolierten Balken** auf. (Ausreißer)
Interpretation JB-Test: X-squared = 17.451 > 6 und der p-Wert = 0.0001624 < 0.05 => **H0** (Normalverteiltheit) **wird** daher

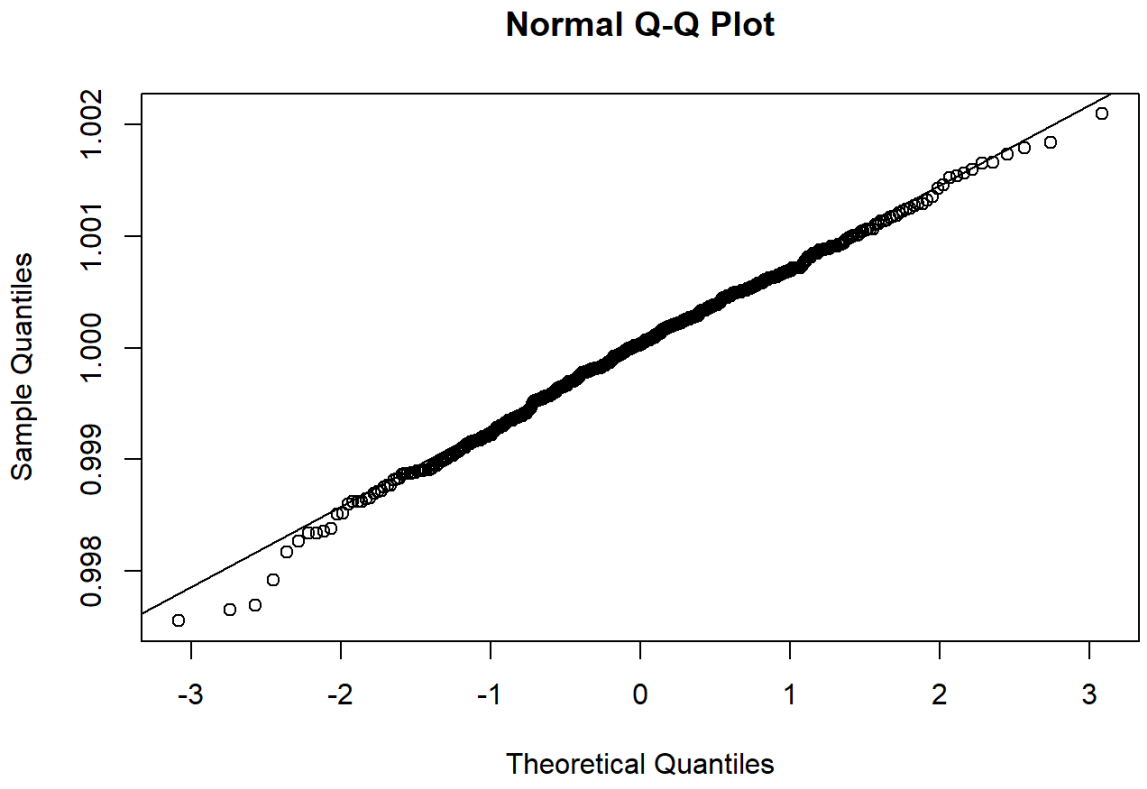
verworfen.
Ergebnis: Die irreguläre Komponente ist in unserem additiven Model für unseren Zeitreihenausschnitt ist nicht normalverteil t.

Zeitreihenkomponenten - Analyse:

Trend Komponente (μt):
Die Zeitreihe besitzt einen **positiven Trend** μt. Die Zeitreihe beginnt deutlich unterhalb dem Mittelwert und bleibt überhalb Mittelwert sobald dieser überschritten wurde, ausgenommen von saisonalen Schwankungen, welche im darauffolgenden Jahr dazu führen, dass ein Wert noch unterhalb des bereits überschrittenen Mittelwert landet.
Saisonale Komponente (st):
Die Zeitreihe besitzt **saisonale Schwankungen**. Die tiefsten Werte werden meistens im September erreicht.
Irreguläre Komponente (et):
Entfernt man die Trendkomponente und Saisonkomponente bleiben **irreguläre Fluktuationen** ü ber. Diese scheinen lt. QQ-Plot, Histogram und Jarque-Bera-Statistik (X-squared > 6) im Modell mit additiver saisonaler Komponente nicht normalverteilt zu sein. Ich glaube, dass dieses Ergebnis an den letzten Werten der Zeitreihe liegt und das gerade wegen den letzten Werten der Zeitreihe wir zu einem Modell mit **multiplikativer saisonaler Komponente** greifen müssen.
Nicht saisonale Schwankungen (ct): Keine /Nicht vorh anden

Überprüfung obiger Annahmen zur irrelären Komponente **et**:

```
# decomposed2 <- decompose(window(cardox_ts, end=2015), type = "additive")
# jarque.bera.test(na.omit(decomposed2$random)) # X-squared  = 6.9479 > 6 => H1 => Nicht Normalverteilt

# decomposed3 <- decompose(window(cardox_ts, end=2000), type = "additive")
# jarque.bera.test(na.omit(decomposed3$random)) # X-squared  = X-squared = 2.5633 < 6 => H0 => Normalver
teilt

decomposed_mult <- decompose(cardox_ts, type = "multiplicative")
# plot(decomposed_mult)
qqnorm(decomposed_mult$random)
qqline(decomposed_mult$random)
```



```
# hist(decomposed_mult$random)
jarque.bera.test(na.omit(decomposed_mult$random))
```

```
##
##  Jarque Bera Test
##
## data:  na.omit(decomposed_mult$random)
## X-squared = 4.0563, df = 2, p-value = 0.1316
```

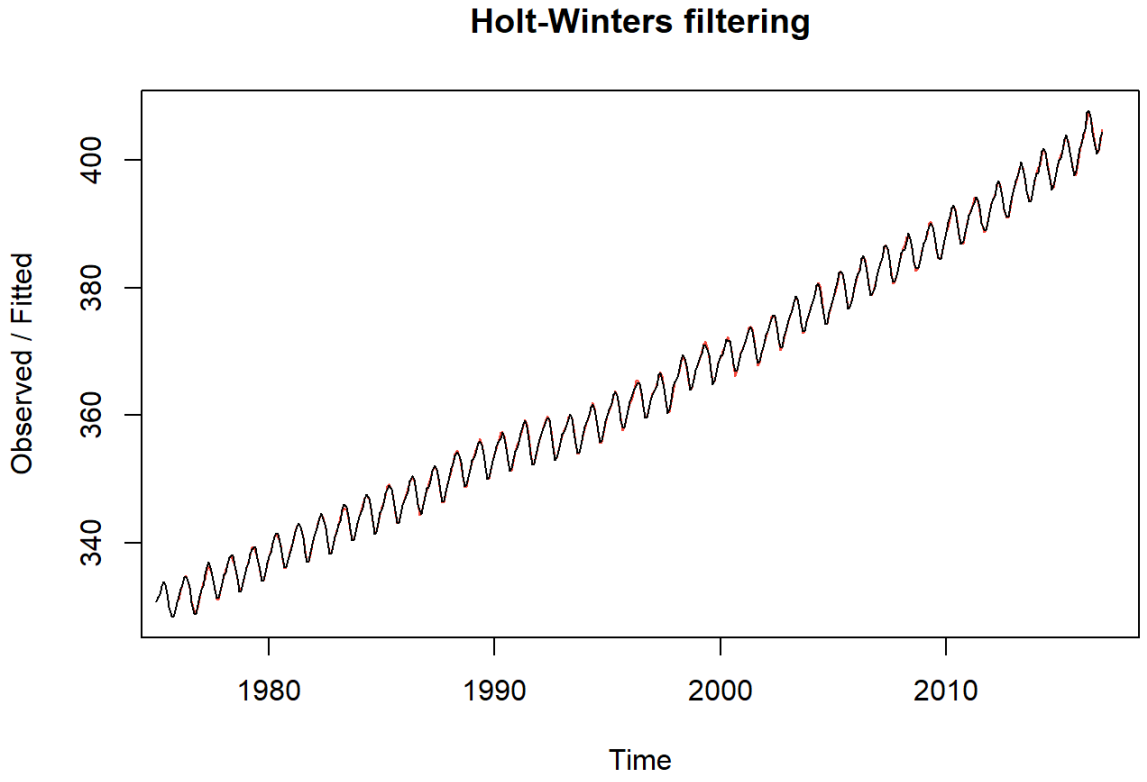
Ergebnis: Die irreguläre Komponente im Modell **mit multiplikativer saisonale Komponente** ist für den **gesamten Zeitreihenausschnitt normalverteilt**!

3. Analyse mit geeigneter Methode des exponentiellen Glättens

Wir haben festgestellt, dass wir eine lineare Trendkomponente und multiplikative saisonale Komponente haben. Daher wählen wir **Holt-Winters Forecasting mit multiplikativer saisonaler Komponente** als geeignete Methode zum exponentiellen Glätten aus.

```
cardox_ts_exp_m <- HoltWinters(cardox_ts, seasonal = "multiplicative")
# Nur zum Vergleich noch HoltWinters mit additiver saisonale Komponente
cardox_ts_exp_a <- HoltWinters(cardox_ts, seasonal = "additive")
```

```
# plot(cardox_ts_exp_a)
plot(cardox_ts_exp_m)
```



Beide Modelle sehen ganz gut aus!
Aber schneidet unser Modell mit multiplikativer saisonaler Komponente auch besser ab?
Erinnerung: Überprüfung mit den in der näher liegenden vergangenen Werten der Zeitreihe, sollte zu deutlich besseren Ergebnissen für unser multiplikatives Modell führen .

```
# Modelle durch Splitting in Training und Test Set validieren
cardox_len <- length(cardox_ts)

test_set_yrs <- 4 # die letzten 4 Jahre als Test Set zur Validierung entsprich 48 von 504 Beobachtungen
 (~10%)
test_set_len <- test_set_yrs * freq
train_set_len <- cardox_len - test_set_len
cut <- J_end - test_set_yrs

training_set <- window(cardox_ts, start = J, end = cut)
test_set <- window(cardox_ts, start = cut, end = J_end)

cardox_ts_train_exp_m <- HoltWinters(training_set, seasonal = "multiplicative")
cardox_ts_train_exp_a <- HoltWinters(training_set, seasonal = "additive")

cardox_ts_train_exp_m_pred = predict(cardox_ts_train_exp_m, n.ahead = test_set_len, prediction.interval
 = T)  # Konfidenzintervalle sind größer beim Modell mit multiplikativer Saisonkomponente
cardox_ts_train_exp_a_pred = predict(cardox_ts_train_exp_a, n.ahead = test_set_len, prediction.interval
 = T)

# plot(cardox_ts_train_exp_m, cardox_ts_train_exp_m_pred, lwd = 2)
# plot(cardox_ts_train_exp_a, cardox_ts_train_exp_a_pred, lwd = 2)

actual_values <- tail(cardox_ts, test_set_len)

hw_m_predicted <- cardox_ts_train_exp_m_pred[,1]
hw_a_predicted <- cardox_ts_train_exp_a_pred[,1]
```

Vergleich der Fehler zu den tatsächlichen Werten für die beiden Modelle:

```
# Compare ME, RMSE, MAE, MAPE  Errors
rbind(accuracy(actual_values, hw_m_predicted),
      accuracy(actual_values, hw_a_predicted))
```

```
##                  ME     RMSE       MAE        MPE      MAPE      ACF1 Theil's U
## Test set -0.6983341 1.029386 0.7383716 -0.1739875 0.1840413 0.7803418 0.8259731
## Test set -0.8216280 1.126819 0.8407560 -0.2048207 0.2096245 0.7886920 0.9383617
```

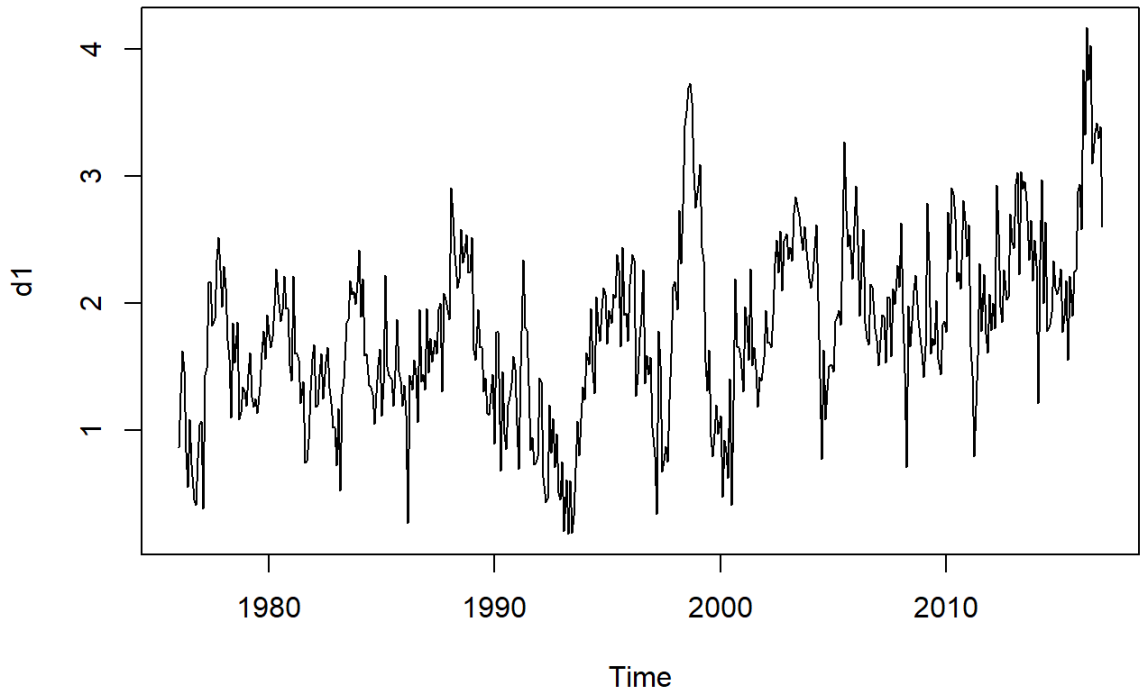
Bei allen drei gängigen Vorhersage-Fehlern *(MAE, MAPE, MSE)* schneidet das Modell mit der multiplikativen saisonalen Kompoenente besser ab.

4. Auswahl eines geeigneten ARIMA Modells

Eine Zeitreihe mit einem Trend μt ist nicht stationär, weil der Erwartungswert nicht konstant ist, sondern eben von t abhängt.

Bildung von Differenzenprozesse: Wir “differenzieren” die Zeitreihe bei Lag 12 um die Saisonalität und anschließend bei Lag 1 um den Trend zu entfernen .

```
d1 <- diff(cardox_ts, lag = freq)
plot(d1) # sieht noch nicht stationär aus
```



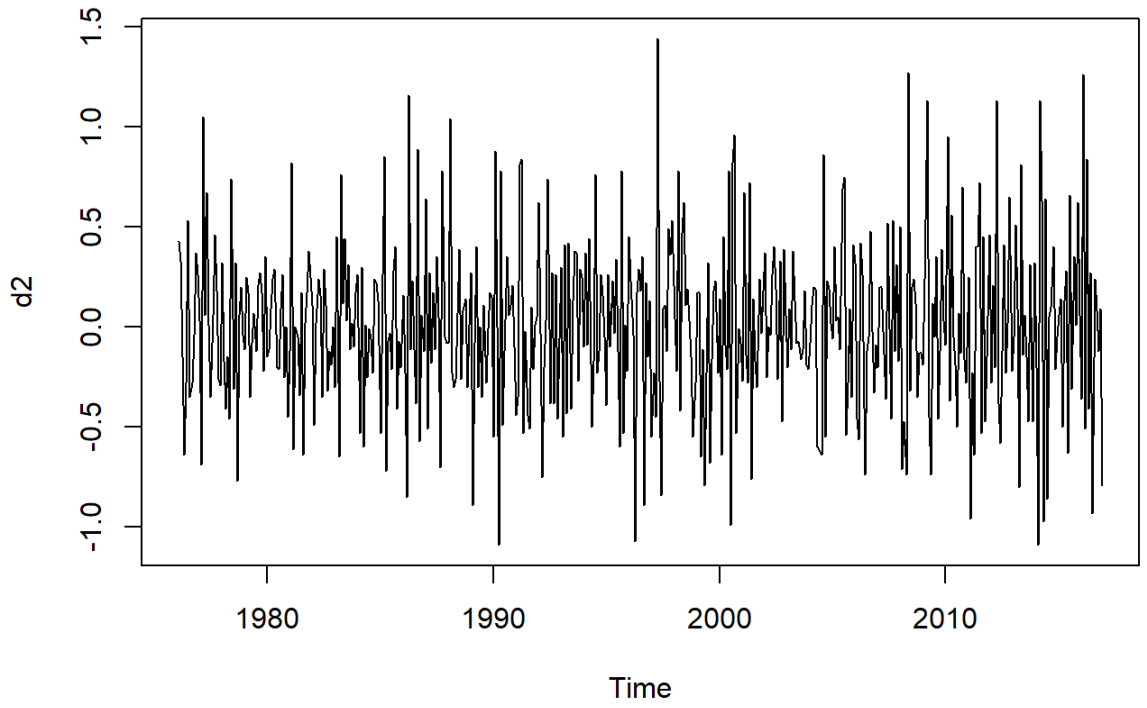
Scheint noch einen Trend zu haben und kehrt nicht regelmäßig zum Erwartungswert zurück.

Einheitswurzeltests für Stationarität von d1:

```
# adf.test(d1) # p-value = 0.01 < 0.05 => H1 => Stationary
# kpss.test(d1, null= "Level") # p-value = 0.01 < 0.05 => H1 => No Level Stationarity
# kpss.test(d1, null = "Trend") # p-value = 0.023 < 0.05 => H1 => No Trend Stationarity
```

Laut KPSS Test wie vermutet noch nicht stationär!

```
d2 <- diff(d1)
plot(d2)
```



Einheitswurzeltests für Stationarität von d2:

```
adf.test(d2) # p-value = 0.01 < 0.05 => H1 => Stationary
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  d2
## Dickey-Fuller = -8.3168, Lag order = 7, p-value = 0.01
## alternative hypothesis: stationary
```

```
kpss.test(d2, null= "Level") # p-value = 0.1 > 0.05 => H0 => Level Stationarity
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  d2
## KPSS Level = 0.01124, Truncation lag parameter = 5, p-value = 0.1
```

```
kpss.test(d2, null= "Trend") # p-value = 0.1 > 0.05 => H0 => Trend Stationarity
```

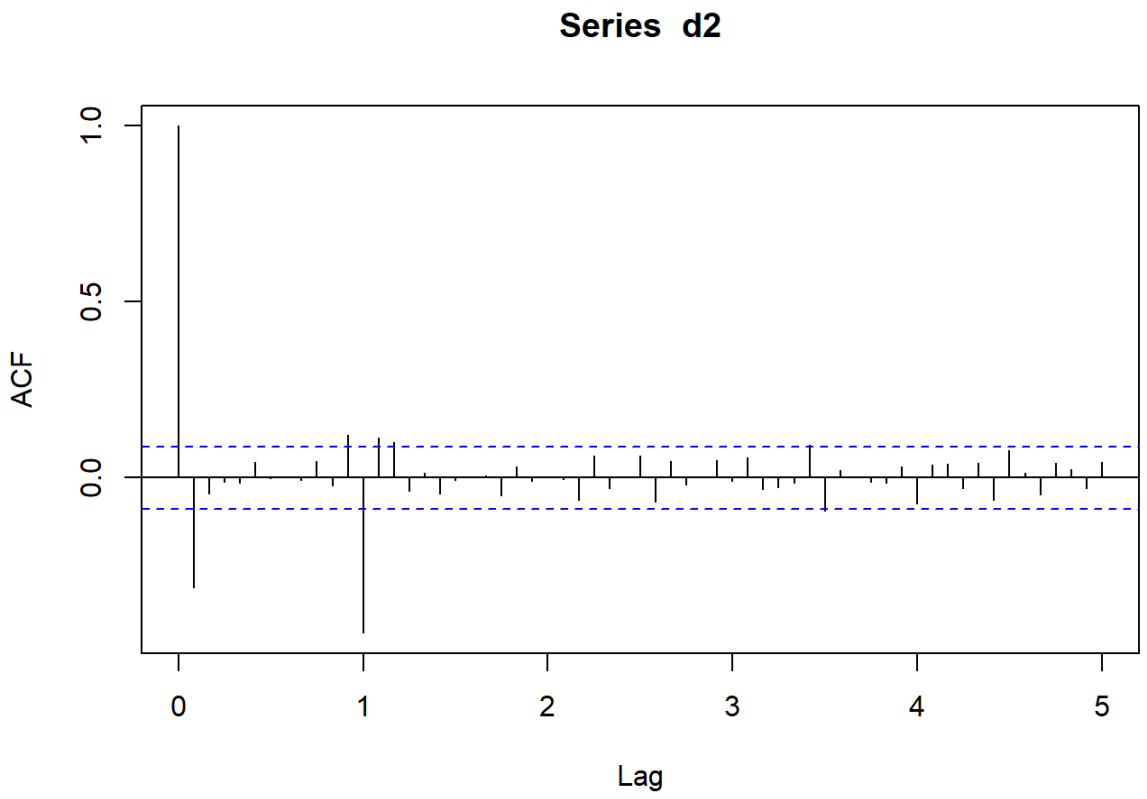
```
##
##  KPSS Test for Trend Stationarity
##
## data:  d2
## KPSS Trend = 0.011354, Truncation lag parameter = 5, p-value = 0.1
```

ADF und KPSS Tests des 2. Differenzenprozesses schlagen vor, dass dieser Prozess nun stationär ist.

Im Plot sieht dieser Prozess auch schon ziemlich stationär aus ! (Kehrt regelmäßig zum Erwartungswert zurück. Varianz sieht auch ok aus.)

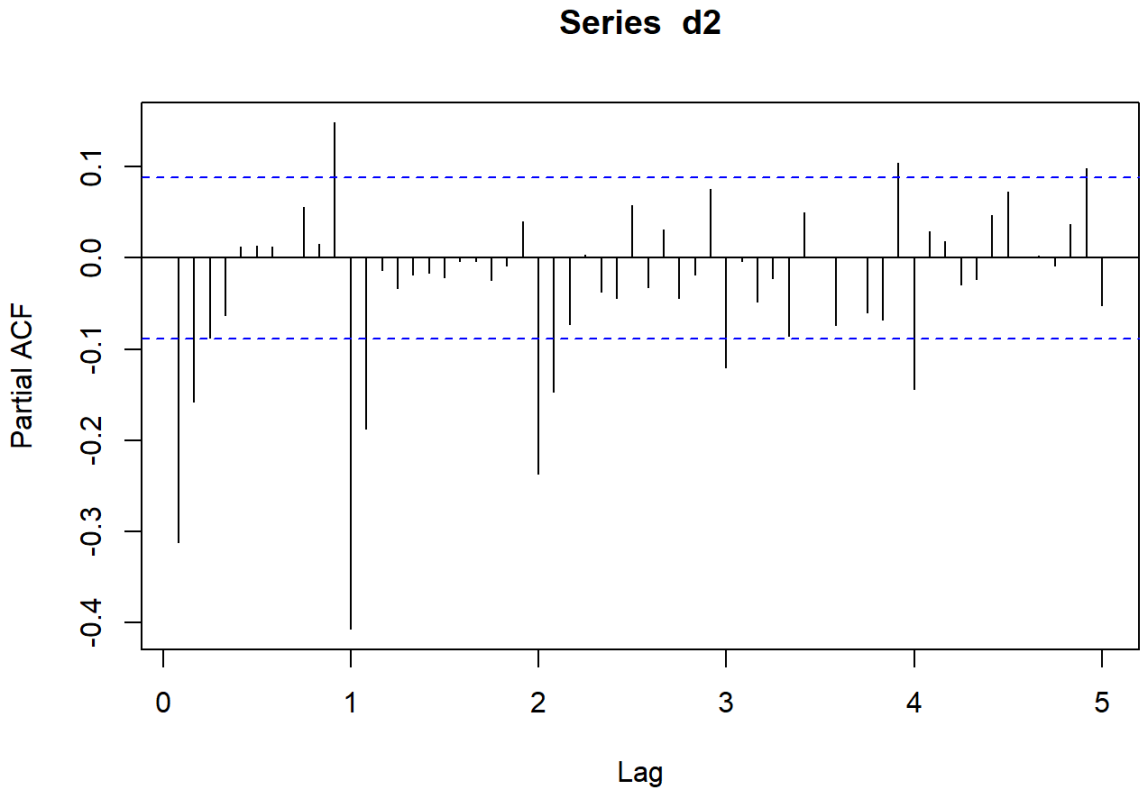
ACF und PACF zur Bestimmung der Ordnung von p, q, P und Q

```
# acf(d2)
acf(d2, lag.max = 60)
```



Die ACF schlägt bei Lag 1 aus und danach nicht mehr regelmäßig. Bzw. meist nur noch ganz leicht über den Grenzen, daher entscheide ich mich für eine Ordnung von p = 1 und P = 0 für das saisonale Modell.

```
# pacf(d2)
pacf(d2, lag.max = 60)
```



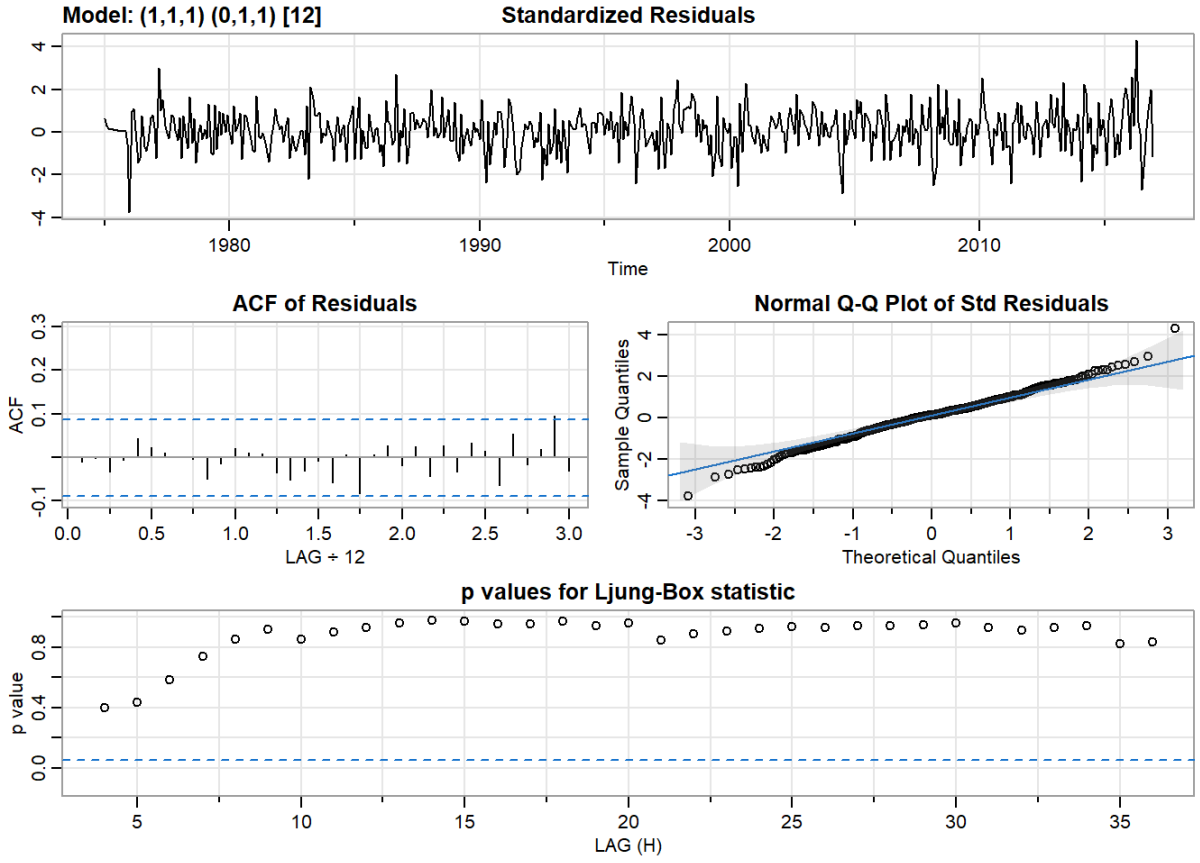
Die PACF zeigt regelmäßige Ausschläge beim 1. bzw. danach jeden 12ten bzw. 1. Lag, Daher entscheide ich mich für q = 1 und Q = 1 für das saisonale Modell.

Ich beabsichtige also ein SARIMA(1,1,1)x(0,1,1) mit Lag = 12 Modell zu fitten.

Da wir eine Zeitreihe mit saisonaler Komponente haben können wir entweder ein ARIMA Modell mit seasonalen Dummies und ARIMA Fehlern fitten oder ein SARIMA Modell verwenden. Ich verwende letzteres, da dieses nicht explizit durch die Angabe ausgeschlossen wird.

```
# Fit eines geeigneten Modells
cardox_sarima_m1 <- sarima(xdata = cardox_ts, p = 1, d = 1, q = 1, P = 0, D = 1, Q = 1, S = freq)
```

```
## initial  value -0.854640
## iter   2 value -1.049945
## iter   3 value -1.135364
## iter   4 value -1.141453
## iter   5 value -1.146061
## iter   6 value -1.148608
## iter   7 value -1.152914
## iter   8 value -1.155337
## iter   9 value -1.155539
## iter  10 value -1.156573
## iter  11 value -1.157583
## iter  12 value -1.157727
## iter  13 value -1.157796
## iter  14 value -1.157799
## iter  15 value -1.157800
## iter  15 value -1.157800
## final  value -1.157800
## converged
## initial  value -1.161249
## iter   2 value -1.162101
## iter   3 value -1.162472
## iter   4 value -1.162561
## iter   5 value -1.162611
## iter   6 value -1.162665
## iter   7 value -1.162723
## iter   8 value -1.162723
## iter   8 value -1.162723
## final  value -1.162723
## converged
```



Beurteilung der Anpassung:

Alle p-Werte in der Ljung-Box statistik sind > 0.05 das heißt wir können H0 verwerfen sprich die Fehler sind voneinander unabhängig, was wir für unser Modell wollen. Die Residuale sind im QQ-Plot leider nicht ganz normalverteilt wegen ein paar Ausreißer außerhalb der Intervalle (“dicke Enden”) In der ACF der ist sind keine hohen Ausschläge mehr sichtbar. Sprich

keine Korrelationen zwischen den Fehlern

Residualanalyse:

```
residuals <- cardox_sarima_m1$fit$residuals
# qqnorm(residuals)
# qqline(residuals)
# hist(residuals)
# jarque.bera.test(residuals)
# X-squared = 17.154 > 6 und p-value = 0.0001884 < 0.05 => Residuale nicht normalverteilt
```

Gängige Informationskritieren (AIC/BIC):

Geringere Werte sprich weniger Information bedeutet, dass unser Modell besser passt!

```
AIC(cardox_sarima_m1$fit)
```

```
## [1] 259.6033
```

```
BIC(cardox_sarima_m1$fit)
```

```
## [1] 276.389
```

Modellvergleich:

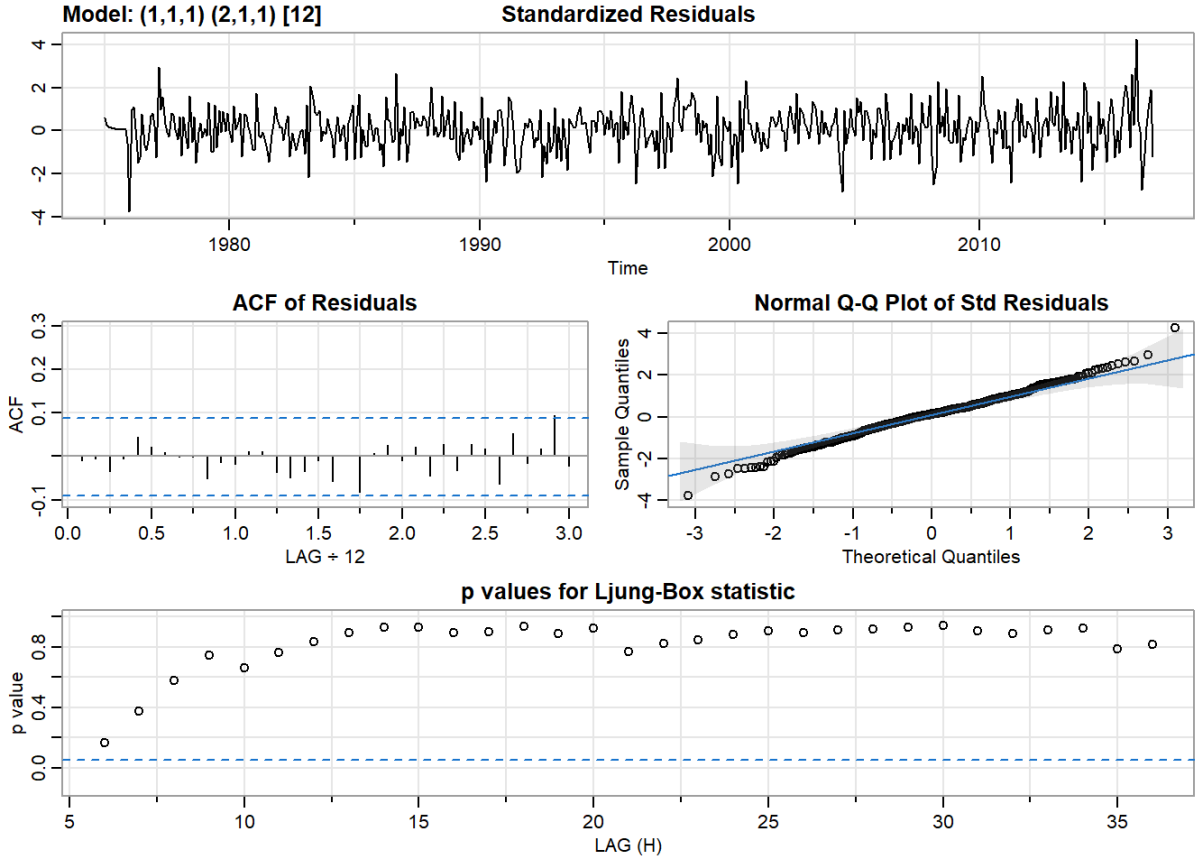
Aus Interesse möchte ich das gewählte Modell mit **auto.sarima** vergleichen:

```
# install.packages('bayesforecast')
# require(bayesforecast)
# auto.sarima(cardox_ts, seasonal = TRUE)
```

auto.sarima schlägt das Modell folgendes Modell vor: SARIMA(1,1,1)x(2,1,1)[12]

```
cardox_auto_sarima_m <- sarima(xdata = cardox_ts, p = 1, d = 1, q = 1, P = 2, D = 1, Q = 1, S = freq)
```

```
## initial  value -0.852561
## iter   2 value -1.059605
## iter   3 value -1.117546
## iter   4 value -1.124468
## iter   5 value -1.133919
## iter   6 value -1.141904
## iter   7 value -1.146237
## iter   8 value -1.148282
## iter   9 value -1.149656
## iter  10 value -1.149990
## iter  11 value -1.150324
## iter  12 value -1.150763
## iter  13 value -1.150780
## iter  14 value -1.150786
## iter  15 value -1.150791
## iter  16 value -1.150798
## iter  17 value -1.150799
## iter  18 value -1.150799
## iter  19 value -1.150799
## iter  19 value -1.150799
## iter  19 value -1.150799
## final  value -1.150799
## converged
## initial  value -1.155026
## iter   2 value -1.161002
## iter   3 value -1.163195
## iter   4 value -1.163408
## iter   5 value -1.163715
## iter   6 value -1.163732
## iter   7 value -1.163742
## iter   7 value -1.163742
## iter   7 value -1.163742
## final  value -1.163742
## converged
```



```
AIC(cardox_auto_sarima_m$fit)
```

```
## [1] 262.6029
```

```
BIC(cardox_auto_sarima_m$fit)
```

```
## [1] 287.7816
```

Tatsächlich hat mein gewähltes **Modell** aber einen **geringerem Wert beim AIC und BIC** als das von **auto.sarima** gewählte Modell.

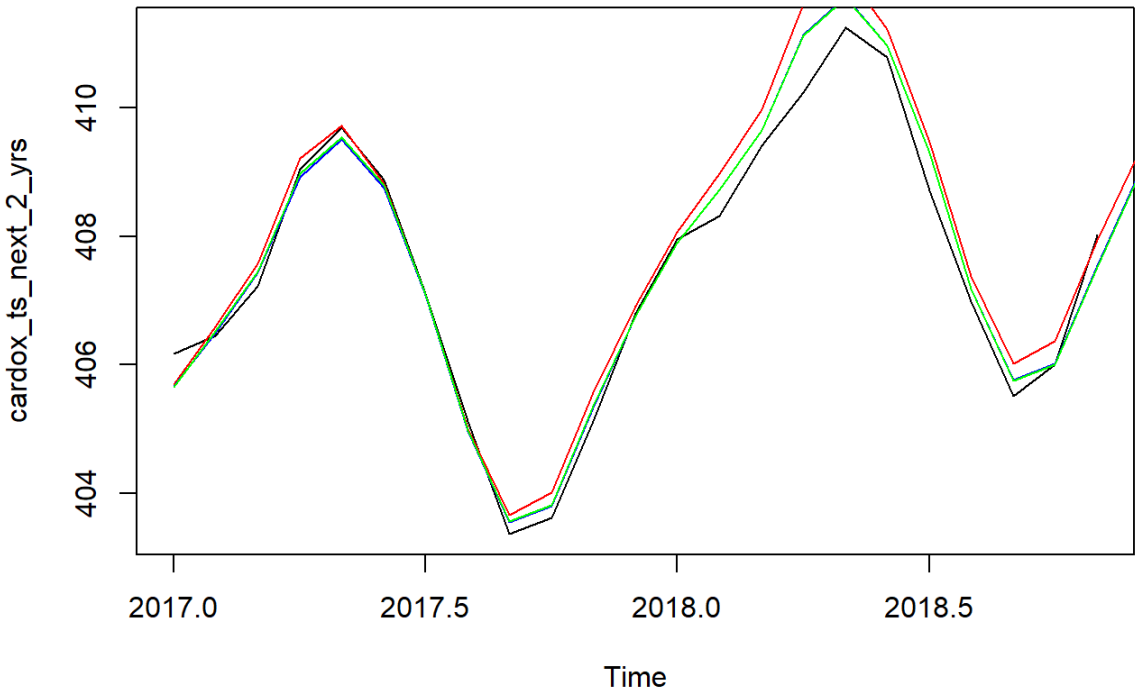
5. Vergleich der Vorhersagen

Farbe	Variable	Modell
Rot	cardox_ts_exp_m	Holt-Winters mit multiplikativer Saisonkomponente
Blau	cardox_sarima_m1	SARIMA(1,1,1)x(0,1,1) Lag=12
Grün	cardox_auto_sarima_m	SARIMA(1,1,1)x(2,1,1) Lag=12

```
# Die nächsten 2 Jahre als Zeitreihenobjekt
cardox_ts_next_2_yrs <- window(x = full_cardox_ts, start = 2017)

holtWinters_m_pred <- predict(cardox_ts_exp_m, n.ahead = 24, prediction.interval = T)
sarima_pred <- predict(cardox_sarima_m1$fit, n.ahead = 24)
auto_sarima_pred <- predict(cardox_auto_sarima_m$fit, n.ahead = 24)
```

Plotten der Vorhersagen:

```
# Schwarz = echte Werte, Rot = HoltWinters, Blau = SARIMA, Grün = Auto-SARIMA
ts.plot(cardox_ts_next_2_yrs)
lines(holtWinters_m_pred[,1], col="red")
lines(sarima_pred$pred, col="blue")
lines(auto_sarima_pred$pred, col="green")
```



Vergleich der Vorhersagefehler:

```
rbind(
  accuracy(cardox_ts_next_2_yrs, holtWinters_m_pred[,1]),
  accuracy(cardox_ts_next_2_yrs, sarima_pred$pred),
  accuracy(cardox_ts_next_2_yrs, auto_sarima_pred$pred))
```

```
##                ME      RMSE       MAE        MPE       MAPE      ACF1
## Test set 0.3136145 0.4935137 0.3798856 0.07665533 0.09297405 0.4766149
## Test set 0.1023666 0.3260811 0.2501929 0.02496080 0.06127057 0.3420497
## Test set 0.1082584 0.3254880 0.2474726 0.02641025 0.06061750 0.3076728
##          Theil's U
## Test set 0.3761882
## Test set 0.2462259
## Test set 0.2446509
```

Die beiden SARIMA Modelle schneiden annähernd gleich und schneiden hier deutlich besser als HoltWinters Exponentielles Glätten ab.

6. Volatilitätsmodelle

Es ist meiner Meinung hier nicht notwendig nach Volatilität zu modellieren, weil es hier um eine Kohlendioxid Menge in der Luft gemessen am Mauna Loa Observatory, Hawaii handelt. CO2 zu ca. 4% in der Atemluft und verteilt sich. Wenn der Ort nicht selbst CO2 Emissionen ausstößt, kann es also keine starken Veränderungen in der Varianz (sprich Schocks) geben.

Volatiliätsmodelle sind sinnvoll bei
- starken Veränderungen in der Varianz
- Perioden mit höheren und niedrigeren Schwankungen (Varianz)
- “Volatitäts-Cluster”

z.B. Aktienkurse (unterliegen ökonomischen Schocks wie etwa bei der Finanzkrise 2008)