Overview

Julian Dietlmeier

March 29, 2019

Das ist ein kurzer Überblick über meine Bachelorarbeit mit Code und Ergebnissen. Ziel ist es, am Beispiel Microsoft ein mal durch all meine Berechnungen zu führen. Begonnen wird mit dem laden der benötigten Paketen und meiner eigenen Funktionen (aus functions.R, dienen überwiegend der Vereinfachung von Import und plotten von Verteilungen oder Markierung von extrem Werten, die Funktionen werden später vorgestellt). Kommentare im Code sind häufig alternative Methodiken.

## 0) Set Up

rm(list = ls())  
library(readxl)  
library(xts)  
library(forecast)  
library(aTSA)  
  
source("functions.R")

## 1) Import

Import der Fundamental- (BV) und Preisdaten (MV). Diese werden in einer eigenen Enviroment gespeichert. Hiermit wird versucht die Daten von verschiedenen Firmen abzugrenzen. Außerdem macht es den Import mittels einer Funktion überhaupt erst möglich.

P\_start\_date <- "1986-03-13"  
BV\_start\_date <- "/1990-03-29"  
count\_of\_days <- 12017  
  
MSFT <- new.env()  
MSFT$BV <- read\_excel("Data\_Eikon/American\_Electronics/Microsoft.xlsx",   
 sheet = "Tabelle1", col\_types = c("date",   
 "numeric", "numeric", "numeric",   
 "numeric", "numeric", "numeric", "numeric", "numeric"))  
MSFT$MV <- read\_excel("Data\_Eikon/American\_Electronics/Microsoft.xlsx",   
 sheet = "Prices", col\_types = c("date",   
 "numeric"))

## 2) Vorbereitung

Nachdem die Daten importiert wurden, wird eine weitere wichtige Kennzahl berechnet. Der “twelve months trailing Earnings per Share”:

MSFT$BV$EPS\_ttm <- rep(0)  
for(i in c((nrow(MSFT$BV)-3):1)){  
 MSFT$BV$EPS\_ttm[i] <- sum(MSFT$BV$NI[c((i+3):i)]) / MSFT$BV$Shares[i]   
}  
  
#calculation of changes  
MSFT$BV$d.BPS\_E <- c((ts(MSFT$BV$BPS\_E) - lag(ts(MSFT$BV$BPS\_E)))/lag(ts(MSFT$BV$BPS\_E)),0)  
MSFT$BV$d.EPS\_ttm <- c((ts(MSFT$BV$EPS\_ttm) - lag(ts(MSFT$BV$EPS\_ttm)))/lag(ts(MSFT$BV$EPS\_ttm)),0)

Im Anschluss werden die Preis- und Fundamentaldaten auf eine einheitliche Zeitskala abgebildet. Dazu wird ein xts-Objekt erstellt, welches alle Tage von Anfang der Aufzeichnungen bis zum 4.2.2019 enthält, die Daten werden dann an der entsprechenden Stelle zugeordnet. BPS = Book Value per Share:

BPS\_E ist dabei auf das Equity bezogen, BPS\_D auf das Debt Capital und BPS\_T auf den kompletten Firmen Wert.

EPS = Earnings per Share:

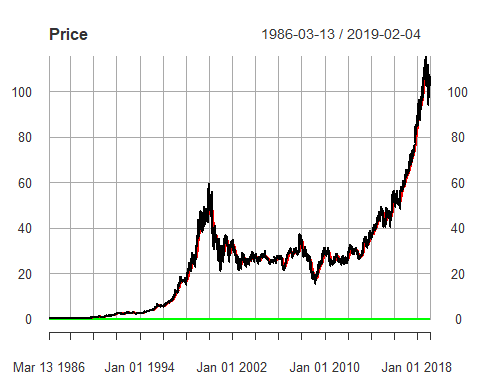
MSFT$Data <- data.frame(matrix(0, nrow = count\_of\_days, ncol = 13))  
names(MSFT$Data) <- c("Date", "PPS", "BV\_E", "BV\_T", "Shares", "BPS\_E", "BPS\_D", "BPS\_T", "NI", "EPS", "EPS\_ttm", "d.BPS\_E", "d.EPS\_ttm")  
MSFT$Data[,1] <- seq(as.Date(P\_start\_date),length=count\_of\_days ,by="days")  
  
  
  
MSFT$Data[as.character.Date(MSFT$Data[,1]) %in% as.character.Date(MSFT$MV$Date) ,2] <-   
 as.numeric(rev(MSFT$MV$Close))  
MSFT$Data[as.character.Date(MSFT$Data[,1]) %in% as.character.Date(MSFT$BV$Date) ,3] <-   
 as.numeric(rev(MSFT$BV$BV\_E))  
MSFT$Data[as.character.Date(MSFT$Data[,1]) %in% as.character.Date(MSFT$BV$Date) ,4] <-   
 as.numeric(rev(MSFT$BV$BV\_T))  
MSFT$Data[as.character.Date(MSFT$Data[,1]) %in% as.character.Date(MSFT$BV$Date) ,5] <-   
 as.numeric(rev(MSFT$BV$Shares))  
MSFT$Data[as.character.Date(MSFT$Data[,1]) %in% as.character.Date(MSFT$BV$Date) ,6] <-   
 as.numeric(rev(MSFT$BV$BPS\_E))  
MSFT$Data[as.character.Date(MSFT$Data[,1]) %in% as.character.Date(MSFT$BV$Date) ,7] <-   
 as.numeric(rev(MSFT$BV$BPS\_D))  
MSFT$Data[as.character.Date(MSFT$Data[,1]) %in% as.character.Date(MSFT$BV$Date) ,8] <-   
 as.numeric(rev(MSFT$BV$BPS\_T))  
MSFT$Data[as.character.Date(MSFT$Data[,1]) %in% as.character.Date(MSFT$BV$Date) ,9] <-   
 as.numeric(rev(MSFT$BV$NI))  
MSFT$Data[as.character.Date(MSFT$Data[,1]) %in% as.character.Date(MSFT$BV$Date) ,10] <-   
 as.numeric(rev(MSFT$BV$EPS))  
MSFT$Data[as.character.Date(MSFT$Data[,1]) %in% as.character.Date(MSFT$BV$Date) ,11] <-   
 as.numeric(rev(MSFT$BV$EPS\_ttm))  
MSFT$Data[as.character.Date(MSFT$Data[,1]) %in% as.character.Date(MSFT$BV$Date) ,12] <-   
 as.numeric(rev(MSFT$BV$d.BPS\_E))  
MSFT$Data[as.character.Date(MSFT$Data[,1]) %in% as.character.Date(MSFT$BV$Date) ,13] <-   
 as.numeric(rev(MSFT$BV$d.EPS\_ttm))  
  
MSFT$Data$d.BPS\_E[is.nan(MSFT$Data$d.BPS\_E)] <- 0  
MSFT$Data$d.BPS\_E[is.infinite(MSFT$Data$d.BPS\_E)] <- 0  
  
MSFT$Data$d.EPS\_ttm[is.nan(MSFT$Data$d.EPS\_ttm)] <- 0  
MSFT$Data$d.EPS\_ttm[is.infinite(MSFT$Data$d.EPS\_ttm)] <- 0

Die nicht aufgezeichneten Zeitpunkte (per Definition alle Punkte mit Wert = 0) werden dann “geforwarded”, sprich durch die letzte aufgezeichnete Beobachtung ersetzt.

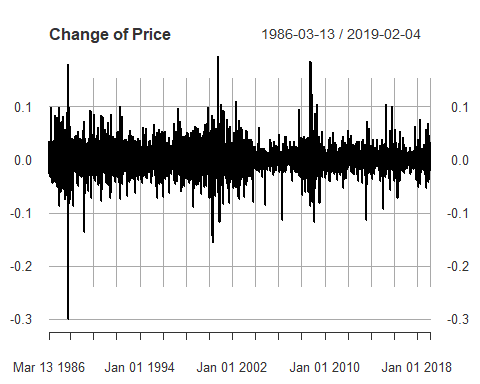
for(i in c(2:count\_of\_days)){  
 if(MSFT$Data[i, 2] == 0) MSFT$Data[i, 2] <- MSFT$Data[i-1, 2]  
 if(MSFT$Data[i, 9] == 0) MSFT$Data[i, 9] <- MSFT$Data[i-1, 9]  
 if(MSFT$Data[i, 10] == 0) MSFT$Data[i, 10] <- MSFT$Data[i-1, 10]  
 if(MSFT$Data[i, 11] == 0) MSFT$Data[i, 11] <- MSFT$Data[i-1, 11]  
 if(MSFT$Data[i, 12] == 0) MSFT$Data[i, 12] <- MSFT$Data[i-1, 12]  
 if(MSFT$Data[i, 13] == 0) MSFT$Data[i, 13] <- MSFT$Data[i-1, 13]  
}

Es geht weiter mit den spezifischen Daten des Preises. Hier werden zusätzliche Variablen wie die Veränderung (immer mit d.\* gekennzeichnet) und der MA berechnet. Diese werden in einem xts-Objekt zusammengefasst. Das dient der besseren Darstellbarkeit und Übersicht. Die Veränderung wird als prozentuale Veränderung zum Vortag berechnet:

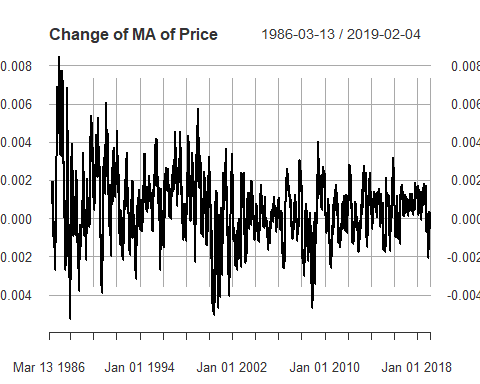
MSFT$P\_xts <- xts(MSFT$Data$PPS, order.by = MSFT$Data$Date)  
colnames(MSFT$P\_xts) <- c("P")  
#MSFT$P\_xts$MA <- xts(ma(MSFT$P\_xts$P, 181), order.by = MSFT$Data$Date)  
#MSFT$P\_xts$MA2 <- xts(ma(MSFT$P\_xts$P, 913), order.by = MSFT$Data$Date)  
  
MSFT$P\_xts$MA <- xts(c(rep(NA, 89), rollmeanr(MSFT$P\_xts$P, 90)), order.by = MSFT$Data$Date)  
#MSFT$P\_xts$MA <- xts(ma(MSFT$P\_xts$P, 90), order.by = MSFT$Data$Date)  
MSFT$P\_xts$d.P <- (MSFT$P\_xts$P - lag(MSFT$P\_xts$P, 1))/lag(MSFT$P\_xts$P, 1)  
MSFT$P\_xts$d.P[is.nan(MSFT$P\_xts$d.P)] <- NA  
MSFT$P\_xts$d.P[is.infinite(MSFT$P\_xts$d.P)] <- NA  
MSFT$P\_xts$d.P\_MA <- (MSFT$P\_xts$MA - lag(MSFT$P\_xts$MA, 1))/lag(MSFT$P\_xts$MA, 1)  
MSFT$P\_xts$d.P\_MA[is.nan(MSFT$P\_xts$d.P\_MA)] <- NA  
MSFT$P\_xts$d.P\_MA[is.infinite(MSFT$P\_xts$d.P\_MA)] <- NA  
  
plot(MSFT$P\_xts, col = c("black", "red", "green"), main = c("Price"))



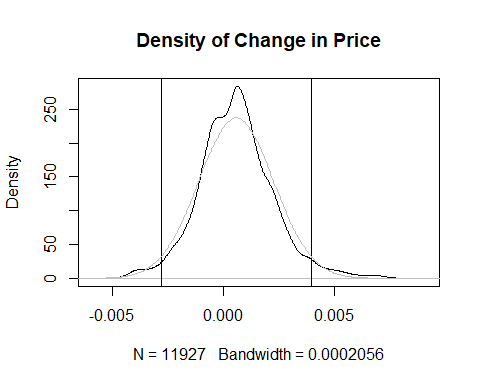
plot(MSFT$P\_xts$d.P, main = c("Change of Price"))



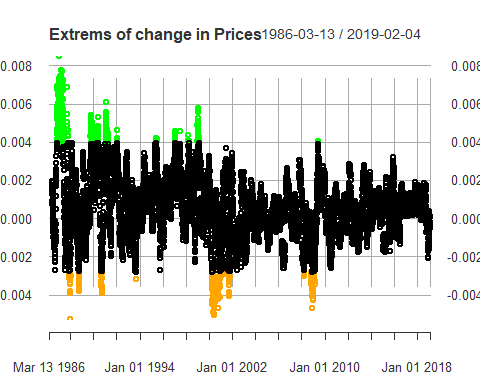
plot(MSFT$P\_xts$d.P\_MA, main = c("Change of MA of Price"))



plot.dens(dataset = MSFT$P\_xts$d.P\_MA, a = 2, title = c("Density of Change in Price"), plot.norm = TRUE)

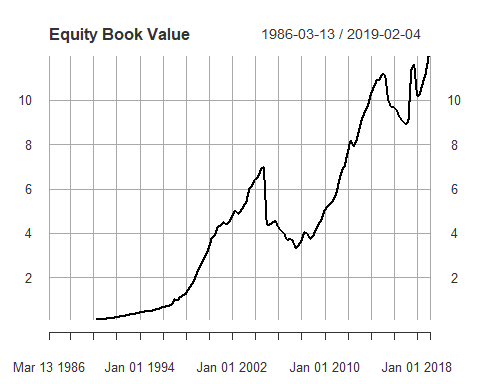


plot.ext(MSFT$P\_xts$d.P\_MA, a = 2, title = c("Extrems of change in Prices"))



Man beachte hier, dass der MA ein backwardsfacing rolling mean ist. Ähnliches geschieht nun mit den Buchwertdaten (BV) und den Einkommensdaten (NI = Net Income). Auch diese werden wieder in getrennten xts-Objekten abgespeichert. Man beachte, dass die Buchwertdaten jetzt erst von ihren “Lücken” bereinigt werden. Nicht zugewiesene Beobachtungen werden Interpoliert.

#BV---------------------------------------------------------------------------  
MSFT$BV\_xts <- xts(MSFT$Data$BPS\_E, order.by = MSFT$Data$Date)  
colnames(MSFT$BV\_xts) <- c("BPS\_E")  
MSFT$BV\_xts$BPS\_E[MSFT$BV\_xts$BPS\_E == 0] <- NA  
MSFT$BV\_xts$BPS\_E[1] <- 0  
MSFT$BV\_xts$BPS\_E[BV\_start\_date] <- 0  
MSFT$BV\_xts$BPS\_E <- xts(na.interp(MSFT$BV\_xts$BPS\_E), order.by = MSFT$Data$Date)  
MSFT$BV\_xts$BPS\_E[BV\_start\_date] <- NA  
  
MSFT$BV\_xts$d.BPS\_E <- xts(MSFT$Data$d.BPS\_E, order.by = MSFT$Data$Date)  
MSFT$BV\_xts$d.EPS\_ttm <- xts(MSFT$Data$d.EPS\_ttm, order.by = MSFT$Data$Date)  
#MSFT$BV\_xts$d.BPS\_E[is.nan(MSFT$BV\_xts$d.BPS\_E)] <- NA  
#MSFT$BV\_xts$d.BPS\_E[is.na(MSFT$BV\_xts$d.BPS\_E)] <- 0  
#MSFT$BV\_xts$d.BPS\_E[is.infinite(MSFT$BV\_xts$d.BPS\_E)] <- NA  
  
MSFT$BV\_xts$BPS\_D <- xts(MSFT$Data$BPS\_D, order.by = MSFT$Data$Date)  
MSFT$BV\_xts$BPS\_D[MSFT$BV\_xts$BPS\_D == 0] <- NA  
MSFT$BV\_xts$BPS\_D[1] <- 0  
MSFT$BV\_xts$BPS\_D[BV\_start\_date] <- 0  
MSFT$BV\_xts$BPS\_D <- xts(na.interp(MSFT$BV\_xts$BPS\_D), order.by = MSFT$Data$Date)  
  
plot(MSFT$BV\_xts$BPS\_E, main = c("Equity Book Value"))



#NI---------------------------------------------------------------------------  
MSFT$EPS\_xts <- xts(MSFT$Data$EPS, order.by = MSFT$Data$Date)  
colnames(MSFT$EPS\_xts) <- c("EPS")  
MSFT$EPS\_xts$EPS[MSFT$EPS\_xts$EPS <= 0] <- NA  
  
MSFT$EPS\_xts$EPS\_ttm <- xts(MSFT$Data$EPS\_ttm, order.by = MSFT$Data$Date)  
MSFT$EPS\_xts$EPS\_ttm[MSFT$EPS\_xts$EPS\_ttm <= 0] <- NA

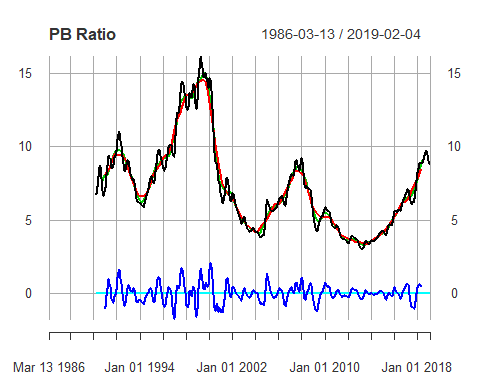
### Multiples

#### P/B Ratio

Kommen wir nun zur Berechnung der Multiples. Begonnen wird mit dem P/B Ratio:

ist dabei der BPS des Equity (BPS\_E).

MSFT$Ratios.PB <- xts(MSFT$P\_xts$MA / MSFT$BV\_xts$BPS\_E, order.by = MSFT$Data$Date)  
colnames(MSFT$Ratios.PB) <- c("PB")  
MSFT$Ratios.PB$PB[MSFT$Ratios.PB$PB == Inf] <- 0  
  
MSFT$Ratios.PB$MA <- xts(ma(MSFT$Ratios.PB$PB, 540), order.by = MSFT$Data$Date)  
#MSFT$Ratios.PB$MA <- xts(c(rep(NA, 539), rollmeanr(MSFT$Ratios.PB$P, 540)), order.by = MSFT$Data$Date)  
MSFT$Ratios.PB$MA2 <- xts(ma(MSFT$Ratios.PB$PB, 360), order.by = MSFT$Data$Date)  
#MSFT$Ratios.PB$MA2 <- xts(c(rep(NA, 99), rollmeanr(MSFT$Ratios.PB$PB, 100)), order.by = MSFT$Data$Date)  
  
MSFT$Ratios.PB$NivCleanded <- MSFT$Ratios.PB$PB - MSFT$Ratios.PB$MA  
MSFT$Ratios.PB$d.PB <- (MSFT$Ratios.PB$PB - lag(MSFT$Ratios.PB$PB, 1))/lag(MSFT$Ratios.PB$PB, 1)  
MSFT$Ratios.PB$d.PB[is.nan(MSFT$Ratios.PB$d.PB)] <- NA  
MSFT$Ratios.PB$d.PB[is.infinite(MSFT$Ratios.PB$d.PB)] <- NA  
  
plot(MSFT$Ratios.PB, main = c("PB Ratio"))



Hier lässt sich leider kein rolling mean verwenden, sondern nur der symetrische MA. Das macht eine Prognose des Modells vorerst unmöglich. Evtl. bräuchte man hier noch eine bessere Idee um das Niveau des PBs zu bestimmen. Alternative Methodiken die bereits getestet wurden:

* Polynomialer Trend
* ARMA mit sehr kleinen Parametern
* rolling mean

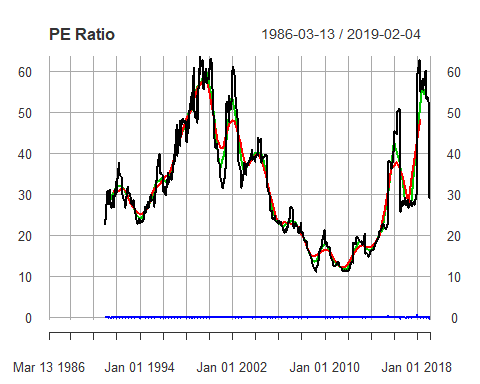
Alternativen die noch getested werden müssen:

* eine kombination von mehreren rolling means, mit unterschiedlichen längen, um kurz und langfristige Trends im Niveau ab zu fangen
* Konstrucktion eines Schwingugnsintervalls mithilfe der verteilung der Änderungen
* ein Stochastisches Modell, welches jede Änderung mit einer Wahrscheinlichkeit gewichted

A.d.A.: Mir gefällt der MA an der Stelle sowieso noch nicht. Extrem hohe Werte des PBs werden zwar unterschätzt (und gleichzeitig extrem niedrige Werte überschätzt), was so auch beabsichtig ist. Der MA (egal welcher Art) hat allerdings keine Strafe für extreme, niveautechnische Ausreißer (also besonders hohe und besonders niedrige Werte). Was ich damit meine ist, dass eine Funktion gesucht ist, die (eine umso stärkere dämpfung hat)in sachen Anpassung) um so stärker gedämpft ist, je höher ihr aktuelles Niveau ist. Um einen solchen Effekt, mit einem MA, zu Simulieren, müsste man eine noch viel größere Breite des MAs benutzen als ich es getan habe.

#### P/E Ratio

MSFT$Ratios.PE <- xts((MSFT$P\_xts$MA + MSFT$BV\_xts$BPS\_D)/MSFT$EPS\_xts$EPS\_ttm, order.by = MSFT$Data$Date)  
colnames(MSFT$Ratios.PE) <- c("PE")  
MSFT$Ratios.PE$PE[MSFT$Ratios.PE$PE == Inf] <- 0  
  
#MSFT$Ratios.PE$MA <- xts(c(rep(NA, 199), rollmeanr(MSFT$Ratios.PE$PE, 200)), order.by = MSFT$Data$Date)  
MSFT$Ratios.PE$MA <- xts(ma(MSFT$Ratios.PE$PE, 540), order.by = MSFT$Data$Date)  
#MSFT$Ratios.PE$MA2 <- xts(c(rep(NA, 99), rollmeanr(MSFT$Ratios.PE$PE, 100)), order.by = MSFT$Data$Date)  
MSFT$Ratios.PE$MA2 <- xts(ma(MSFT$Ratios.PE$PE, 360), order.by = MSFT$Data$Date)  
  
  
MSFT$Ratios.PE$d.PE <- (MSFT$Ratios.PE$PE - lag(MSFT$Ratios.PE$PE, 1))/lag(MSFT$Ratios.PE$PE, 1)  
MSFT$Ratios.PE$d.PE[is.nan(MSFT$Ratios.PE$d.PE)] <- NA  
MSFT$Ratios.PE$d.PE[is.infinite(MSFT$Ratios.PE$d.PE)] <- NA  
  
plot(MSFT$Ratios.PE, main = c("PE Ratio"))



## 3) Aufstellen des ersten Modells

### Univariate Regression

#### Berechnung

Modell:

mit .

ist dabei die Veränderung des 90 Tage MAs des Preises (Berechnung siehe oben in der Datenvorbereitung).

Das Modell wird für alle geschätzt. Dabei werden die Kennzahlen AIC und BIC aufgezeichnet. Zur Bestimmung des optimalen Modells, wird das i mit dem geringsten AIC bzw. BIC selektiert.

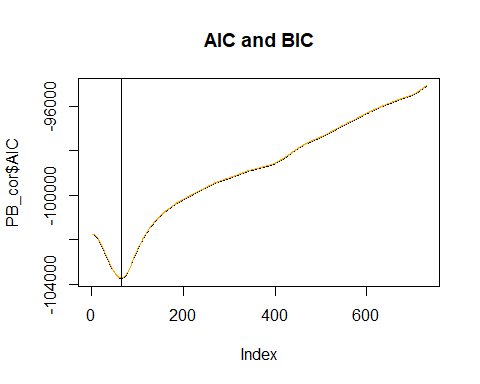
MSFT$NivClean\_Ind <- index(MSFT$Ratios.PB$NivCleanded[!is.na(MSFT$Ratios.PB$NivCleanded)])  
  
PB\_cor <- data.frame(matrix(NA, nrow = 730, ncol = 2))  
colnames(PB\_cor) <- c("AIC", "BIC")  
  
data.xts <- xts(MSFT$Ratios.PB$NivCleanded[MSFT$NivClean\_Ind], order.by = MSFT$NivClean\_Ind)  
colnames(data.xts) <- c("PB")  
data.xts$P <- MSFT$P\_xts$d.P\_MA[MSFT$NivClean\_Ind]  
  
for(i in c(1:730)){  
 model <- lm(data.xts$P ~ lag(data.xts$PB, i))  
 PB\_cor$AIC[i] <- AIC(model)  
 PB\_cor$BIC[i] <- BIC(model)  
}  
  
  
(best\_i <- order(PB\_cor$AIC)[1])

## [1] 65

(order(PB\_cor$BIC)[1])

## [1] 65

par(mfrow = c(1, 1))  
plot(PB\_cor$AIC, type = c("l"), main = c("AIC and BIC"))  
lines(PB\_cor$BIC, col = c("orange"))  
abline(v = best\_i)



Anschließend wird das Modell für das beste i neu berechnet:

data.xts$PB <- lag(MSFT$Ratios.PB$NivCleanded[MSFT$NivClean\_Ind], 65)  
model <- lm(data.xts$P ~ data.xts$PB, na.action = na.exclude)  
summary(model)

##   
## Call:  
## lm(formula = data.xts$P ~ data.xts$PB, na.action = na.exclude)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.0062531 -0.0008172 0.0000374 0.0008148 0.0044778   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 4.353e-04 1.310e-05 33.22 <2e-16 \*\*\*  
## data.xts$PB -1.125e-03 2.206e-05 -50.97 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.001306 on 9932 degrees of freedom  
## (65 observations deleted due to missingness)  
## Multiple R-squared: 0.2073, Adjusted R-squared: 0.2072   
## F-statistic: 2598 on 1 and 9932 DF, p-value: < 2.2e-16

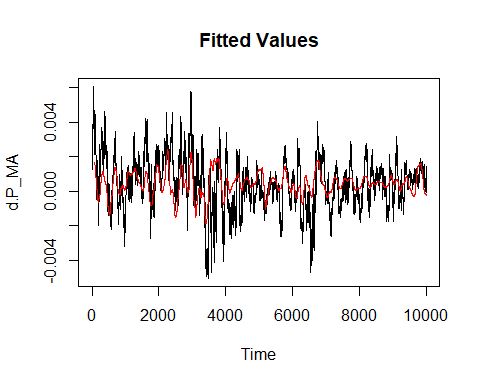
AIC(model)

## [1] -103743.1

BIC(model)

## [1] -103721.5

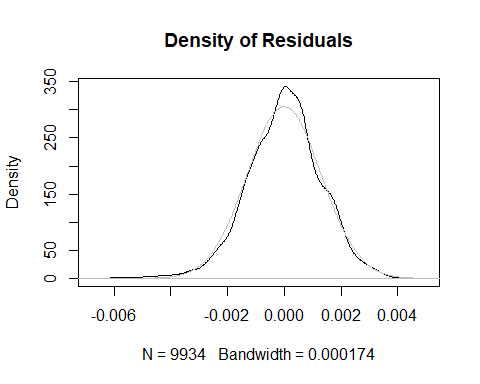
plot(ts(MSFT$P\_xts$d.P\_MA[MSFT$NivClean\_Ind]), main = c("Fitted Values"))  
lines(c(rep(NA, 65), ts(model$fitted.values)), col = c("red"))



#### Betrachtung der Residuen

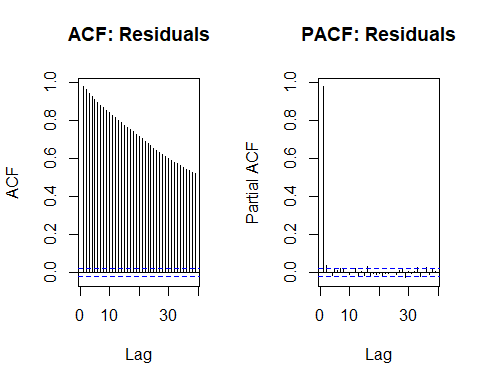
Kurze Betrachtung der Residuen des Modells:

plot.dens(as.numeric(model$residuals), title = c("Density of Residuals"), plot.norm = TRUE, plot.lines = FALSE)



Wie man sieht, sind sie nicht perfekt normal verteilt. Ein weiteres Problem, welches sich häufig bei Finanzdaten ergibt, sind Autokorrelationen in den Residuen.

par(mfrow = c(1, 2))  
Acf(as.numeric(model$residuals[!is.na(model$residuals)]), main = c("ACF: Residuals"))  
Pacf(as.numeric(model$residuals[!is.na(model$residuals)]), main = c("PACF: Residuals"))



par(mfrow = c(1, 1))

Die langsam abklingende ACF kann auf zwei Dinge hindeuten: einen AR-Prozess oder einen Random Walk Prozess. Im folgenden werden die beiden Möglichkeiten mithilfe von auto.arima() verglichen. A.d.A.: Zuvor hatten wir ja, neben der abklingenden ACF, auch noch schwingende Strukturen in der PACF gesehen. Diesen bin ich Herr geworden, indem ich statt einem normalen MA, einen backwardsfacing rolling mean verwendet habe.

(model.autoarima <- auto.arima(ts(model$residuals[!is.na(model$residuals)])))

## Series: ts(model$residuals[!is.na(model$residuals)])   
## ARIMA(0,1,1)   
##   
## Coefficients:  
## ma1  
## -0.0521  
## s.e. 0.0101  
##   
## sigma^2 estimated as 6.55e-08: log likelihood=68058.2  
## AIC=-136112.4 AICc=-136112.4 BIC=-136098

Die auto.arima() Funktion testet mithilfe des ADF Test ob eine Einheitswurzel vorliegt. Nachdem es einen ARIMA(0, 1, 1) Prozess vorschlägt, liegt die Vermutung eines Random Walks sehr nahe. Trotzdem wird als zweiter Ansatz nun ein AR Prozess gefitted, indem auto.arima dazu gezwungen wird ein ARMA Modell zu fitten.

(model.stationary <- auto.arima(ts(model$residuals[!is.na(model$residuals)]), d = 0))

## Series: ts(model$residuals[!is.na(model$residuals)])   
## ARIMA(2,0,0) with zero mean   
##   
## Coefficients:  
## ar1 ar2  
## 0.9395 0.0419  
## s.e. 0.0100 0.0100  
##   
## sigma^2 estimated as 6.494e-08: log likelihood=68106.27  
## AIC=-136206.5 AICc=-136206.5 BIC=-136184.9

Die auto.arima() Funktion schlägt in diesem Fall ein ARMA(2, 0) oder ein AR(2) Prozess vor. Beide Möglichkeiten werden im folgenden nun genauer untersucht.

### ADL

Als ersten Versuch, wurde ein ADL Modell mit einem AR(2) Prozess in den Residuen hergeleitet.

Nach dem Umformen ergibt dies ein ADL(2, 2) Modell:

mit

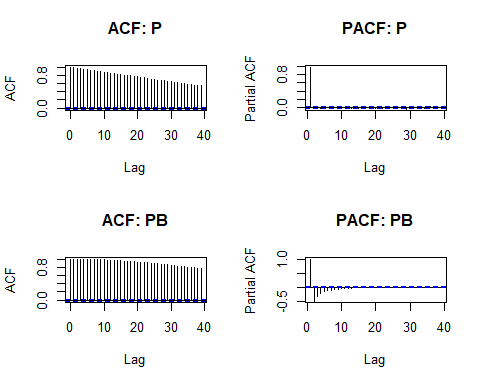
und

, , , , , .

#### Vorraussetzungen:

Damit das ADL Modell gefitted werden kann, muss zuerst sichergestellt werden, dass die Variablen Stationär sind. Hierzu wird der ADF - Test, sowie die ACF Funktion heran gezogen. Es wird die adf.test() Funktion aus dem aTSA Packet verwendet.

MSFT$NivClean\_Ind <- index(MSFT$Ratios.PB$NivCleanded[!is.na(MSFT$Ratios.PB$NivCleanded)])  
  
AIC\_BIC <- data.frame(matrix(NA, nrow = 730, ncol = 2))  
colnames(AIC\_BIC) <- c("AIC", "BIC")  
  
data.xts <- xts(MSFT$Ratios.PB$NivCleanded[MSFT$NivClean\_Ind], order.by = MSFT$NivClean\_Ind)  
colnames(data.xts) <- c("PB")  
data.xts$P <- MSFT$P\_xts$d.P\_MA[MSFT$NivClean\_Ind]  
  
#Test for the stationary requirement of P and PB  
par(mfrow = c(2, 2))  
acf(data.xts$P, main = c("ACF: P"))  
pacf(data.xts$P, main = c("PACF: P"))  
acf(data.xts$PB, main = c("ACF: PB"))  
pacf(data.xts$PB, main = c("PACF: PB"))



Die Plots legen nahe, dass keine Stationarität vorliegt.

adf.test(data.xts$P)

## Augmented Dickey-Fuller Test   
## alternative: stationary   
##   
## Type 1: no drift no trend   
## lag ADF p.value  
## [1,] 0 8.21 0.99  
## [2,] 1 8.60 0.99  
## [3,] 2 8.71 0.99  
## [4,] 3 8.62 0.99  
## [5,] 4 8.85 0.99  
## [6,] 5 9.04 0.99  
## [7,] 6 9.24 0.99  
## [8,] 7 9.36 0.99  
## [9,] 8 9.39 0.99  
## [10,] 9 9.38 0.99  
## [11,] 10 9.44 0.99  
## [12,] 11 9.63 0.99  
## Type 2: with drift no trend   
## lag ADF p.value  
## [1,] 0 8.61 0.99  
## [2,] 1 9.03 0.99  
## [3,] 2 9.14 0.99  
## [4,] 3 9.04 0.99  
## [5,] 4 9.29 0.99  
## [6,] 5 9.49 0.99  
## [7,] 6 9.70 0.99  
## [8,] 7 9.83 0.99  
## [9,] 8 9.86 0.99  
## [10,] 9 9.85 0.99  
## [11,] 10 9.92 0.99  
## [12,] 11 10.12 0.99  
## Type 3: with drift and trend   
## lag ADF p.value  
## [1,] 0 8.76 0.99  
## [2,] 1 9.19 0.99  
## [3,] 2 9.29 0.99  
## [4,] 3 9.20 0.99  
## [5,] 4 9.45 0.99  
## [6,] 5 9.64 0.99  
## [7,] 6 9.86 0.99  
## [8,] 7 9.99 0.99  
## [9,] 8 10.03 0.99  
## [10,] 9 10.02 0.99  
## [11,] 10 10.09 0.99  
## [12,] 11 10.30 0.99  
## ----   
## Note: in fact, p.value = 0.01 means p.value <= 0.01

Die Nullhypothese lässt sich nicht ablehnen (Ich hoffe ich habe das richtig Interpretiert).

Das selbe, wird nun für PB durchgeführt.

adf.test(data.xts$PB)

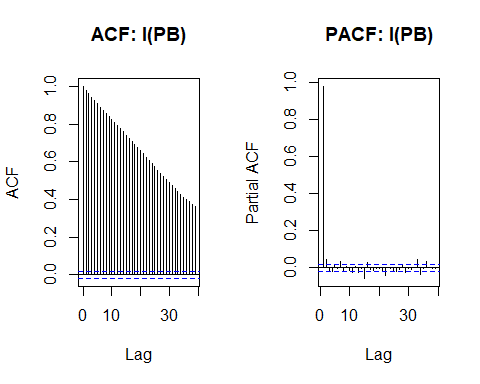
## Augmented Dickey-Fuller Test   
## alternative: stationary   
##   
## Type 1: no drift no trend   
## lag ADF p.value  
## [1,] 0 0.272 0.722  
## [2,] 1 -9.499 0.010  
## [3,] 2 -10.051 0.010  
## [4,] 3 -9.899 0.010  
## [5,] 4 -9.752 0.010  
## [6,] 5 -9.995 0.010  
## [7,] 6 -9.999 0.010  
## [8,] 7 -10.416 0.010  
## [9,] 8 -10.326 0.010  
## [10,] 9 -10.460 0.010  
## [11,] 10 -10.376 0.010  
## [12,] 11 -10.160 0.010  
## Type 2: with drift no trend   
## lag ADF p.value  
## [1,] 0 0.273 0.976  
## [2,] 1 -9.499 0.010  
## [3,] 2 -10.050 0.010  
## [4,] 3 -9.898 0.010  
## [5,] 4 -9.751 0.010  
## [6,] 5 -9.994 0.010  
## [7,] 6 -9.999 0.010  
## [8,] 7 -10.416 0.010  
## [9,] 8 -10.325 0.010  
## [10,] 9 -10.459 0.010  
## [11,] 10 -10.376 0.010  
## [12,] 11 -10.159 0.010  
## Type 3: with drift and trend   
## lag ADF p.value  
## [1,] 0 0.238 0.99  
## [2,] 1 -9.506 0.01  
## [3,] 2 -10.059 0.01  
## [4,] 3 -9.907 0.01  
## [5,] 4 -9.760 0.01  
## [6,] 5 -10.004 0.01  
## [7,] 6 -10.009 0.01  
## [8,] 7 -10.426 0.01  
## [9,] 8 -10.336 0.01  
## [10,] 9 -10.470 0.01  
## [11,] 10 -10.387 0.01  
## [12,] 11 -10.170 0.01  
## ----   
## Note: in fact, p.value = 0.01 means p.value <= 0.01

Hier muss die Nullhypothese des ADF Test abgelehnt werden, es liegt also keine Stationarität vor. PB muss (mindestens) ein mal differenziert werden.

data.xts$IPB <- data.xts$PB-lag(data.xts$PB, 1)  
adf.test(data.xts$IPB)

## Augmented Dickey-Fuller Test   
## alternative: stationary   
##   
## Type 1: no drift no trend   
## lag ADF p.value  
## [1,] 0 10.1 0.99  
## [2,] 1 10.6 0.99  
## [3,] 2 10.6 0.99  
## [4,] 3 10.5 0.99  
## [5,] 4 10.8 0.99  
## [6,] 5 10.9 0.99  
## [7,] 6 11.4 0.99  
## [8,] 7 11.4 0.99  
## [9,] 8 11.6 0.99  
## [10,] 9 11.6 0.99  
## [11,] 10 11.5 0.99  
## [12,] 11 11.6 0.99  
## Type 2: with drift no trend   
## lag ADF p.value  
## [1,] 0 10.1 0.99  
## [2,] 1 10.6 0.99  
## [3,] 2 10.6 0.99  
## [4,] 3 10.5 0.99  
## [5,] 4 10.8 0.99  
## [6,] 5 10.9 0.99  
## [7,] 6 11.4 0.99  
## [8,] 7 11.4 0.99  
## [9,] 8 11.6 0.99  
## [10,] 9 11.6 0.99  
## [11,] 10 11.5 0.99  
## [12,] 11 11.6 0.99  
## Type 3: with drift and trend   
## lag ADF p.value  
## [1,] 0 10.1 0.99  
## [2,] 1 10.6 0.99  
## [3,] 2 10.6 0.99  
## [4,] 3 10.5 0.99  
## [5,] 4 10.8 0.99  
## [6,] 5 10.9 0.99  
## [7,] 6 11.4 0.99  
## [8,] 7 11.4 0.99  
## [9,] 8 11.6 0.99  
## [10,] 9 11.6 0.99  
## [11,] 10 11.5 0.99  
## [12,] 11 11.6 0.99  
## ----   
## Note: in fact, p.value = 0.01 means p.value <= 0.01

par(mfrow = c(1, 2))  
acf(data.xts$IPB, na.action = na.exclude, main = c("ACF: I(PB)"))  
pacf(data.xts$IPB, na.action = na.exclude, main = c("PACF: I(PB)"))



Nach dem Differenzieren, wird der ADF Test erneut durchgeführt um sicherzustellen, dass Stationarität vorliegt. Die Nullhypothese kann an dieser Stelle nicht abgelehnt werden.

Als nächstes wird das Modell gefitted:

model <- lm(data.xts$P ~ data.xts$IPB + lag(data.xts$IPB, 1) + lag(data.xts$IPB, 2) + lag(data.xts$P, 1) + lag(data.xts$P, 2))  
summary(model)

##   
## Call:  
## lm(formula = data.xts$P ~ data.xts$IPB + lag(data.xts$IPB, 1) +   
## lag(data.xts$IPB, 2) + lag(data.xts$P, 1) + lag(data.xts$P,   
## 2))  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.0027193 -0.0000383 -0.0000013 0.0000354 0.0039712   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 3.863e-06 1.434e-06 2.694 0.00707 \*\*   
## data.xts$IPB 9.369e-02 5.557e-04 168.577 < 2e-16 \*\*\*  
## lag(data.xts$IPB, 1) -9.202e-02 1.207e-03 -76.229 < 2e-16 \*\*\*  
## lag(data.xts$IPB, 2) -8.986e-04 1.090e-03 -0.824 0.40974   
## lag(data.xts$P, 1) 9.779e-01 1.000e-02 97.763 < 2e-16 \*\*\*  
## lag(data.xts$P, 2) 1.334e-02 1.000e-02 1.334 0.18221   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.0001304 on 9990 degrees of freedom  
## (3 observations deleted due to missingness)  
## Multiple R-squared: 0.9923, Adjusted R-squared: 0.9923   
## F-statistic: 2.588e+05 on 5 and 9990 DF, p-value: < 2.2e-16

AIC(model)

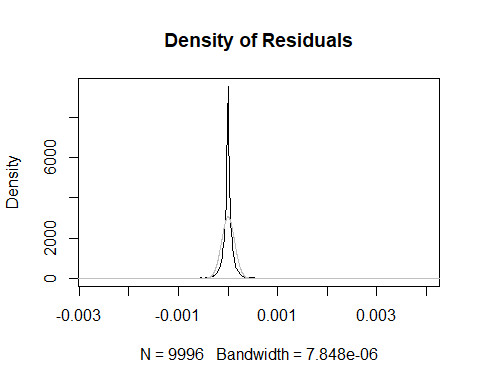
## [1] -150447.9

BIC(model)

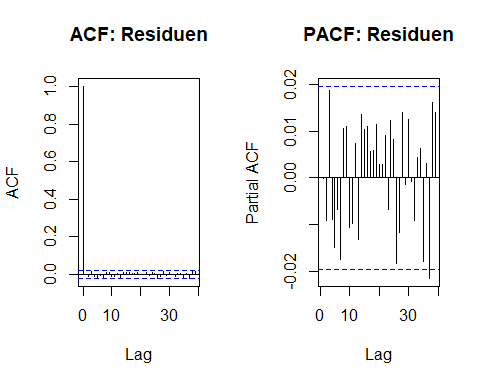
## [1] -150397.4

Man beachte das sehr hohe R-squared und die beiden Kennzahlen AIC und BIC, welche wesentlich niedriger sind als bei den Vorherigen Modellen. Es folgt die Betrachtung der Residuen.

plot.dens(as.numeric(model$residuals), title = c("Density of Residuals"), plot.norm = TRUE, plot.lines = FALSE)



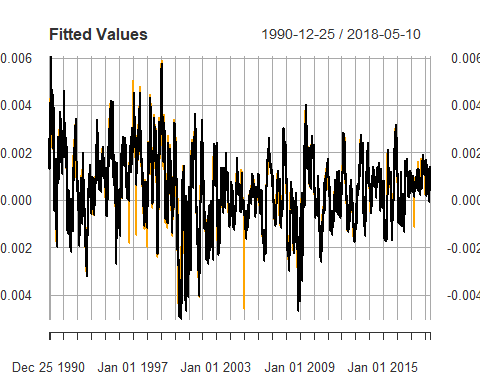
par(mfrow = c(1, 2))  
acf(model$residuals, main = c("ACF: Residuen"))  
pacf(model$residuals, main = c("PACF: Residuen"))



par(mfrow = c(1, 1))

Sie sind wieder nicht normal verteilt, die ACF und die PACF legen aber die Vermutung nahe, dass es keine vergessene Systematik in den Residuen gibt.

plot.data <- data.xts$P  
colnames(plot.data) <- c("P")  
plot.data$fit <- model$fitted.values  
plot(plot.data, type = c("l"), col = c("black", "orange"), main = c("Fitted Values"))



# Vorstellung der Funktionen aus functions.R

plot.dens <- function(dataset, a = 2, title, color = c("black", "black", "black"), plot.norm = FALSE, plot.lines = TRUE){  
 m <- mean(dataset, na.rm = TRUE)  
 s <- sd(dataset, na.rm = TRUE)  
 pp <- plot(density(dataset, na.rm = TRUE), main = title, col = color[1])  
 if(plot.lines){  
 pp <- abline(v = m + a\*s, col = color[2])  
 pp <- abline(v = m - a\*s, col = color[3])  
 }  
 if(plot.norm){  
 pp <- lines(density(rnorm(n = 1000000, mean = m, sd = s)), col = c("grey"))  
 }  
 invisible(pp)  
}

Diese Funktion plottet die Verteilung eines Datensatzes und zeichnet zwei Linien bei der a-fachen Standardabweichung ein.

plot.ext <- function(dataset, a = 2, title, color = c("black", "green", "orange")){  
 m <- mean(dataset, na.rm = TRUE)  
 s <- sd(dataset, na.rm = TRUE)  
 ind\_up <- index(dataset[dataset > (m + a \* s)])  
 ind\_down <- index(dataset[dataset < (m - a \* s)])  
 plot\_data <- dataset  
 colnames(plot\_data) <- c("a")  
 plot\_data$b <- dataset  
 plot\_data$c <- dataset  
 plot\_data$a[index(plot\_data) %in% ind\_up] <- NA  
 plot\_data$a[index(plot\_data) %in% ind\_down] <- NA  
 plot\_data$b[!index(plot\_data) %in% ind\_up] <- NA  
 plot\_data$c[!index(plot\_data) %in% ind\_down] <- NA  
 plot(plot\_data, col = color, type = c("p"), main = title)  
}

Diese Funktion plottet den Datensatz und markiert alle Werte mit einer Abweichung die größer ist als die a-fache Standardabweichung.

# Auswertung bei weiteren Firmen

## Apple

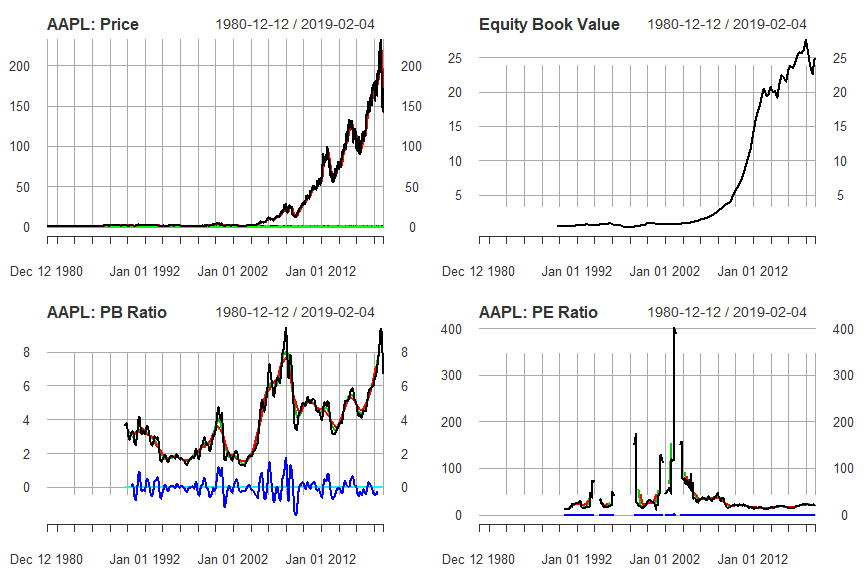
### Import

AAPL <- new.env()  
AAPL$MV <- read\_excel("Data\_Eikon/American\_Electronics/Apple.xlsx",   
 sheet = "Prices", col\_types = c("date",   
 "numeric"))  
  
AAPL$BV <- read\_excel("Data\_Eikon/American\_Electronics/Apple.xlsx",   
 sheet = "Data", col\_types = c("date",   
 "numeric", "numeric", "numeric",   
 "numeric", "numeric", "numeric", "numeric", "numeric"))  
P\_st\_da <- "1980-12-12"  
BV\_st\_da <- "/1989-09-28"  
day\_c <- 13934  
import(AAPL, P\_start\_date = P\_st\_da, BV\_start\_date = BV\_st\_da, count\_of\_days = day\_c)

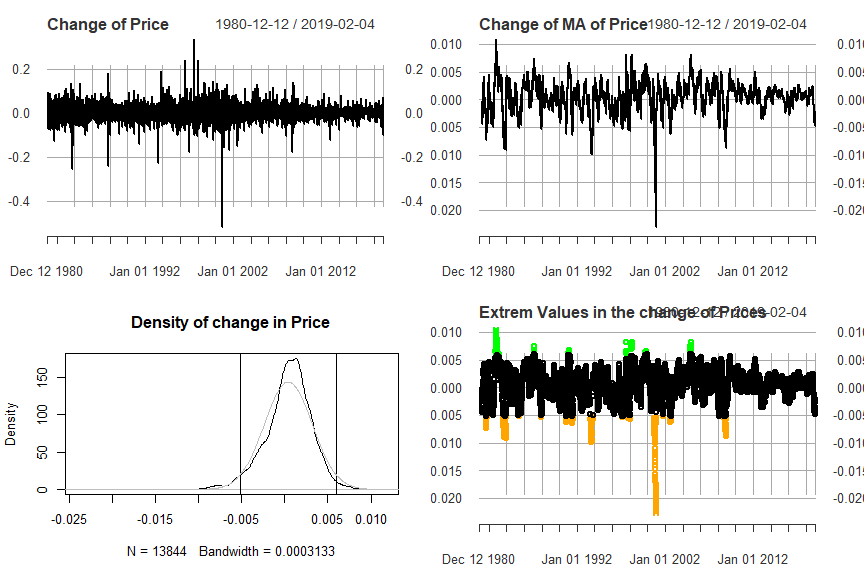
Die Funktion Import übernimmt den gesamten Import und die Vorbereitung der Daten. Die komplette Funktion wurde am Beispiel von Microsoft weiter oben durchgesprochen.

### Überblick

par(mfrow = c(2,2))  
plot(AAPL$P\_xts, col = c("black", "red", "green"), main = c("AAPL: Price"))  
plot(AAPL$BV\_xts$BPS\_E, main = c("Equity Book Value"))  
plot(AAPL$Ratios.PB, main = c("AAPL: PB Ratio"))  
plot(AAPL$Ratios.PE, main = c("AAPL: PE Ratio"))



plot(AAPL$P\_xts$d.P, main = c("Change of Price"))  
plot(AAPL$P\_xts$d.P\_MA, main = c("Change of MA of Price"))  
plot.dens(AAPL$P\_xts$d.P\_MA, a = 2, title = c("Density of change in Price"), plot.norm = TRUE)  
  
plot.ext(AAPL$P\_xts$d.P\_MA, a = 2, title = c("Extrem Values in the change of Prices"))



par(mfrow = c(1,1))

### Auswertung

#### Univariates Model

##### Berechnung:

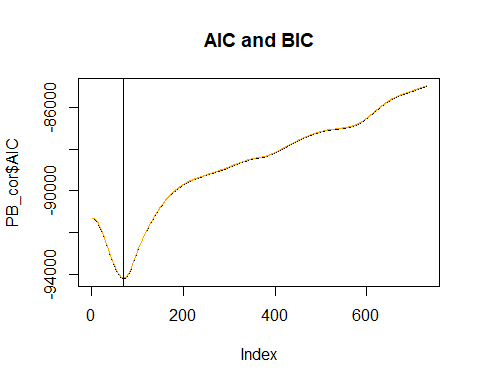
AAPL$NivClean\_Ind <- index(AAPL$Ratios.PB$NivCleanded[!is.na(AAPL$Ratios.PB$NivCleanded)])  
  
PB\_cor <- data.frame(matrix(NA, nrow = 730, ncol = 2))  
colnames(PB\_cor) <- c("AIC", "BIC")  
  
data.xts <- xts(AAPL$Ratios.PB$NivCleanded[AAPL$NivClean\_Ind], order.by = AAPL$NivClean\_Ind)  
colnames(data.xts) <- c("PB")  
data.xts$P <- AAPL$P\_xts$d.P\_MA[AAPL$NivClean\_Ind]  
  
#find the best fitting model  
for(i in c(1:730)){  
 model <- lm(data.xts$P ~ lag(data.xts$PB, i))  
 PB\_cor$AIC[i] <- AIC(model)  
 PB\_cor$BIC[i] <- BIC(model)  
}  
  
(best\_i <- order(PB\_cor$AIC)[1])

## [1] 70

(order(PB\_cor$BIC)[1])

## [1] 70

par(mfrow = c(1, 1))  
plot(PB\_cor$AIC, type = c("l"), main = c("AIC and BIC"))  
lines(PB\_cor$BIC, col = c("orange"))  
abline(v = best\_i)



Man beachte hier, dass das optimale i 70 ist und nicht wie davor 65. Es wird dennoch weiter mit 65 gerechnet.

data.xts$PB <- lag(AAPL$Ratios.PB$NivCleanded[AAPL$NivClean\_Ind], 65)  
model <- lm(data.xts$P ~ data.xts$PB, na.action = na.exclude)  
summary(model)

##   
## Call:  
## lm(formula = data.xts$P ~ data.xts$PB, na.action = na.exclude)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.0210247 -0.0011323 0.0002232 0.0013196 0.0080120   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 4.462e-04 2.294e-05 19.45 <2e-16 \*\*\*  
## data.xts$PB -3.118e-03 4.869e-05 -64.03 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.002307 on 10114 degrees of freedom  
## (65 observations deleted due to missingness)  
## Multiple R-squared: 0.2884, Adjusted R-squared: 0.2884   
## F-statistic: 4100 on 1 and 10114 DF, p-value: < 2.2e-16

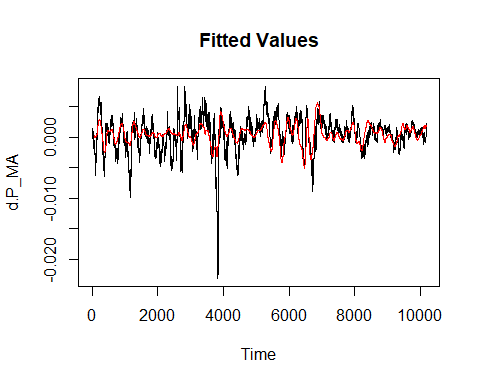
AIC(model)

## [1] -94130.8

BIC(model)

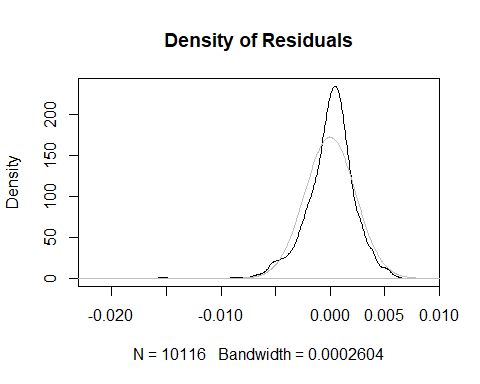
## [1] -94109.14

plot(ts(AAPL$P\_xts$d.P\_MA[AAPL$NivClean\_Ind]), main = c("Fitted Values"))  
lines(c(rep(NA, 65), ts(model$fitted.values)), col = c("red"))

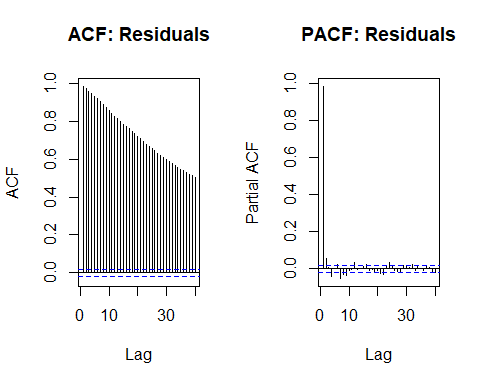


##### Betrachtung der Residuen:

plot.dens(as.numeric(model$residuals), title = c("Density of Residuals"), plot.norm = TRUE, plot.lines = FALSE)



par(mfrow = c(1, 2))  
Acf(as.numeric(model$residuals[!is.na(model$residuals)]), main = c("ACF: Residuals"))  
Pacf(as.numeric(model$residuals[!is.na(model$residuals)]), main = c("PACF: Residuals"))



par(mfrow = c(1, 1))  
  
(model.autoarima <- auto.arima(ts(model$residuals[!is.na(model$residuals)])))

## Series: ts(model$residuals[!is.na(model$residuals)])   
## ARIMA(2,1,2)   
##   
## Coefficients:  
## ar1 ar2 ma1 ma2  
## -0.5581 -0.6627 0.4988 0.6213  
## s.e. 0.0874 0.0692 0.0908 0.0723  
##   
## sigma^2 estimated as 1.597e-07: log likelihood=64799.52  
## AIC=-129589.1 AICc=-129589 BIC=-129552.9

(model.stationary <- auto.arima(ts(model$residuals[!is.na(model$residuals)]), d = 0))

## Series: ts(model$residuals[!is.na(model$residuals)])   
## ARIMA(1,0,1) with zero mean   
##   
## Coefficients:  
## ar1 ma1  
## 0.9866 -0.0566  
## s.e. 0.0016 0.0101  
##   
## sigma^2 estimated as 1.591e-07: log likelihood=64822.17  
## AIC=-129638.4 AICc=-129638.3 BIC=-129616.7

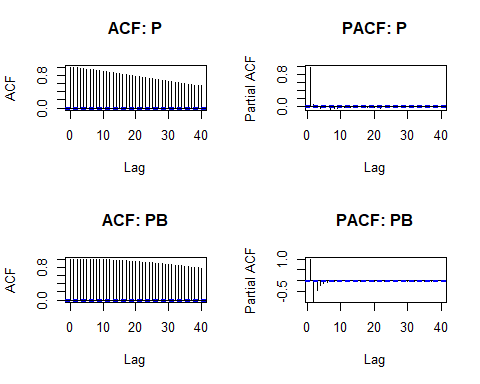
Auch hier ergibt sich wieder ein AR Prozess oder eine Einheitswurzel in den Residuen. Sie sind wieder nicht perfekt Normal verteilt.

#### ADL

##### Berechnung:

Es wird das selbe ADL Modell wie bei Microsoft benutzt.

AAPL$NivClean\_Ind <- index(AAPL$Ratios.PB$NivCleanded[!is.na(AAPL$Ratios.PB$NivCleanded)])  
  
AIC\_BIC <- data.frame(matrix(NA, nrow = 730, ncol = 2))  
colnames(AIC\_BIC) <- c("AIC", "BIC")  
  
data.xts <- xts(AAPL$Ratios.PB$NivCleanded[AAPL$NivClean\_Ind], order.by = AAPL$NivClean\_Ind)  
colnames(data.xts) <- c("PB")  
data.xts$P <- AAPL$P\_xts$d.P\_MA[AAPL$NivClean\_Ind]  
  
#Test for the stationary requirement of P and PB  
par(mfrow = c(2, 2))  
acf(data.xts$P, main = c("ACF: P"))  
pacf(data.xts$P, main = c("PACF: P"))  
acf(data.xts$PB, main = c("ACF: PB"))  
pacf(data.xts$PB, main = c("PACF: PB"))



adf.test(data.xts$P)

## Augmented Dickey-Fuller Test   
## alternative: stationary   
##   
## Type 1: no drift no trend   
## lag ADF p.value  
## [1,] 0 7.36 0.99  
## [2,] 1 7.76 0.99  
## [3,] 2 7.83 0.99  
## [4,] 3 7.53 0.99  
## [5,] 4 7.56 0.99  
## [6,] 5 7.74 0.99  
## [7,] 6 7.36 0.99  
## [8,] 7 7.12 0.99  
## [9,] 8 6.87 0.99  
## [10,] 9 6.82 0.99  
## [11,] 10 6.81 0.99  
## [12,] 11 7.10 0.99  
## Type 2: with drift no trend   
## lag ADF p.value  
## [1,] 0 7.46 0.99  
## [2,] 1 7.86 0.99  
## [3,] 2 7.94 0.99  
## [4,] 3 7.64 0.99  
## [5,] 4 7.67 0.99  
## [6,] 5 7.85 0.99  
## [7,] 6 7.46 0.99  
## [8,] 7 7.23 0.99  
## [9,] 8 6.98 0.99  
## [10,] 9 6.92 0.99  
## [11,] 10 6.92 0.99  
## [12,] 11 7.21 0.99  
## Type 3: with drift and trend   
## lag ADF p.value  
## [1,] 0 7.49 0.99  
## [2,] 1 7.90 0.99  
## [3,] 2 7.97 0.99  
## [4,] 3 7.67 0.99  
## [5,] 4 7.70 0.99  
## [6,] 5 7.88 0.99  
## [7,] 6 7.49 0.99  
## [8,] 7 7.26 0.99  
## [9,] 8 7.00 0.99  
## [10,] 9 6.95 0.99  
## [11,] 10 6.94 0.99  
## [12,] 11 7.23 0.99  
## ----   
## Note: in fact, p.value = 0.01 means p.value <= 0.01

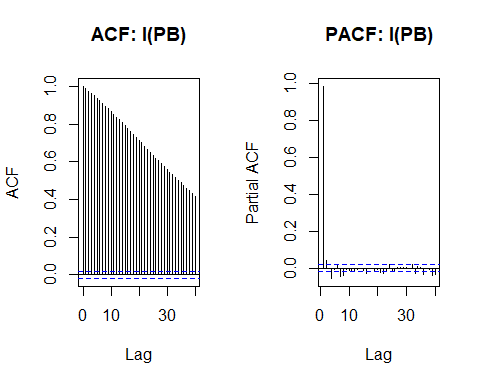
adf.test(data.xts$PB)

## Augmented Dickey-Fuller Test   
## alternative: stationary   
##   
## Type 1: no drift no trend   
## lag ADF p.value  
## [1,] 0 1.0 0.915  
## [2,] 1 -11.4 0.010  
## [3,] 2 -12.0 0.010  
## [4,] 3 -12.0 0.010  
## [5,] 4 -11.4 0.010  
## [6,] 5 -11.4 0.010  
## [7,] 6 -11.6 0.010  
## [8,] 7 -11.2 0.010  
## [9,] 8 -10.7 0.010  
## [10,] 9 -10.7 0.010  
## [11,] 10 -10.5 0.010  
## [12,] 11 -10.5 0.010  
## Type 2: with drift no trend   
## lag ADF p.value  
## [1,] 0 0.995 0.99  
## [2,] 1 -11.447 0.01  
## [3,] 2 -12.038 0.01  
## [4,] 3 -12.046 0.01  
## [5,] 4 -11.417 0.01  
## [6,] 5 -11.426 0.01  
## [7,] 6 -11.631 0.01  
## [8,] 7 -11.151 0.01  
## [9,] 8 -10.726 0.01  
## [10,] 9 -10.662 0.01  
## [11,] 10 -10.544 0.01  
## [12,] 11 -10.457 0.01  
## Type 3: with drift and trend   
## lag ADF p.value  
## [1,] 0 0.985 0.99  
## [2,] 1 -11.446 0.01  
## [3,] 2 -12.036 0.01  
## [4,] 3 -12.043 0.01  
## [5,] 4 -11.414 0.01  
## [6,] 5 -11.423 0.01  
## [7,] 6 -11.630 0.01  
## [8,] 7 -11.149 0.01  
## [9,] 8 -10.723 0.01  
## [10,] 9 -10.659 0.01  
## [11,] 10 -10.541 0.01  
## [12,] 11 -10.455 0.01  
## ----   
## Note: in fact, p.value = 0.01 means p.value <= 0.01

data.xts$IPB <- data.xts$PB-lag(data.xts$PB, 1)  
adf.test(data.xts$IPB)

## Augmented Dickey-Fuller Test   
## alternative: stationary   
##   
## Type 1: no drift no trend   
## lag ADF p.value  
## [1,] 0 8.28 0.99  
## [2,] 1 8.67 0.99  
## [3,] 2 8.69 0.99  
## [4,] 3 8.27 0.99  
## [5,] 4 8.30 0.99  
## [6,] 5 8.47 0.99  
## [7,] 6 8.15 0.99  
## [8,] 7 7.85 0.99  
## [9,] 8 7.84 0.99  
## [10,] 9 7.79 0.99  
## [11,] 10 7.77 0.99  
## [12,] 11 7.75 0.99  
## Type 2: with drift no trend   
## lag ADF p.value  
## [1,] 0 8.28 0.99  
## [2,] 1 8.67 0.99  
## [3,] 2 8.69 0.99  
## [4,] 3 8.27 0.99  
## [5,] 4 8.30 0.99  
## [6,] 5 8.47 0.99  
## [7,] 6 8.15 0.99  
## [8,] 7 7.85 0.99  
## [9,] 8 7.84 0.99  
## [10,] 9 7.79 0.99  
## [11,] 10 7.77 0.99  
## [12,] 11 7.75 0.99  
## Type 3: with drift and trend   
## lag ADF p.value  
## [1,] 0 8.28 0.99  
## [2,] 1 8.67 0.99  
## [3,] 2 8.69 0.99  
## [4,] 3 8.27 0.99  
## [5,] 4 8.30 0.99  
## [6,] 5 8.48 0.99  
## [7,] 6 8.15 0.99  
## [8,] 7 7.85 0.99  
## [9,] 8 7.84 0.99  
## [10,] 9 7.79 0.99  
## [11,] 10 7.77 0.99  
## [12,] 11 7.75 0.99  
## ----   
## Note: in fact, p.value = 0.01 means p.value <= 0.01

par(mfrow = c(1, 2))  
acf(data.xts$IPB, na.action = na.exclude, main = c("ACF: I(PB)"))  
pacf(data.xts$IPB, na.action = na.exclude, main = c("PACF: I(PB)"))



par(mfrow = c(1, 1))  
  
  
#Fitting of the Model  
model <- lm(data.xts$P ~ data.xts$IPB + lag(data.xts$IPB, 1) + lag(data.xts$IPB, 2) + lag(data.xts$P, 1) + lag(data.xts$P, 2))  
summary(model)

##   
## Call:  
## lm(formula = data.xts$P ~ data.xts$IPB + lag(data.xts$IPB, 1) +   
## lag(data.xts$IPB, 2) + lag(data.xts$P, 1) + lag(data.xts$P,   
## 2))  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.0029189 -0.0000449 -0.0000019 0.0000375 0.0088281   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 3.458e-06 2.195e-06 1.575 0.11519   
## data.xts$IPB 2.427e-01 1.511e-03 160.567 < 2e-16 \*\*\*  
## lag(data.xts$IPB, 1) -2.355e-01 3.199e-03 -73.615 < 2e-16 \*\*\*  
## lag(data.xts$IPB, 2) -5.042e-03 2.841e-03 -1.775 0.07601 .   
## lag(data.xts$P, 1) 9.636e-01 9.912e-03 97.224 < 2e-16 \*\*\*  
## lag(data.xts$P, 2) 2.863e-02 9.913e-03 2.889 0.00388 \*\*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.0002128 on 10172 degrees of freedom  
## (3 observations deleted due to missingness)  
## Multiple R-squared: 0.9939, Adjusted R-squared: 0.9939   
## F-statistic: 3.323e+05 on 5 and 10172 DF, p-value: < 2.2e-16

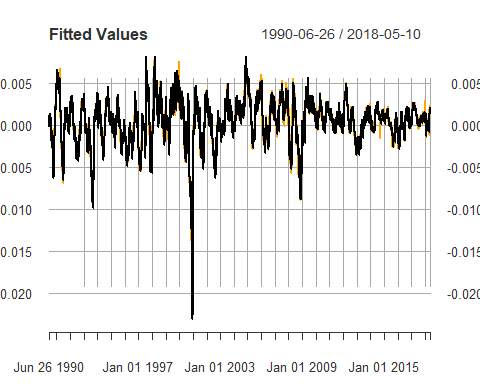
AIC(model)

## [1] -143218.5

BIC(model)

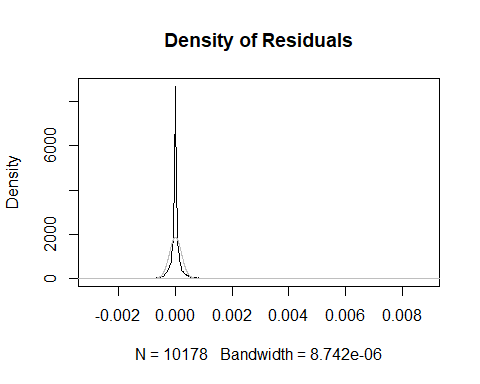
## [1] -143167.9

plot.data <- data.xts$P  
colnames(plot.data) <- c("P")  
plot.data$fit <- model$fitted.values  
plot(plot.data, type = c("l"), col = c("black", "orange"), main = c("Fitted Values"))

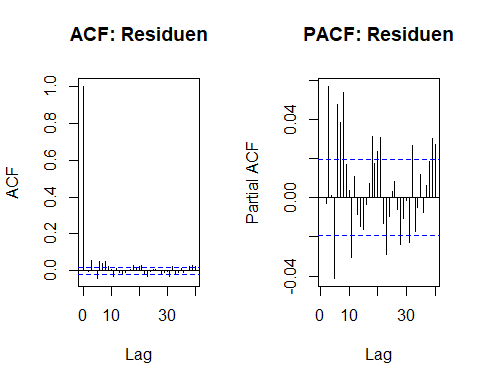


##### Betrachtung der Residuen:

plot.dens(as.numeric(model$residuals), title = c("Density of Residuals"), plot.norm = TRUE, plot.lines = FALSE)



par(mfrow = c(1, 2))  
acf(model$residuals, main = c("ACF: Residuen"))  
pacf(model$residuals, main = c("PACF: Residuen"))



par(mfrow = c(1, 1))

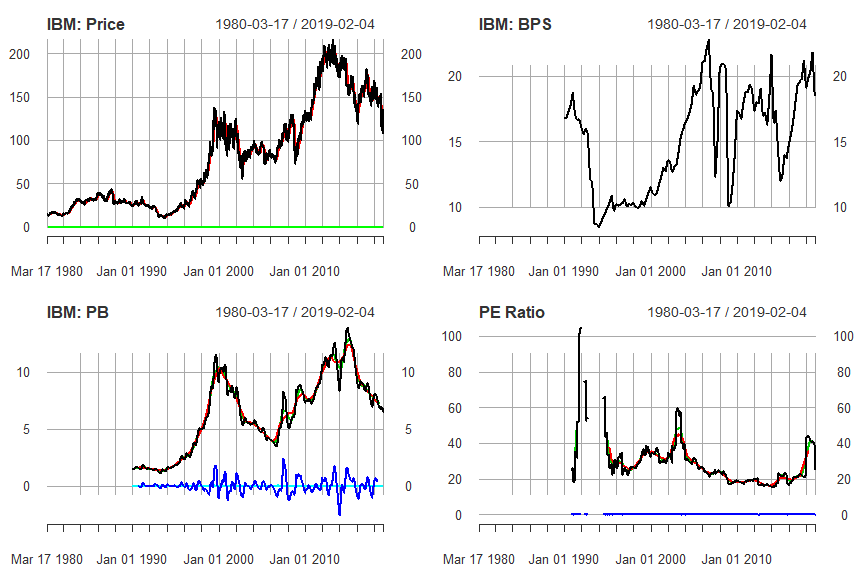
## IBM

### Import

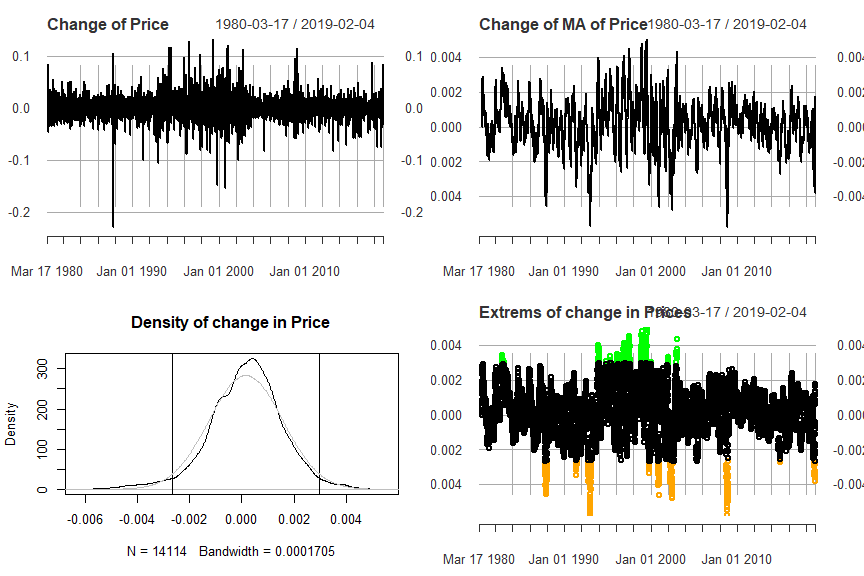
IBM <- new.env()  
  
IBM$MV <- read\_excel("Data\_Eikon/American\_Electronics/IBM.xlsx",   
 sheet = "Prices", col\_types = c("date",   
 "numeric"))  
  
IBM$BV <- read\_excel("Data\_Eikon/American\_Electronics/IBM.xlsx",   
 sheet = "Data", col\_types = c("date",   
 "numeric", "numeric", "numeric",   
 "numeric", "numeric", "numeric", "numeric", "numeric"))  
  
P\_st\_da <- "1980-03-17"  
BV\_st\_da <- "/1989-12-30"  
day\_c <- 14204  
import(IBM, P\_start\_date = P\_st\_da, BV\_start\_date = BV\_st\_da, count\_of\_days = day\_c)

### Überblick

par(mfrow = c(2,2))  
plot(IBM$P\_xts, col = c("black", "red", "green"), main = c("IBM: Price"))  
plot(IBM$BV\_xts$BPS\_E, main = c("IBM: BPS"))  
plot(IBM$Ratios.PB, main = c("IBM: PB"))  
plot(IBM$Ratios.PE, main = c("PE Ratio"))



plot(IBM$P\_xts$d.P, main = c("Change of Price"))  
plot(IBM$P\_xts$d.P\_MA, main = c("Change of MA of Price"))  
plot.dens(IBM$P\_xts$d.P\_MA, a = 2, title = c("Density of change in Price"), plot.norm = TRUE)  
  
plot.ext(IBM$P\_xts$d.P\_MA, a = 2, title = c("Extrems of change in Prices"))



par(mfrow = c(1,1))

### Auswertung

#### Univariates Model

##### Berechnung:

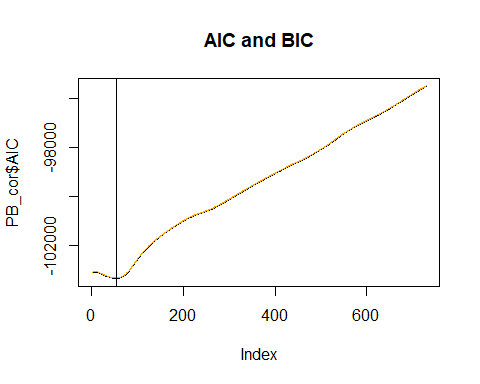
IBM$NivClean\_Ind <- index(IBM$Ratios.PB$NivCleanded[!is.na(IBM$Ratios.PB$NivCleanded)])  
  
PB\_cor <- data.frame(matrix(NA, nrow = 730, ncol = 2))  
colnames(PB\_cor) <- c("AIC", "BIC")  
  
data.xts <- xts(IBM$Ratios.PB$NivCleanded[IBM$NivClean\_Ind], order.by = IBM$NivClean\_Ind)  
colnames(data.xts) <- c("PB")  
data.xts$P <- IBM$P\_xts$d.P\_MA[IBM$NivClean\_Ind]  
  
#find the best fitting model  
for(i in c(1:730)){  
 model <- lm(data.xts$P ~ lag(data.xts$PB, i))  
 PB\_cor$AIC[i] <- AIC(model)  
 PB\_cor$BIC[i] <- BIC(model)  
}  
  
(best\_i <- order(PB\_cor$AIC)[1])

## [1] 54

(order(PB\_cor$BIC)[1])

## [1] 54

par(mfrow = c(1, 1))  
plot(PB\_cor$AIC, type = c("l"), main = c("AIC and BIC"))  
lines(PB\_cor$BIC, col = c("orange"))  
abline(v = best\_i)



#fitting of the best the model with the lowest AIC/BIC  
data.xts$PB <- lag(IBM$Ratios.PB$NivCleanded[IBM$NivClean\_Ind], 65)  
model <- lm(data.xts$P ~ data.xts$PB, na.action = na.exclude)  
summary(model)

##   
## Call:  
## lm(formula = data.xts$P ~ data.xts$PB, na.action = na.exclude)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.0064686 -0.0008355 0.0000257 0.0008363 0.0048858   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 1.750e-04 1.396e-05 12.54 <2e-16 \*\*\*  
## data.xts$PB -7.000e-04 2.337e-05 -29.95 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.001397 on 10021 degrees of freedom  
## (65 observations deleted due to missingness)  
## Multiple R-squared: 0.08218, Adjusted R-squared: 0.08208   
## F-statistic: 897.2 on 1 and 10021 DF, p-value: < 2.2e-16

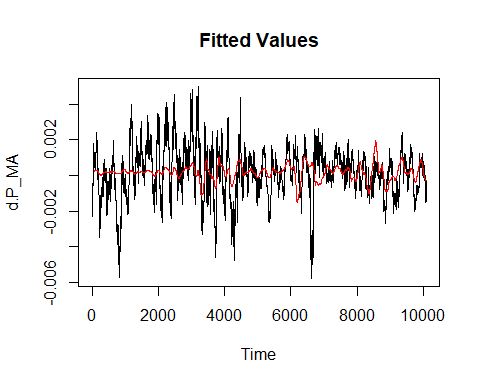
AIC(model)

## [1] -103316.3

BIC(model)

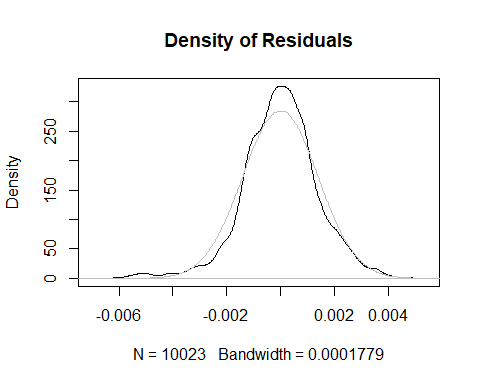
## [1] -103294.7

plot(ts(IBM$P\_xts$d.P\_MA[IBM$NivClean\_Ind]), main = c("Fitted Values"))  
lines(c(rep(NA, 65), ts(model$fitted.values)), col = c("red"))

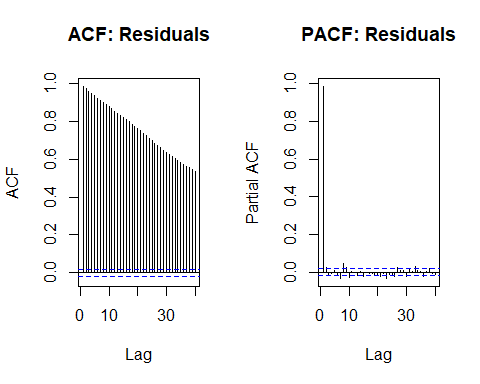


##### Betrachtung der Residuen:

plot.dens(as.numeric(model$residuals), title = c("Density of Residuals"), plot.norm = TRUE, plot.lines = FALSE)



par(mfrow = c(1, 2))  
Acf(as.numeric(model$residuals[!is.na(model$residuals)]), main = c("ACF: Residuals"))  
Pacf(as.numeric(model$residuals[!is.na(model$residuals)]), main = c("PACF: Residuals"))



par(mfrow = c(1, 1))  
  
(model.autoarima <- auto.arima(ts(model$residuals[!is.na(model$residuals)])))

## Series: ts(model$residuals[!is.na(model$residuals)])   
## ARIMA(0,1,1)   
##   
## Coefficients:  
## ma1  
## -0.0298  
## s.e. 0.0099  
##   
## sigma^2 estimated as 5.071e-08: log likelihood=69950.29  
## AIC=-139896.6 AICc=-139896.6 BIC=-139882.2

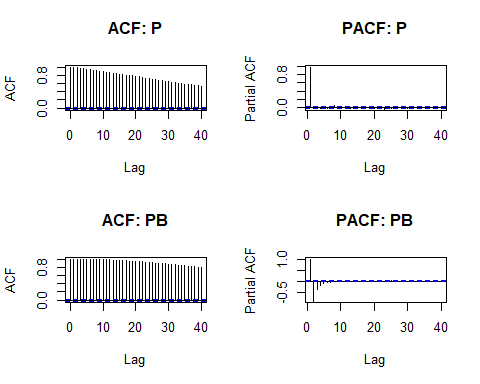
(model.stationary <- auto.arima(ts(model$residuals[!is.na(model$residuals)]), d = 0))

## Series: ts(model$residuals[!is.na(model$residuals)])   
## ARIMA(3,0,0) with zero mean   
##   
## Coefficients:  
## ar1 ar2 ar3  
## 0.9636 0.0409 -0.0175  
## s.e. 0.0100 0.0139 0.0100  
##   
## sigma^2 estimated as 5.039e-08: log likelihood=69988.38  
## AIC=-139968.8 AICc=-139968.8 BIC=-139939.9

#### ADL

##### Berechnung:

IBM$NivClean\_Ind <- index(IBM$Ratios.PB$NivCleanded[!is.na(IBM$Ratios.PB$NivCleanded)])  
  
AIC\_BIC <- data.frame(matrix(NA, nrow = 730, ncol = 2))  
colnames(AIC\_BIC) <- c("AIC", "BIC")  
  
data.xts <- xts(IBM$Ratios.PB$NivCleanded[IBM$NivClean\_Ind], order.by = IBM$NivClean\_Ind)  
colnames(data.xts) <- c("PB")  
data.xts$P <- IBM$P\_xts$d.P\_MA[IBM$NivClean\_Ind]  
  
#Test for the stationary requirement of P and PB  
par(mfrow = c(2, 2))  
acf(data.xts$P, main = c("ACF: P"))  
pacf(data.xts$P, main = c("PACF: P"))  
acf(data.xts$PB, main = c("ACF: PB"))  
pacf(data.xts$PB, main = c("PACF: PB"))



adf.test(data.xts$P)

## Augmented Dickey-Fuller Test   
## alternative: stationary   
##   
## Type 1: no drift no trend   
## lag ADF p.value  
## [1,] 0 7.72 0.99  
## [2,] 1 7.95 0.99  
## [3,] 2 7.86 0.99  
## [4,] 3 7.85 0.99  
## [5,] 4 7.99 0.99  
## [6,] 5 7.95 0.99  
## [7,] 6 7.76 0.99  
## [8,] 7 8.15 0.99  
## [9,] 8 8.37 0.99  
## [10,] 9 8.21 0.99  
## [11,] 10 8.28 0.99  
## [12,] 11 8.32 0.99  
## Type 2: with drift no trend   
## lag ADF p.value  
## [1,] 0 7.77 0.99  
## [2,] 1 8.00 0.99  
## [3,] 2 7.91 0.99  
## [4,] 3 7.90 0.99  
## [5,] 4 8.05 0.99  
## [6,] 5 8.01 0.99  
## [7,] 6 7.81 0.99  
## [8,] 7 8.21 0.99  
## [9,] 8 8.43 0.99  
## [10,] 9 8.26 0.99  
## [11,] 10 8.33 0.99  
## [12,] 11 8.38 0.99  
## Type 3: with drift and trend   
## lag ADF p.value  
## [1,] 0 7.76 0.99  
## [2,] 1 8.00 0.99  
## [3,] 2 7.91 0.99  
## [4,] 3 7.90 0.99  
## [5,] 4 8.05 0.99  
## [6,] 5 8.01 0.99  
## [7,] 6 7.81 0.99  
## [8,] 7 8.21 0.99  
## [9,] 8 8.43 0.99  
## [10,] 9 8.26 0.99  
## [11,] 10 8.33 0.99  
## [12,] 11 8.38 0.99  
## ----   
## Note: in fact, p.value = 0.01 means p.value <= 0.01

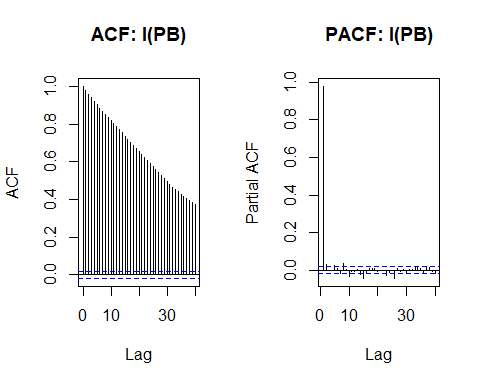
adf.test(data.xts$PB)

## Augmented Dickey-Fuller Test   
## alternative: stationary   
##   
## Type 1: no drift no trend   
## lag ADF p.value  
## [1,] 0 1.01 0.916  
## [2,] 1 -8.50 0.010  
## [3,] 2 -8.81 0.010  
## [4,] 3 -8.83 0.010  
## [5,] 4 -8.84 0.010  
## [6,] 5 -9.09 0.010  
## [7,] 6 -9.23 0.010  
## [8,] 7 -9.08 0.010  
## [9,] 8 -9.48 0.010  
## [10,] 9 -9.58 0.010  
## [11,] 10 -9.31 0.010  
## [12,] 11 -9.28 0.010  
## Type 2: with drift no trend   
## lag ADF p.value  
## [1,] 0 1.01 0.99  
## [2,] 1 -8.50 0.01  
## [3,] 2 -8.81 0.01  
## [4,] 3 -8.83 0.01  
## [5,] 4 -8.84 0.01  
## [6,] 5 -9.09 0.01  
## [7,] 6 -9.23 0.01  
## [8,] 7 -9.08 0.01  
## [9,] 8 -9.48 0.01  
## [10,] 9 -9.58 0.01  
## [11,] 10 -9.31 0.01  
## [12,] 11 -9.28 0.01  
## Type 3: with drift and trend   
## lag ADF p.value  
## [1,] 0 1.00 0.99  
## [2,] 1 -8.50 0.01  
## [3,] 2 -8.81 0.01  
## [4,] 3 -8.83 0.01  
## [5,] 4 -8.84 0.01  
## [6,] 5 -9.10 0.01  
## [7,] 6 -9.23 0.01  
## [8,] 7 -9.08 0.01  
## [9,] 8 -9.48 0.01  
## [10,] 9 -9.58 0.01  
## [11,] 10 -9.31 0.01  
## [12,] 11 -9.28 0.01  
## ----   
## Note: in fact, p.value = 0.01 means p.value <= 0.01

data.xts$IPB <- data.xts$PB-lag(data.xts$PB, 1)  
adf.test(data.xts$IPB)

## Augmented Dickey-Fuller Test   
## alternative: stationary   
##   
## Type 1: no drift no trend   
## lag ADF p.value  
## [1,] 0 10.5 0.99  
## [2,] 1 11.0 0.99  
## [3,] 2 11.1 0.99  
## [4,] 3 11.2 0.99  
## [5,] 4 11.6 0.99  
## [6,] 5 11.8 0.99  
## [7,] 6 11.8 0.99  
## [8,] 7 12.2 0.99  
## [9,] 8 12.4 0.99  
## [10,] 9 12.3 0.99  
## [11,] 10 12.3 0.99  
## [12,] 11 12.4 0.99  
## Type 2: with drift no trend   
## lag ADF p.value  
## [1,] 0 10.5 0.99  
## [2,] 1 11.0 0.99  
## [3,] 2 11.1 0.99  
## [4,] 3 11.2 0.99  
## [5,] 4 11.6 0.99  
## [6,] 5 11.8 0.99  
## [7,] 6 11.8 0.99  
## [8,] 7 12.2 0.99  
## [9,] 8 12.4 0.99  
## [10,] 9 12.3 0.99  
## [11,] 10 12.3 0.99  
## [12,] 11 12.4 0.99  
## Type 3: with drift and trend   
## lag ADF p.value  
## [1,] 0 10.5 0.99  
## [2,] 1 11.0 0.99  
## [3,] 2 11.1 0.99  
## [4,] 3 11.2 0.99  
## [5,] 4 11.6 0.99  
## [6,] 5 11.8 0.99  
## [7,] 6 11.8 0.99  
## [8,] 7 12.2 0.99  
## [9,] 8 12.4 0.99  
## [10,] 9 12.3 0.99  
## [11,] 10 12.3 0.99  
## [12,] 11 12.4 0.99  
## ----   
## Note: in fact, p.value = 0.01 means p.value <= 0.01

par(mfrow = c(1, 2))  
acf(data.xts$IPB, na.action = na.exclude, main = c("ACF: I(PB)"))  
pacf(data.xts$IPB, na.action = na.exclude, main = c("PACF: I(PB)"))



par(mfrow = c(1, 1))  
  
  
#Fitting of the Model  
model <- lm(data.xts$P ~ data.xts$IPB + lag(data.xts$IPB, 1) + lag(data.xts$IPB, 2) + lag(data.xts$P, 1) + lag(data.xts$P, 2))  
summary(model)

##   
## Call:  
## lm(formula = data.xts$P ~ data.xts$IPB + lag(data.xts$IPB, 1) +   
## lag(data.xts$IPB, 2) + lag(data.xts$P, 1) + lag(data.xts$P,   
## 2))  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.0024308 -0.0000540 -0.0000014 0.0000466 0.0060918   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 1.685e-06 1.741e-06 0.968 0.333   
## data.xts$IPB 6.293e-02 7.588e-04 82.932 <2e-16 \*\*\*  
## lag(data.xts$IPB, 1) -6.294e-02 1.224e-03 -51.431 <2e-16 \*\*\*  
## lag(data.xts$IPB, 2) 5.169e-04 9.845e-04 0.525 0.600   
## lag(data.xts$P, 1) 9.922e-01 9.961e-03 99.602 <2e-16 \*\*\*  
## lag(data.xts$P, 2) -1.603e-03 9.961e-03 -0.161 0.872   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.0001732 on 10079 degrees of freedom  
## (3 observations deleted due to missingness)  
## Multiple R-squared: 0.9859, Adjusted R-squared: 0.9859   
## F-statistic: 1.407e+05 on 5 and 10079 DF, p-value: < 2.2e-16

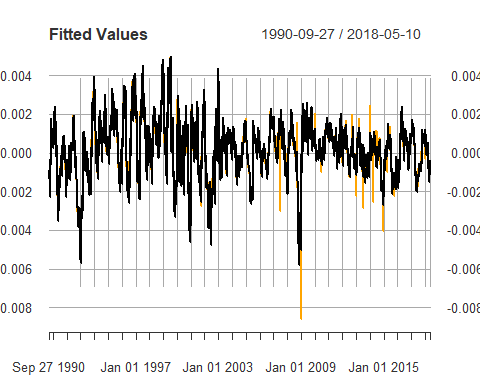
AIC(model)

## [1] -146060.6

BIC(model)

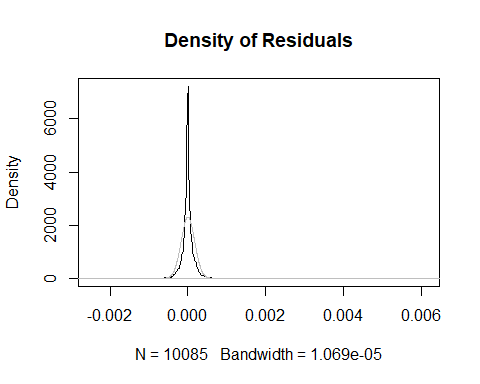
## [1] -146010.1

plot.data <- data.xts$P  
colnames(plot.data) <- c("P")  
plot.data$fit <- model$fitted.values  
plot(plot.data, type = c("l"), col = c("black", "orange"), main = c("Fitted Values"))

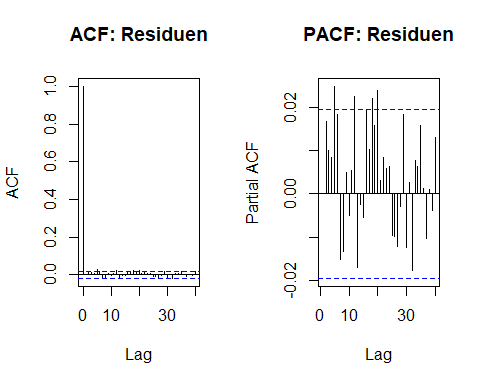


##### Betrachtung der Residuen:

plot.dens(as.numeric(model$residuals), title = c("Density of Residuals"), plot.norm = TRUE, plot.lines = FALSE)



par(mfrow = c(1, 2))  
acf(model$residuals, main = c("ACF: Residuen"))  
pacf(model$residuals, main = c("PACF: Residuen"))



par(mfrow = c(1, 1))

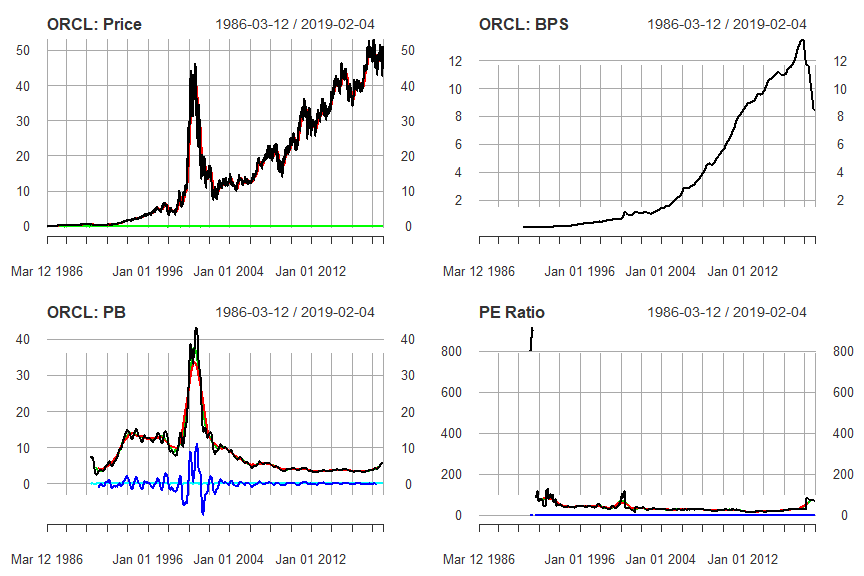
## Oracle

### Import

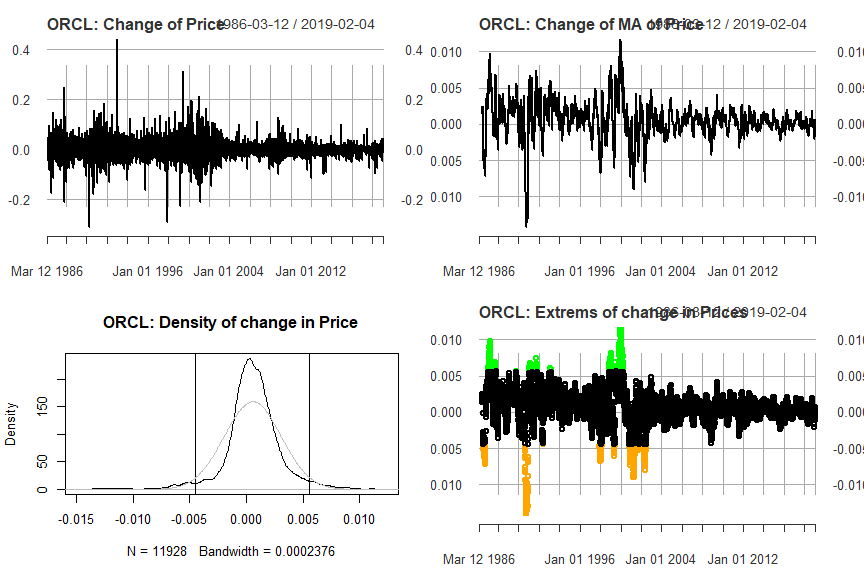
ORCL <- new.env()  
  
ORCL$MV <- read\_excel("Data\_Eikon/American\_Electronics/Oracle.xlsx",   
 sheet = "Prices", col\_types = c("date",   
 "numeric"))  
  
ORCL$BV <- read\_excel("Data\_Eikon/American\_Electronics/Oracle.xlsx",   
 sheet = "Data", col\_types = c("date",   
 "numeric", "numeric", "numeric",   
 "numeric", "numeric", "numeric", "numeric", "numeric"))  
  
P\_st\_da <- "1986-03-12"  
BV\_st\_da <- "/1990-05-30"  
day\_c <- 12018  
import(ORCL, P\_start\_date = P\_st\_da, BV\_start\_date = BV\_st\_da, count\_of\_days = day\_c)

### Überblick

par(mfrow = c(2,2))  
plot(ORCL$P\_xts, col = c("black", "red", "green"), main = c("ORCL: Price"))  
plot(ORCL$BV\_xts$BPS\_E, main = c("ORCL: BPS"))  
plot(ORCL$Ratios.PB, main = c("ORCL: PB"))  
plot(ORCL$Ratios.PE, main = c("PE Ratio"))



plot(ORCL$P\_xts$d.P, main = c("ORCL: Change of Price"))  
plot(ORCL$P\_xts$d.P\_MA, main = c("ORCL: Change of MA of Price"))  
plot.dens(ORCL$P\_xts$d.P\_MA, a = 2, title = c("ORCL: Density of change in Price"), plot.norm = TRUE)  
  
plot.ext(ORCL$P\_xts$d.P\_MA, a = 2, title = c("ORCL: Extrems of change in Prices"))



par(mfrow = c(1,1))

### Auswertung

#### Univariates Modell

##### Berechnung:

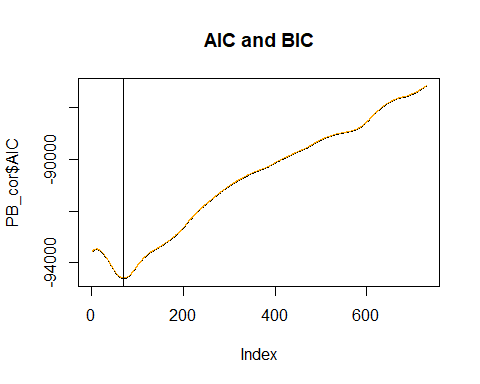
ORCL$NivClean\_Ind <- index(ORCL$Ratios.PB$NivCleanded[!is.na(ORCL$Ratios.PB$NivCleanded)])  
  
PB\_cor <- data.frame(matrix(NA, nrow = 730, ncol = 2))  
colnames(PB\_cor) <- c("AIC", "BIC")  
  
data.xts <- xts(ORCL$Ratios.PB$NivCleanded[ORCL$NivClean\_Ind], order.by = ORCL$NivClean\_Ind)  
colnames(data.xts) <- c("PB")  
data.xts$P <- ORCL$P\_xts$d.P\_MA[ORCL$NivClean\_Ind]  
  
#find the best fitting model  
for(i in c(1:730)){  
 model <- lm(data.xts$P ~ lag(data.xts$PB, i))  
 PB\_cor$AIC[i] <- AIC(model)  
 PB\_cor$BIC[i] <- BIC(model)  
}  
  
(best\_i <- order(PB\_cor$AIC)[1])

## [1] 70

(order(PB\_cor$BIC)[1])

## [1] 70

par(mfrow = c(1, 1))  
plot(PB\_cor$AIC, type = c("l"), main = c("AIC and BIC"))  
lines(PB\_cor$BIC, col = c("orange"))  
abline(v = best\_i)



#fitting of the best the model with the lowest AIC/BIC  
data.xts$PB <- lag(ORCL$Ratios.PB$NivCleanded[ORCL$NivClean\_Ind], 65)  
model <- lm(data.xts$P ~ data.xts$PB, na.action = na.exclude)  
summary(model)

##   
## Call:  
## lm(formula = data.xts$P ~ data.xts$PB, na.action = na.exclude)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.0080522 -0.0009832 -0.0000293 0.0009556 0.0088123   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 5.388e-04 2.023e-05 26.63 <2e-16 \*\*\*  
## data.xts$PB -4.422e-04 1.072e-05 -41.26 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.00201 on 9870 degrees of freedom  
## (65 observations deleted due to missingness)  
## Multiple R-squared: 0.1471, Adjusted R-squared: 0.147   
## F-statistic: 1703 on 1 and 9870 DF, p-value: < 2.2e-16

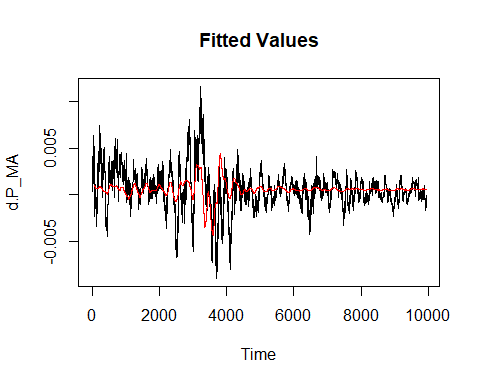
AIC(model)

## [1] -94584.2

BIC(model)

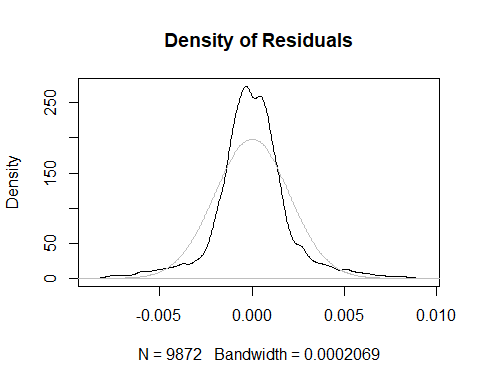
## [1] -94562.61

plot(ts(ORCL$P\_xts$d.P\_MA[ORCL$NivClean\_Ind]), main = c("Fitted Values"))  
lines(c(rep(NA, 65), ts(model$fitted.values)), col = c("red"))

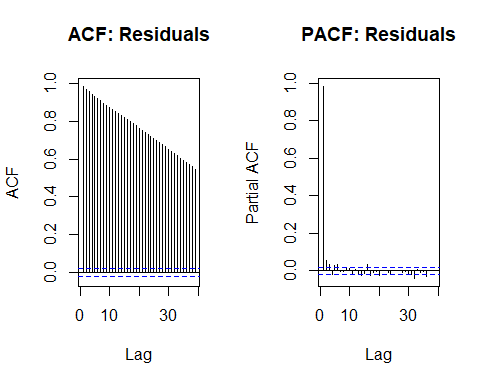


##### Betrachtung der Residuen

plot.dens(as.numeric(model$residuals), title = c("Density of Residuals"), plot.norm = TRUE, plot.lines = FALSE)



par(mfrow = c(1, 2))  
Acf(as.numeric(model$residuals[!is.na(model$residuals)]), main = c("ACF: Residuals"))  
Pacf(as.numeric(model$residuals[!is.na(model$residuals)]), main = c("PACF: Residuals"))



par(mfrow = c(1, 1))  
  
(model.autoarima <- auto.arima(ts(model$residuals[!is.na(model$residuals)])))

## Series: ts(model$residuals[!is.na(model$residuals)])   
## ARIMA(5,1,5)   
##   
## Coefficients:

## Warning in sqrt(diag(x$var.coef)): NaNs produced

## ar1 ar2 ar3 ar4 ar5 ma1 ma2 ma3  
## 0.0768 0.0626 0.4220 0.2399 -0.3469 -0.1441 -0.0942 -0.4088  
## s.e. NaN 0.1023 0.0984 0.1295 0.0904 NaN 0.1226 0.1006  
## ma4 ma5  
## -0.2466 0.3407  
## s.e. 0.1218 0.0878  
##   
## sigma^2 estimated as 1.33e-07: log likelihood=64143.02  
## AIC=-128264.1 AICc=-128264 BIC=-128184.9

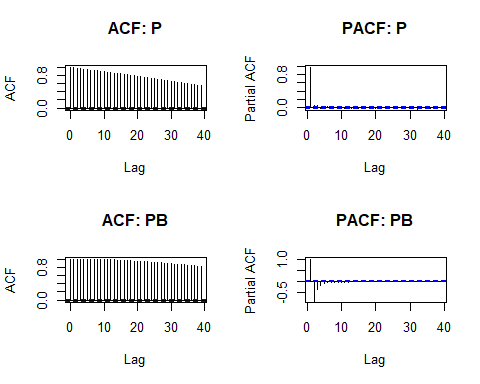
(model.stationary <- auto.arima(ts(model$residuals[!is.na(model$residuals)]), d = 0))

## Series: ts(model$residuals[!is.na(model$residuals)])   
## ARIMA(4,0,2) with zero mean   
##   
## Coefficients:  
## ar1 ar2 ar3 ar4 ma1 ma2  
## 0.3138 -0.2385 0.8547 0.0318 0.6135 0.8474  
## s.e. 0.0461 0.0548 0.0662 0.0137 0.0452 0.0785  
##   
## sigma^2 estimated as 1.323e-07: log likelihood=64171.69  
## AIC=-128329.4 AICc=-128329.4 BIC=-128279

#### ADL

##### Berechnung:

ORCL$NivClean\_Ind <- index(ORCL$Ratios.PB$NivCleanded[!is.na(ORCL$Ratios.PB$NivCleanded)])  
  
AIC\_BIC <- data.frame(matrix(NA, nrow = 730, ncol = 2))  
colnames(AIC\_BIC) <- c("AIC", "BIC")  
  
data.xts <- xts(ORCL$Ratios.PB$NivCleanded[ORCL$NivClean\_Ind], order.by = ORCL$NivClean\_Ind)  
colnames(data.xts) <- c("PB")  
data.xts$P <- ORCL$P\_xts$d.P\_MA[ORCL$NivClean\_Ind]  
  
#Test for the stationary requirement of P and PB  
par(mfrow = c(2, 2))  
acf(data.xts$P, main = c("ACF: P"))  
pacf(data.xts$P, main = c("PACF: P"))  
acf(data.xts$PB, main = c("ACF: PB"))  
pacf(data.xts$PB, main = c("PACF: PB"))



adf.test(data.xts$P)

## Augmented Dickey-Fuller Test   
## alternative: stationary   
##   
## Type 1: no drift no trend   
## lag ADF p.value  
## [1,] 0 8.09 0.99  
## [2,] 1 8.53 0.99  
## [3,] 2 8.89 0.99  
## [4,] 3 8.83 0.99  
## [5,] 4 9.07 0.99  
## [6,] 5 9.36 0.99  
## [7,] 6 9.39 0.99  
## [8,] 7 9.40 0.99  
## [9,] 8 9.46 0.99  
## [10,] 9 9.54 0.99  
## [11,] 10 9.46 0.99  
## [12,] 11 9.56 0.99  
## Type 2: with drift no trend   
## lag ADF p.value  
## [1,] 0 8.38 0.99  
## [2,] 1 8.84 0.99  
## [3,] 2 9.20 0.99  
## [4,] 3 9.13 0.99  
## [5,] 4 9.38 0.99  
## [6,] 5 9.68 0.99  
## [7,] 6 9.71 0.99  
## [8,] 7 9.73 0.99  
## [9,] 8 9.79 0.99  
## [10,] 9 9.88 0.99  
## [11,] 10 9.80 0.99  
## [12,] 11 9.90 0.99  
## Type 3: with drift and trend   
## lag ADF p.value  
## [1,] 0 8.57 0.99  
## [2,] 1 9.04 0.99  
## [3,] 2 9.40 0.99  
## [4,] 3 9.32 0.99  
## [5,] 4 9.57 0.99  
## [6,] 5 9.88 0.99  
## [7,] 6 9.90 0.99  
## [8,] 7 9.92 0.99  
## [9,] 8 10.00 0.99  
## [10,] 9 10.10 0.99  
## [11,] 10 10.01 0.99  
## [12,] 11 10.11 0.99  
## ----   
## Note: in fact, p.value = 0.01 means p.value <= 0.01

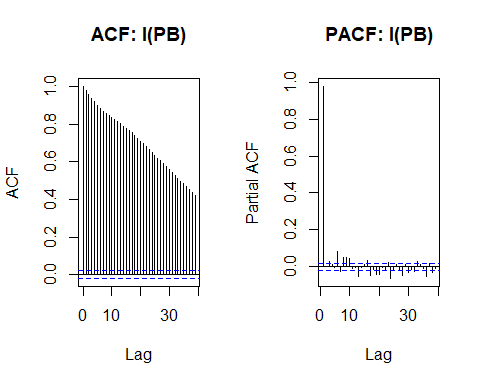
adf.test(data.xts$PB)

## Augmented Dickey-Fuller Test   
## alternative: stationary   
##   
## Type 1: no drift no trend   
## lag ADF p.value  
## [1,] 0 0.792 0.872  
## [2,] 1 -7.839 0.010  
## [3,] 2 -7.875 0.010  
## [4,] 3 -8.125 0.010  
## [5,] 4 -8.275 0.010  
## [6,] 5 -8.257 0.010  
## [7,] 6 -9.055 0.010  
## [8,] 7 -8.863 0.010  
## [9,] 8 -9.384 0.010  
## [10,] 9 -9.925 0.010  
## [11,] 10 -10.465 0.010  
## [12,] 11 -10.398 0.010  
## Type 2: with drift no trend   
## lag ADF p.value  
## [1,] 0 0.793 0.99  
## [2,] 1 -7.839 0.01  
## [3,] 2 -7.875 0.01  
## [4,] 3 -8.125 0.01  
## [5,] 4 -8.274 0.01  
## [6,] 5 -8.256 0.01  
## [7,] 6 -9.055 0.01  
## [8,] 7 -8.863 0.01  
## [9,] 8 -9.384 0.01  
## [10,] 9 -9.924 0.01  
## [11,] 10 -10.465 0.01  
## [12,] 11 -10.398 0.01  
## Type 3: with drift and trend   
## lag ADF p.value  
## [1,] 0 0.791 0.99  
## [2,] 1 -7.838 0.01  
## [3,] 2 -7.874 0.01  
## [4,] 3 -8.125 0.01  
## [5,] 4 -8.274 0.01  
## [6,] 5 -8.256 0.01  
## [7,] 6 -9.055 0.01  
## [8,] 7 -8.863 0.01  
## [9,] 8 -9.383 0.01  
## [10,] 9 -9.924 0.01  
## [11,] 10 -10.465 0.01  
## [12,] 11 -10.397 0.01  
## ----   
## Note: in fact, p.value = 0.01 means p.value <= 0.01

data.xts$IPB <- data.xts$PB-lag(data.xts$PB, 1)  
adf.test(data.xts$IPB)

## Augmented Dickey-Fuller Test   
## alternative: stationary   
##   
## Type 1: no drift no trend   
## lag ADF p.value  
## [1,] 0 10.5 0.99  
## [2,] 1 10.7 0.99  
## [3,] 2 11.1 0.99  
## [4,] 3 11.4 0.99  
## [5,] 4 11.5 0.99  
## [6,] 5 12.4 0.99  
## [7,] 6 12.3 0.99  
## [8,] 7 12.8 0.99  
## [9,] 8 13.3 0.99  
## [10,] 9 13.7 0.99  
## [11,] 10 13.7 0.99  
## [12,] 11 13.8 0.99  
## Type 2: with drift no trend   
## lag ADF p.value  
## [1,] 0 10.5 0.99  
## [2,] 1 10.7 0.99  
## [3,] 2 11.1 0.99  
## [4,] 3 11.4 0.99  
## [5,] 4 11.5 0.99  
## [6,] 5 12.4 0.99  
## [7,] 6 12.3 0.99  
## [8,] 7 12.8 0.99  
## [9,] 8 13.3 0.99  
## [10,] 9 13.7 0.99  
## [11,] 10 13.7 0.99  
## [12,] 11 13.8 0.99  
## Type 3: with drift and trend   
## lag ADF p.value  
## [1,] 0 10.5 0.99  
## [2,] 1 10.7 0.99  
## [3,] 2 11.1 0.99  
## [4,] 3 11.4 0.99  
## [5,] 4 11.5 0.99  
## [6,] 5 12.4 0.99  
## [7,] 6 12.3 0.99  
## [8,] 7 12.8 0.99  
## [9,] 8 13.3 0.99  
## [10,] 9 13.7 0.99  
## [11,] 10 13.7 0.99  
## [12,] 11 13.8 0.99  
## ----   
## Note: in fact, p.value = 0.01 means p.value <= 0.01

par(mfrow = c(1, 2))  
acf(data.xts$IPB, na.action = na.exclude, main = c("ACF: I(PB)"))  
pacf(data.xts$IPB, na.action = na.exclude, main = c("PACF: I(PB)"))



par(mfrow = c(1, 1))  
  
  
#Fitting of the Model  
model <- lm(data.xts$P ~ data.xts$IPB + lag(data.xts$IPB, 1) + lag(data.xts$IPB, 2) + lag(data.xts$P, 1) + lag(data.xts$P, 2))  
summary(model)

##   
## Call:  
## lm(formula = data.xts$P ~ data.xts$IPB + lag(data.xts$IPB, 1) +   
## lag(data.xts$IPB, 2) + lag(data.xts$P, 1) + lag(data.xts$P,   
## 2))  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.0074611 -0.0000970 -0.0000020 0.0000916 0.0082006   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 6.661e-06 2.770e-06 2.405 0.016203 \*   
## data.xts$IPB 3.774e-02 3.853e-04 97.937 < 2e-16 \*\*\*  
## lag(data.xts$IPB, 1) -3.655e-02 6.656e-04 -54.919 < 2e-16 \*\*\*  
## lag(data.xts$IPB, 2) -5.038e-04 5.399e-04 -0.933 0.350816   
## lag(data.xts$P, 1) 9.501e-01 1.003e-02 94.744 < 2e-16 \*\*\*  
## lag(data.xts$P, 2) 3.745e-02 1.002e-02 3.736 0.000188 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.0002618 on 9928 degrees of freedom  
## (3 observations deleted due to missingness)  
## Multiple R-squared: 0.9856, Adjusted R-squared: 0.9856   
## F-statistic: 1.361e+05 on 5 and 9928 DF, p-value: < 2.2e-16

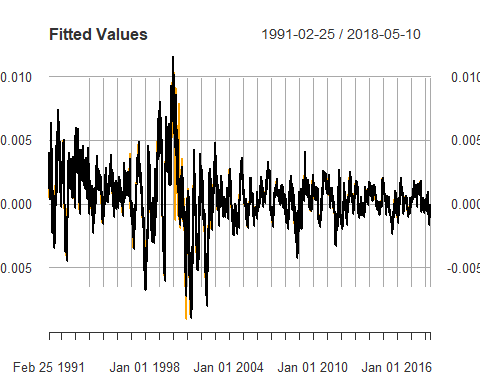
AIC(model)

## [1] -135668.7

BIC(model)

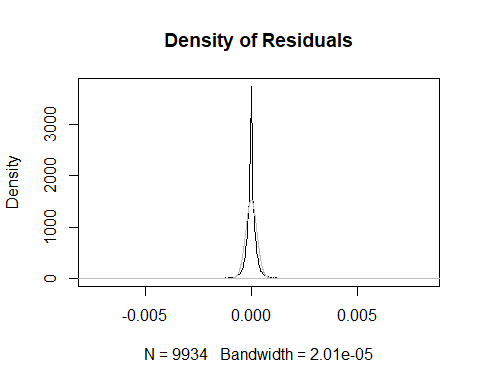
## [1] -135618.3

plot.data <- data.xts$P  
colnames(plot.data) <- c("P")  
plot.data$fit <- model$fitted.values  
plot(plot.data, type = c("l"), col = c("black", "orange"), main = c("Fitted Values"))

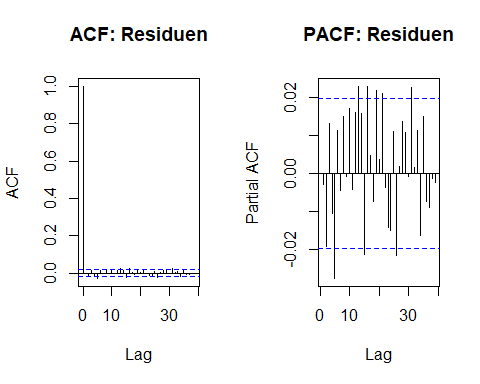


##### Betrachtung der Residuen:

plot.dens(as.numeric(model$residuals), title = c("Density of Residuals"), plot.norm = TRUE, plot.lines = FALSE)



par(mfrow = c(1, 2))  
acf(model$residuals, main = c("ACF: Residuen"))  
pacf(model$residuals, main = c("PACF: Residuen"))



par(mfrow = c(1, 1))