TA FINAL

sella

2025-07-18

```r
#MASS digunakan untuk fungsi plot boxcox()
library(MASS)

# Tseries digunakan untuk uji adf
library(tseries)
```

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

```r
#untuk Arima
library(forecast)

# lmtest untuk signifikansi parameter
library(lmtest)
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```r
# FinTS untuk ArchTest
library(FinTS)
```

```
##
## Attaching package: 'FinTS'
```

```
## The following object is masked from 'package:forecast':
##
##     Acf
```

```r
# TSA untuk acf dan pacf
library(TSA)
```

```
## Registered S3 methods overwritten by 'TSA':
##   method       from
##   fitted.Arima forecast
##   plot.Arima   forecast
```

```
##
## Attaching package: 'TSA'
```

```
## The following objects are masked from 'package:stats':
##
##     acf, arima
```

```
## The following object is masked from 'package:utils':
##
##     tar
```

```
# pysch describe di statistik deskriptif
library(psych)

# rygarch untuk model GARCH dan GJR-GARCH
library(rugarch)
```

```
## Loading required package: parallel
```

```
##
## Attaching package: 'rugarch'
```

```
## The following object is masked from 'package:stats':
##
##     sigma
```

```
# Plot
library(ggplot2)
```

```
##
## Attaching package: 'ggplot2'
```

```
## The following objects are masked from 'package:psych':
##
##     %+%, alpha
```

```
# dplyr (manipulasi kaya select dan filter)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following object is masked from 'package:MASS':
##
##     select
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
# scale untuk sumbu grafik
library(scales)
```

```
##
## Attaching package: 'scales'
```

```
## The following objects are masked from 'package:psych':
##
##     alpha, rescale
```

```
# untuk manajemen tanggal dan waktu
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
# untuk menginput data
library(readxl)
```

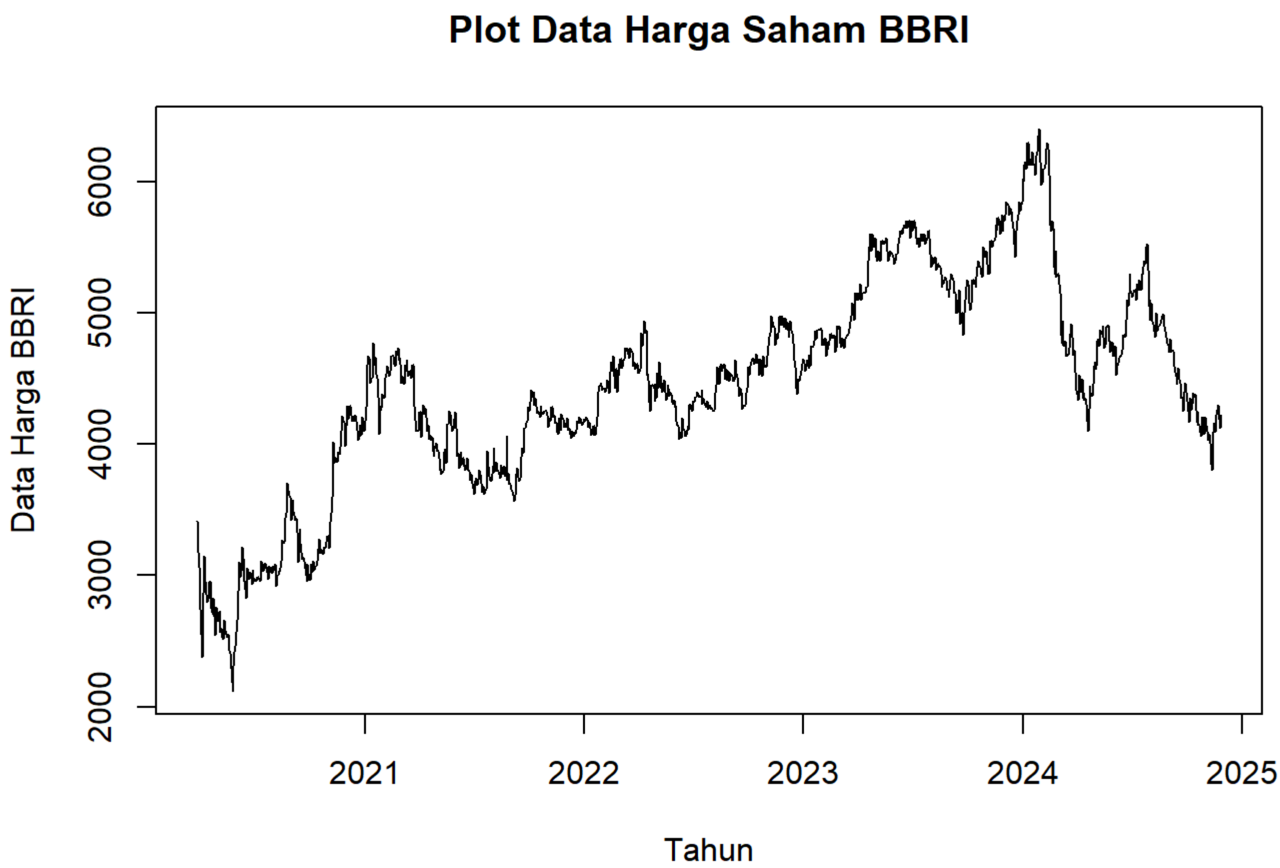
```
# data awal BBRI
library(readxl)
bbri <- read_excel("C:/Users/Lenovo/Downloads/data saham bbri.xlsx")
head(bbri, n=10)
```

```
## # A tibble: 10 × 6
##    Date                 Open  High   Low Close    Volume
##    <dttm>              <dbl> <dbl> <dbl> <dbl>     <dbl>
##  1 2020-03-16 16:00:00  3658  3658  3376  3415 182883087
##  2 2020-03-17 16:00:00  3415  3483  3180  3180 213785249
##  3 2020-03-18 16:00:00  3190  3356  3015  3063 254309677
##  4 2020-03-19 16:00:00  3054  3063  2859  2859 141397167
##  5 2020-03-20 16:00:00  2859  2898  2663  2741 613329030
##  6 2020-03-23 16:00:00  2585  2634  2556  2556 168516636
##  7 2020-03-24 16:00:00  2517  2683  2380  2380 507513269
##  8 2020-03-26 16:00:00  2439  2927  2439  2868 650460333
##  9 2020-03-27 16:00:00  2927  3580  2927  3151 492959353
## 10 2020-03-30 16:00:00  3073  3073  2937  2937 222949462
```

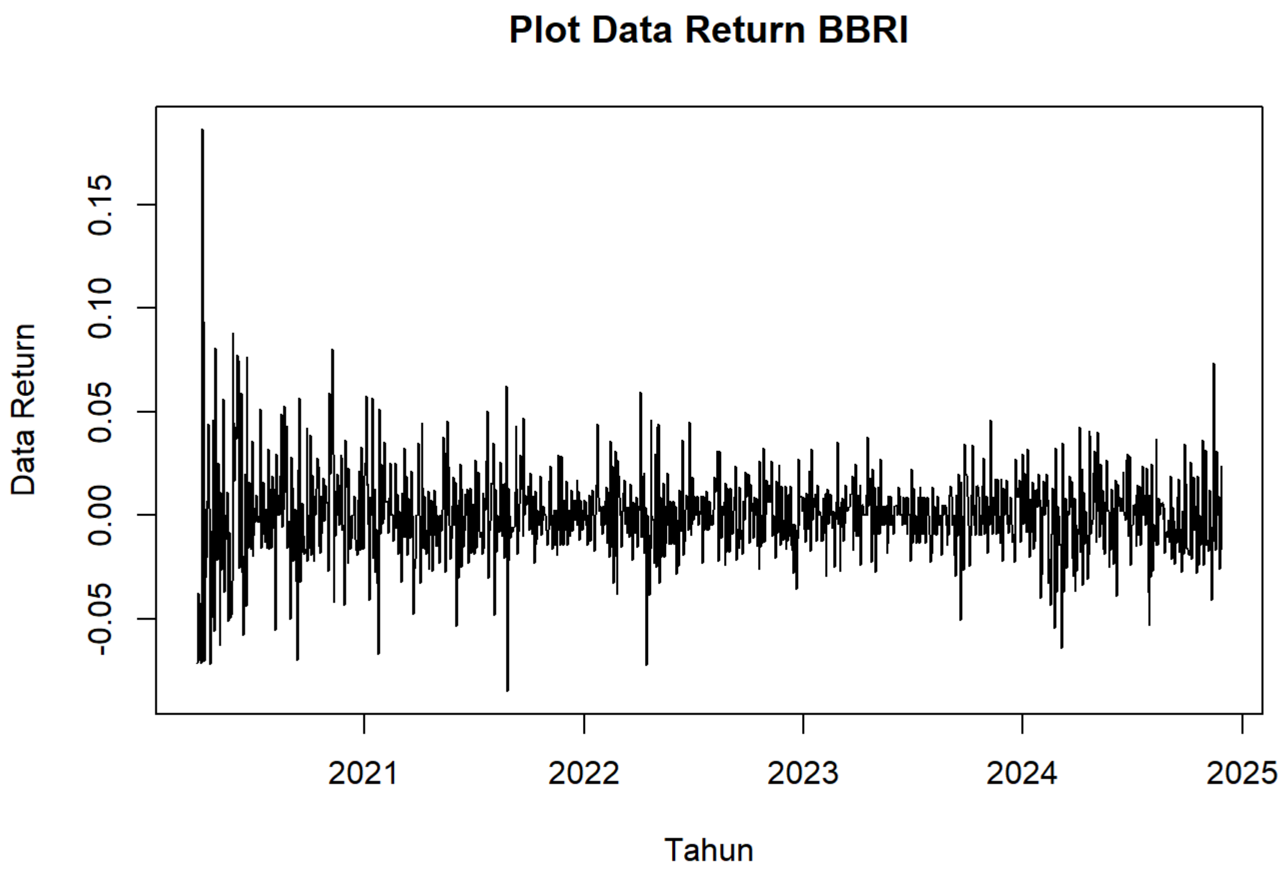
```
tail(bbri, n=10)
```

```
## # A tibble: 10 × 6
##    Date                 Open  High   Low Close    Volume
##    <dttm>              <dbl> <dbl> <dbl> <dbl>     <dbl>
##  1 2025-01-15 16:00:00  3840  4090  3840  4090 407813500
##  2 2025-01-16 16:00:00  4210  4240  4130  4160 349387500
##  3 2025-01-17 16:00:00  4150  4190  4080  4090 268758600
##  4 2025-01-20 16:00:00  4130  4220  4130  4220 167348600
##  5 2025-01-21 16:00:00  4270  4380  4250  4260 345223800
##  6 2025-01-22 16:00:00  4310  4320  4230  4260 193218900
##  7 2025-01-23 16:00:00  4260  4340  4260  4300 216018100
##  8 2025-01-24 16:00:00  4350  4350  4190  4190 276421900
##  9 2025-01-30 16:00:00  4130  4170  4090  4120 220222400
## 10 2025-01-31 16:00:00  4170  4280  4150  4220 192622700
```

```
# Data Harga Saham BBRI yang diubah ke bentuk time series
bbri$Date <- as.Date(bbri$Date, format="%Y-%m-%d")
bbri_ts <- ts(bbri$Close, start = c(2020, 60), frequency = 252)
plot(bbri_ts, type="l", xlab="Tahun", ylab="Data Harga BBRI", main="Plot Data Harga Saham BBR
I")
```




```
# Data return yang berasal dari data harga saham BBRI
returnss <- diff(log(bbri_ts))
returns <- ts(returnss, start = c(2020, 60))
plot(returnss, type="l", xlab="Tahun", ylab="Data Return", main="Plot Data Return BBRI")
```




```r
# 5 nilai terakhir data return
tail(returnss, n=5)
```

```
## Time Series:
## Start = c(2024, 225)
## End = c(2024, 229)
## Frequency = 252
## [1]  0.000000000  0.009345862 -0.025914289 -0.016847571  0.023981965
```

```r
# Statistik deskriptif data return saham BBRI
library(psych)
statistik <- describe(returns)
statistik_deskriptif <- statistik[, c("n", "mean", "skew", "kurtosis")]
options(scipen = 999)
statistik_deskriptif <- as.data.frame(lapply(statistik_deskriptif, function(x) round (x,4)))
print(statistik_deskriptif)
```

```
##      n   mean   skew kurtosis
## 1 1177 0.0002 0.7049   7.0256
```

```r
library(tseries)
# Uji ADF digunakan untuk mengecek stasioneritas rata-rata
adf.test(returns)
```

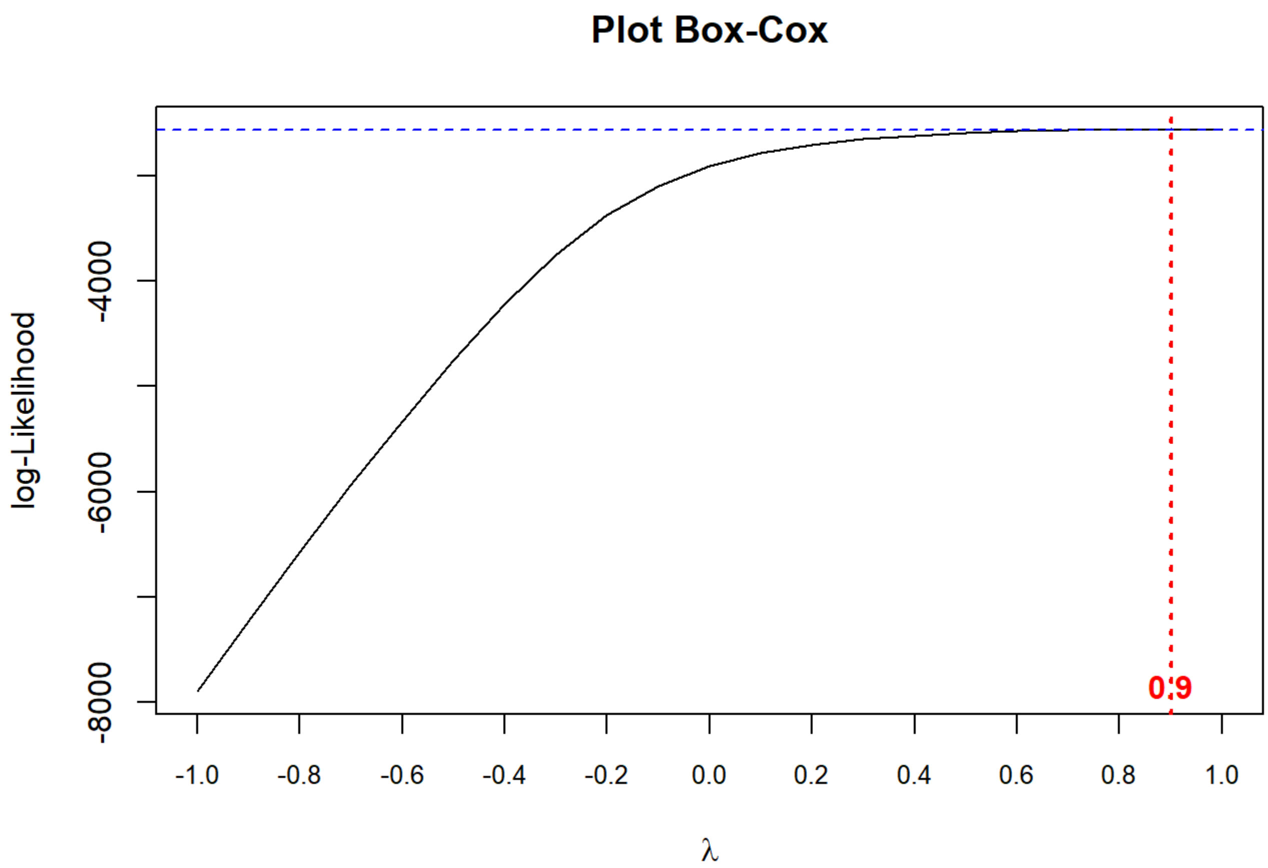
```
## Warning in adf.test(returns): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  returns
## Dickey-Fuller = -10.934, Lag order = 10, p-value = 0.01
## alternative hypothesis: stationary
```

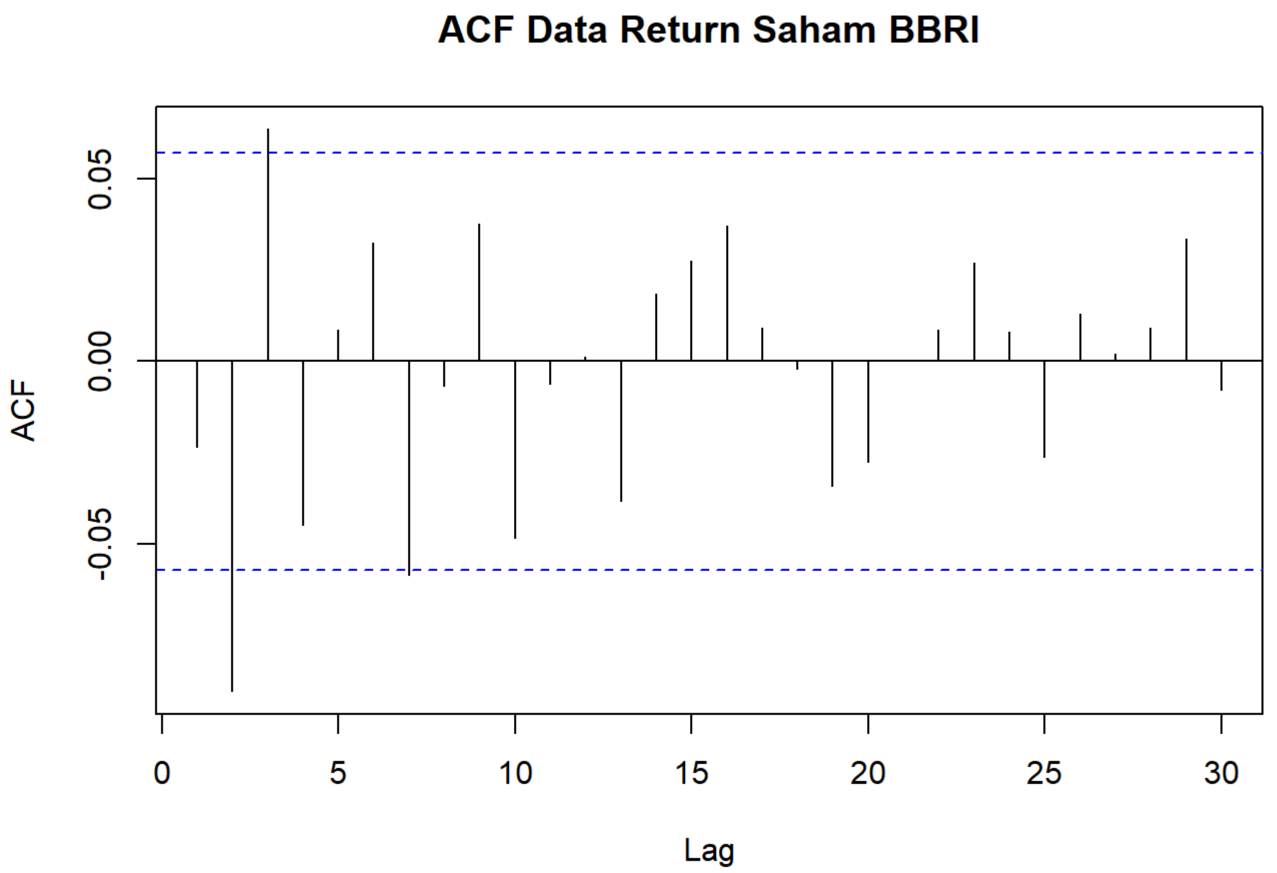
```r
library(MASS)
library(forecast)
# jika terdapat nilai negatif atau nol pada return maka diubah ke positif dahulu
returns_min <- min(returns)
epsilon <- 0.0001
returns_positif <- if (returns_min <= 0) returns + abs(returns_min) + epsilon else returns

# Lambda optimal
lambda <- BoxCox.lambda(returns_positif, method = "loglik")
print(paste("Nilai Lambda Optimal:", lambda))
```

```
## [1] "Nilai Lambda Optimal: 0.9"
```

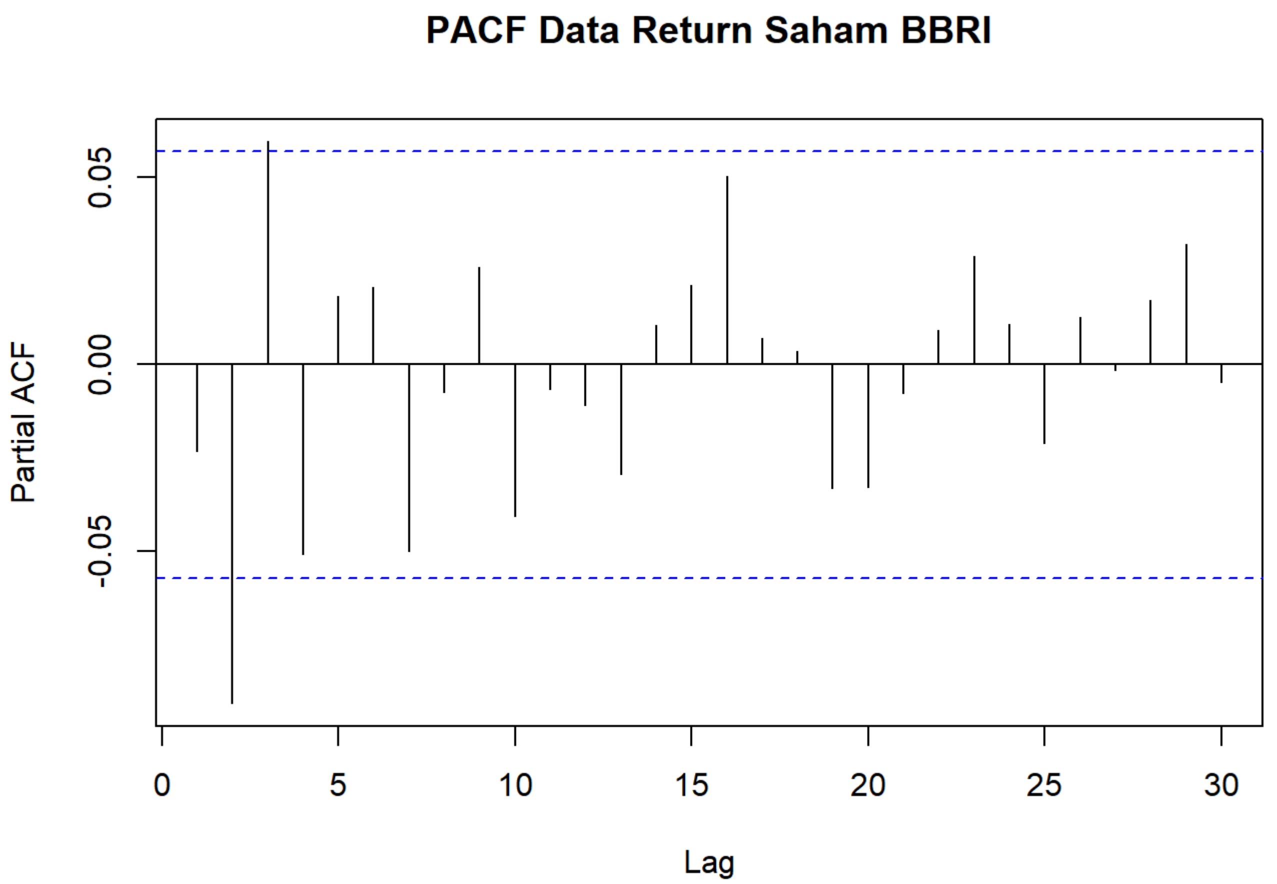
```r
boxcox_result <- boxcox(lm(returns_positif ~ 1),
                        lambda = seq(-1, 1, 0.1),
                        plotit = FALSE)

# Plot Boxcox
plot(boxcox_result$x, boxcox_result$y, type = "l",
     xlab = expression(lambda), ylab = "log-Likelihood",
     main = "Plot Box-Cox", xaxt = "n")
axis(1, at = seq(-1, 1, 0.2), las = 1, cex.axis = 0.8)
abline(h = max(boxcox_result$y) - qchisq(0.95, df = 1)/2,
       lty = 2, col = "blue")
abline(v = lambda, col = "red", lwd = 2, lty = 3)
text(lambda, min(boxcox_result$y) + 50,
     labels = round(lambda, 2),
     col = "red", cex = 1, font = 2)
```




```
# Plot ACF dan PACF untuk menetukan model ARIMA terbaik
library(TSA)
acf(returns, main="ACF Data Return Saham BBRI")
```




```
pacf(returns, main="PACF Data Return Saham BBRI")
```



```r
library(forecast)
# Membangun model ARIMA berdasarkan lag signifikan ACF dan PACF
arma1 = Arima(returns, order=c(0,0,1))
arma2 = Arima(returns, order=c(1,0,0))
arma3 = Arima(returns, order=c(1,0,1))
arma4 = Arima(returns, order=c(2,0,1))
arma5 = Arima(returns, order=c(1,0,2))
arma6 = Arima(returns, order=c(2,0,2))
arma7 = Arima(returns, order=c(3,0,1))
arma8 = Arima(returns, order=c(1,0,3))
arma9 = Arima(returns, order=c(3,0,2))
arma10 = Arima(returns, order=c(2,0,3))
arma11 = Arima(returns, order=c(3,0,3))

# Membandingkan nilai AIC unutuk memilih model terbaik
model_arima <- list(arma1, arma2, arma3, arma4, arma5, arma6, arma7, arma8, arma9, arma10, ar
ma11)
for (i in 1:length(model_arima)) {
  cat("Model ARMA", i, "\n")
  cat("AIC:", AIC(model_arima[[i]]), "\n")
  cat("------------------------\n")
}
```

```
## Model ARMA 1
## AIC: -5735.442
## ------------------------
## Model ARMA 2
## AIC: -5735.3
## ------------------------
## Model ARMA 3
## AIC: -5736.142
## ------------------------
## Model ARMA 4
## AIC: -5747.196
## ------------------------
## Model ARMA 5
## AIC: -5746.56
## ------------------------
## Model ARMA 6
## AIC: -5748.917
## ------------------------
## Model ARMA 7
## AIC: -5745.573
## ------------------------
## Model ARMA 8
## AIC: -5745.004
## ------------------------
## Model ARMA 9
## AIC: -5743.197
## ------------------------
## Model ARMA 10
## AIC: -5742.562
## ------------------------
## Model ARMA 11
## AIC: -5745.76
## ------------------------
```

```
model_terbaik <- which.min(sapply(model_arima, AIC))
cat("Model terbaik: ARMA", model_terbaik, "\n")
```

```
## Model terbaik: ARMA 6
```

```
library(lmtest)
# Signifikansi Koefisien Parameter Model ARMA
print(arma6)
```

```
## Series: returns
## ARIMA(2,0,2) with non-zero mean
##
## Coefficients:
##           ar1      ar2     ma1     ma2    mean
##       -1.0479  -0.7184  1.0378  0.6392  0.0002
## s.e.   0.1174   0.1325  0.1280  0.1452  0.0006
##
## sigma^2 = 0.0004402:  log likelihood = 2880.46
## AIC=-5748.92   AICc=-5748.85   BIC=-5718.49
```

```
options(scipen = 0)
coeftest(arma6)
```

```
##
## z test of coefficients:
##
##              Estimate  Std. Error z value  Pr(>|z|)
## ar1       -1.04794204  0.11744234 -8.9230 < 2.2e-16 ***
## ar2       -0.71838380  0.13254532 -5.4199 5.963e-08 ***
## ma1        1.03776447  0.12797037  8.1094 5.087e-16 ***
## ma2        0.63918321  0.14520360  4.4020 1.073e-05 ***
## intercept  0.00018379  0.00059129  0.3108    0.7559
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Uji diagnostik residual model ARMA(2,2)
residuals_arima <- residuals(arma6)

# Uji Ljung-Box terhadap residual ARMA
ljung1_arma <- Box.test(residuals_arima, lag=1, type="Ljung-Box")
print(ljung1_arma)
```

```
##
##  Box-Ljung test
##
## data:  residuals_arima
## X-squared = 0.019449, df = 1, p-value = 0.8891
```

```
ljung2_arma <- Box.test(residuals_arima, lag=11, type="Ljung-Box")
print(ljung2_arma)
```

```
##
##  Box-Ljung test
##
## data:  residuals_arima
## X-squared = 4.59, df = 11, p-value = 0.9494
```

```
ljung3_arma <- Box.test(residuals_arima, lag=19, type="Ljung-Box")
print(ljung3_arma)
```

```
##
##  Box-Ljung test
##
## data:  residuals_arima
## X-squared = 10.583, df = 19, p-value = 0.9371
```

```
library(tseries)
# Uji Jarque-Bera terhadap residual ARMA
jarque_bera <- jarque.bera.test(residuals_arima)
print(jarque_bera)
```

```
##
##  Jarque Bera Test
##
## data:  residuals_arima
## X-squared = 2431.2, df = 2, p-value < 2.2e-16
```

```
library(FinTS)
# Uji Lagrange-Multiplier terhadap residual ARMA
lm1_arma <- ArchTest(residuals_arima, lags = 3)
print(lm1_arma)
```

```
##
##  ARCH LM-test; Null hypothesis: no ARCH effects
##
## data:  residuals_arima
## Chi-squared = 141, df = 3, p-value < 2.2e-16
```

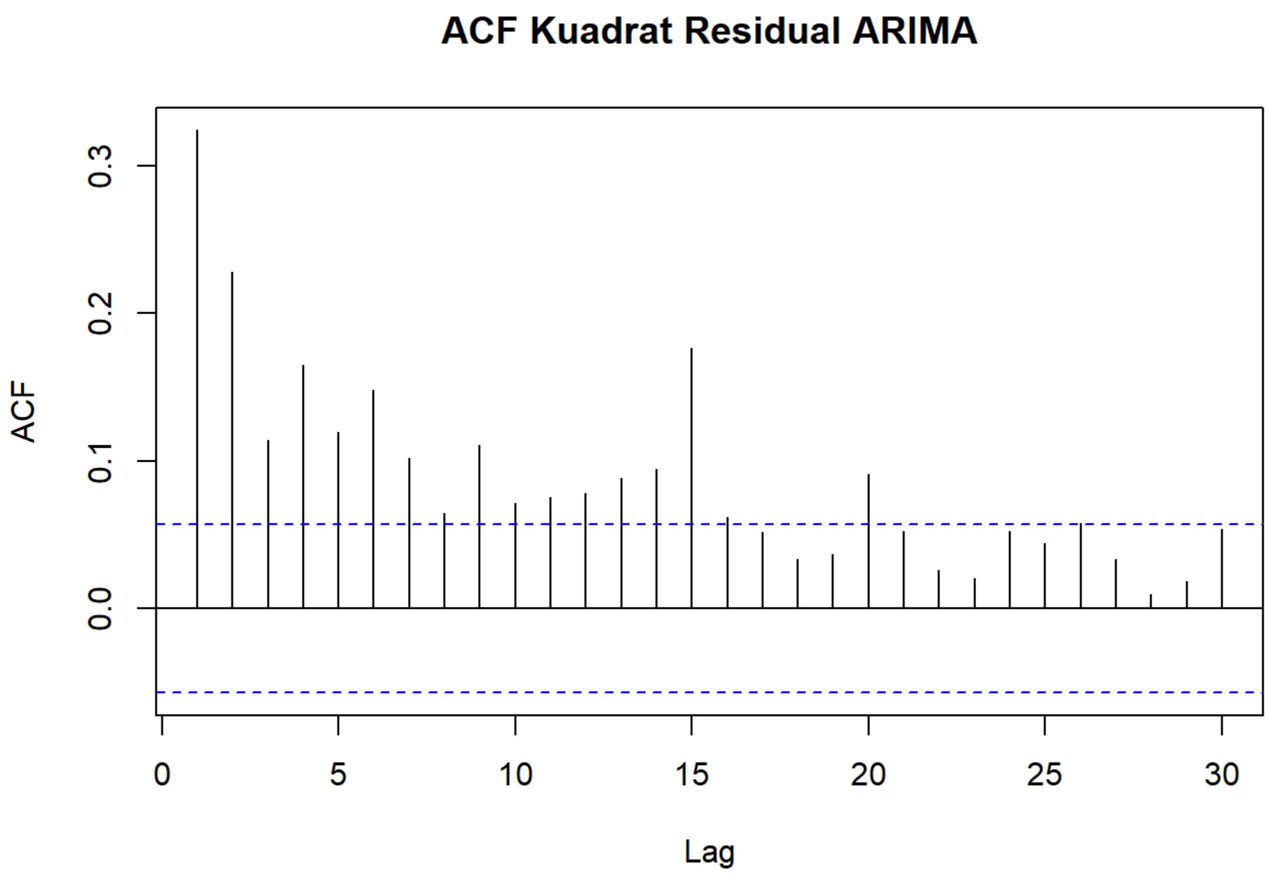
```
lm2_arma <- ArchTest(residuals_arima, lags = 5)
print(lm2_arma)
```

```
##
##  ARCH LM-test; Null hypothesis: no ARCH effects
##
## data:  residuals_arima
## Chi-squared = 150.25, df = 5, p-value < 2.2e-16
```

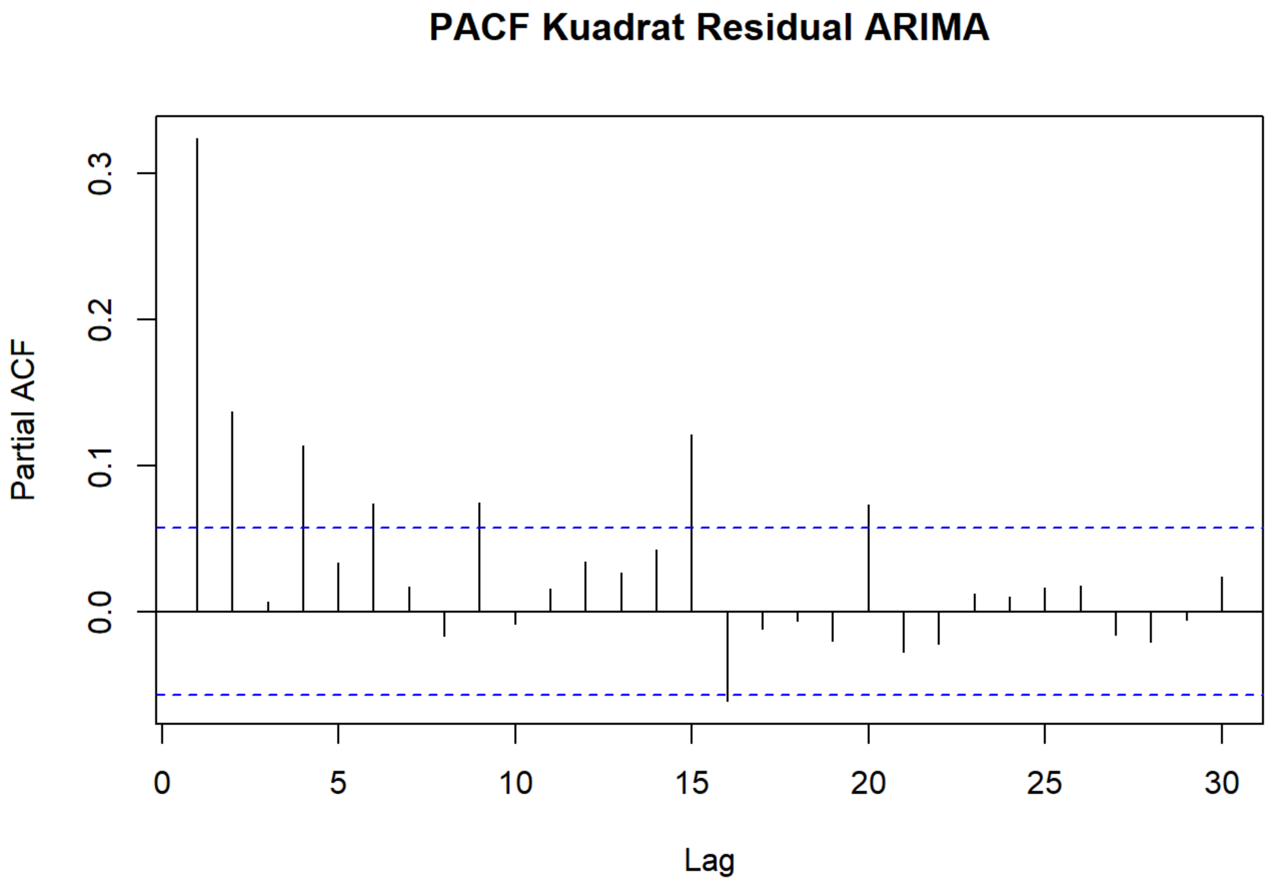
```
lm3_arma <- ArchTest(residuals_arima, lags = 7)
print(lm3_arma)
```

```
##
##  ARCH LM-test; Null hypothesis: no ARCH effects
##
## data:  residuals_arima
## Chi-squared = 176.12, df = 7, p-value < 2.2e-16
```

```
# ACF dan PACF kuadrat residual ARIMA digunakan untuk menentukan orde model GARCH

library(TSA)

residual_kuadrat_arma <- residuals_arima^2
acf(residual_kuadrat_arma, main="ACF Kuadrat Residual ARIMA")
```




```
pacf(residual_kuadrat_arma, main="PACF Kuadrat Residual ARIMA")
```


PACF Kuadrat Residual ARIMA
Partial ACF
0.0
0.1
0.2
0.3
0
5
10
15
20
25
30
Lag

```r
# Membangun model ARMA(2,2)-GARCH dengan orde ARCH(1) hingga GARCH(2,2)
library(rugarch)

# Model GARCH 1
#ARIMA(2,0,2)-GARCH(1,0)
garch1 <- ugarchspec(
  variance.model = list(model = "sGARCH", garchOrder = c(1,0)),
  mean.model = list(armaOrder = c(2,2)),
  distribution.model = "std"
)
model1_garch <- ugarchfit(spec = garch1, data = returns)

#Model GARCH 2
#ARIMA(2,0,2)-GARCH(0,1)
garch2 <- ugarchspec(
  variance.model = list(model = "sGARCH", garchOrder= c(0,1)),
  mean.model = list(armaOrder = c(2,2)),
  distribution.model = "std"
)
model2_garch <- ugarchfit(spec = garch2, data = returns)

#Model GARCH 3
#ARIMA(2,0,2)-GARCH(1,1)
garch3 <- ugarchspec(
  variance.model = list(model = "sGARCH", garchOrder= c(1,1)),
  mean.model = list(armaOrder = c(2,2)),
  distribution.model = "std"
)
model3_garch <- ugarchfit(spec = garch3, data = returns)

#Model GARCH 4
#ARIMA(2,0,2)-GARCH(1,2)
garch4 <- ugarchspec(
  variance.model = list(model = "sGARCH", garchOrder= c(1,2)),
  mean.model = list(armaOrder = c(2,2)),
  distribution.model = "std"
)
model4_garch <- ugarchfit(spec = garch4, data = returns)

#Model GARCH 5
#ARIMA(2,0,2)-GARCH(2,1)
garch5 <- ugarchspec(
  variance.model = list(model = "sGARCH", garchOrder= c(2,1)),
  mean.model = list(armaOrder = c(2,2)),
  distribution.model = "std"
)
model5_garch <- ugarchfit(spec = garch5, data = returns)

#Model GARCH 6
#ARIMA(2,0,2)-GARCH(2,2)
garch6 <- ugarchspec(
  variance.model = list(model = "sGARCH", garchOrder= c(2,2)),
  mean.model = list(armaOrder = c(2,2)),
  distribution.model = "std"
)
```

```r
model6_garch <- ugarchfit(spec = garch6, data = returns)


# Gunakan perbandingan AIC untuk memilih model terbaik
GARCH <- list(model1_garch, model2_garch, model3_garch, model4_garch, model5_garch, model6_ga
rch)
for (i in 1:length(GARCH)) {
  cat("Model GARCH", i, "\n")
  cat("AIC:", infocriteria(GARCH[[i]])["Akaike", ], "\n")
  cat("------------------------\n")
}
```

```
## Model GARCH 1
## AIC: -5.113247
## ------------------------
## Model GARCH 2
## AIC: -5.075625
## ------------------------
## Model GARCH 3
## AIC: -5.167723
## ------------------------
## Model GARCH 4
## AIC: -5.166465
## ------------------------
## Model GARCH 5
## AIC: -5.166214
## ------------------------
## Model GARCH 6
## AIC: -5.167054
## ------------------------
```

```r
# Uji Signifikansi GARCH
koefisien <- coef(model3_garch)
rse <- sqrt(diag(vcov(model3_garch, robust = TRUE)))

uji_t <- koefisien / rse
df <- length(returns) - length(koefisien)
p_values <- 2 * pt(-abs(uji_t), df = df)

keputusan <- ifelse(p_values < 0.05, "Signifikan", "Tidak Signifikan")

hasil <- data.frame(
  Parameter  = names(koefisien),
  Koefisien  = round(koefisien, 5),
  P_Value    = signif(p_values, digits = 2),
  Keputusan  = keputusan
)

print(hasil, row.names = FALSE)
```

```
##  Parameter Koefisien  P_Value        Keputusan
##         mu   0.00019  6.4e-01 Tidak Signifikan
##        ar1  -0.84264  4.3e-12       Signifikan
##        ar2  -0.76103  1.5e-12       Signifikan
##        ma1   0.75372  9.1e-09       Signifikan
##        ma2   0.67847  7.0e-07       Signifikan
##      omega   0.00002  8.5e-02 Tidak Signifikan
##     alpha1   0.11613  3.6e-03       Signifikan
##      beta1   0.84570 1.3e-147       Signifikan
##      shape   6.07738  1.2e-08       Signifikan
```

```
# Uji diagnostik residual ARIMA-GARCH
# Uji Ljung-Box, ARCH LM dan Sign Size Bias dapat dilihat pada model berikut
print(model3_garch)
```

```
## 
## *---------------------------------*
## *          GARCH Model Fit        *
## *---------------------------------*
## 
## Conditional Variance Dynamics
## -----------------------------------
## GARCH Model  : sGARCH(1,1)
## Mean Model   : ARFIMA(2,0,2)
## Distribution : std
## 
## Optimal Parameters
## ------------------------------------
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.000188    0.000428  0.43879 0.660812
## ar1    -0.842635    0.100794 -8.36000 0.000000
## ar2    -0.761034    0.098729 -7.70833 0.000000
## ma1     0.753722    0.109873  6.85993 0.000000
## ma2     0.678469    0.121158  5.59985 0.000000
## omega   0.000016    0.000006  2.74876 0.005982
## alpha1  0.116131    0.026521  4.37890 0.000012
## beta1   0.845698    0.025993 32.53547 0.000000
## shape   6.077377    1.074816  5.65434 0.000000
## 
## Robust Standard Errors:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.000188    0.000404  0.46585 0.641320
## ar1    -0.842635    0.120344 -7.00190 0.000000
## ar2    -0.761034    0.106390 -7.15327 0.000000
## ma1     0.753722    0.130198  5.78903 0.000000
## ma2     0.678469    0.135999  4.98879 0.000001
## omega   0.000016    0.000009  1.72581 0.084382
## alpha1  0.116131    0.039830  2.91567 0.003549
## beta1   0.845698    0.028120 30.07408 0.000000
## shape   6.077377    1.058042  5.74399 0.000000
## 
## LogLikelihood : 3050.205 
## 
## Information Criteria
## ------------------------------------
##                     
## Akaike       -5.1677
## Bayes        -5.1289
## Shibata      -5.1678
## Hannan-Quinn -5.1531
## 
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                          statistic p-value
## Lag[1]                       1.260  0.2616
## Lag[2*(p+q)+(p+q)-1][11]     4.165  0.9996
## Lag[4*(p+q)+(p+q)-1][19]     8.158  0.7718
## d.o.f=4
## H0 : No serial correlation
## 
```

```
## Weighted Ljung-Box Test on Standardized Squared Residuals
## ------------------------------------
##                          statistic p-value
## Lag[1]                       1.814  0.1780
## Lag[2*(p+q)+(p+q)-1][5]     3.319  0.3519
## Lag[4*(p+q)+(p+q)-1][9]     6.014  0.2965
## d.o.f=2
##
## Weighted ARCH LM Tests
## ------------------------------------
##             Statistic Shape Scale P-Value
## ARCH Lag[3]    0.4303 0.500 2.000  0.5118
## ARCH Lag[5]    1.5821 1.440 1.667  0.5710
## ARCH Lag[7]    4.1463 2.315 1.543  0.3260
##
## Nyblom stability test
## ------------------------------------
## Joint Statistic:  5.3792
## Individual Statistics:
## mu     0.05414
## ar1    0.11808
## ar2    0.04228
## ma1    0.15949
## ma2    0.05957
## omega  0.58518
## alpha1 0.36211
## beta1  0.20823
## shape  0.16068
##
## Asymptotic Critical Values (10% 5% 1%)
## Joint Statistic:          2.1 2.32 2.82
## Individual Statistic:     0.35 0.47 0.75
##
## Sign Bias Test
## ------------------------------------
##                    t-value     prob sig
## Sign Bias          1.35986 0.174136
## Negative Sign Bias 2.92668 0.003492 ***
## Positive Sign Bias 0.02291 0.981729
## Joint Effect       8.62112 0.034776  **
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## ------------------------------------
##   group statistic p-value(g-1)
## 1    20     14.01       0.7830
## 2    30     23.08       0.7730
## 3    40     28.13       0.9016
## 4    50     44.54       0.6544
##
##
## Elapsed time : 0.6163869
```

```r
# Membangun model ARMA(2,2)-GJR GARCH
# Model ARMA(2,2)-GJR GARCH(1,0)
gjr1 <- ugarchspec(
  variance.model = list(model = "gjrGARCH", garchOrder = c(1, 0)),
  mean.model = list(armaOrder = c(2, 2)),
  distribution.model = "std"
)
model_gjr1 <- ugarchfit(spec = gjr1, data = returns)

# Model ARMA(2,2)-GJR GARCH(0,1)
gjr2 <- ugarchspec(
  variance.model = list(model = "gjrGARCH", garchOrder = c(0, 1)),
  mean.model = list(armaOrder = c(2, 2)),
  distribution.model = "std"
)
model_gjr2 <- ugarchfit(spec = gjr2, data = returns)

# Model ARMA(2,2)-GJR GARCH(1,1)
gjr3 <- ugarchspec(
  variance.model = list(model = "gjrGARCH", garchOrder = c(1, 1)),
  mean.model = list(armaOrder = c(2, 2)),
  distribution.model = "std"
)
model_gjr3 <- ugarchfit(spec = gjr3, data = returns)

# Model ARMA(2,2)-GJR GARCH(1,2)
gjr4 <- ugarchspec(
  variance.model = list(model = "gjrGARCH", garchOrder = c(1, 2)),
  mean.model = list(armaOrder = c(2, 2)),
  distribution.model = "std"
)

model_gjr4 <- ugarchfit(spec = gjr4, data = returns)

# Model ARMA(2,2)-GJR GARCH(2,1)
gjr5 <- ugarchspec(
  variance.model = list(model = "gjrGARCH", garchOrder = c(2, 1)),
  mean.model = list(armaOrder = c(2, 2)),
  distribution.model = "std"
)

model_gjr5 <- ugarchfit(spec = gjr5, data = returns)

# Model ARMA(2,2)-GJR GARCH(2,2)
gjr6 <- ugarchspec(
  variance.model = list(model = "gjrGARCH", garchOrder = c(2, 2)),
  mean.model = list(armaOrder = c(2, 2)),
  distribution.model = "std"
)

model_gjr6 <- ugarchfit(spec = gjr6, data = returns)


# Gunakan perbandingan AIC untuk memilih model terbaik
GJR <- list(model_gjr1, model_gjr2, model_gjr3, model_gjr4, model_gjr5 ,model_gjr6)
```

```r
for (i in 1:length(GJR)) {
  cat("Model GJR-GARCH", i, "\n")
  cat("AIC:", infocriteria(GJR[[i]])["Akaike", ], "\n")
  cat("-------------------------\n")
}
```

```
## Model GJR-GARCH 1
## AIC: -5.100045
## -------------------------
## Model GJR-GARCH 2
## AIC: -5.075625
## -------------------------
## Model GJR-GARCH 3
## AIC: -5.172156
## -------------------------
## Model GJR-GARCH 4
## AIC: -5.170804
## -------------------------
## Model GJR-GARCH 5
## AIC: -5.169608
## -------------------------
## Model GJR-GARCH 6
## AIC: -5.17019
## -------------------------
```

```r
# Uji signifikansi parameter model ARIMA-GJR GARCH
model_gjr3 <- ugarchfit(spec = gjr3, data = returns)

koefisien_gjr <- coef(model_gjr3)
rse <- sqrt(diag(vcov(model_gjr3, robust = TRUE)))

uji_T <- koefisien_gjr / rse
df <- length(returns) - length(koefisien_gjr)
nilai_pvalue <- 2 * pt(-abs(uji_T), df = df)

keputusan_gjr <- ifelse(nilai_pvalue < 0.05, "Signifikan", "Tidak Signifikan")

results_rse <- data.frame(
  Parameter  = names(koefisien_gjr),
  Koefisien  = round(koefisien_gjr, 5),
  P_Value    = signif(nilai_pvalue, digits = 2),
  Keputusan  = keputusan_gjr
)

print(results_rse, row.names = FALSE)
```

```
##  Parameter Koefisien  P_Value        Keputusan
##         mu  -0.00002  9.7e-01 Tidak Signifikan
##        ar1  -0.83564  2.3e-12       Signifikan
##        ar2  -0.75660  2.8e-10       Signifikan
##        ma1   0.74657  3.8e-09       Signifikan
##        ma2   0.67201  1.2e-05       Signifikan
##      omega   0.00002  5.6e-02 Tidak Signifikan
##     alpha1   0.07328  2.4e-02       Signifikan
##      beta1   0.84329 6.4e-135       Signifikan
##     gamma1   0.09741  2.9e-02       Signifikan
##      shape   6.16912  1.0e-08       Signifikan
```

```
# Uji diagnostik residual ARIMA-GJR GARCH
# Uji Ljung-Box dan ARCH LM dapat dilihat pada model berikut
print(model_gjr3)
```

```
## 
## *---------------------------------*
## *          GARCH Model Fit        *
## *---------------------------------*
## 
## Conditional Variance Dynamics
## -----------------------------------
## GARCH Model  : gjrGARCH(1,1)
## Mean Model   : ARFIMA(2,0,2)
## Distribution : std
## 
## Optimal Parameters
## ------------------------------------
##         Estimate  Std. Error   t value Pr(>|t|)
## mu     -0.000017    0.000435 -0.039368 0.968597
## ar1    -0.835644    0.099381 -8.408480 0.000000
## ar2    -0.756599    0.103895 -7.282360 0.000000
## ma1     0.746571    0.107924  6.917591 0.000000
## ma2     0.672010    0.128122  5.245071 0.000000
## omega   0.000016    0.000006  2.847831 0.004402
## alpha1  0.073278    0.025397  2.885348 0.003910
## beta1   0.843293    0.026873 31.380449 0.000000
## gamma1  0.097414    0.038885  2.505176 0.012239
## shape   6.169119    1.098293  5.617005 0.000000
## 
## Robust Standard Errors:
##         Estimate  Std. Error   t value Pr(>|t|)
## mu     -0.000017    0.000431 -0.039811 0.968244
## ar1    -0.835644    0.117851 -7.090688 0.000000
## ar2    -0.756599    0.118844 -6.366331 0.000000
## ma1     0.746571    0.125763  5.936338 0.000000
## ma2     0.672010    0.153075  4.390061 0.000011
## omega   0.000016    0.000009  1.913213 0.055721
## alpha1  0.073278    0.032408  2.261111 0.023752
## beta1   0.843293    0.029748 28.347465 0.000000
## gamma1  0.097414    0.044594  2.184431 0.028931
## shape   6.169119    1.068556  5.773325 0.000000
## 
## LogLikelihood : 3053.814
## 
## Information Criteria
## ------------------------------------
## 
## Akaike       -5.1722
## Bayes        -5.1291
## Shibata      -5.1723
## Hannan-Quinn -5.1559
## 
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                           statistic p-value
## Lag[1]                        1.225  0.2683
## Lag[2*(p+q)+(p+q)-1][11]      4.081  0.9998
## Lag[4*(p+q)+(p+q)-1][19]      8.300  0.7509
## d.o.f=4
```

```
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## ------------------------------------
##                          statistic p-value
## Lag[1]                       2.206  0.1375
## Lag[2*(p+q)+(p+q)-1][5]      2.951  0.4160
## Lag[4*(p+q)+(p+q)-1][9]      7.134  0.1882
## d.o.f=2
##
## Weighted ARCH LM Tests
## ------------------------------------
##             Statistic Shape Scale P-Value
## ARCH Lag[3]    0.5221 0.500 2.000  0.4700
## ARCH Lag[5]    1.3398 1.440 1.667  0.6354
## ARCH Lag[7]    5.9009 2.315 1.543  0.1480
##
## Nyblom stability test
## ------------------------------------
## Joint Statistic:  5.5295
## Individual Statistics:
## mu     0.03819
## ar1    0.12541
## ar2    0.05388
## ma1    0.16522
## ma2    0.07452
## omega  0.70475
## alpha1 0.49554
## beta1  0.25200
## gamma1 0.44901
## shape  0.15195
##
## Asymptotic Critical Values (10% 5% 1%)
## Joint Statistic:          2.29 2.54 3.05
## Individual Statistic:     0.35 0.47 0.75
##
## Sign Bias Test
## ------------------------------------
##                    t-value    prob sig
## Sign Bias           0.9283 0.35347
## Negative Sign Bias  1.6707 0.09505   *
## Positive Sign Bias  0.7798 0.43565
## Joint Effect        3.7447 0.29038
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## ------------------------------------
##   group statistic p-value(g-1)
## 1    20     11.94       0.8883
## 2    30     20.78       0.8669
## 3    40     31.39       0.8017
## 4    50     54.14       0.2847
##
##
## Elapsed time : 0.638453
```

```r
peramalan <- ugarchforecast(model_gjr3, n.ahead = 3)
peramalan_volatilitas <- sigma(peramalan)
print(peramalan_volatilitas)
```

```
##     3255-01-01
## T+1 0.02088469
## T+2 0.02091702
## T+3 0.02094819
```

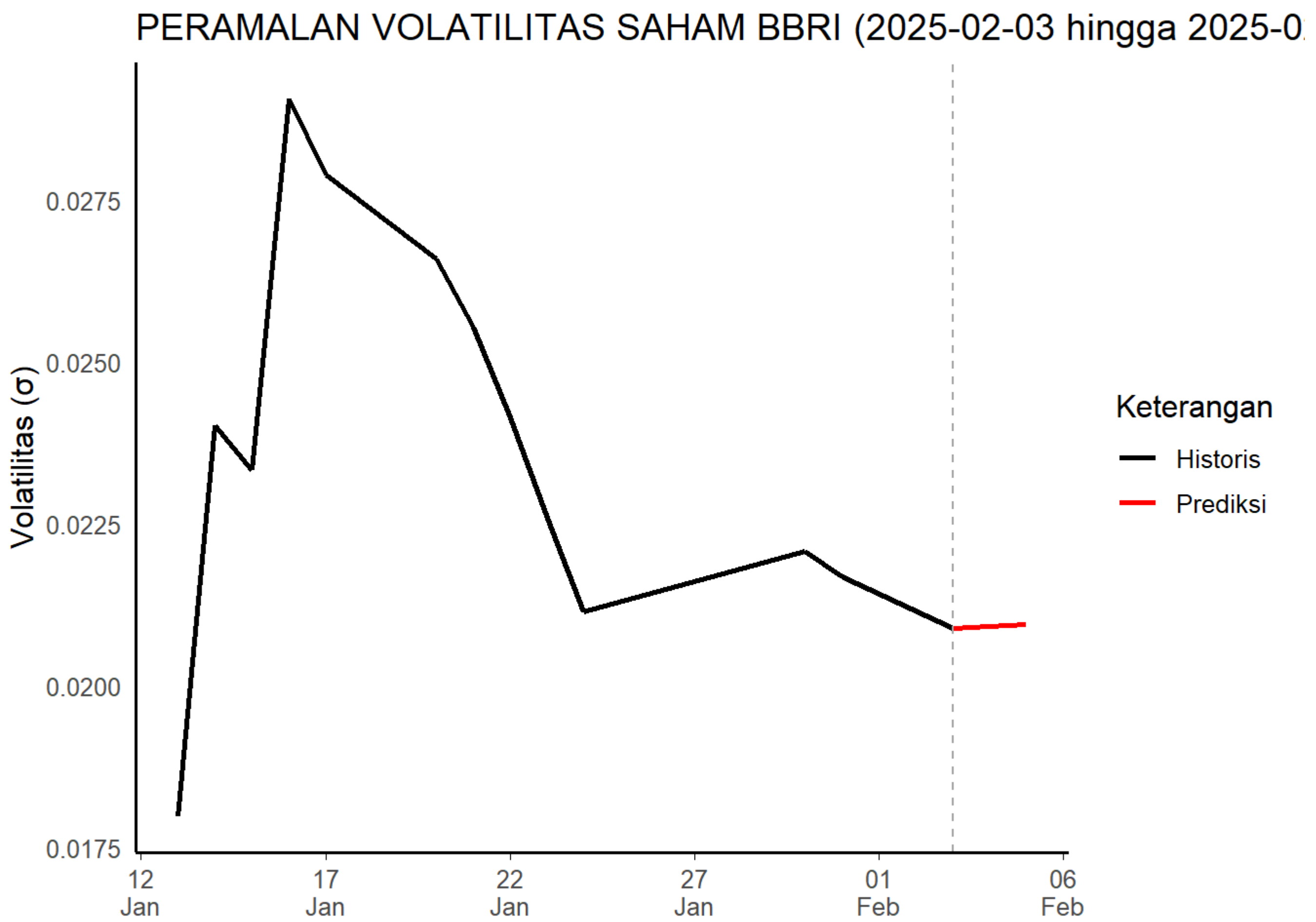
```r
# Volatilitas standar
volatilitas_sd <- sd(returnss)
print(volatilitas_sd)
```

```
## [1] 0.02112724
```

```r
library(dplyr)
library(scales)
library(lubridate)
library(ggplot2)
# Plot Volatilitas Historis dan Peramalan Volatilitas

sigma_volatilitas <- sigma(model_gjr3)
tanggal_volatilitas <- bbri$Date[-1]

df_volatilitas <- data.frame(Date = tanggal_volatilitas, Volatilitas = sigma_volatilitas)

df_vol_hist <- df_volatilitas %>%
  filter(Date >= as.Date("2025-01-12") & Date <= as.Date("2025-01-31")) %>%
  mutate(Tipe = "Historis")

tgl_prediksi <- as.Date(c("2025-02-03", "2025-02-04", "2025-02-05"))
sigma_prediksi <- c(0.02088469, 0.02091702, 0.02094819)

df_prediksi <- data.frame(
  Date = tgl_prediksi,
  Volatilitas = sigma_prediksi,
  Tipe = "Prediksi"
)

df_gabungann <- bind_rows(df_vol_hist, df_prediksi)
ggplot(df_gabungann, aes(x = Date, y = Volatilitas, color = Tipe, group = 1)) +
  geom_line(size = 1) +
  scale_color_manual(values = c("Historis" = "black", "Prediksi" = "red")) +
  scale_x_date(
    date_breaks = "5 days",
    labels = date_format("%d\n%b")
  ) +
  labs(
    title = "PERAMALAN VOLATILITAS SAHAM BBRI (2025-02-03 hingga 2025-02-05)",
    x = NULL,
    y = "Volatilitas (σ)",
    color = "Keterangan"
  ) +
  theme_minimal(base_size = 12) +
  theme(
    panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(),
    panel.background = element_blank(),
    axis.line = element_line(color = "black"),
    axis.text.x = element_text(angle = 0, vjust = 0.5),
    axis.ticks.x = element_line(color = "black", size = 0.3),
    legend.position = "right"
  ) +
  geom_vline(xintercept = as.numeric(as.Date("2025-02-03")),
             linetype = "dashed", color = "darkgray") +
  annotate("text", x = as.Date("2025-02-03"),
           y = min(df_gabungann$Volatilitas),
           label = "2025", vjust = 5.5, size = 3)
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## ℹ Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
## Warning: The `size` argument of `element_line()` is deprecated as of ggplot2 3.4.0.
## ℹ Please use the `linewidth` argument instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



```r
# Nilai Volatilitas Dengan Return Terbaru
library(readxl)

data_aktual <- read_excel("C:/Users/Lenovo/Downloads/data terbaru bri.xlsx") %>%
  select(Date, Close)

data_aktual$Date <- as.Date(data_aktual$Date)

log_returns <- diff(log(data_aktual$Close))

gjr_terbaru <- ugarchspec(
  variance.model = list(model = "gjrGARCH", garchOrder = c(1, 1)),
  mean.model = list(armaOrder = c(2, 2)),
  distribution.model = "std"
)

model_gjr_terbaru <- ugarchfit(spec = gjr_terbaru, data = log_returns)

sigma_terbaru <- sigma(model_gjr_terbaru)
tanggal <- data_aktual$Date[-1]

dataframe_vol <- data.frame(Date = tanggal, Volatilitas = sigma_terbaru)
tail(dataframe_vol, 3)
```

```
##                  Date Volatilitas
## 1973-03-24 2025-02-03  0.02097339
## 1973-03-25 2025-02-04  0.01994591
## 1973-03-26 2025-02-05  0.01883635
```

```r
# MAPE
peramalan_volatilitas <- as.numeric(sigma(peramalan))

aktual_volatilitas <- tail(dataframe_vol$Volatilitas, 3)
mape <- mean(abs((aktual_volatilitas - peramalan_volatilitas) / aktual_volatilitas)) * 100

cat("MAPE:", round(mape, 2), "%\n")
```

```
## MAPE: 5.5 %
```