502_HW7

Shengbo Jin

3/2/2022

Q1

```
load("Homework 7 Data.Rdata")
head(futures)
```

```
##   lnfuture  lnspot
## 1  6.08382 6.08618
## 2  6.08404 6.08623
## 3  6.08473 6.08630
## 4  6.08450 6.08630
## 5  6.08450 6.08623
## 6  6.08439 6.08625
```

```
attach(futures)
```

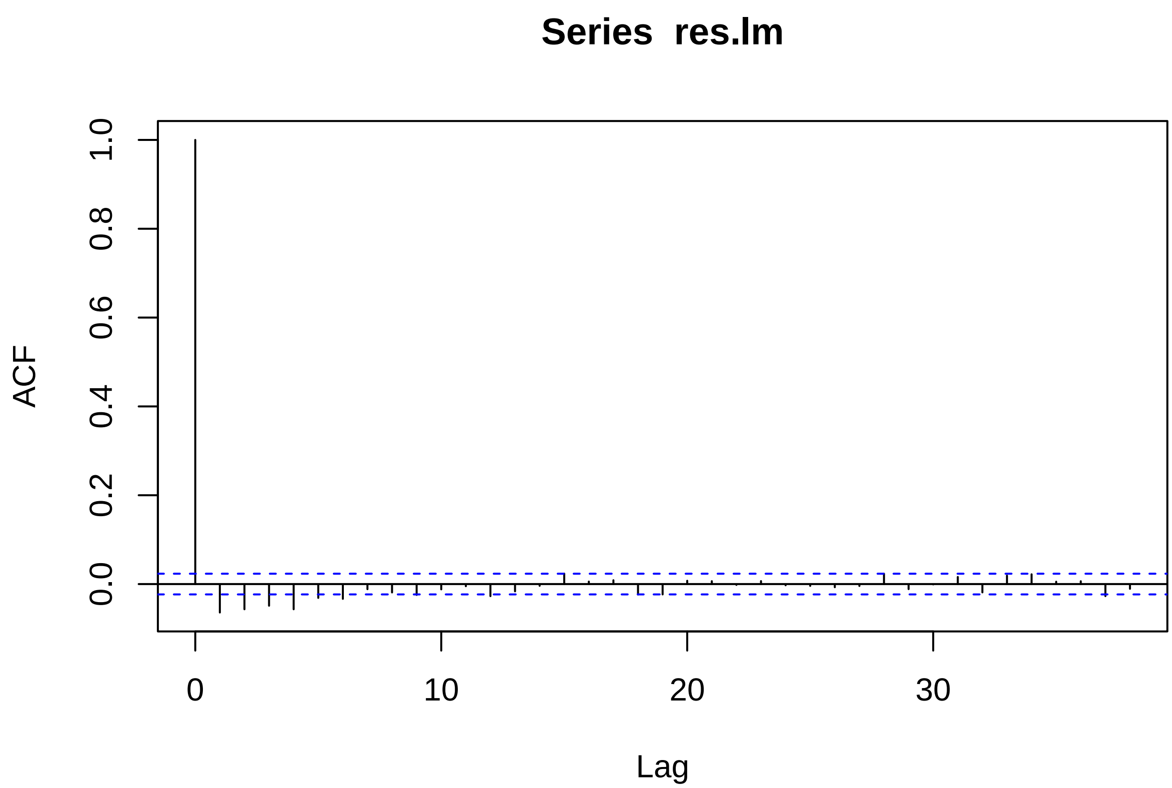
(a)

```
Y <- diff(lnfuture)
X <- diff(lnspot)
lm <- lm(Y~X)
summary(lm)
```

```
##
## Call:
## lm(formula = Y ~ X)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -0.0038484 -0.0001568 -0.0000014  0.0001612  0.0026256
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1.354e-06  3.509e-06   0.386      0.7
## X           6.212e-01  1.754e-02  35.420   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 0.0002948 on 7058 degrees of freedom
## Multiple R-squared:  0.1509, Adjusted R-squared:  0.1508
## F-statistic:  1255 on 1 and 7058 DF,  p-value: < 2.2e-16
```

```r
res.lm <- resid(lm)
acf(res.lm)
```




```r
Box.test(res.lm, lag=10, type="Ljung-Box", fitdf=2)
```

```
##
##  Box-Ljung test
##
## data:  res.lm
## X-squared = 114.79, df = 8, p-value < 2.2e-16
```

A plot of the sample autocorrelation function shows the residuals have significant autocorrelation, and a Ljung-Box test also confirms this.

(b)

```r
library(forecast)
```

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

```
auto.arima(res.lm)
```

```
## Series: res.lm
## ARIMA(1,0,1) with zero mean
##
## Coefficients:
##          ar1      ma1
##       0.8179  -0.9196
## s.e.  0.0152   0.0102
##
## sigma^2 = 8.429e-08:  log likelihood = 47483.26
## AIC=-94960.51   AICc=-94960.51   BIC=-94939.93
```

```
(mod.arimax1 <- Arima(Y, xreg=X, order=c(1,0,1)))
```

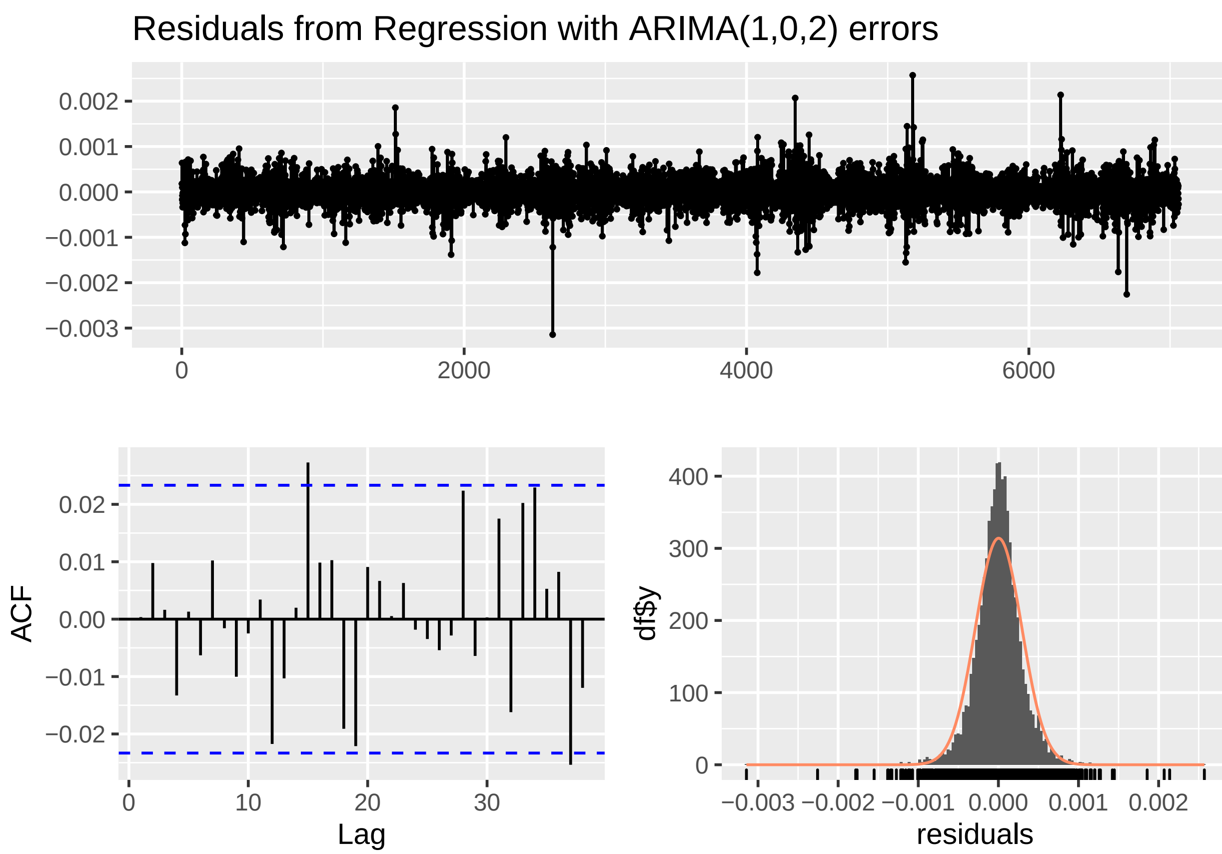
```
## Series: Y
## Regression with ARIMA(1,0,1) errors
##
## Coefficients:
##          ar1      ma1  intercept    xreg
##       0.8223  -0.9375          0  0.7229
## s.e.  0.0125   0.0081          0  0.0177
##
## sigma^2 = 8.393e-08:  log likelihood = 47499.45
## AIC=-94988.9   AICc=-94988.89   BIC=-94954.59
```

```
(mod.arimax2 <- auto.arima(Y, xreg=X))
```

```
## Series: Y
## Regression with ARIMA(1,0,2) errors
##
## Coefficients:
##          ar1      ma1      ma2    xreg
##       0.8084  -0.9106  -0.0212  0.7264
## s.e.  0.0158   0.0196   0.0141  0.0177
##
## sigma^2 = 8.391e-08:  log likelihood = 47500.14
## AIC=-94990.28   AICc=-94990.27   BIC=-94955.97
```

The fitted model is ARIMAX(1,0,2), and has lower AICc than the previous model as expected.

```
checkresiduals(mod.arimax2)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from Regression with ARIMA(1,0,2) errors
## Q* = 3.7473, df = 6, p-value = 0.7108
##
## Model df: 4.   Total lags used: 10
```

To a large extent, both plots and tests show the residuals are white noise, so the model is a good fit.

(c)

```
forecast(mod.arimax2, xreg=0.0002, level=0.95)
```

```
##      Point Forecast         Lo 95        Hi 95
## 7061   0.0001934793 -0.0003742788 0.0007612375
```

Q2

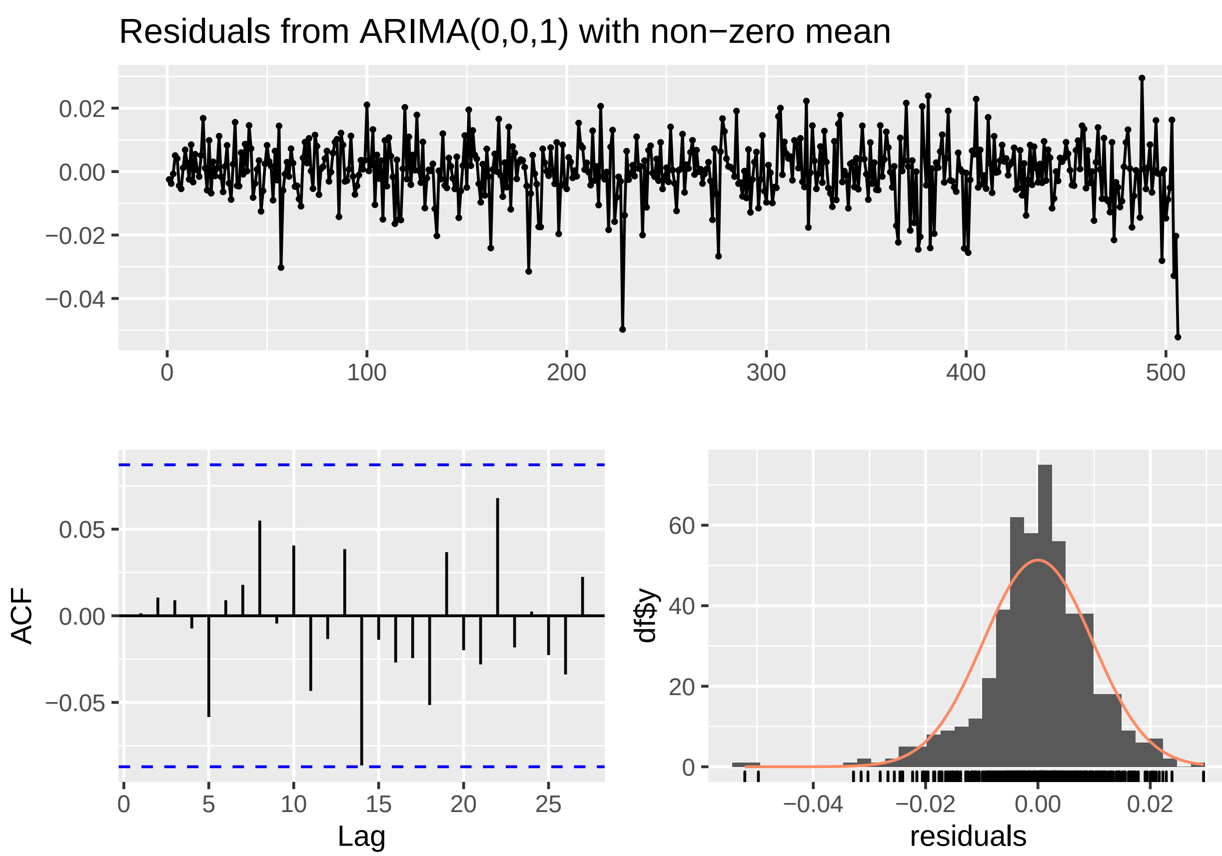
```
data(SP500, package="Ecdat")
ret <- SP500$r500[(1804-2*253+1):1804]
ret.black.mon <- SP500$r500[1805]
```

(a)

```r
(mean.fit <- auto.arima(ret))
```

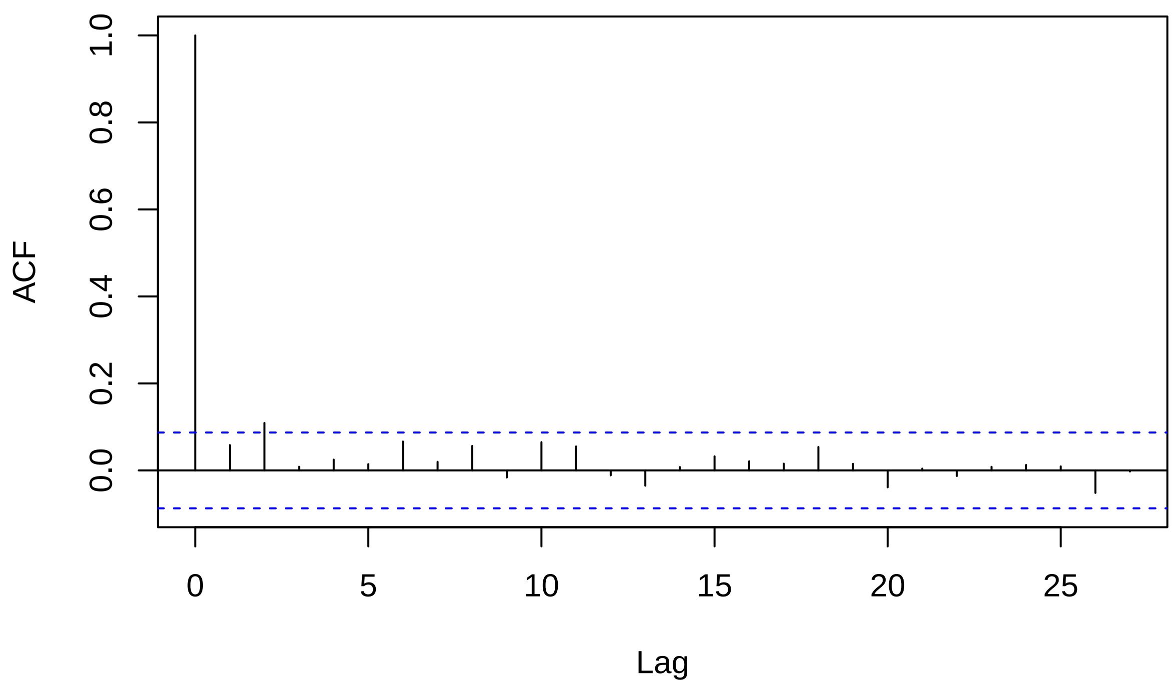
```
## Series: ret
## ARIMA(0,0,1) with non-zero mean
##
## Coefficients:
##          ma1   mean
##       0.1266  8e-04
## s.e.  0.0449  5e-04
##
## sigma^2 = 9.489e-05:  log likelihood = 1626.49
## AIC=-3246.98   AICc=-3246.93   BIC=-3234.3
```

```r
checkresiduals(mean.fit)
```

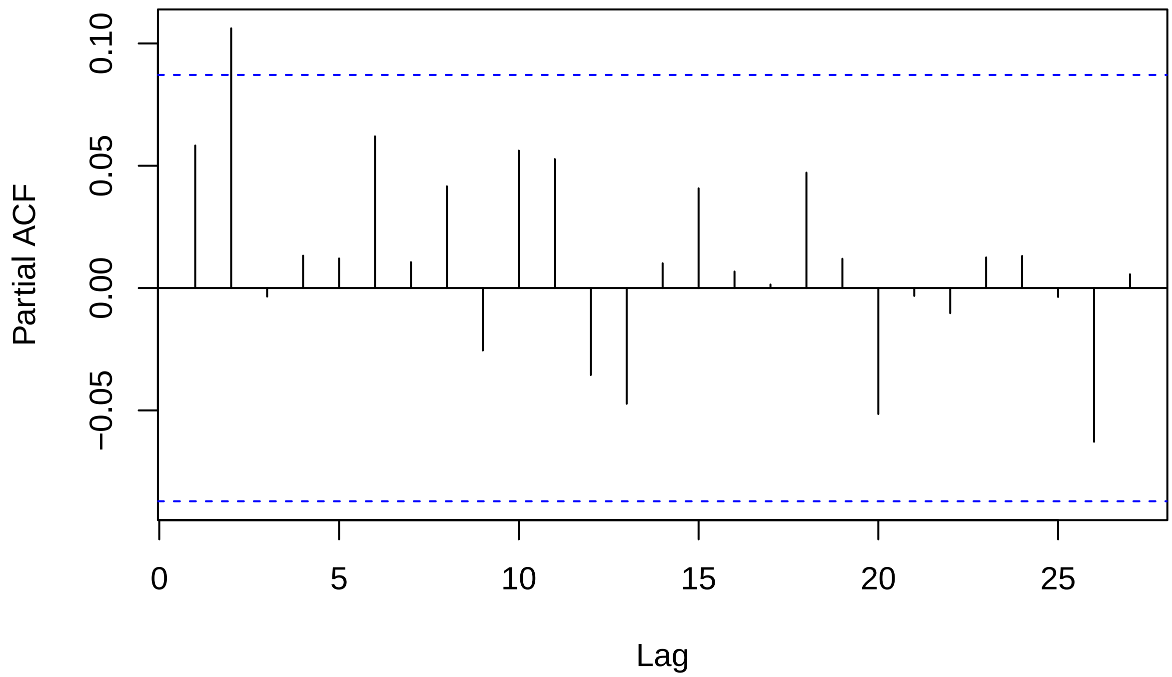



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(0,0,1) with non-zero mean
## Q* = 4.5035, df = 8, p-value = 0.8091
##
## Model df: 2.   Total lags used: 10
```

```r
acf(resid(mean.fit)^2, main="")
```




```r
pacf(resid(mean.fit)^2, main="")
```



The residuals of the fitted MA(1) appear to be white noise, but have volatility clustering as the time series plot and sample acf of squared residuals shows. Sample pacf of squared residuals suggests a GARCH(1,1) model.

(b)

```
library(fGarch)
```

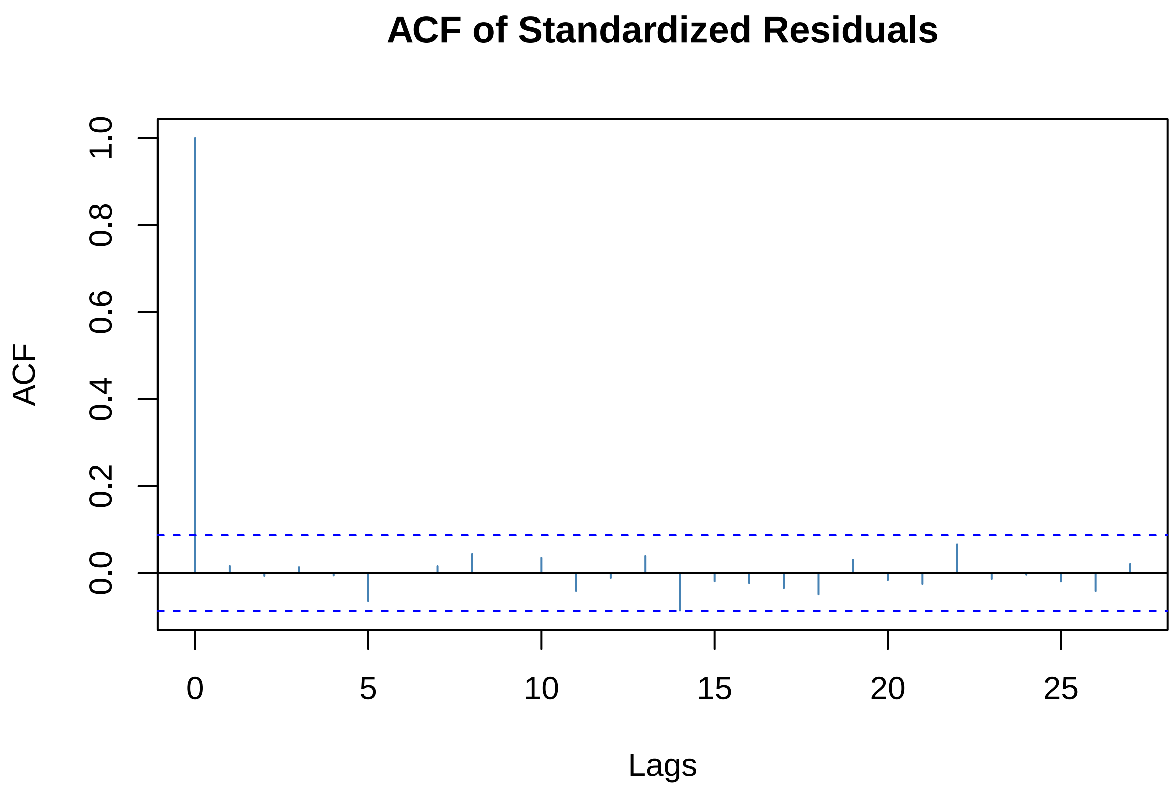
```
## Loading required package: timeDate

## Loading required package: timeSeries

## Loading required package: fBasics
```

```
fit.armagarch <- garchFit(~arma(0,1)+garch(1,1), data=ret,
                          cond.dist = "std", trace=FALSE)
summary(fit.armagarch)
```

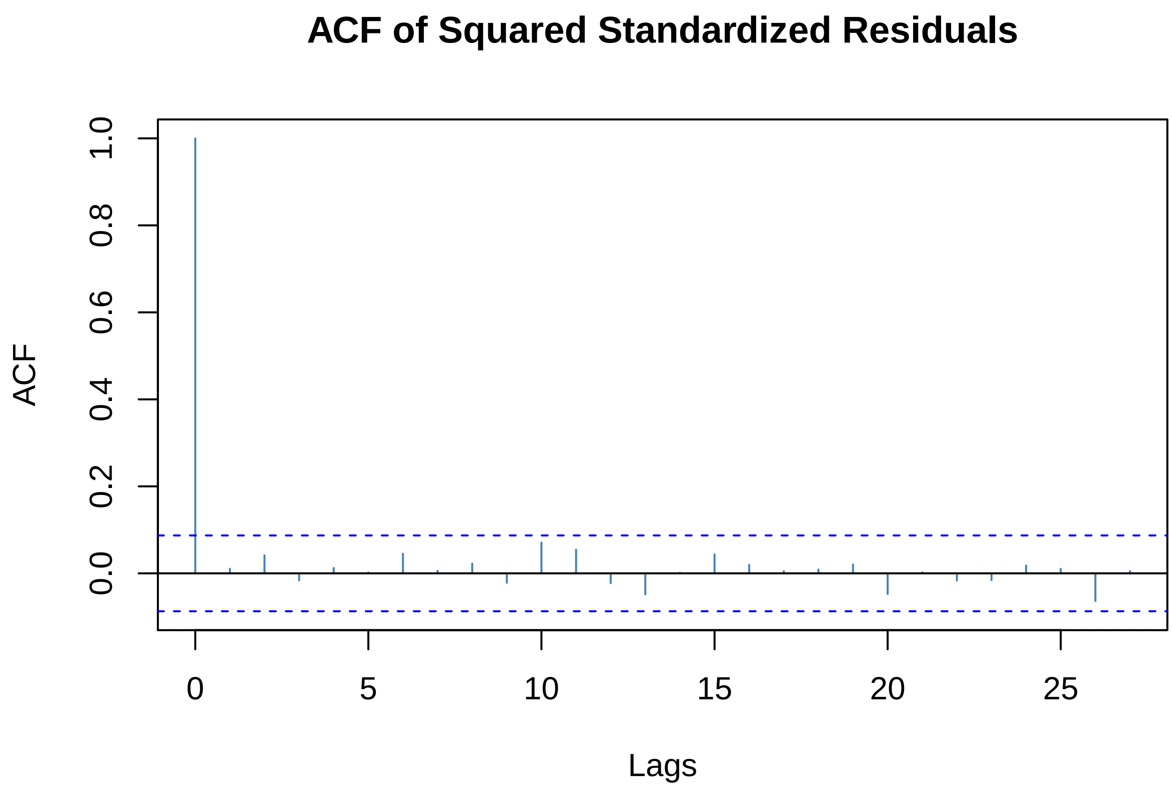
```
##
## Title:
##  GARCH Modelling
##
## Call:
```

```
##  garchFit(formula = ~arma(0, 1) + garch(1, 1), data = ret, cond.dist = "std",
##     trace = FALSE)
##
## Mean and Variance Equation:
##  data ~ arma(0, 1) + garch(1, 1)
## <environment: 0x00000000261dc038>
##  [data = ret]
##
## Conditional Distribution:
##  std
##
## Coefficient(s):
##         mu         ma1       omega      alpha1       beta1       shape
## 0.00141651  0.09477637  0.00001075  0.04388364  0.85698004  4.08099919
##
## Std. Errors:
##  based on Hessian
##
## Error Analysis:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu     1.417e-03   4.086e-04    3.467 0.000526 ***
## ma1    9.478e-02   4.443e-02    2.133 0.032898 *
## omega  1.075e-05   8.591e-06    1.251 0.210846
## alpha1 4.388e-02   3.283e-02    1.337 0.181384
## beta1  8.570e-01   9.709e-02    8.826  < 2e-16 ***
## shape  4.081e+00   9.239e-01    4.417 9.99e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Log Likelihood:
##  1655.48    normalized:  3.271699
##
## Description:
##  Mon Mar 07 21:29:19 2022 by user: King48
##
##
## Standardised Residuals Tests:
##                                 Statistic p-Value
##  Jarque-Bera Test   R    Chi^2  150.6831  0
##  Shapiro-Wilk Test  R    W      0.9663883 2.324834e-09
##  Ljung-Box Test     R    Q(10)  4.173439  0.9391838
##  Ljung-Box Test     R    Q(15)  9.965657  0.8218946
##  Ljung-Box Test     R    Q(20)  12.7511   0.8878061
##  Ljung-Box Test     R^2  Q(10)  5.311719  0.8694062
##  Ljung-Box Test     R^2  Q(15)  9.316155  0.860415
##  Ljung-Box Test     R^2  Q(20)  10.98605  0.9465835
##  LM Arch Test       R    TR^2   8.304252  0.7609251
##
## Information Criterion Statistics:
##       AIC       BIC       SIC      HQIC
## -6.519682 -6.469565 -6.519959 -6.500026
```

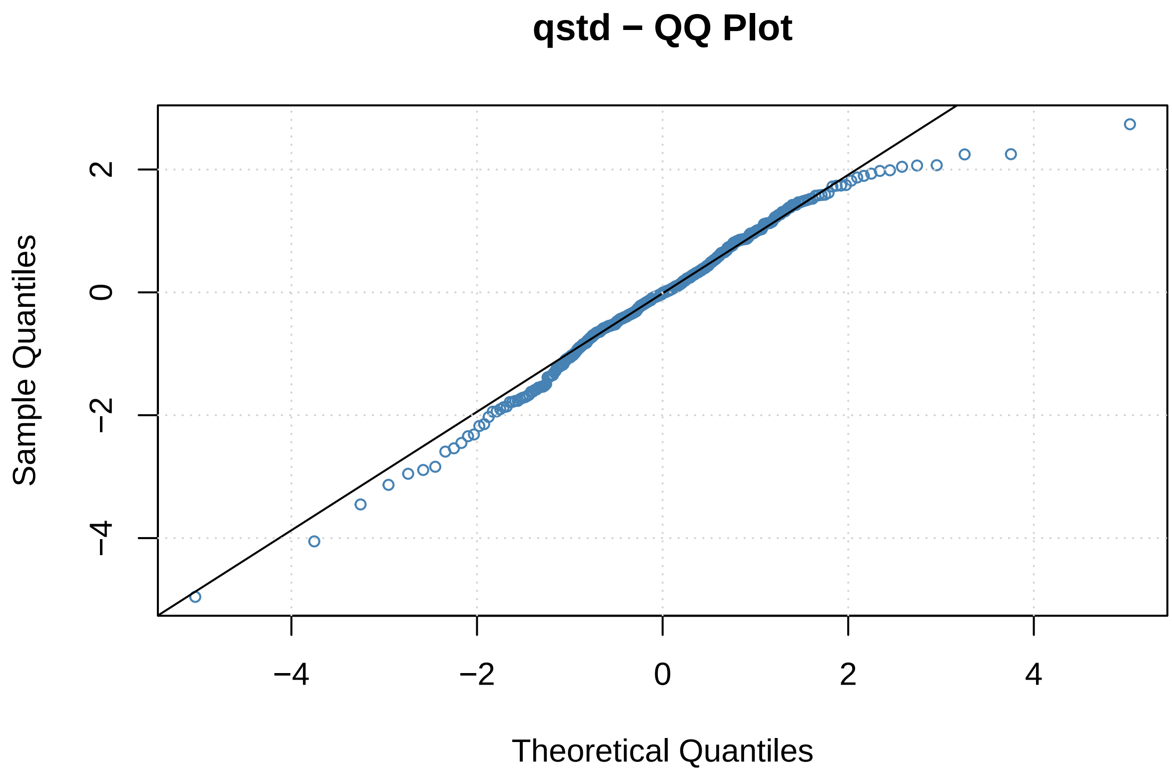
```
plot(fit.armagarch, which=10)
```




```
plot(fit.armagarch, which=11)
```




```
plot(fit.armagarch, which=13)
```



According to the Ljung-Box test on R, for lags up 10, 15, 20, we fail to reject zero autocorrelation; According to the Ljung-Box test on R^2, for lags up 10, 15, 20, we fail to reject zero autocorrelation. Based on the LM Arch test, we fail to reject the null that all ARCH coefficients of the standardized residuals are 0. All these test results and plots suggest that the model is a good fit for capturing the volatility clustering.

(c)

```
(armagarch.pred <- predict(fit.armagarch))
```

```
##    meanForecast  meanError standardDeviation
## 1  -0.003634847 0.01683942        0.01683942
## 2   0.001416508 0.01639363        0.01631576
## 3   0.001416508 0.01590454        0.01582919
## 4   0.001416508 0.01545068        0.01537767
## 5   0.001416508 0.01503009        0.01495926
## 6   0.001416508 0.01464085        0.01457204
## 7   0.001416508 0.01428112        0.01421418
## 8   0.001416508 0.01394911        0.01388390
## 9   0.001416508 0.01364309        0.01357949
## 10  0.001416508 0.01336142        0.01329929
```

```
(a <- armagarch.pred[1, 1])
```

```
## [1] -0.003634847
```

```r
(b <- armagarch.pred[1, 2])
```

```
## [1] 0.01683942
```

```r
(v <- coef(fit.armagarch)["shape"])
```

```
##    shape
## 4.080999
```

(d)

```r
q <- as.numeric(qstd(0.001, nu=coef(fit.armagarch)["shape"]))
(VaR <- -(q*b + a))
```

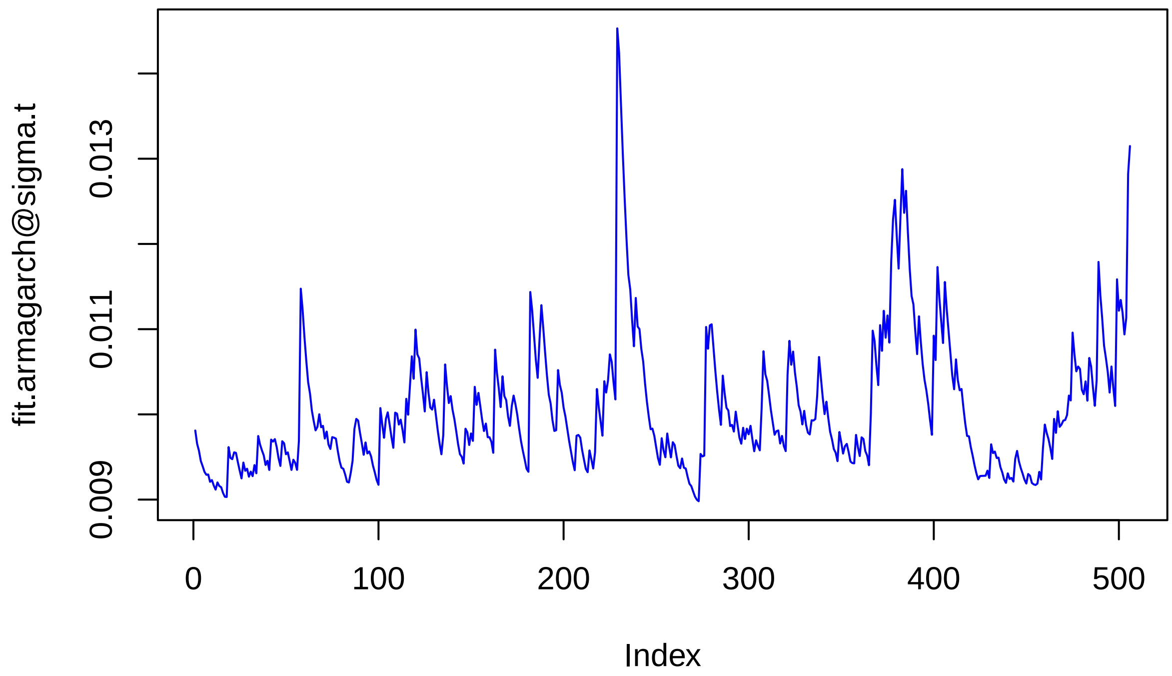
```
## [1] 0.08817832
```

```r
-(ret.black.mon) > VaR
```

```
## [1] TRUE
```

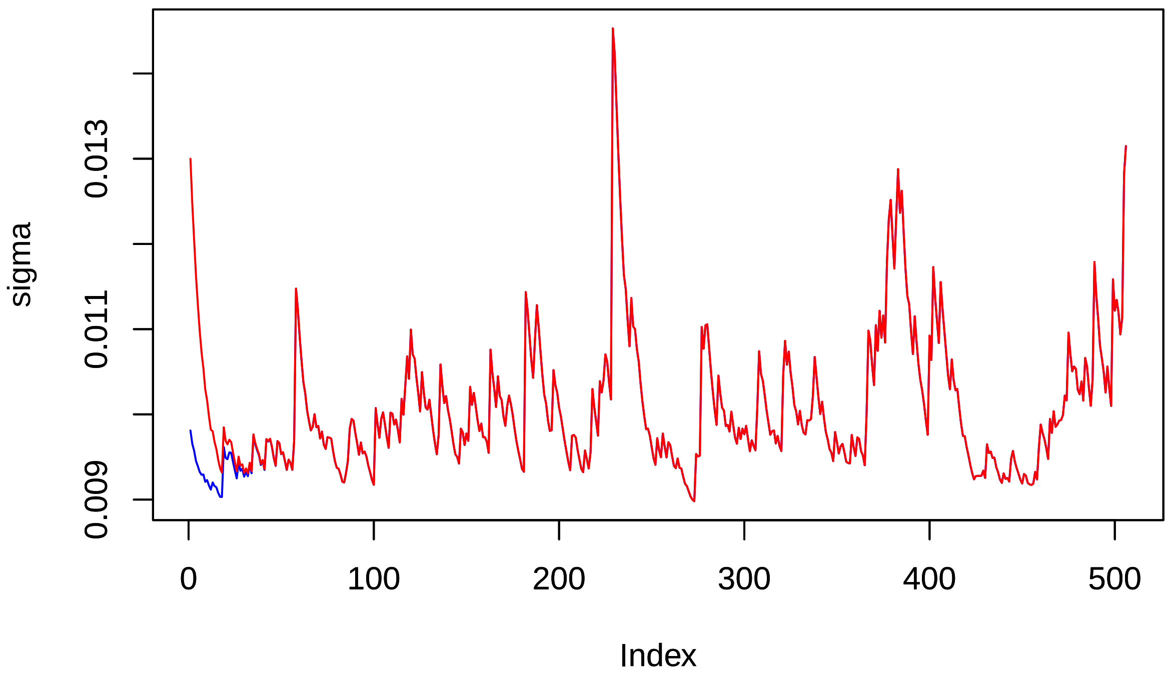
(e)

```r
plot(fit.armagarch@sigma.t, type='l', col='blue')
```



(f)

```
sigma <- rep(1, 506)
sigma[1] <- 0.013
at <- residuals(fit.armagarch, standardize=FALSE)

for (i in seq(2, 506)) {
  sigma[i] <- sqrt(coef(fit.armagarch)["omega"] +
                   coef(fit.armagarch)["alpha1"]*at[i-1]**2 +
                   coef(fit.armagarch)["beta1"]*sigma[i-1]**2)
}

plot(fit.armagarch@sigma.t, type='l', col='blue', ylab='')
par(new=TRUE)
plot(sigma, type='l', col='red')
```



sigma
0.009
0.011
0.013
0
100
200
300
400
500
Index

(g)

$$\sigma_t^2 = w + \alpha a_{t-1}^2 + \beta \sigma_{t-1}^2$$

$$= w + \alpha a_{t-1}^2 + \beta (w + \alpha a_{t-2}^2 + \beta \sigma_{t-2}^2)$$

$$= w(1+\beta) + \alpha a_{t-1}^2 + \alpha\beta a_{t-2}^2 + \beta^2 \sigma_{t-2}^2$$

$$\vdots$$

$$= w(1+\beta+\beta^2+\cdots \beta^{t-2}) + \alpha a_{t-1}^2 + \alpha\beta a_{t-2}^2 + \alpha\beta^2 a_{t-3}^2 + \cdots + \alpha\beta^{t-2} a_1^2$$

$$+ \beta^{t-1} \sigma_1^2$$

As t increases, the weight of σ_1^2 — β^{t-1} quickly decreases, so the value of σ_1 is less important and both converge to the same values.

Q3

Based on the lecture notes,

$$E[\sigma^2_{n+h} \mid \mathcal{F}_{n-1}] = \omega \frac{1-(\alpha+\beta)^h}{1-\alpha-\beta} + (\alpha+\beta)^h \sigma_n^2$$

$$= \frac{\omega}{1-\alpha-\beta} + (\alpha+\beta)^h \left(\sigma_n^2 - \frac{\omega}{1-\alpha-\beta} \right)$$

$$= v + e^{-ah} (\sigma_n^2 - v)$$

$$E[\sigma_n^2 \mid \mathcal{F}_{n-1}] = v + e^{0} (\sigma_n^2 - v)$$

$$E[\sigma_{n+1}^2 \mid \mathcal{F}_{n-1}] = v + e^{-a} (\sigma_n^2 - v)$$

$$E[\sigma_{n+2}^2 \mid \mathcal{F}_{n-1}] = v + e^{-2a} (\sigma_n^2 - v)$$

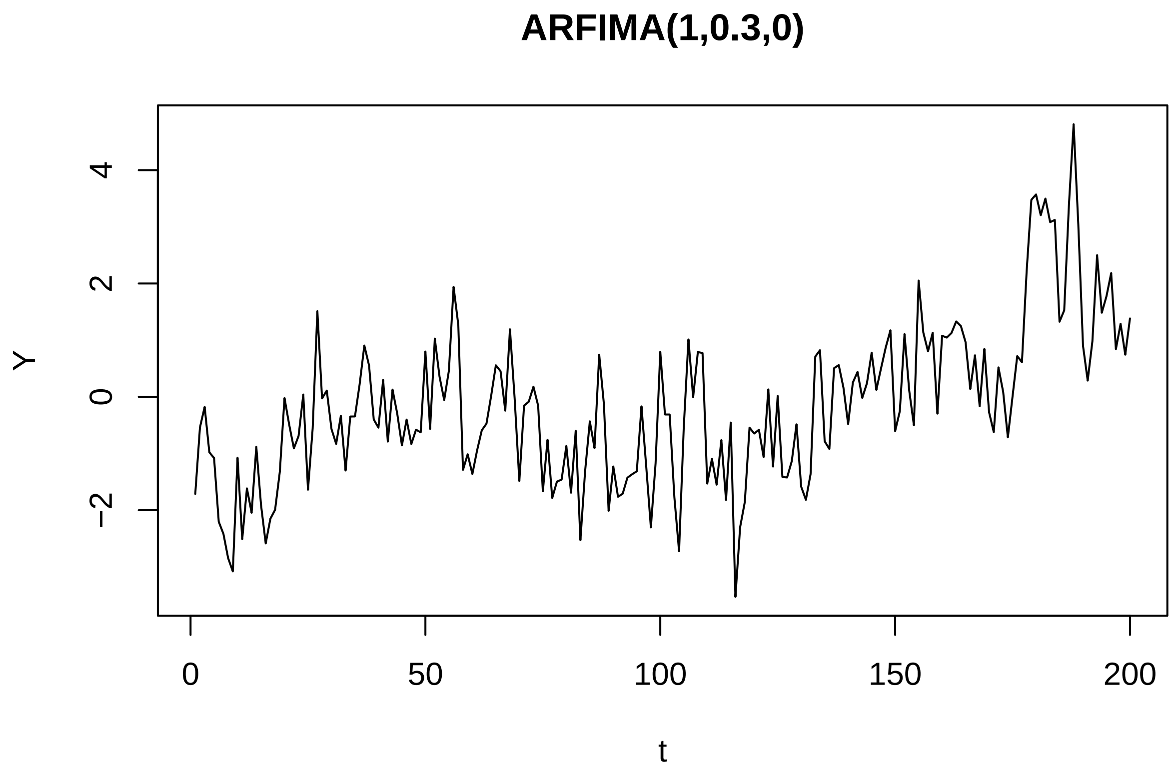
$$\vdots$$

$$\sum_{h=0}^{T-1} E[\sigma_{n+h}^2 \mid \mathcal{F}_{n-1}] = Tv + \frac{1-e^{-aT}}{1-e^{-a}} (\sigma_n^2 - v)$$

$$\frac{\sum_{h=0}^{T-1} E[\sigma_{n+h}^2 \mid \mathcal{F}_{n-1}]}{T} = v + \frac{1-e^{-aT}}{T(1-e^{-a})} (\sigma_n^2 - v)$$

Q4

```
library(fracdiff)
set.seed(343)

Y <- fracdiff.sim(200, ar=0.35, d=0.3, sd=1)$series
plot(Y, type="l", xlab="t", ylab="Y", main="ARFIMA(1,0.3,0)")
```



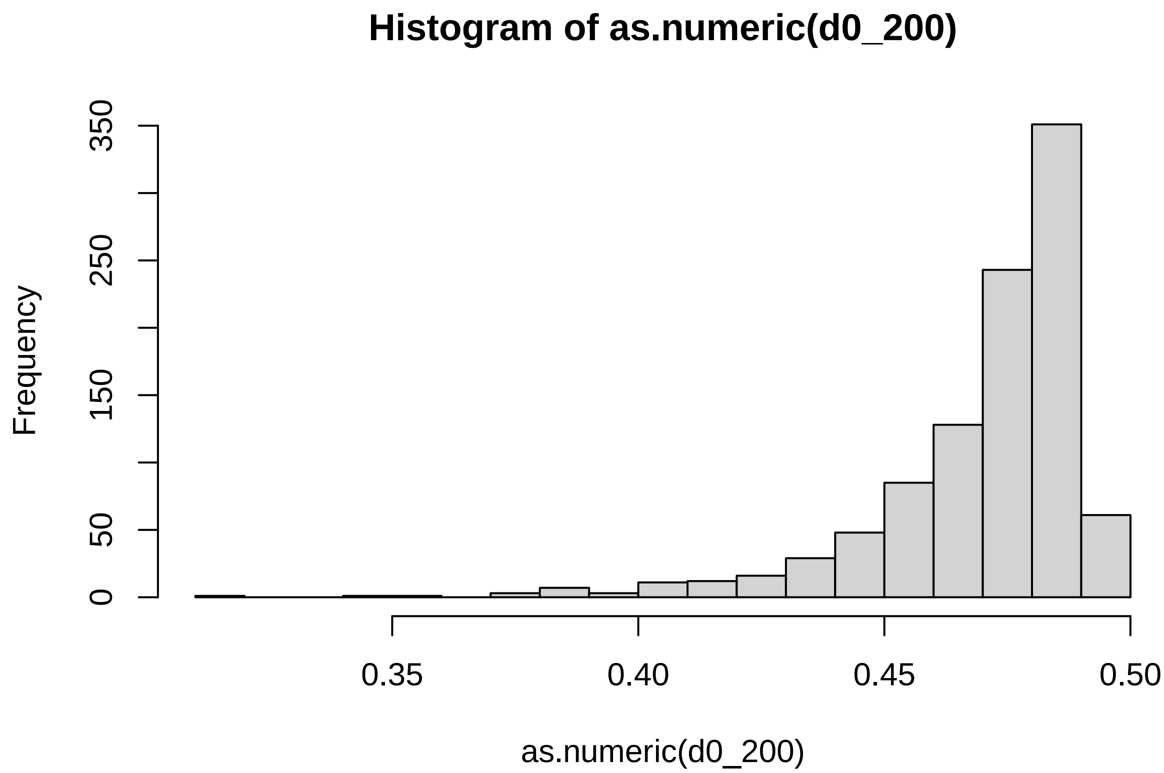

```
(d0 <- coef(fracdiff(Y, nar=0, nma=0))[1])
```

```
##         d
## 0.4812387
```

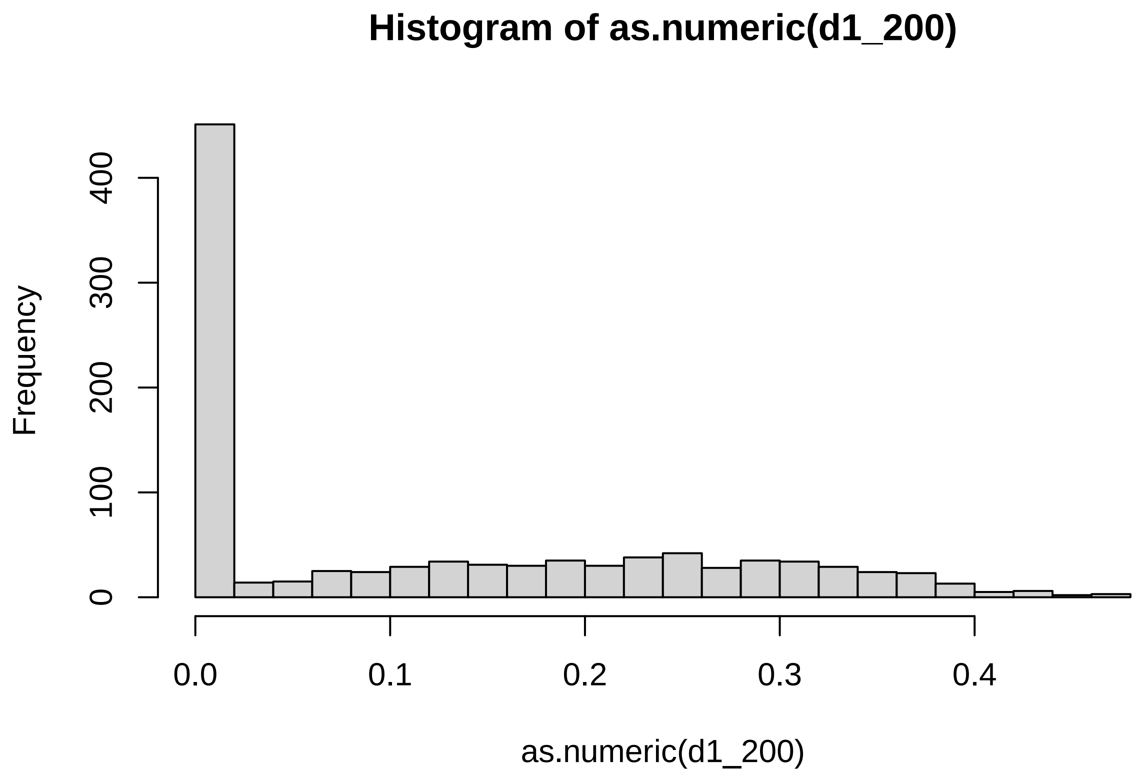
```
(d1 <- coef(fracdiff(Y, nar=1, nma=1))[1])
```

```
##         d
## 0.4338088
```

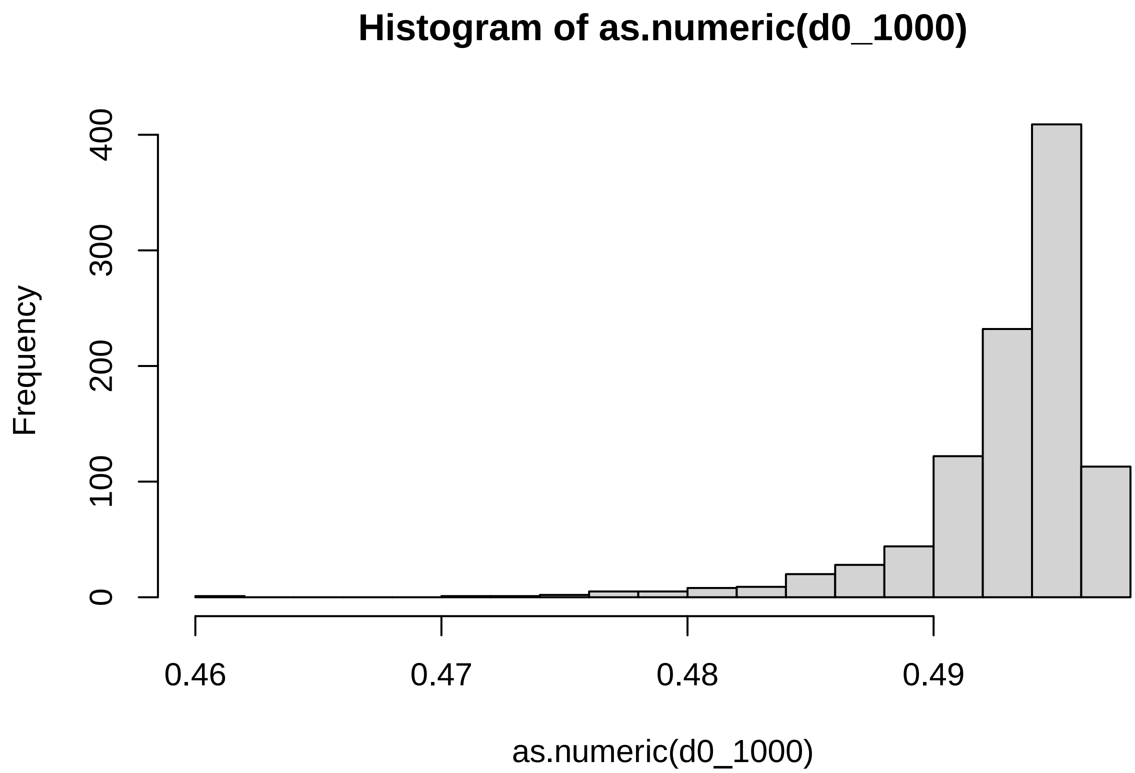
```
d0_200 <- rep(1, 1000)
d1_200 <- rep(1, 1000)

for (i in seq(1,1000)) {
  Y <- fracdiff.sim(200, ar=0.35, d=0.3, sd=1)$series
  d0_200[i] <- coef(fracdiff(Y, nar=0, nma=0))[1]
  d1_200[i] <- coef(fracdiff(Y, nar=1, nma=1))[1]
}

hist(as.numeric(d0_200), breaks=20)
```



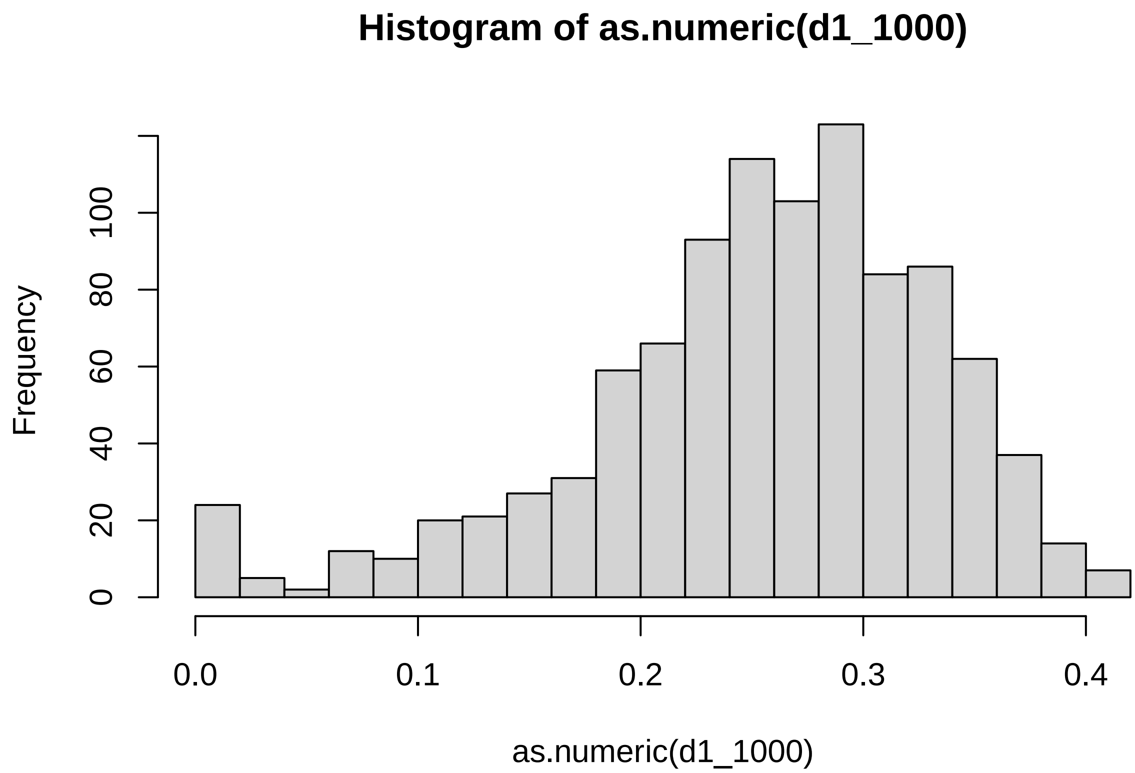

```
hist(as.numeric(d1_200), breaks=20)
```




```
d0_1000 <- rep(1, 1000)
d1_1000 <- rep(1, 1000)

for (i in seq(1,1000)) {
  Y <- fracdiff.sim(1000, ar=0.35, d=0.3, sd=1)$series
  d0_1000[i] <- coef(fracdiff(Y, nar=0, nma=0))[1]
  d1_1000[i] <- coef(fracdiff(Y, nar=1, nma=1))[1]
}

hist(as.numeric(d0_1000), breaks=20)
```




```
hist(as.numeric(d1_1000), breaks=20)
```



As the sample size increases, fitting an ARFIMA(1, d, 1) process is much better than fitting an ARFIMA(0, d, 0) in estimating d.