

MATH 8090: Univariate Volatility Modeling

Whitney Huang, Clemson University

10/24-10/26/2023

Contents

Introductory Example: Apple Stock Data	2
Load the Apple stock data via <i>quantmod</i>	2
Visualize the Apple stock time series.	3
ARCH Engle (1982)	9
Simulation	9
Intel Stock Example	11
Load and plot the monthly log returns of Intel stock	11
Examine the mean structure	12
Testing ARCH effect	14
Fitting ARCH	15
ARCH(1) with Student-t Innovations for 5-step predictions	21
GARCH Bollerslev (1986)	25
Simulation	25
IGARCH	31
Simulation	31
EGARCH Nelson (1991)	34
IBM monthly returns	34
Fit EGARCH using <i>ugarch</i>	36
Stochastic Volatility (SV) Model Melino and Turnbull (1990); Harvey, Ruiz, and Shephard (1994); Jacquier, Polson, and Rossi (2002)	38
Simulation	38
Euro exchange rate example	39
Perform Markov Chain Monte Carlo (MCMC) sampling for the Stochastic Volatility (SV) Model	40
References	44

Introductory Example: Apple Stock Data

Load the Apple stock data via *quantmod*

```
library(quantmod)
getSymbols("AAPL", src = "yahoo")
```

```
## [1] "AAPL"
```

```
dim(AAPL)
```

```
## [1] 4232    6
```

```
head(AAPL); tail(AAPL)
```

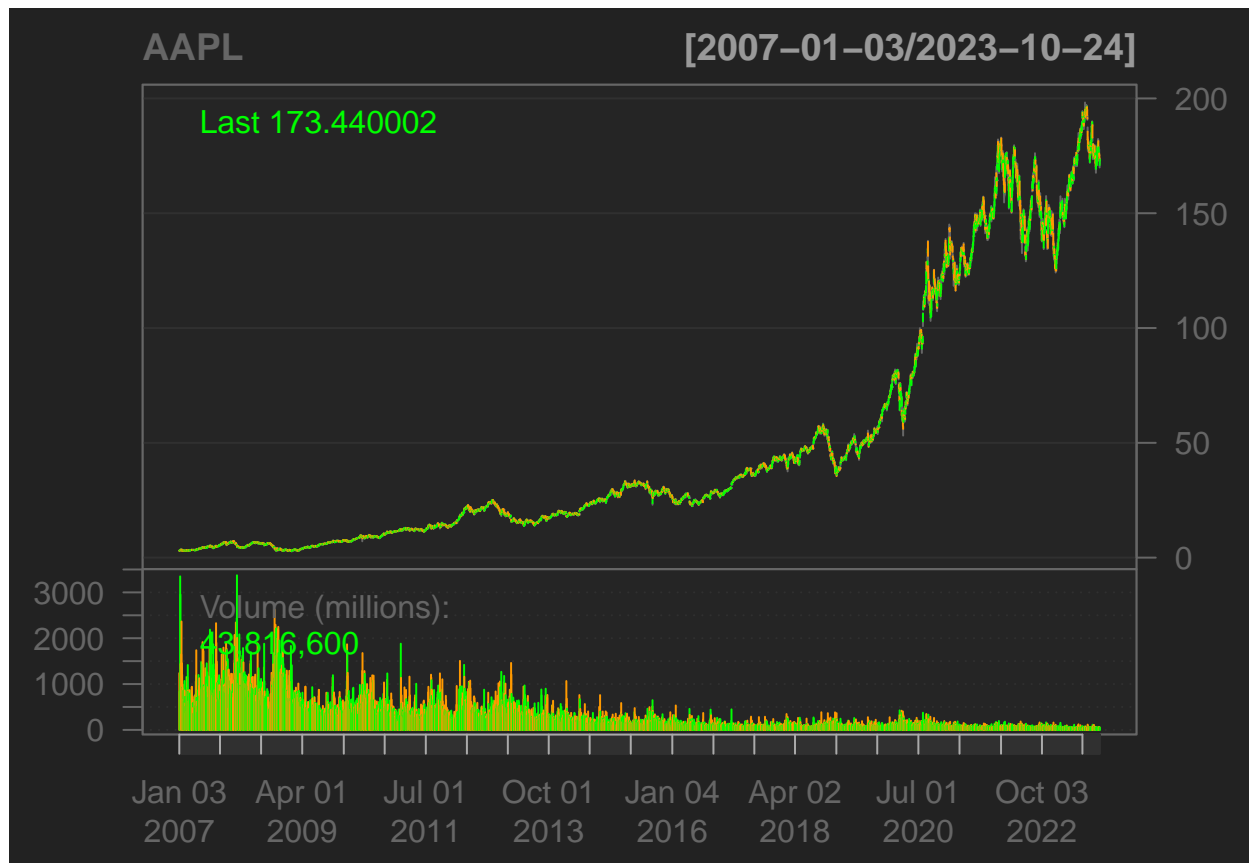
```
##           AAPL.Open AAPL.High AAPL.Low AAPL.Close AAPL.Volume AAPL.Adjusted
## 2007-01-03  3.081786  3.092143  2.925000  2.992857  1238319600    2.540326
## 2007-01-04  3.001786  3.069643  2.993571  3.059286   847260400    2.596712
## 2007-01-05  3.063214  3.078571  3.014286  3.037500   834741600    2.578219
## 2007-01-08  3.070000  3.090357  3.045714  3.052500   797106800    2.590951
## 2007-01-09  3.087500  3.320714  3.041071  3.306071  3349298400    2.806181
## 2007-01-10  3.383929  3.492857  3.337500  3.464286  2952880000    2.940474
```

```
##           AAPL.Open AAPL.High AAPL.Low AAPL.Close AAPL.Volume AAPL.Adjusted
## 2023-10-17   176.65   178.42   174.80   177.15   57549400    177.15
## 2023-10-18   175.58   177.58   175.11   175.84   54764400    175.84
## 2023-10-19   176.04   177.84   175.19   175.46   59302900    175.46
## 2023-10-20   175.31   175.42   172.64   172.88   64189300    172.88
## 2023-10-23   170.91   174.01   169.93   173.00   55980100    173.00
## 2023-10-24   173.05   173.67   171.45   173.44   43816600    173.44
```

```
summary(AAPL)
```

```
##           Index           AAPL.Open           AAPL.High           AAPL.Low
## Min.      :2007-01-03  Min.      : 2.835  Min.      : 2.929  Min.      : 2.793
## 1st Qu.: 2011-03-15  1st Qu.: 12.101  1st Qu.: 12.216  1st Qu.: 11.971
## Median   :2015-05-30  Median   : 26.704  Median   : 26.916  Median   : 26.407
## Mean     :2015-05-29  Mean     : 48.687  Mean     : 49.223  Mean     : 48.171
## 3rd Qu.:2019-08-12  3rd Qu.: 55.043  3rd Qu.: 55.645  3rd Qu.: 54.331
## Max.     :2023-10-24  Max.     :196.240  Max.     :198.230  Max.     :195.280
##           AAPL.Close           AAPL.Volume           AAPL.Adjusted
## Min.      : 2.793  Min.      :3.146e+07  Min.      : 2.371
## 1st Qu.: 12.099  1st Qu.:1.045e+08  1st Qu.: 10.270
## Median   : 26.684  Median :2.069e+08  Median   : 24.331
## Mean     : 48.719  Mean     :3.608e+08  Mean     : 46.977
## 3rd Qu.: 55.119  3rd Qu.:4.875e+08  3rd Qu.: 53.181
## Max.     :196.450  Max.     :3.373e+09  Max.     :196.185
```

```
chartSeries(AAPL)
```



Visualize the Apple stock time series.

First, let's plot the daily closing values

```
closing <- AAPL$AAPL.Close  
plot(closing)
```

closing

2007-01-03 / 2023-10-24



Next, apply a log transformation to stabilize the variance.

```
plot(log(closing))
```

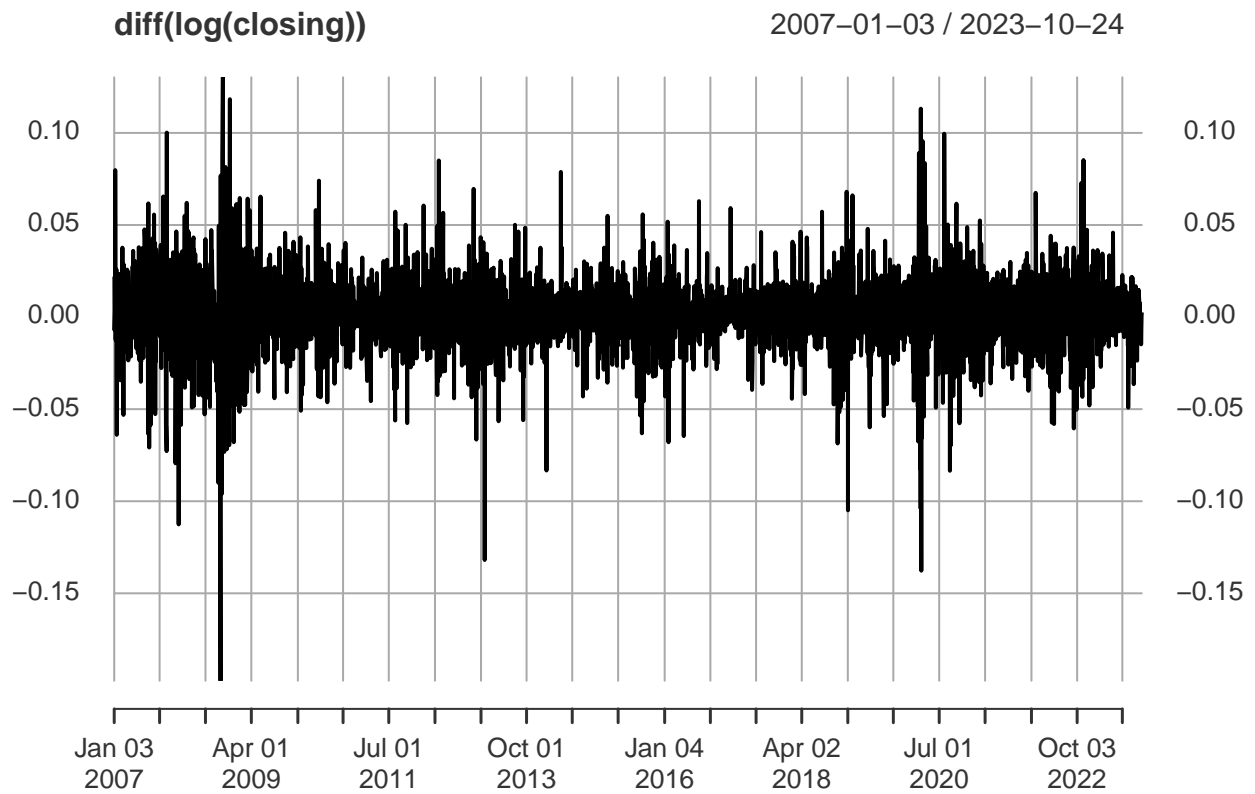
log(closing)

2007-01-03 / 2023-10-24



Perform first-order differencing to make the series approximately stationary

```
plot(diff(log(closing)))
```



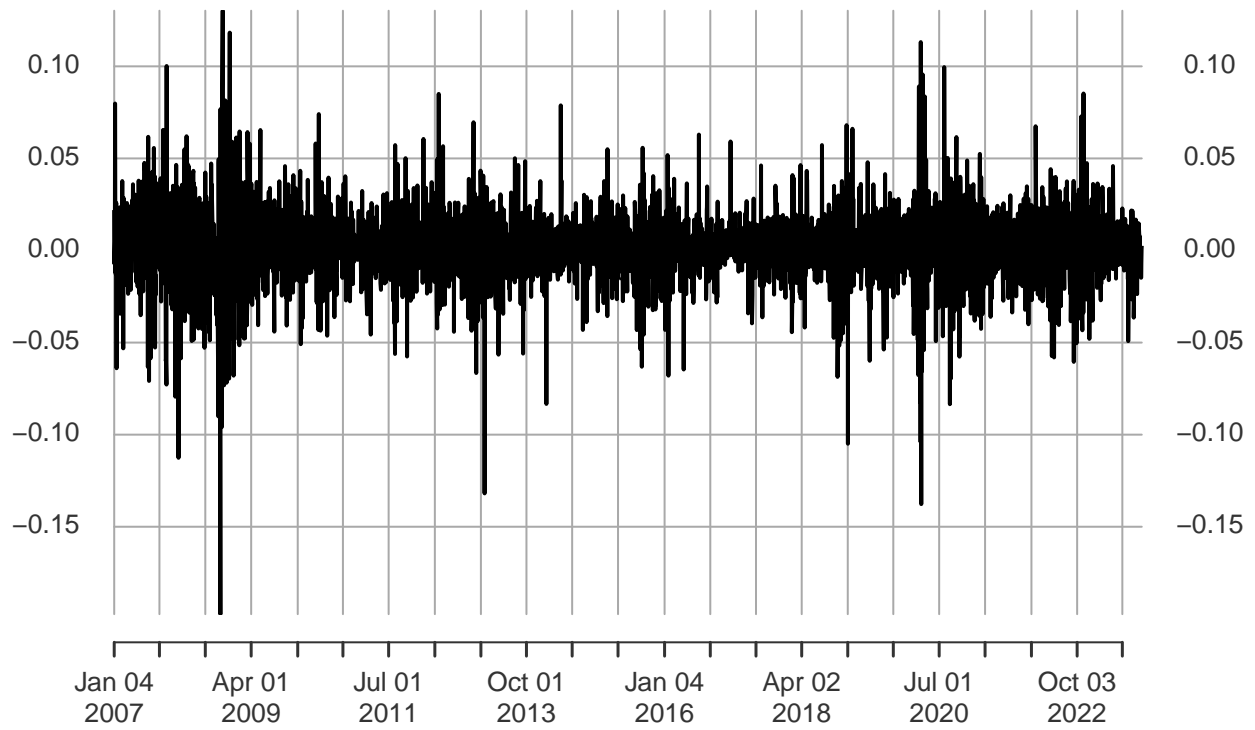
These series of operations lead to the log-return $r_t = \log(y_t) - \log(y_{t-1})$

```
temp <- diff(log(closing))
return <- temp[!is.na(temp)]

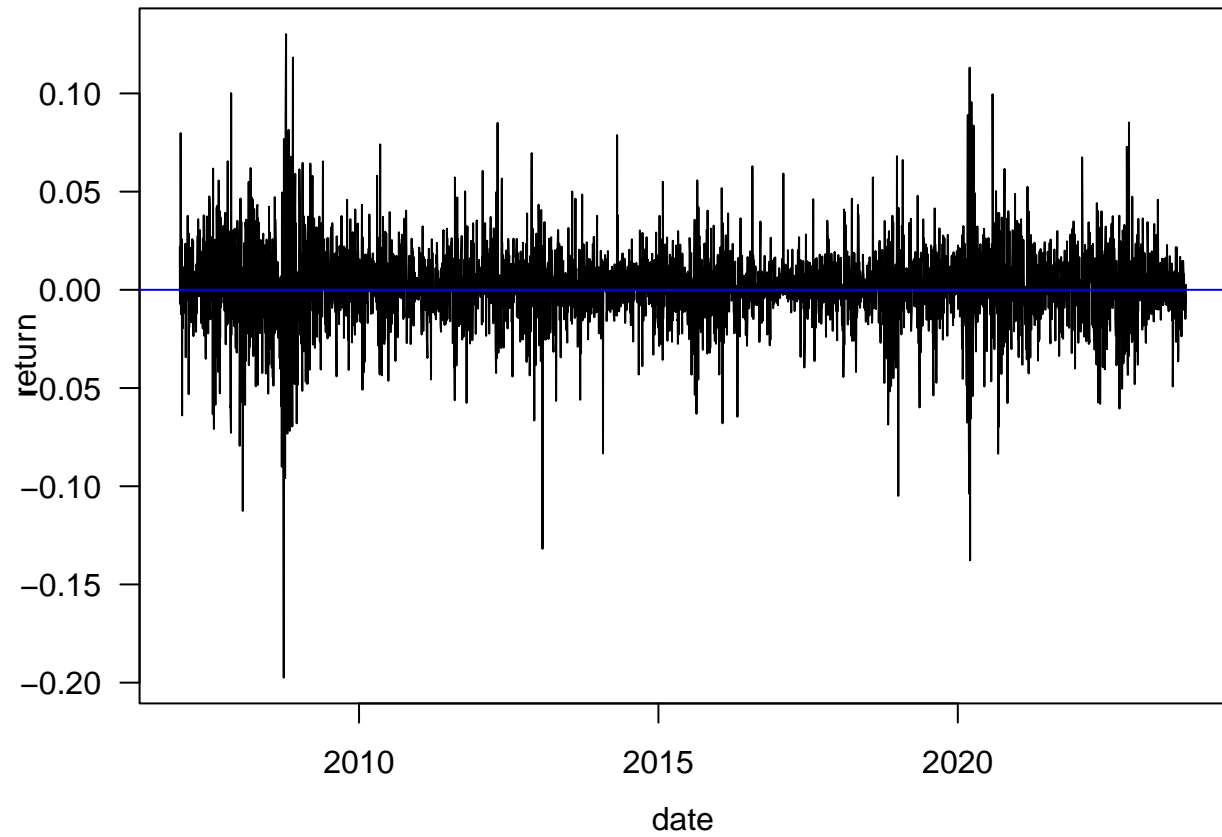
par(las = 1, mgp = c(2.2, 1, 0), mar = c(3.6, 3.6, 0.8, 0.6))
plot(return)
```

return

2007-01-04 / 2023-10-24



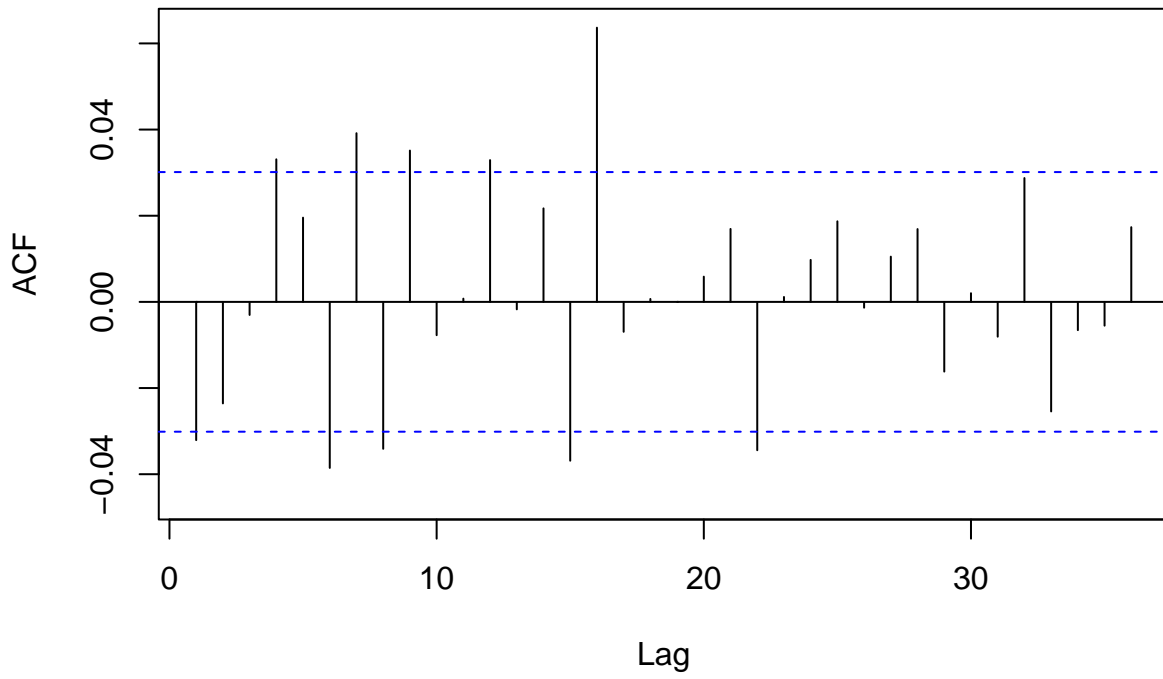
```
date <- time(return)
par(las = 1, mgp = c(2.5, 1, 0), mar = c(3.6, 3.6, 0.8, 0.6))
plot(date, return, type = "l")
abline(h = 0, col = "blue", lwd = 1)
```



The resulting series is nearly uncorrelated but clearly dependent

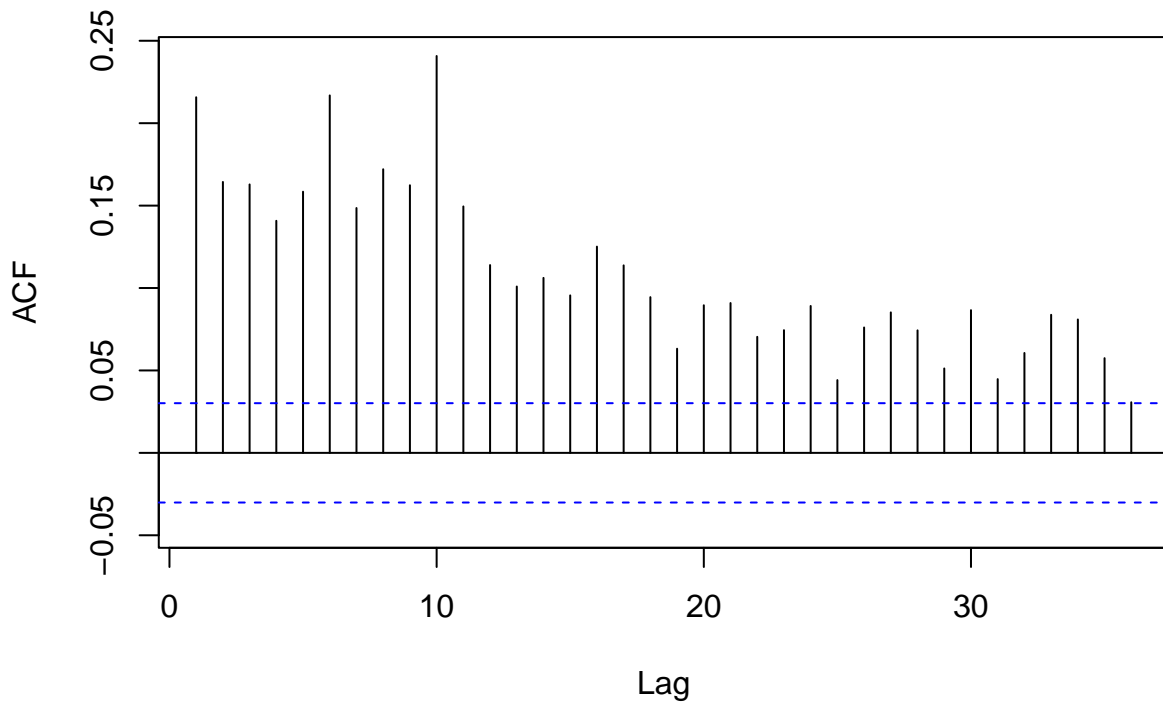
```
library(forecast)  
Acf(return)
```

Series return



```
Acf(return^2)
```

Series return^2



ARCH Engle (1982)

An ARCH(m) model:

$$a_t = \sigma_t \epsilon_t, \quad \sigma_t^2 = \alpha_0 + \alpha_1 a_{t-1}^2 + \cdots + \alpha_m a_{t-m}^2,$$

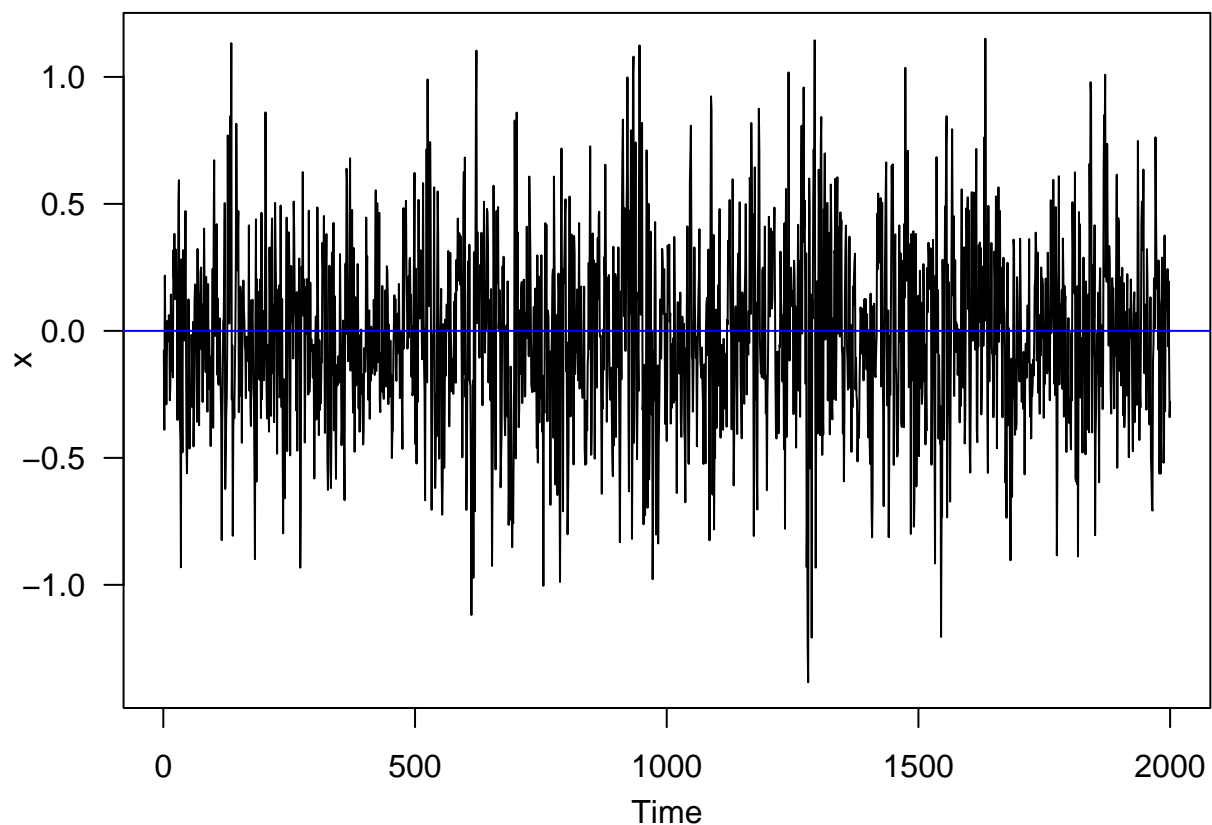
where $\{\epsilon_t\}$ is a sequence of i.i.d. r.v. with

- $\mathbb{E}(\epsilon_t) = 0$
- $\text{Var}(\epsilon_t) = 1$
- $\alpha_i \geq 0$ for $1 \leq i \leq m$
- *Distribution*: standard normal, standardize Student-t, generalized error distribution, or their skewed counterparts

Simulation

We will use both the *fGarch* and *rugarch* packages for volatility modeling.

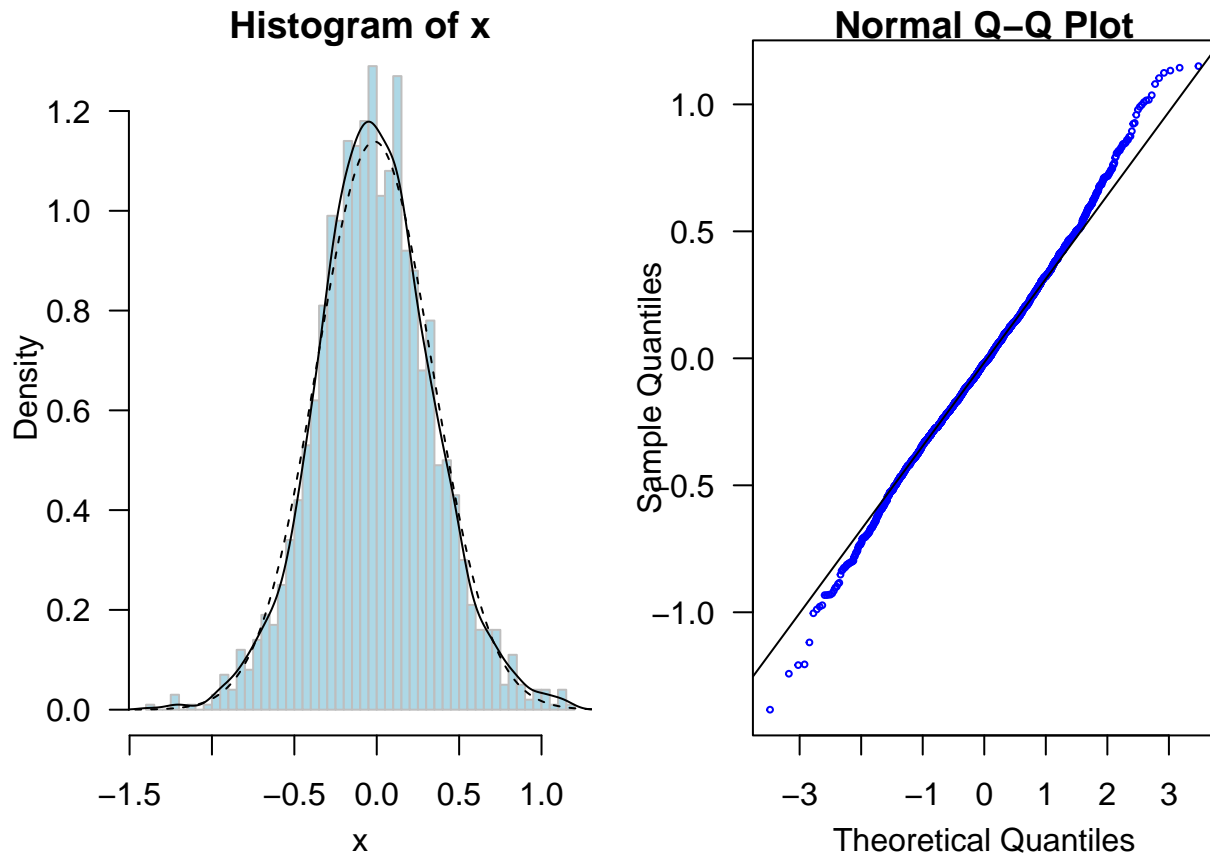
```
library(fGarch)
mod_spec <- garchSpec(model = list(ar = c(.35), omega = 0.01))
set.seed(124)
x <- garchSim(spec = mod_spec, n = 2000)
par(las = 1, mgp = c(2.2, 1, 0), mar = c(3.6, 3.6, 0.8, 0.6))
ts.plot(x)
abline(h = 0, col = "blue")
```



```

par(las = 1, mgp = c(2.2, 1, 0), mar = c(3.6, 3.6, 0.8, 0.6), mfrow = c(1, 2))
hist(x, nclass = 40, col = "lightblue", border = "gray", prob = T)
den <- density(x)
rg <- 1.2 * range(x)
xg <- seq(rg[1], rg[2], .001)
yg <- dnorm(xg, mean(x), stdev(x))
lines(den$x, den$y, xlab = "", ylab = "Density", type = "l")
lines(xg, yg, lty = 2)
qqnorm(x, col = "blue", cex = 0.4); qqline(x)

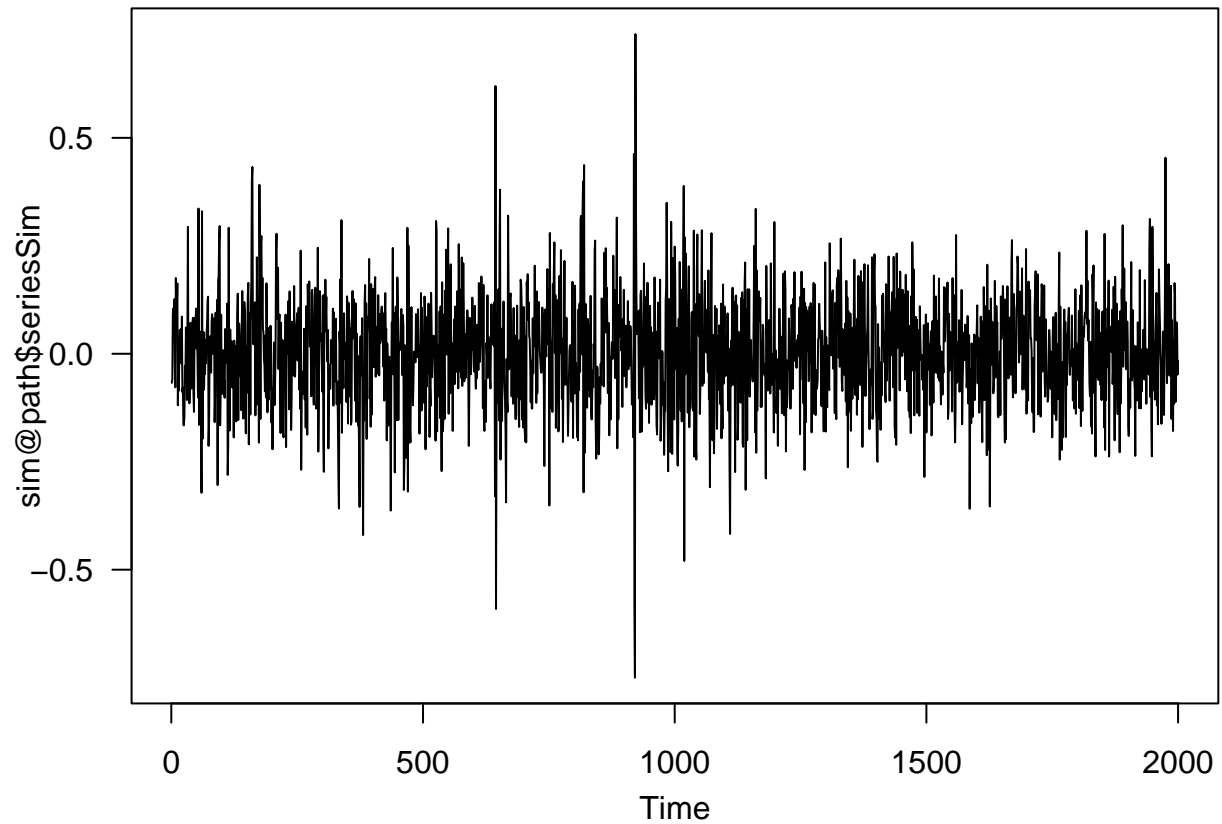
```



```

library(rugarch)
par(las = 1, mgp = c(2.2, 1, 0), mar = c(3.6, 3.6, 0.8, 0.6))
mod_spec <- ugarchspec(variance.model = list(model = "sGARCH", garchOrder = c(1, 0)), mean.model = list
sim <- ugarchpath(mod_spec, n.sim = 2000)
ts.plot(sim@path$seriesSim)

```



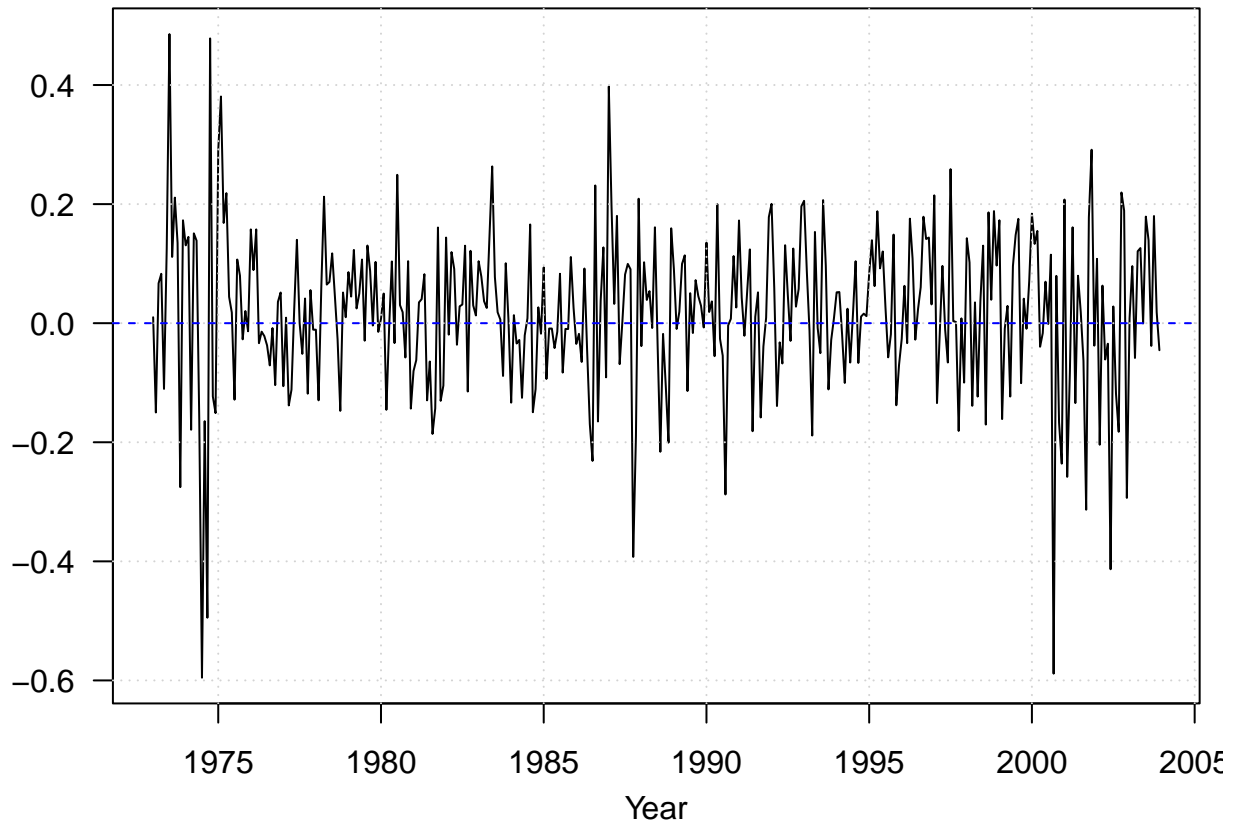
Intel Stock Example

Load and plot the monthly log returns of Intel stock

```
url <- "https://www.chicagobooth.edu/-/media/faculty/ruey-s-tsay/teaching/fts2//m-intc7303.txt"
dat1 <- read.table(url)
names(dat1) <- c("Date", "Return"); head(dat1)
```

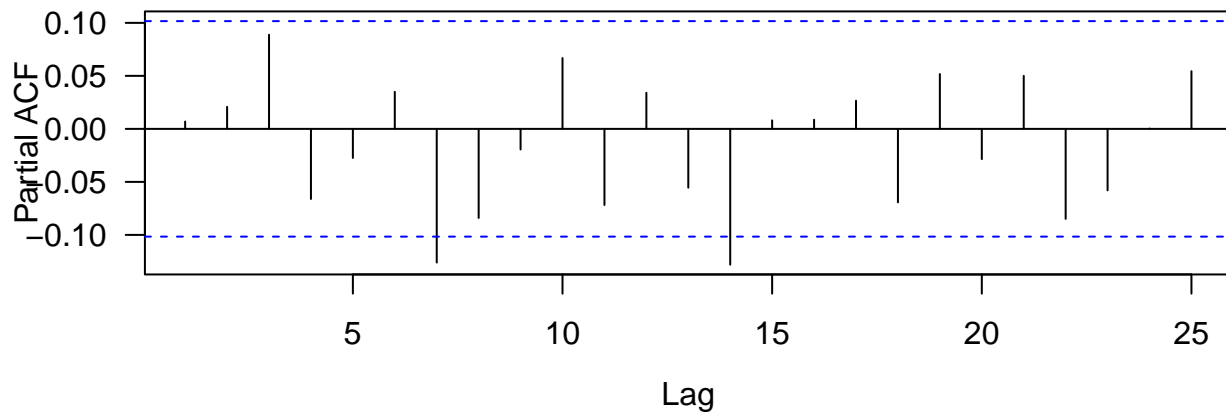
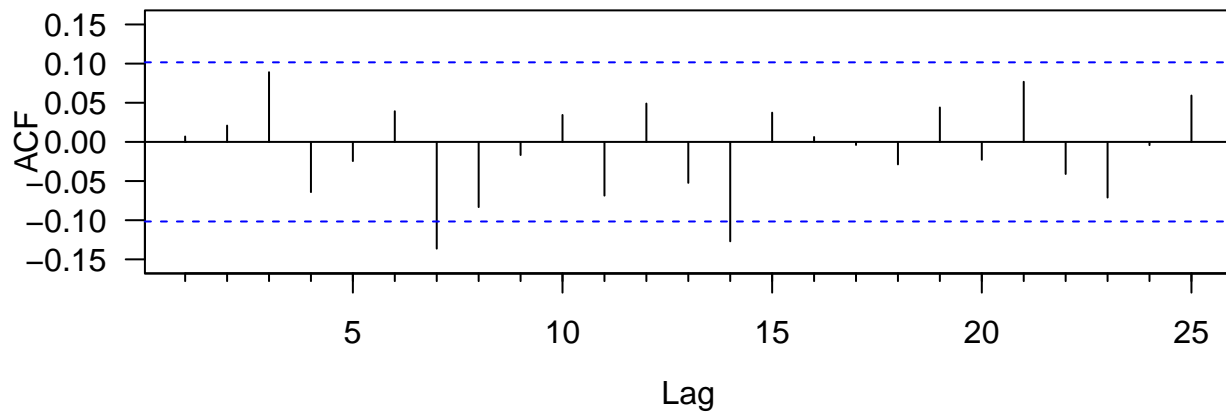
```
##      Date   Return
## 1 19730131  0.01005
## 2 19730228 -0.13930
## 3 19730330  0.06936
## 4 19730430  0.08649
## 5 19730531 -0.10448
## 6 19730629  0.13333
```

```
intc <- log(dat1$Return + 1)
return <- ts(intc, frequency = 12, start = c(1973, 1))
par(las = 1, mgp = c(2.2, 1, 0), mar = c(3.6, 3.6, 0.8, 0.6))
plot(return, type = "l", xlab = "Year", ylab = "")
grid()
abline(h = 0, lty = 2, col = "blue")
```



Examine the mean structure

```
par(las = 1, mgp = c(2.6, 1, 0), mar = c(3.6, 3.6, 0.8, 0.6), mfrow = c(2, 1))  
Acf(intc); pacf(intc)
```



```
Box.test(intc, lag = 24, type = "Ljung-Box")
```

```
##
## Box-Ljung test
##
## data: intc
## X-squared = 32.764, df = 24, p-value = 0.1092
```

```
t.test(intc)
```

```
##
## One Sample t-test
##
## data: intc
## t = 2.5944, df = 371, p-value = 0.00985
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
## 0.004354971 0.031624693
## sample estimates:
## mean of x
## 0.01798983
```

```
y <- intc - mean(intc)
```

Testing ARCH effect

```
Box.test(y^2, lag = 24, type = 'Ljung')
```

```
##  
## Box-Ljung test  
##  
## data: y^2  
## X-squared = 79.837, df = 24, p-value = 6.459e-08
```

```
# LM test for ARCH effects  
source("archTest.R")  
archTest(y, 12)
```

```
##  
## Call:  
## lm(formula = atsq ~ x)  
##  
## Residuals:  
##      Min       1Q   Median       3Q      Max  
## -0.07368 -0.01295 -0.00729  0.00450  0.35621  
##  
## Coefficients:  
##              Estimate Std. Error t value Pr(>|t|)  
## (Intercept)  0.007029   0.002752   2.554  0.01107 *  
## x1           0.090001   0.052911   1.701  0.08984 .  
## x2           0.155741   0.052830   2.948  0.00342 **  
## x3           0.148341   0.053414   2.777  0.00578 **  
## x4           0.020289   0.053994   0.376  0.70732  
## x5           0.004670   0.053971   0.087  0.93110  
## x6           0.007733   0.051753   0.149  0.88131  
## x7           0.055361   0.051756   1.070  0.28552  
## x8           0.009982   0.051805   0.193  0.84731  
## x9           0.002042   0.051674   0.040  0.96850  
## x10          -0.021888   0.051218  -0.427  0.66939  
## x11          -0.057741   0.050622  -1.141  0.25481  
## x12           0.162048   0.050563   3.205  0.00148 **  
## ---  
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1  
##  
## Residual standard error: 0.03689 on 347 degrees of freedom  
## Multiple R-squared:  0.1189, Adjusted R-squared:  0.0884  
## F-statistic: 3.901 on 12 and 347 DF, p-value: 1.236e-05
```

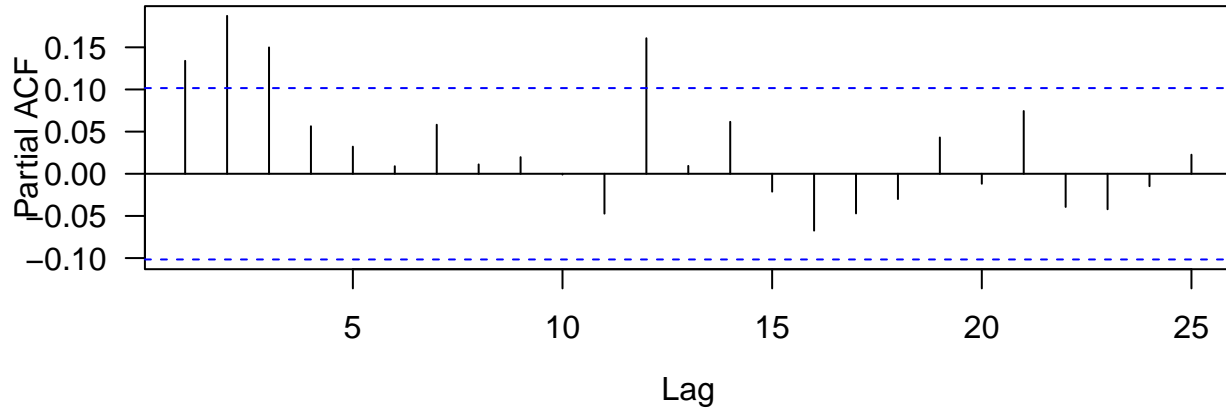
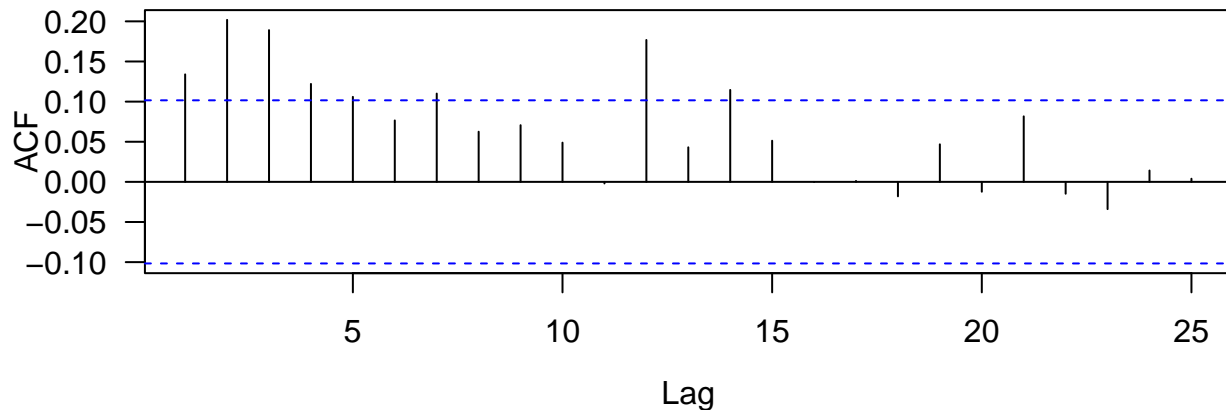
```
library(FinTS)
```

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

```
ArchTest(y)
```

```
##  
## ARCH LM-test; Null hypothesis: no ARCH effects  
##  
## data: y  
## Chi-squared = 42.794, df = 12, p-value = 2.446e-05
```

```
par(las = 1, mgp = c(2.6, 1, 0), mar = c(3.6, 3.6, 0.8, 0.6), mfrow = c(2, 1))  
Acf(y^2)  
pacf(y^2)
```



Fitting ARCH

First, let's fit an ARCH(3) model

```
# fGarch  
Intel_m1 <- garchFit(~ 1 + garch(3, 0), data = intc, trace = F)  
summary(Intel_m1)
```

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

```

## Call:
## garchFit(formula = ~1 + garch(3, 0), data = intc, trace = F)
##
## Mean and Variance Equation:
## data ~ 1 + garch(3, 0)
## <environment: 0x7faad5d989e8>
## [data = intc]
##
## Conditional Distribution:
## norm
##
## Coefficient(s):
##      mu      omega    alpha1    alpha2    alpha3
## 0.016572 0.012043 0.208649 0.071837 0.049045
##
## Std. Errors:
## based on Hessian
##
## Error Analysis:
##      Estimate Std. Error t value Pr(>|t|)
## mu      0.016572 0.006423 2.580 0.00988 **
## omega   0.012043 0.001579 7.627 2.4e-14 ***
## alpha1  0.208649 0.129177 1.615 0.10626
## alpha2  0.071837 0.048551 1.480 0.13897
## alpha3  0.049045 0.048847 1.004 0.31536
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Log Likelihood:
## 233.4286 normalized: 0.6274962
##
## Description:
## Wed Oct 25 21:37:41 2023 by user:
##
##
## Standardised Residuals Tests:
##
##      Statistic p-Value
## Jarque-Bera Test R Chi^2 169.773 0
## Shapiro-Wilk Test R W 0.960696 1.970626e-08
## Ljung-Box Test R Q(10) 10.97025 0.3598405
## Ljung-Box Test R Q(15) 19.59024 0.1882211
## Ljung-Box Test R Q(20) 20.82192 0.40768
## Ljung-Box Test R^2 Q(10) 5.376602 0.8646439
## Ljung-Box Test R^2 Q(15) 22.7346 0.08993974
## Ljung-Box Test R^2 Q(20) 23.70577 0.255481
## LM Arch Test R TR^2 20.48506 0.05844884
##
## Information Criterion Statistics:
##      AIC      BIC      SIC      HQIC
## -1.228111 -1.175437 -1.228466 -1.207193

```

```

# rugarch
M1 = ugarchspec(variance.model = list(model = "sGARCH", garchOrder = c(3, 0)),
                mean.model = list(armaOrder = c(0, 0), include.mean = T),

```



```
distribution.model = "norm", fixed.pars = list())
(Intel_m1 <- ugarchfit(M1, data = intc))
```

```
##
## *-----*
## *          GARCH Model Fit          *
## *-----*
##
## Conditional Variance Dynamics
## -----
## GARCH Model   : sGARCH(3,0)
## Mean Model    : ARFIMA(0,0,0)
## Distribution   : norm
##
## Optimal Parameters
## -----
##      Estimate  Std. Error  t value  Pr(>|t|)
## mu      0.016560   0.006419   2.57981  0.009885
## omega   0.012050   0.001591   7.57490  0.000000
## alpha1  0.212953   0.131646   1.61763  0.105743
## alpha2  0.071933   0.048928   1.47016  0.141519
## alpha3  0.049129   0.049221   0.99813  0.318219
##
## Robust Standard Errors:
##      Estimate  Std. Error  t value  Pr(>|t|)
## mu      0.016560   0.006895   2.4016  0.016322
## omega   0.012050   0.002493   4.8345  0.000001
## alpha1  0.212953   0.193580   1.1001  0.271297
## alpha2  0.071933   0.036720   1.9590  0.050118
## alpha3  0.049129   0.031957   1.5373  0.124211
##
## LogLikelihood : 233.4331
##
## Information Criteria
## -----
## Akaike          -1.2281
## Bayes           -1.1755
## Shibata         -1.2285
## Hannan-Quinn   -1.2072
##
## Weighted Ljung-Box Test on Standardized Residuals
## -----
##              statistic  p-value
## Lag[1]              0.03327  0.8553
## Lag[2*(p+q)+(p+q)-1][2]  0.06686  0.9435
## Lag[4*(p+q)+(p+q)-1][5]  2.04550  0.6077
## d.o.f=0
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## -----
##              statistic  p-value
```

```

## Lag[1]                                0.5564  0.4557
## Lag[2*(p+q)+(p+q)-1][8]              1.4875  0.9287
## Lag[4*(p+q)+(p+q)-1][14]             6.7064  0.5442
## d.o.f=3
##
## Weighted ARCH LM Tests
## -----
##           Statistic Shape Scale P-Value
## ARCH Lag[4]    0.5757 0.500 2.000  0.4480
## ARCH Lag[6]    0.8634 1.461 1.711  0.7873
## ARCH Lag[8]    1.7352 2.368 1.583  0.7945
##
## Nyblom stability test
## -----
## Joint Statistic:  1.8622
## Individual Statistics:
## mu      0.04824
## omega   0.31431
## alpha1  0.25826
## alpha2  0.57419
## alpha3  0.20981
##
## Asymptotic Critical Values (10% 5% 1%)
## Joint Statistic:      1.28 1.47 1.88
## Individual Statistic:  0.35 0.47 0.75
##
## Sign Bias Test
## -----
##           t-value  prob sig
## Sign Bias      0.22799 0.8198
## Negative Sign Bias 0.07266 0.9421
## Positive Sign Bias 0.27306 0.7850
## Joint Effect    0.10621 0.9911
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## -----
##   group statistic p-value(g-1)
## 1    20     32.95     0.02439
## 2    30     44.29     0.03444
## 3    40     47.78     0.15799
## 4    50     64.29     0.07027
##
##
## Elapsed time : 0.110934

```

Let's fit an ARCH(1) model

```
Intel_m2 <- garchFit(~ 1 + garch(1, 0), data = intc, trace = F)
```

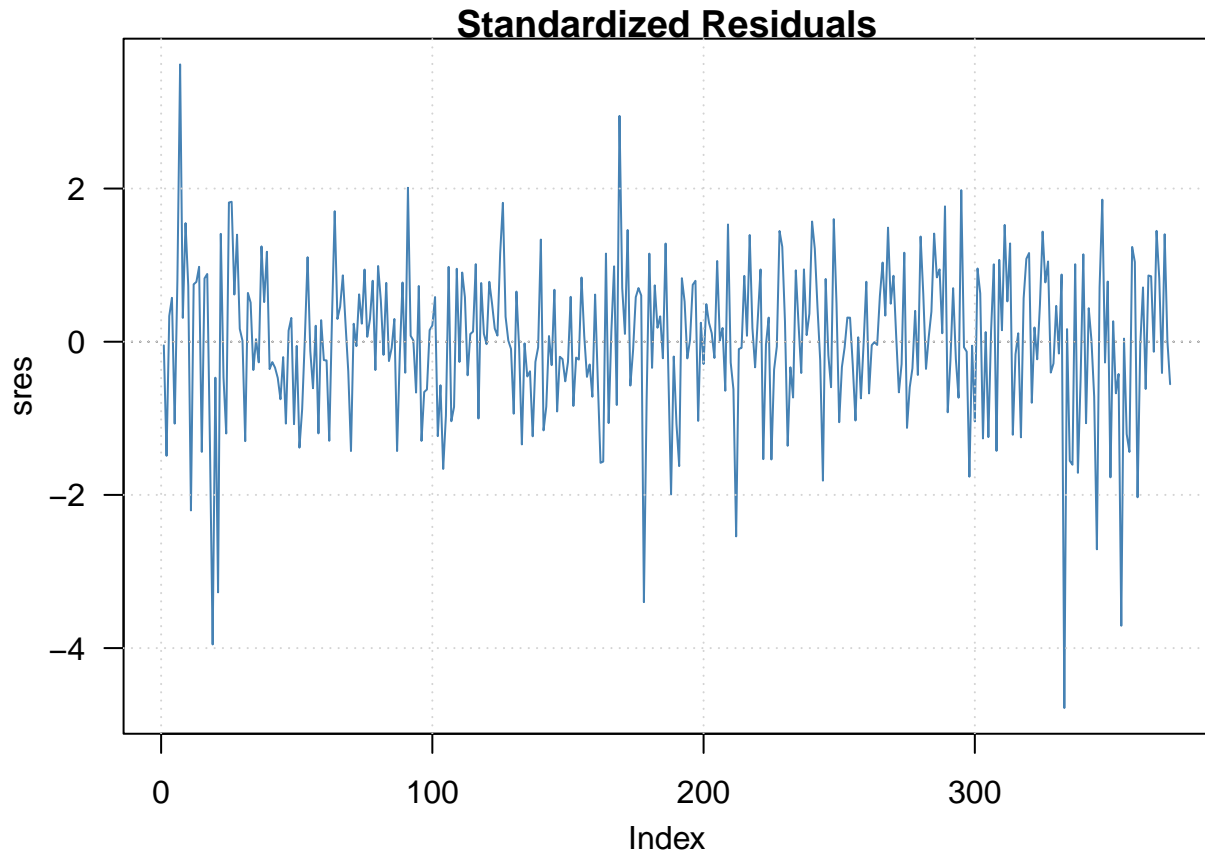
```
## Warning: Using formula(x) is deprecated when x is a character vector of length > 1.
## Consider formula(paste(x, collapse = " ")) instead.
```

```
summary(Intel_m2)
```

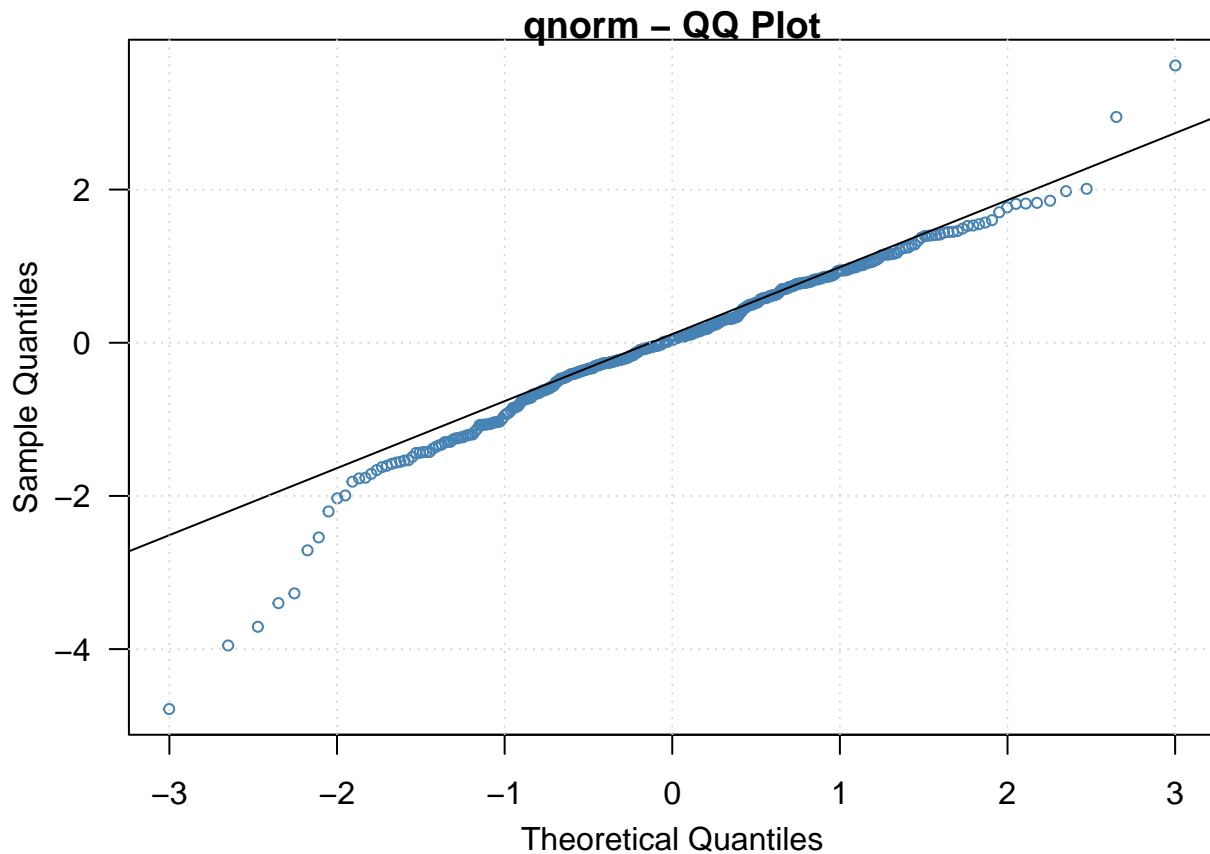
```
##
## Title:
## GARCH Modelling
##
## Call:
## garchFit(formula = ~1 + garch(1, 0), data = intc, trace = F)
##
## Mean and Variance Equation:
## data ~ 1 + garch(1, 0)
## <environment: 0x7faae6967b00>
## [data = intc]
##
## Conditional Distribution:
## norm
##
## Coefficient(s):
##      mu      omega  alpha1
## 0.01657 0.01249 0.36345
##
## Std. Errors:
## based on Hessian
##
## Error Analysis:
##      Estimate Std. Error t value Pr(>|t|)
## mu      0.016570   0.006161   2.689 0.00716 **
## omega   0.012490   0.001549   8.061 6.66e-16 ***
## alpha1  0.363447   0.131598   2.762 0.00575 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Log Likelihood:
## 230.2423      normalized: 0.6189309
##
## Description:
## Wed Oct 25 21:37:41 2023 by user:
##
##
## Standardised Residuals Tests:
##
##      Statistic p-Value
## Jarque-Bera Test  R    Chi^2 122.404 0
## Shapiro-Wilk Test R    W    0.9647625 8.273101e-08
## Ljung-Box Test   R    Q(10) 13.72604 0.1858587
## Ljung-Box Test   R    Q(15) 22.31714 0.09975386
## Ljung-Box Test   R    Q(20) 23.88257 0.2475594
## Ljung-Box Test   R^2  Q(10) 12.50025 0.25297
## Ljung-Box Test   R^2  Q(15) 30.11276 0.01152131
## Ljung-Box Test   R^2  Q(20) 31.46404 0.04935483
## LM Arch Test     R    TR^2 22.036 0.0371183
##
## Information Criterion Statistics:
##      AIC      BIC      SIC      HQIC
```

```
## -1.221733 -1.190129 -1.221861 -1.209182
```

```
par(las = 1, mgp = c(2.2, 1, 0), mar = c(3.6, 3.6, 0.8, 0.6))  
plot(Intel_m2, which = 9)
```



```
plot(Intel_m2, which = 13)
```



ARCH(1) with Student-t Innovations for 5-step predictions

```
Intel_m3 <- garchFit(~ 1 + garch(1, 0), data = intc, cond.dist = "std", trace = F)
```

```
## Warning: Using formula(x) is deprecated when x is a character vector of length > 1.
## Consider formula(paste(x, collapse = " ")) instead.
```

```
summary(Intel_m3)
```

```
##
## Title:
## GARCH Modelling
##
## Call:
## garchFit(formula = ~1 + garch(1, 0), data = intc, cond.dist = "std",
## trace = F)
##
## Mean and Variance Equation:
## data ~ 1 + garch(1, 0)
## <environment: 0x7faaf18ca390>
## [data = intc]
##
## Conditional Distribution:
```

```

## std
##
## Coefficient(s):
##      mu      omega    alpha1    shape
## 0.021571 0.013424 0.259867 5.985979
##
## Std. Errors:
## based on Hessian
##
## Error Analysis:
##      Estimate Std. Error t value Pr(>|t|)
## mu      0.021571 0.006054 3.563 0.000366 ***
## omega   0.013424 0.001968 6.820 9.09e-12 ***
## alpha1  0.259867 0.119901 2.167 0.030209 *
## shape   5.985979 1.660030 3.606 0.000311 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Log Likelihood:
## 242.9678      normalized: 0.6531391
##
## Description:
## Wed Oct 25 21:37:42 2023 by user:
##
##
## Standardised Residuals Tests:
##
##              Statistic p-Value
## Jarque-Bera Test  R      Chi^2 130.8931 0
## Shapiro-Wilk Test R      W      0.9637533 5.744995e-08
## Ljung-Box Test   R      Q(10) 14.31288 0.1591926
## Ljung-Box Test   R      Q(15) 23.34043 0.07717449
## Ljung-Box Test   R      Q(20) 24.87286 0.2063387
## Ljung-Box Test   R^2    Q(10) 15.35917 0.1195054
## Ljung-Box Test   R^2    Q(15) 33.96318 0.003446127
## Ljung-Box Test   R^2    Q(20) 35.46828 0.01774746
## LM Arch Test     R      TR^2 24.11517 0.01961957
##
## Information Criterion Statistics:
##      AIC      BIC      SIC      HQIC
## -1.284773 -1.242634 -1.285001 -1.268039

```

```
predict(Intel_m3, 5)
```

```

##      meanForecast meanError standardDeviation
## 1      0.021571 0.1207911      0.1207911
## 2      0.021571 0.1312069      0.1312069
## 3      0.021571 0.1337810      0.1337810
## 4      0.021571 0.1344418      0.1344418
## 5      0.021571 0.1346130      0.1346130

```

```

M3 = ugarchspec(variance.model = list(model = "sGARCH", garchOrder = c(1, 0)),
                mean.model = list(armaOrder = c(0, 0), include.mean = T),
                distribution.model = "std", fixed.pars = list())
(Intel_m3 <- ugarchfit(M3, data = intc))

```

```

##
## *-----*
## *          GARCH Model Fit          *
## *-----*
##
## Conditional Variance Dynamics
## -----
## GARCH Model   : sGARCH(1,0)
## Mean Model    : ARFIMA(0,0,0)
## Distribution   : std
##
## Optimal Parameters
## -----
##      Estimate  Std. Error  t value  Pr(>|t|)
## mu      0.021565   0.006053   3.5627  0.000367
## omega   0.013476   0.001996   6.7508  0.000000
## alpha1  0.263910   0.121752   2.1676  0.030189
## shape   5.937781   1.655116   3.5875  0.000334
##
## Robust Standard Errors:
##      Estimate  Std. Error  t value  Pr(>|t|)
## mu      0.021565   0.006365   3.3880  0.000704
## omega   0.013476   0.002132   6.3203  0.000000
## alpha1  0.263910   0.153254   1.7220  0.085061
## shape   5.937781   1.475958   4.0230  0.000057
##
## LogLikelihood : 242.9753
##
## Information Criteria
## -----
##
## Akaike          -1.2848
## Bayes           -1.2427
## Shibata         -1.2850
## Hannan-Quinn   -1.2681
##
## Weighted Ljung-Box Test on Standardized Residuals
## -----
##
##              statistic p-value
## Lag[1]                0.003118  0.9555
## Lag[2*(p+q)+(p+q)-1] [2] 0.193986  0.8579
## Lag[4*(p+q)+(p+q)-1] [5] 3.649143  0.3012
## d.o.f=0
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## -----
##
##              statistic p-value
## Lag[1]                0.5263  0.46815
## Lag[2*(p+q)+(p+q)-1] [2] 3.5205  0.10150
## Lag[4*(p+q)+(p+q)-1] [5] 6.6637  0.06253
## d.o.f=1

```

```

##
## Weighted ARCH LM Tests
## -----
##           Statistic Shape Scale P-Value
## ARCH Lag[2]      5.924 0.500 2.000 0.01493
## ARCH Lag[4]      7.325 1.397 1.611 0.02349
## ARCH Lag[6]      8.323 2.222 1.500 0.03480
##
## Nyblom stability test
## -----
## Joint Statistic:  1.2181
## Individual Statistics:
## mu      0.05211
## omega   0.48258
## alpha1  0.37199
## shape   0.13634
##
## Asymptotic Critical Values (10% 5% 1%)
## Joint Statistic:      1.07 1.24 1.6
## Individual Statistic:  0.35 0.47 0.75
##
## Sign Bias Test
## -----
##           t-value  prob sig
## Sign Bias      0.18119 0.8563
## Negative Sign Bias 0.39211 0.6952
## Positive Sign Bias 0.06475 0.9484
## Joint Effect      0.16040 0.9837
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## -----
##   group statistic p-value(g-1)
## 1   20      32.95      0.02439
## 2   30      38.48      0.11195
## 3   40      42.62      0.31802
## 4   50      55.42      0.24547
##
##
## Elapsed time : 0.08172512

```

```
ugarchforecast(Intel_m3, n.ahead = 5)
```

```

##
## *-----*
## *      GARCH Model Forecast      *
## *-----*
## Model: sGARCH
## Horizon: 5
## Roll Steps: 0
## Out of Sample: 0
##
## 0-roll forecast [T0=1971-01-07 19:00:00]:
##   Series Sigma

```



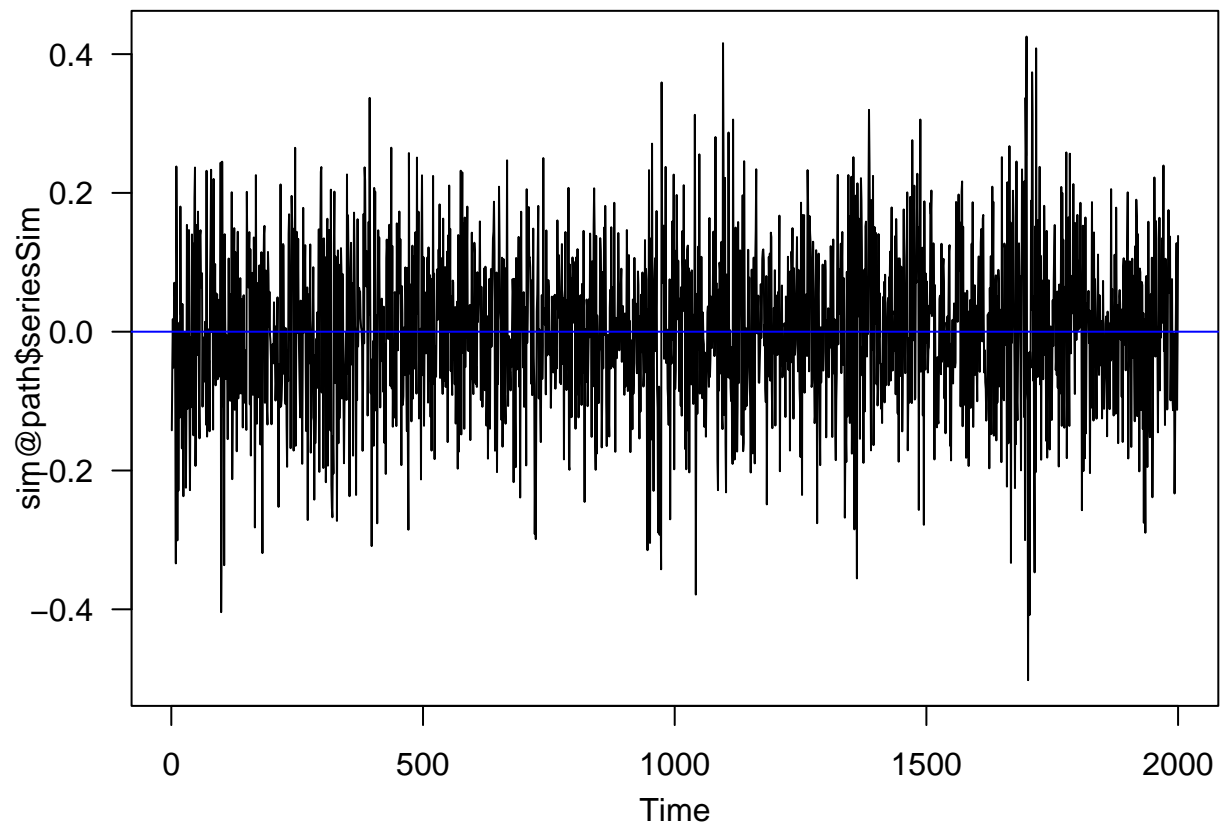
```
## T+1 0.02156 0.1211
## T+2 0.02156 0.1317
## T+3 0.02156 0.1344
## T+4 0.02156 0.1351
## T+5 0.02156 0.1352
```

GARCH Bollerslev (1986)

$$a_t = \sigma_t \epsilon_t, \quad \sigma_t^2 = \alpha_0 + \sum_{i=1}^m \alpha_i a_{t-i}^2 + \sum_{j=1}^s \beta_j \sigma_{t-j}^2.$$

Simulation

```
par(las = 1, mgp = c(2.2, 1, 0), mar = c(3.6, 3.6, 0.8, 0.6))
mod_spec <- ugarchspec(variance.model = list(model = "sGARCH", garchOrder = c(1, 1)),
  mean.model = list(armaOrder = c(0, 0), include.mean = FALSE),
  fixed.pars = list(omega = 0.001, alpha1 = 0.08, beta1 = 0.85))
sim <- ugarchpath(mod_spec, n.sim = 2000)
ts.plot(sim@path$seriesSim)
abline(h = 0, col = "blue")
```



```
### Fit a GARCH(1,1)
```

```
Intel_m4 <- garchFit(~ 1 + garch(1, 1), data = intc, trace = F)
```

```
## Warning: Using formula(x) is deprecated when x is a character vector of length > 1.  
## Consider formula(paste(x, collapse = " ")) instead.
```

```
summary(Intel_m4)
```

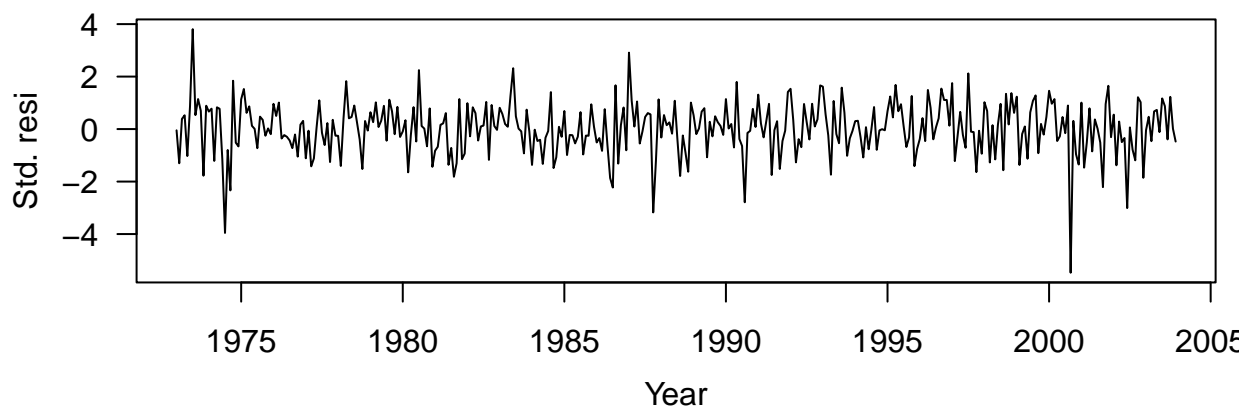
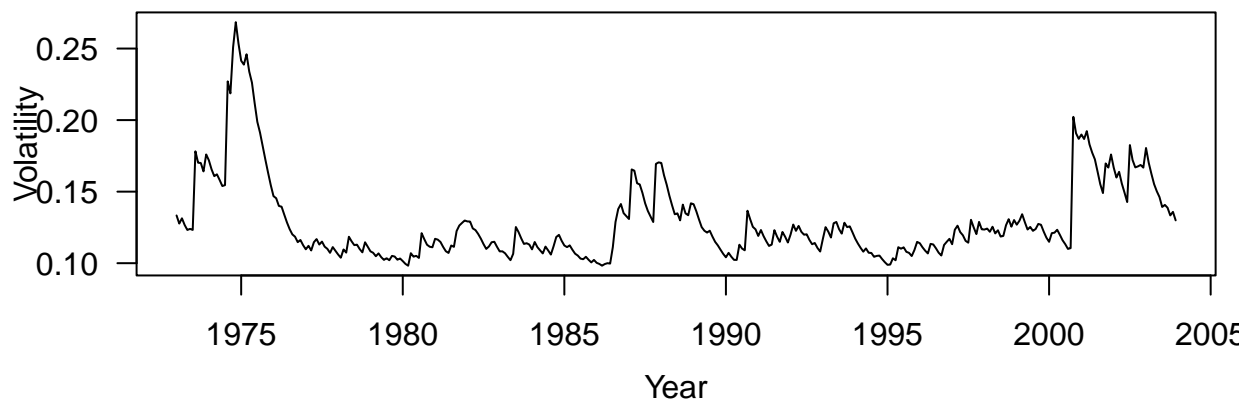
```
##  
## Title:  
## GARCH Modelling  
##  
## Call:  
## garchFit(formula = ~1 + garch(1, 1), data = intc, trace = F)  
##  
## Mean and Variance Equation:  
## data ~ 1 + garch(1, 1)  
## <environment: 0x7faae0f40ef0>  
## [data = intc]  
##  
## Conditional Distribution:  
## norm  
##  
## Coefficient(s):  
## mu omega alpha1 beta1  
## 0.0163276 0.0010918 0.0802716 0.8553014  
##  
## Std. Errors:  
## based on Hessian  
##  
## Error Analysis:  
## Estimate Std. Error t value Pr(>|t|)  
## mu 0.0163276 0.0062624 2.607 0.00913 **  
## omega 0.0010918 0.0005291 2.063 0.03907 *  
## alpha1 0.0802716 0.0281162 2.855 0.00430 **  
## beta1 0.8553014 0.0461374 18.538 < 2e-16 ***  
## ---  
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1  
##  
## Log Likelihood:  
## 239.5189 normalized: 0.6438681  
##  
## Description:  
## Wed Oct 25 21:37:42 2023 by user:  
##  
##  
## Standardised Residuals Tests:  
## Statistic p-Value  
## Jarque-Bera Test R Chi^2 156.5138 0  
## Shapiro-Wilk Test R W 0.9676933 2.471139e-07  
## Ljung-Box Test R Q(10) 9.805485 0.4577215  
## Ljung-Box Test R Q(15) 16.54435 0.346824  
## Ljung-Box Test R Q(20) 17.8005 0.6005484
```

```
## Ljung-Box Test      R^2  Q(10)  0.5130171  0.9999925
## Ljung-Box Test      R^2  Q(15)  10.24557   0.8040151
## Ljung-Box Test      R^2  Q(20)  11.77988   0.9234441
## LM Arch Test        R     TR^2   9.334459   0.6741288
##
## Information Criterion Statistics:
##      AIC      BIC      SIC      HQIC
## -1.266231 -1.224092 -1.266459 -1.249496
```

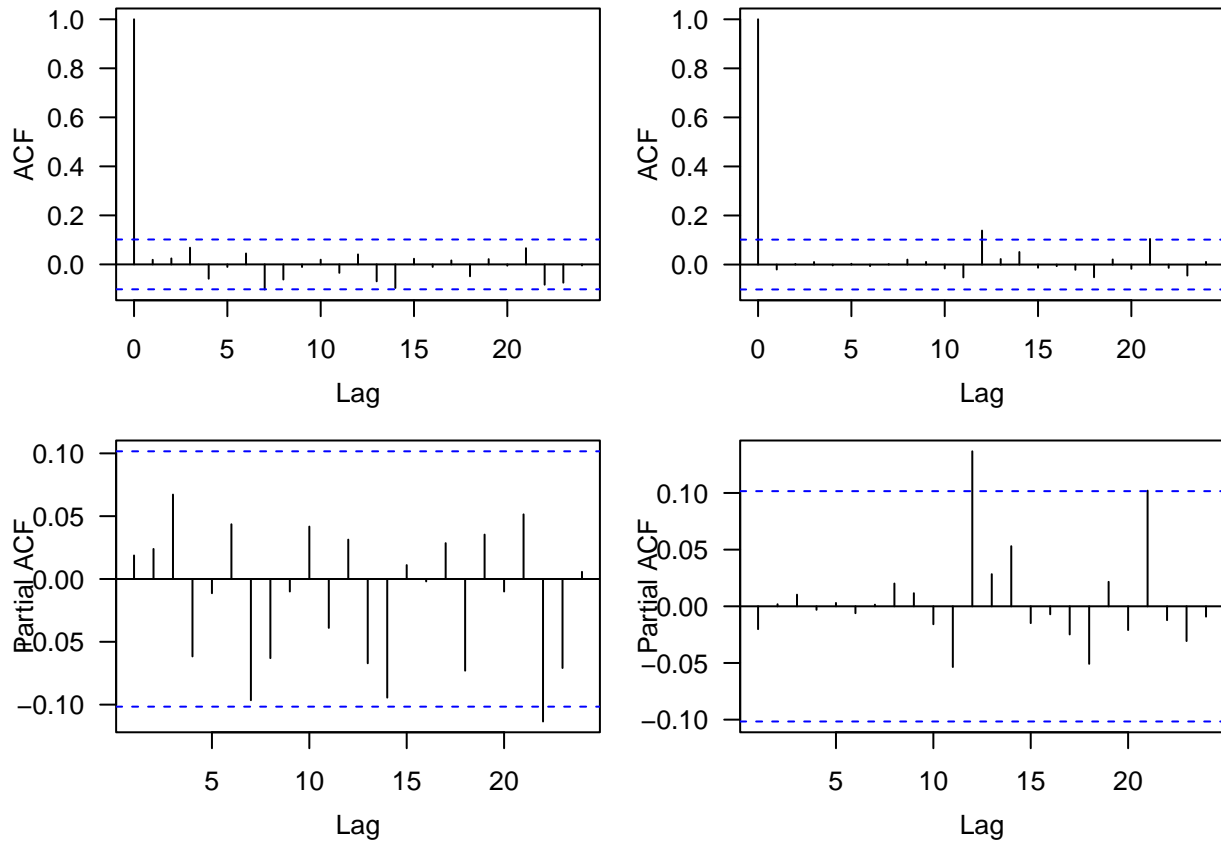
```
mu <- Intel_m4@fit$par[1]

v1 <- volatility(Intel_m4)
resi <- residuals(Intel_m4, standardize = T)
vol <- ts(v1, frequency = 12, start = c(1973, 1))
res <- ts(resi, frequency = 12, start = c(1973, 1))
```

```
par(las = 1, mgp = c(2.4, 1, 0), mar = c(3.6, 3.8, 0.8, 0.6), mfcol = c(2, 1))
plot(vol, xlab = 'Year', ylab = 'Volatility', type = 'l')
plot(res, xlab = 'Year', ylab = 'Std. resi', type = 'l')
```



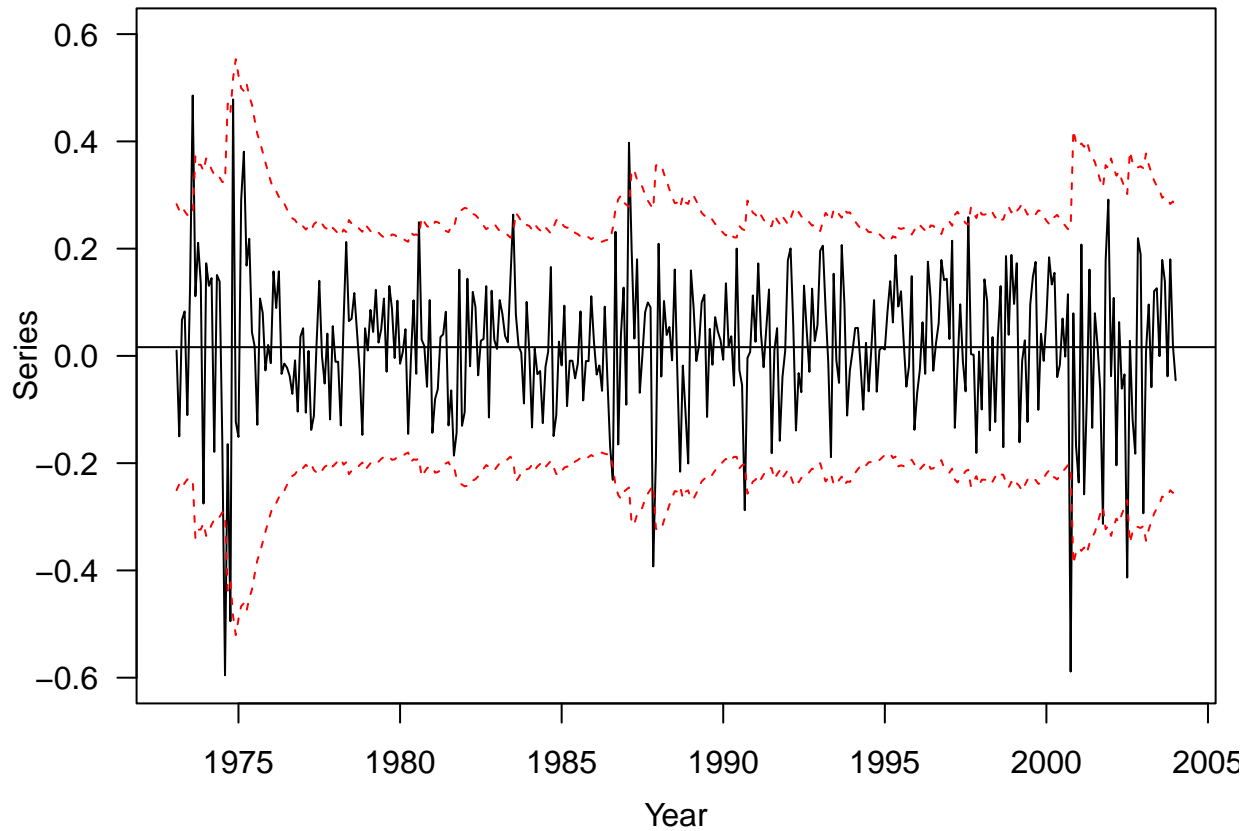
```
par(las = 1, mgp = c(2.4, 1, 0), mar = c(3.6, 3.8, 0.8, 0.6), mfcol = c(2, 2))
acf(resi, lag = 24)
pacf(resi, lag = 24)
acf(resi^2, lag = 24)
pacf(resi^2, lag = 24)
```



```

par(las = 1, mgp = c(2.4, 1, 0), mar = c(3.6, 3.8, 0.8, 0.6))
upp = mu + 2 * v1; low = mu - 2 * v1
tdx <- (1:length(intc)) / 12 + 1973
plot(tdx, intc, xlab = 'Year', ylab = 'Series', type = 'l', ylim = c(-0.6, 0.6))
lines(tdx, upp, lty = 2, col = 'red'); lines(tdx, low, lty = 2, col = 'red')
abline(h = mu)

```



```
M4 = ugarchspec(variance.model = list(model = "sGARCH", garchOrder = c(1, 1)),
                mean.model = list(armaOrder = c(0, 0), include.mean = T),
                distribution.model = "norm", fixed.pars = list())
Intel_m4 <- ugarchfit(M4, data = intc)
Intel_m4
```

```
##
## *-----*
## *          GARCH Model Fit          *
## *-----*
##
## Conditional Variance Dynamics
## -----
## GARCH Model   : sGARCH(1,1)
## Mean Model    : ARFIMA(0,0,0)
## Distribution   : norm
##
## Optimal Parameters
## -----
##      Estimate Std. Error t value Pr(>|t|)
## mu      0.016330  0.006262  2.6079 0.009111
## omega    0.001091  0.000529  2.0619 0.039214
## alpha1   0.079785  0.028023  2.8471 0.004412
## beta1    0.855460  0.046223 18.5074 0.000000
##
## Robust Standard Errors:
##      Estimate Std. Error t value Pr(>|t|)
```

```

## mu      0.016330    0.007332    2.2272 0.025934
## omega   0.001091    0.000636    1.7159 0.086184
## alpha1  0.079785    0.032726    2.4379 0.014772
## beta1   0.855460    0.050233   17.0297 0.000000
##
## LogLikelihood : 239.5281
##
## Information Criteria
## -----
## Akaike      -1.2663
## Bayes       -1.2241
## Shibata     -1.2665
## Hannan-Quinn -1.2495
##
## Weighted Ljung-Box Test on Standardized Residuals
## -----
##                statistic p-value
## Lag[1]                0.1307 0.7177
## Lag[2*(p+q)+(p+q)-1][2] 0.2414 0.8293
## Lag[4*(p+q)+(p+q)-1][5] 1.8780 0.6475
## d.o.f=0
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## -----
##                statistic p-value
## Lag[1]                0.1505 0.6981
## Lag[2*(p+q)+(p+q)-1][5] 0.1792 0.9940
## Lag[4*(p+q)+(p+q)-1][9] 0.2349 0.9999
## d.o.f=2
##
## Weighted ARCH LM Tests
## -----
##                Statistic Shape Scale P-Value
## ARCH Lag[3]    0.04014 0.500 2.000 0.8412
## ARCH Lag[5]    0.04586 1.440 1.667 0.9956
## ARCH Lag[7]    0.05466 2.315 1.543 0.9998
##
## Nyblom stability test
## -----
## Joint Statistic: 1.6271
## Individual Statistics:
## mu      0.04707
## omega   0.19694
## alpha1  0.10765
## beta1   0.20534
##
## Asymptotic Critical Values (10% 5% 1%)
## Joint Statistic:      1.07 1.24 1.6
## Individual Statistic: 0.35 0.47 0.75
##
## Sign Bias Test
## -----

```

```

##              t-value  prob sig
## Sign Bias      0.04526 0.9639
## Negative Sign Bias 0.43619 0.6630
## Positive Sign Bias 0.25615 0.7980
## Joint Effect    0.25672 0.9680
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## -----
##   group statistic p-value(g-1)
## 1    20      24.67      0.1718
## 2    30      36.06      0.1717
## 3    40      50.58      0.1013
## 4    50      51.39      0.3804
##
##
## Elapsed time : 0.06037498

```

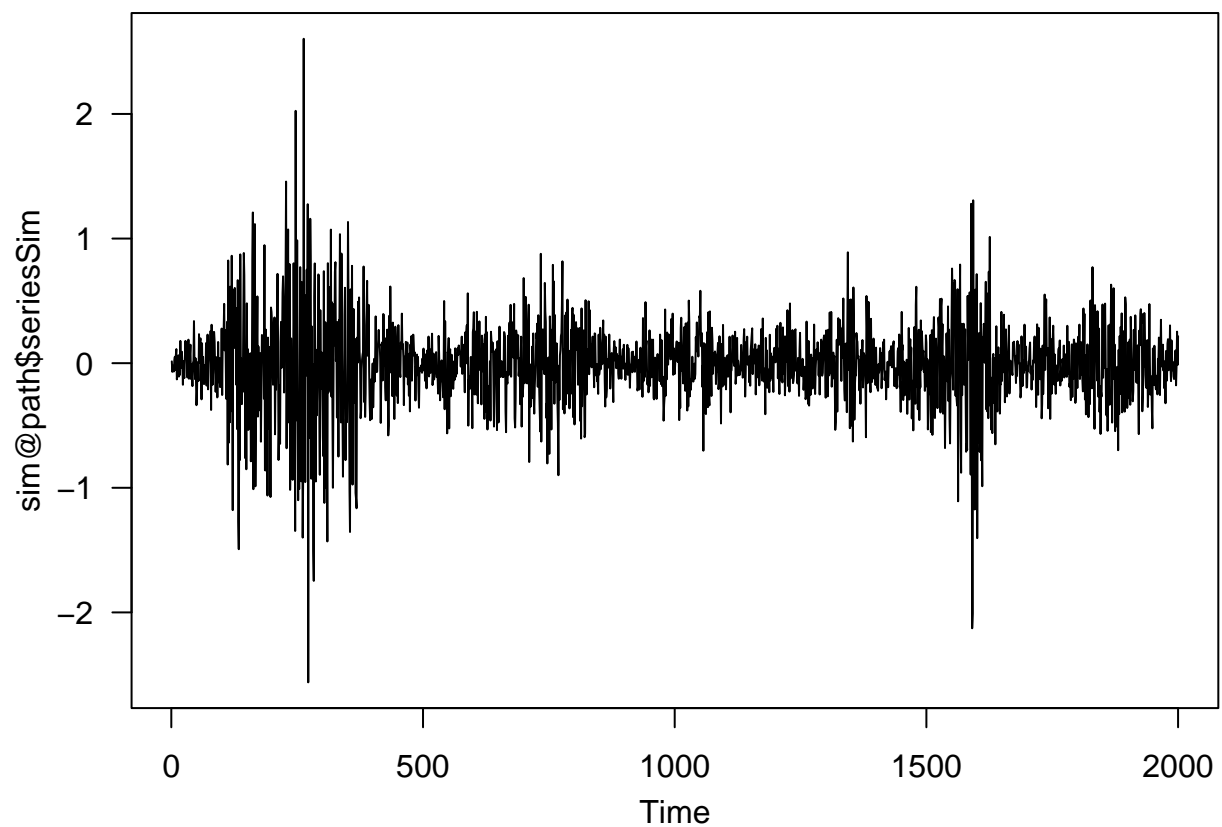
IGARCH

Simulation

```

par(las = 1, mgp = c(2.2, 1, 0), mar = c(3.6, 3.6, 0.8, 0.6))
mod_spec <- ugarchspec(variance.model = list(model = "iGARCH", garchOrder = c(1, 1)), mean.model = list
sim <- ugarchpath(mod_spec, n.sim = 2000)
ts.plot(sim@path$seriesSim)

```



```
source("Igarch.R")
IGARCH_fit <- Igarch(intc, include.mean = T)
```

```
## Estimates: 0.01518916 0.930005
## Maximized log-likelihood: -231.7141
##
## Coefficient(s):
##      Estimate Std. Error t value Pr(>|t|)
## mu    0.01518916 0.00625338 2.42895 0.015143 *
## beta 0.93000501 0.01661928 55.95940 < 2e-16 ***
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
names(IGARCH_fit)
```

```
## [1] "par"      "volatility"
```

```
Intel_m5 = ugarchspec(variance.model = list(model = "iGARCH", garchOrder = c(1, 1)), mean.model = list(
(Intel_m5 <- ugarchfit(Intel_m5, data = intc))
```

```
##
## *-----*
## *          GARCH Model Fit          *
## *-----*
##
## Conditional Variance Dynamics
## -----
## GARCH Model   : iGARCH(1,1)
## Mean Model    : ARFIMA(0,0,0)
## Distribution   : norm
##
## Optimal Parameters
## -----
##      Estimate Std. Error t value Pr(>|t|)
## mu      0.015365 0.006210 2.4741 0.013357
## omega   0.000336 0.000206 1.6318 0.102719
## alpha1  0.113395 0.035860 3.1621 0.001566
## beta1   0.886605      NA      NA      NA
##
## Robust Standard Errors:
##      Estimate Std. Error t value Pr(>|t|)
## mu      0.015365 0.007326 2.0973 0.035968
## omega   0.000336 0.000213 1.5765 0.114903
## alpha1  0.113395 0.037279 3.0418 0.002352
## beta1   0.886605      NA      NA      NA
##
## LogLikelihood : 236.014
##
## Information Criteria
## -----
##
```



```

## Akaike          -1.2528
## Bayes           -1.2212
## Shibata         -1.2529
## Hannan-Quinn   -1.2402
##
## Weighted Ljung-Box Test on Standardized Residuals
## -----
##                statistic p-value
## Lag[1]          0.1910  0.6621
## Lag[2*(p+q)+(p+q)-1][2]  0.3424  0.7726
## Lag[4*(p+q)+(p+q)-1][5]  1.7669  0.6744
## d.o.f=0
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## -----
##                statistic p-value
## Lag[1]          0.1677  0.6822
## Lag[2*(p+q)+(p+q)-1][5]  0.3334  0.9801
## Lag[4*(p+q)+(p+q)-1][9]  0.4189  0.9991
## d.o.f=2
##
## Weighted ARCH LM Tests
## -----
##                Statistic Shape Scale P-Value
## ARCH Lag[3]     0.01600 0.500 2.000 0.8993
## ARCH Lag[5]     0.07147 1.440 1.667 0.9918
## ARCH Lag[7]     0.12253 2.315 1.543 0.9990
##
## Nyblom stability test
## -----
## Joint Statistic:  1.5876
## Individual Statistics:
## mu      0.05671
## omega   0.13078
## alpha1  0.49311
##
## Asymptotic Critical Values (10% 5% 1%)
## Joint Statistic:      0.846 1.01 1.35
## Individual Statistic:  0.35 0.47 0.75
##
## Sign Bias Test
## -----
##                t-value  prob sig
## Sign Bias          0.00311 0.9975
## Negative Sign Bias 0.03265 0.9740
## Positive Sign Bias 0.02437 0.9806
## Joint Effect       0.00404 0.9999
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## -----
## group statistic p-value(g-1)
## 1      20      29.08      0.06481

```

```
## 2    30    35.74    0.18120
## 3    40    46.28    0.19700
## 4    50    55.42    0.24547
##
##
## Elapsed time : 0.03690696
```

```
ugarchforecast(Intel_m5, n.ahead = 10, data = intc)
```

```
## Warning in 'setfixed<-'('*tmp*', value = as.list(pars)): Unrecognized Parameter
## in Fixed Values: beta1...Ignored
```

```
##
## *-----*
## *      GARCH Model Forecast      *
## *-----*
## Model: iGARCH
## Horizon: 10
## Roll Steps: 0
## Out of Sample: 0
##
## 0-roll forecast [T0=1971-01-07 19:00:00]:
##      Series Sigma
## T+1  0.01536 0.1429
## T+2  0.01536 0.1441
## T+3  0.01536 0.1453
## T+4  0.01536 0.1464
## T+5  0.01536 0.1476
## T+6  0.01536 0.1487
## T+7  0.01536 0.1498
## T+8  0.01536 0.1510
## T+9  0.01536 0.1521
## T+10 0.01536 0.1532
```

EGARCH Nelson (1991)

IBM monthly returns

```
source("Egarch.R")
IBM <- read.table("m-ibmsp6709.txt", header = T)
ibm <- log(IBM$ibm + 1)
Box.test(ibm, lag = 12, type = 'Ljung')
```

```
##
## Box-Ljung test
##
## data:  ibm
## X-squared = 7.4042, df = 12, p-value = 0.8298
```

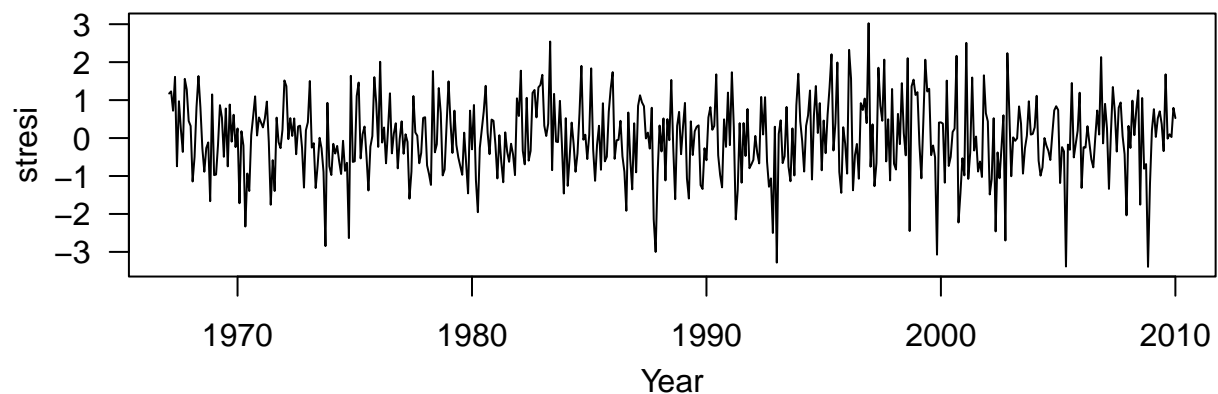
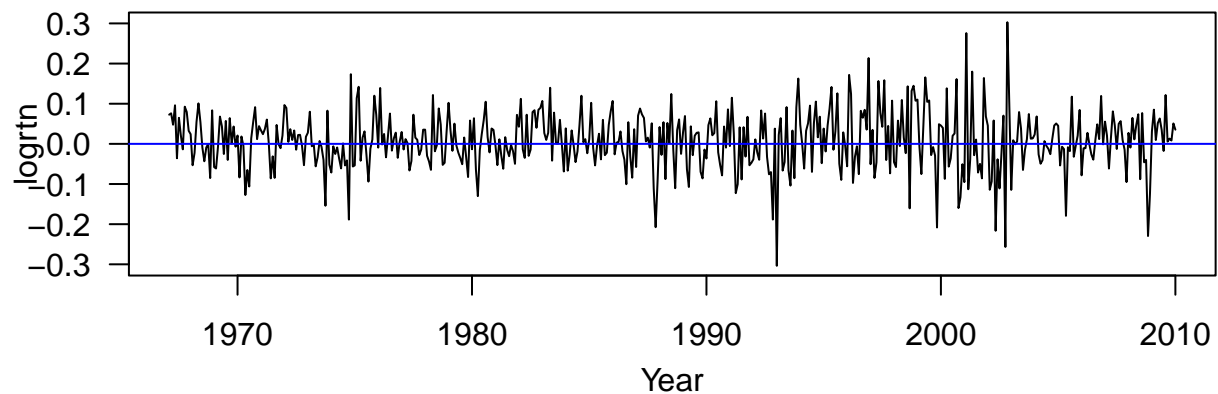
```
EGARCH_fit <- Egarch(ibm)
```

```
##  
## Estimation results of EGARCH(1,1) model:  
## estimates: 0.006732389 -0.5983263 0.217603 -0.4243245 0.92015  
## std.errors: 0.002877666 0.2349172 0.05916528 0.1683064 0.0388656  
## t-ratio: 2.339531 -2.546967 3.677882 -2.521144 23.67518
```

```
names(EGARCH_fit)
```

```
## [1] "residuals" "volatility"
```

```
stresi <- EGARCH_fit$residuals / EGARCH_fit$volatility  
tdx = (1:length(ibm)) / 12 + 1967  
par(las = 1, mgp = c(2.2, 1, 0), mar = c(3.6, 3.6, 0.8, 0.6), mfcol = c(2, 1))  
plot(tdx, ibm, xlab = 'Year', ylab = 'logrtn', type = 'l')  
abline(h = 0, col = "blue")  
plot(tdx, stres, xlab = 'Year', ylab = 'stresi', type = 'l')
```



```
Box.test(stresi, lag = 10, type = 'Ljung')
```

```
##  
## Box-Ljung test  
##  
## data: stres  
## X-squared = 5.2866, df = 10, p-value = 0.8712
```

```
Box.test(stresi, lag = 20, type = 'Ljung')
```

```
##  
## Box-Ljung test  
##  
## data: stres  
## X-squared = 20.983, df = 20, p-value = 0.3981
```

```
Box.test(stresi^2, lag = 10, type = 'Ljung')
```

```
##  
## Box-Ljung test  
##  
## data: stres^2  
## X-squared = 5.0469, df = 10, p-value = 0.888
```

```
Box.test(stresi^2, lag = 20, type = 'Ljung')
```

```
##  
## Box-Ljung test  
##  
## data: stres^2  
## X-squared = 14.261, df = 20, p-value = 0.817
```

Fit EGARCH using *ugarch*

```
IBM_egarch <- ugarchspec(variance.model = list(model = "eGARCH", garchOrder = c(1, 1)), mean.model = li
```

```
(IBM_egarch <- ugarchfit(IBM_egarch, data = ibm))
```

```
##  
## *-----*  
## *          GARCH Model Fit          *  
## *-----*  
##  
## Conditional Variance Dynamics  
## -----  
## GARCH Model   : eGARCH(1,1)  
## Mean Model    : ARFIMA(0,0,0)  
## Distribution   : norm  
##  
## Optimal Parameters  
## -----  
##      Estimate Std. Error t value Pr(>|t|)  
## mu      0.006649  0.002963  2.2442 0.024820  
## omega  -0.423208  0.223673 -1.8921 0.058480  
## alpha1 -0.094813  0.039373 -2.4081 0.016037  
## beta1   0.920485  0.041729 22.0586 0.000000  
## gamma1  0.218711  0.060802  3.5971 0.000322
```

```

##
## Robust Standard Errors:
##      Estimate  Std. Error  t value  Pr(>|t|)
## mu      0.006649   0.003073   2.1635  0.030502
## omega  -0.423208   0.308230  -1.3730  0.169743
## alpha1 -0.094813   0.051835  -1.8291  0.067382
## beta1   0.920485   0.057270  16.0728  0.000000
## gamma1  0.218711   0.061770   3.5407  0.000399
##
## LogLikelihood : 651.634
##
## Information Criteria
## -----
##
## Akaike      -2.5063
## Bayes       -2.4652
## Shibata     -2.5065
## Hannan-Quinn -2.4902
##
## Weighted Ljung-Box Test on Standardized Residuals
## -----
##
##              statistic  p-value
## Lag[1]                1.237  0.2661
## Lag[2*(p+q)+(p+q)-1] [2]  1.344  0.3989
## Lag[4*(p+q)+(p+q)-1] [5]  1.867  0.6501
## d.o.f=0
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## -----
##
##              statistic  p-value
## Lag[1]                0.009633  0.9218
## Lag[2*(p+q)+(p+q)-1] [5]  1.087446  0.8396
## Lag[4*(p+q)+(p+q)-1] [9]  2.511467  0.8360
## d.o.f=2
##
## Weighted ARCH LM Tests
## -----
##
##      Statistic  Shape  Scale  P-Value
## ARCH Lag[3]    0.09128  0.500  2.000  0.7626
## ARCH Lag[5]    1.10479  1.440  1.667  0.7021
## ARCH Lag[7]    2.20197  2.315  1.543  0.6746
##
## Nyblom stability test
## -----
## Joint Statistic:  1.1719
## Individual Statistics:
## mu      0.21948
## omega   0.61756
## alpha1  0.15868
## beta1   0.61824
## gamma1  0.06386
##
## Asymptotic Critical Values (10% 5% 1%)

```

```

## Joint Statistic:          1.28 1.47 1.88
## Individual Statistic:    0.35 0.47 0.75
##
## Sign Bias Test
## -----
##           t-value  prob sig
## Sign Bias      0.1014 0.9192
## Negative Sign Bias 0.2560 0.7980
## Positive Sign Bias 0.1888 0.8503
## Joint Effect    0.3726 0.9458
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## -----
##   group statistic p-value(g-1)
## 1    20     13.07     0.8350
## 2    30     22.26     0.8094
## 3    40     28.03     0.9040
## 4    50     42.53     0.7314
##
##
## Elapsed time : 0.07054901

```

Stochastic Volatility (SV) Model Melino and Turnbull (1990); Harvey, Ruiz, and Shephard (1994); Jacquier, Polson, and Rossi (2002)

Simulation

Let's simulate realization from a stochastic volatility model where $\log(\sigma_t)$ follows an AR(1) process. That is

$$(1 - \phi B) \log(\sigma_t^2) = \mu + \nu_t,$$

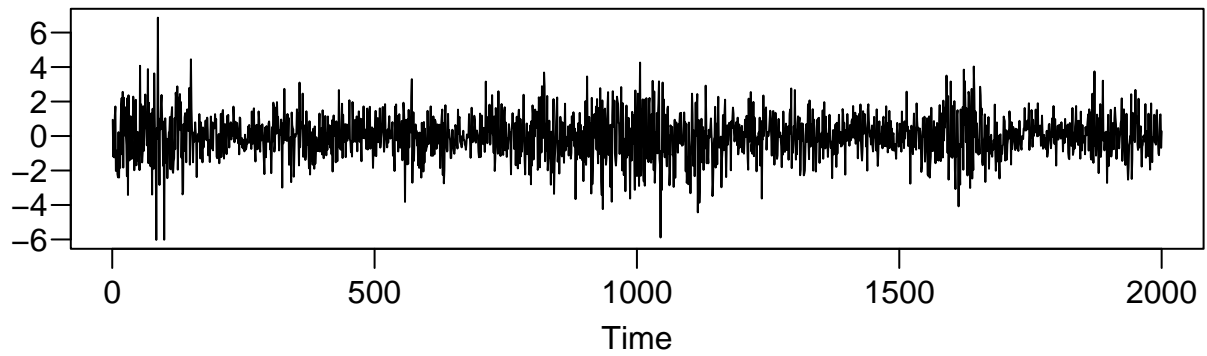
where $\nu_t \sim N(0, \sigma_\nu^2)$.

```

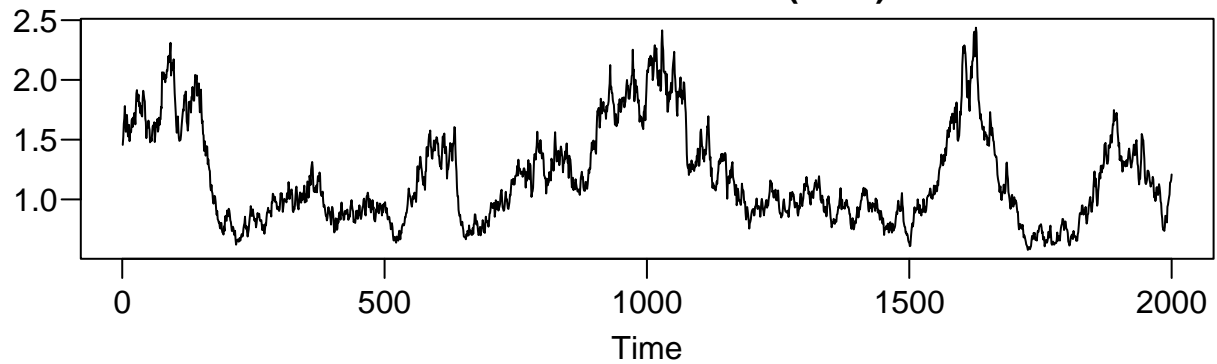
library(stochvol)
sim <- svsim(2000, mu = -9, phi = 0.99, sigma = 0.1)
par(mfrow = c(2, 1), las = 1)
plot(sim)

```

Simulated data: 'log-returns' (in %)

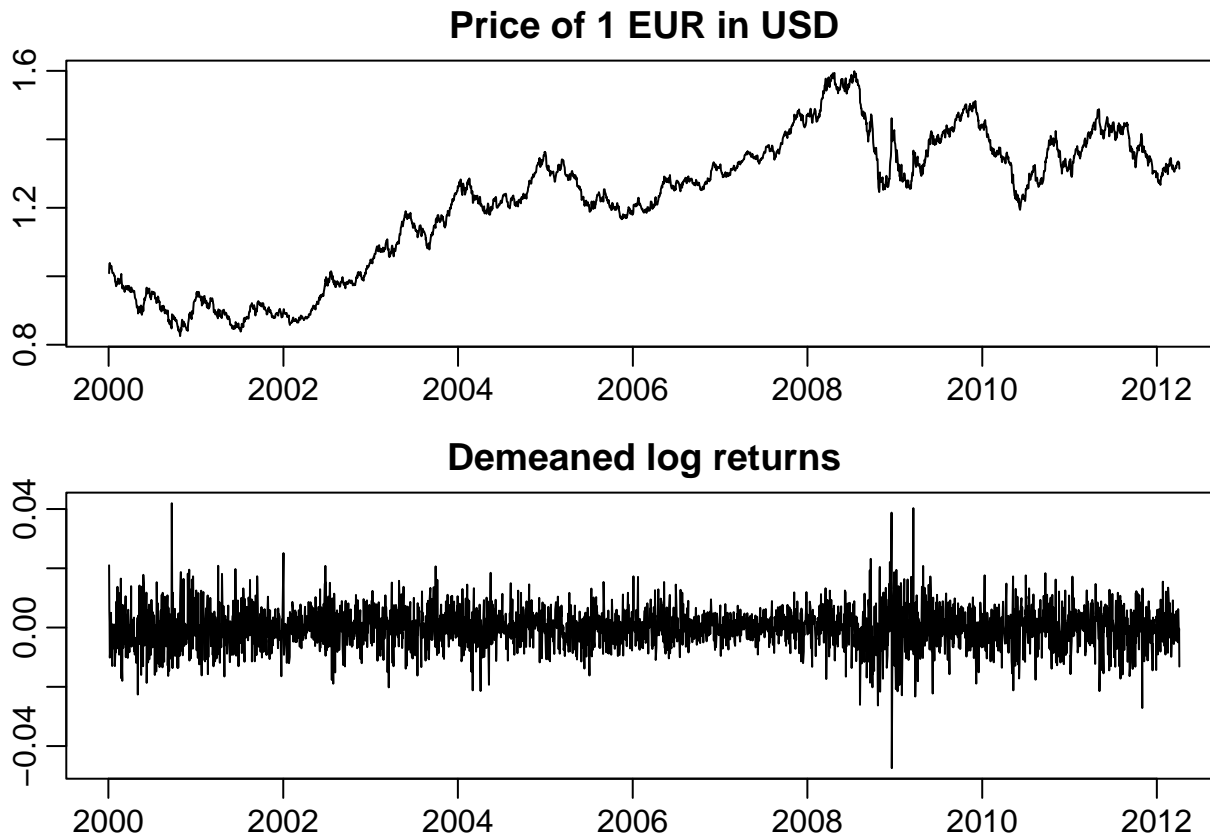


Simulated volatilities (in %)



Euro exchange rate example

```
data(exrates)
# Computes the Log Returns
ret <- logret(exrates$USD, demean = TRUE)
par(mfrow = c(2, 1), mar = c(1.9, 1.9, 1.9, 0.5), mgp = c(2, 0.6, 0))
plot(exrates$date, exrates$USD, type = "l", main = "Price of 1 EUR in USD")
plot(exrates$date[-1], ret, type = "l", main = "Demeaned log returns")
```



Perform Markov Chain Monte Carlo (MCMC) sampling for the Stochastic Volatility (SV) Model

Prior

- $\pi(\mu) \sim N(-10, 1)$
- $\pi(\phi) \sim \text{Beta}(20, 1.1)$

```
res <- svsample(ret, priormu = c(-10, 1), priorphi = c(20, 1.1), priorsigma = 0.1)
```

```
## Done!
```

```
## Summarizing posterior draws...
```

```
summary(res, showlatent = FALSE)
```

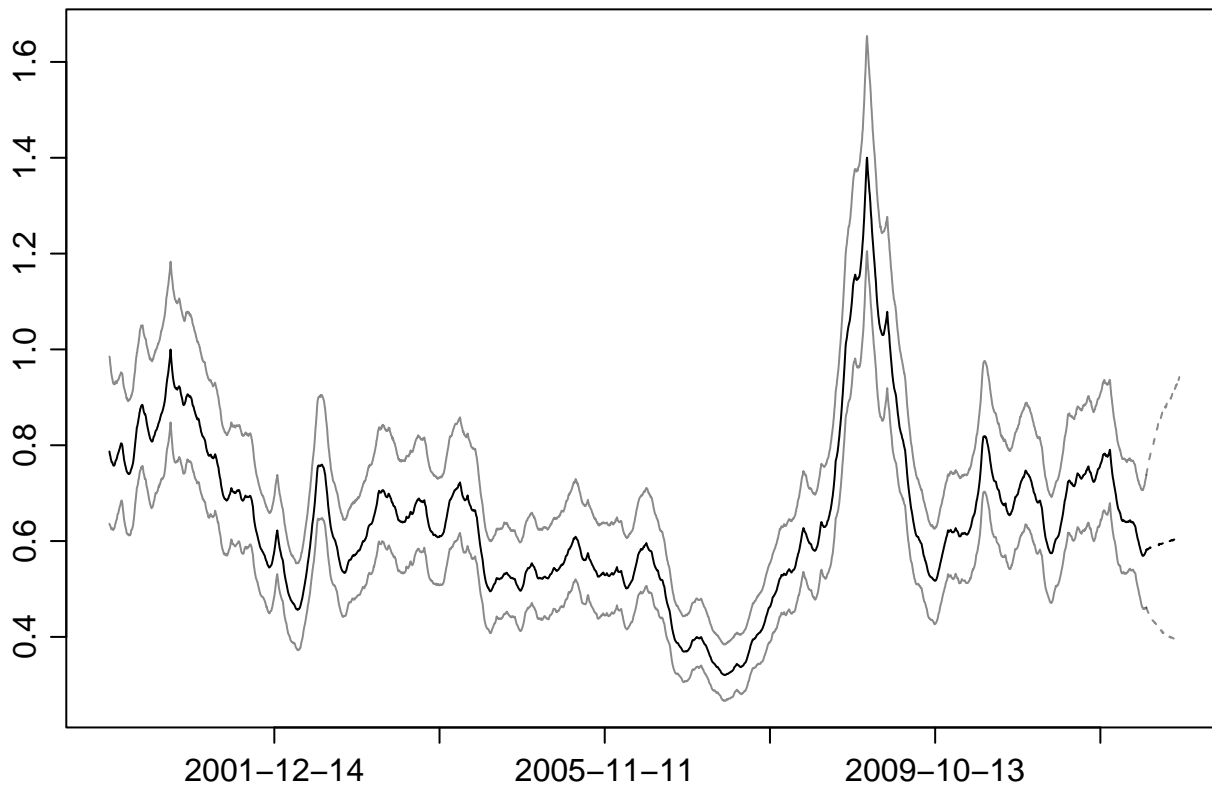
```
##
## Summary of 'svdraws' object
##
## Prior distributions:
## mu      ~ Normal(mean = -10, sd = 1)
## (phi+1)/2 ~ Beta(a = 20, b = 1.1)
## sigma^2 ~ Gamma(shape = 0.5, rate = 5)
## nu      ~ Infinity
```



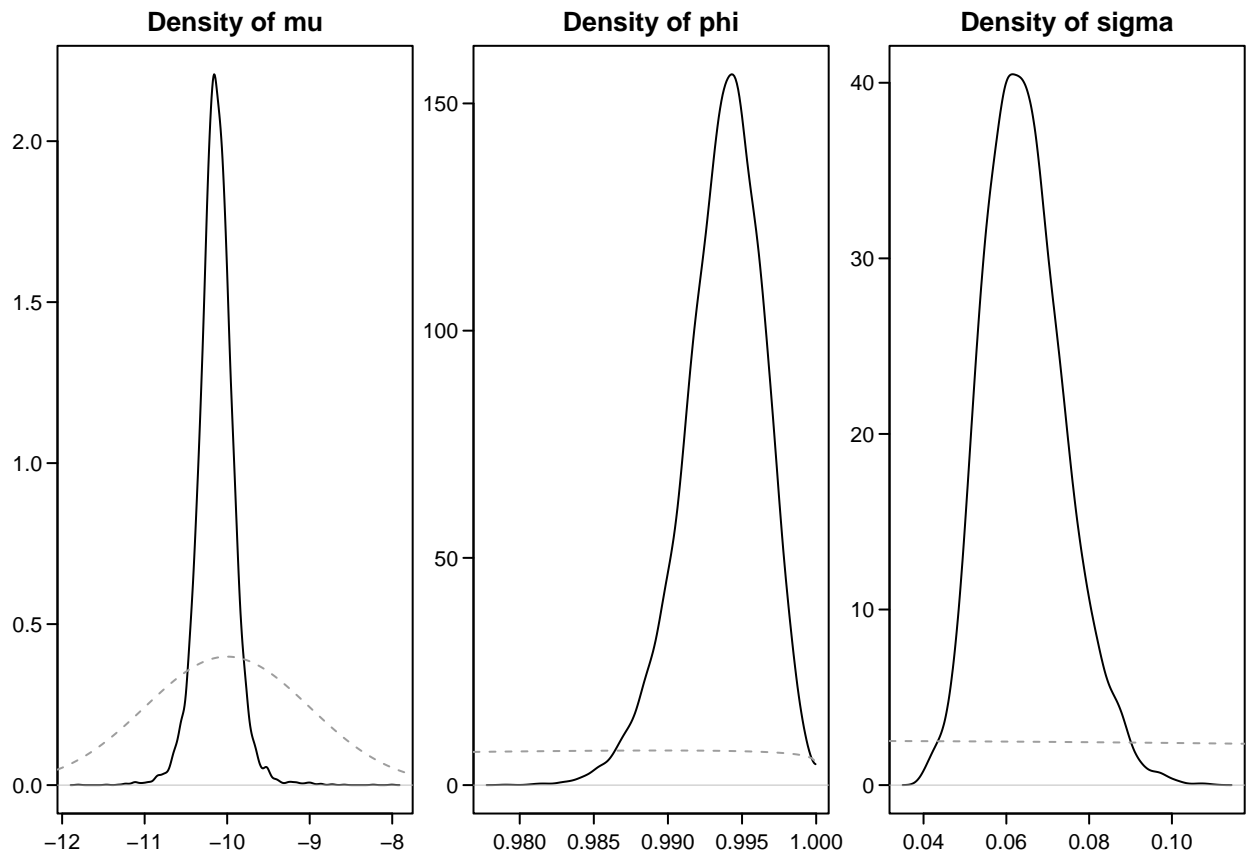
```
## rho      ~ Constant(value = 0)
##
## Stored 10000 MCMC draws after a burn-in of 1000.
## No thinning.
##
## Posterior draws of SV parameters (thinning = 1):
##           mean      sd      5%      50%      95%  ESS
## mu        -10.1343  0.21782 -10.4657 -10.1377 -9.8059 4836
## phi         0.9937  0.00268  0.9889  0.9939  0.9977  330
## sigma       0.0645  0.00981  0.0503  0.0637  0.0821  151
## exp(mu/2)   0.0063  0.00072  0.0053  0.0063  0.0074 4836
## sigma^2     0.0043  0.00132  0.0025  0.0041  0.0067  151
```

```
volplot(res, forecast = 100, dates = exrates$date[-1])
```

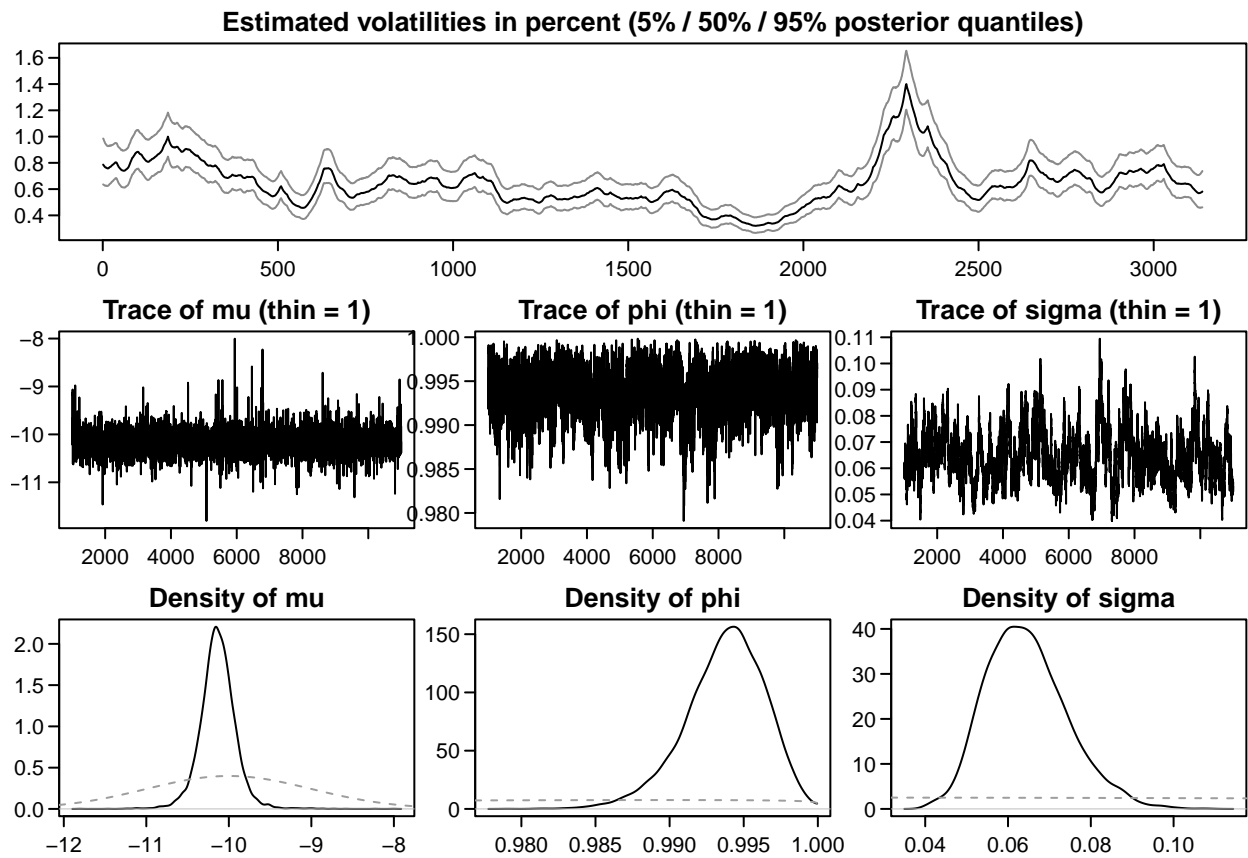
Estimated volatilities in percent (5% / 50% / 95% posterior quantiles)



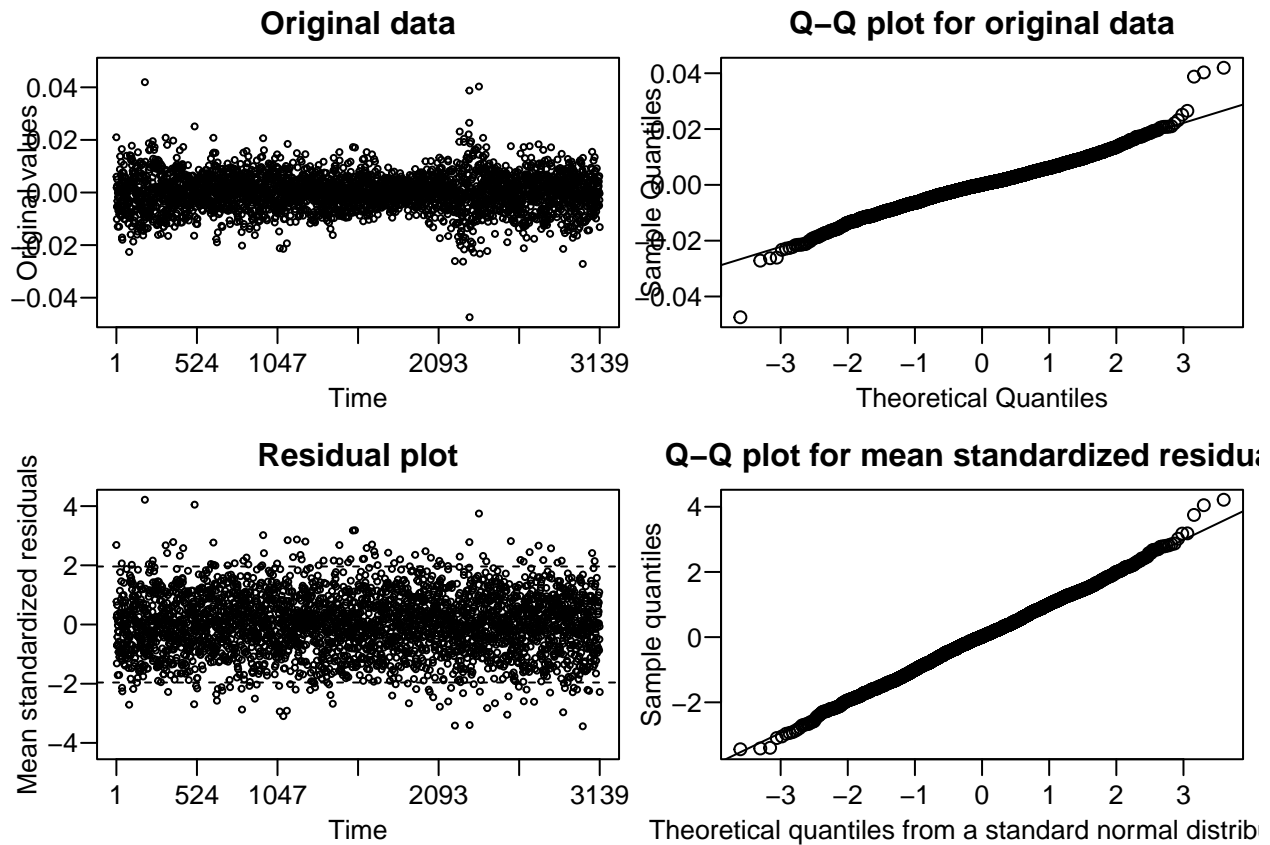
```
par(mfrow = c(1, 3), las = 1, mar = c(3.5, 4, 1, 0.5))
paradensplot(res, showobs = FALSE, cex = 0.5)
```



```
plot(res, showobs = FALSE, cex = 0.5)
```



```
myresid <- resid(res)
plot(myresid, ret, cex = 0.5)
```



References

- Bollerslev, Tim. 1986. "Generalized Autoregressive Conditional Heteroskedasticity." *Journal of Econometrics* 31 (3): 307–27.
- Engle, Robert F. 1982. "Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of United Kingdom Inflation." *Econometrica: Journal of the Econometric Society*, 987–1007.
- Harvey, Andrew, Esther Ruiz, and Neil Shephard. 1994. "Multivariate Stochastic Variance Models." *The Review of Economic Studies* 61 (2): 247–64.
- Jacquier, Eric, Nicholas G Polson, and Peter E Rossi. 2002. "Bayesian Analysis of Stochastic Volatility Models." *Journal of Business & Economic Statistics* 20 (1): 69–87.
- Melino, Angelo, and Stuart M Turnbull. 1990. "Pricing Foreign Currency Options with Stochastic Volatility." *Journal of Econometrics* 45 (1-2): 239–65.
- Nelson, Daniel B. 1991. "Conditional Heteroskedasticity in Asset Returns: A New Approach." *Econometrica: Journal of the Econometric Society*, 347–70.