ARCH models

Jesse Keränen

2/24/2022

Prologue

Many times we are interested in predicting the future return of the asset, but that is only half of the story. Sometimes it is important to be able to estimate future volatility of price of the asset. One can try to model volatility for example with autoregressive conditional heteroskedasticity models. In following we try to model volatility of one Finnish stock using different ARCH models.

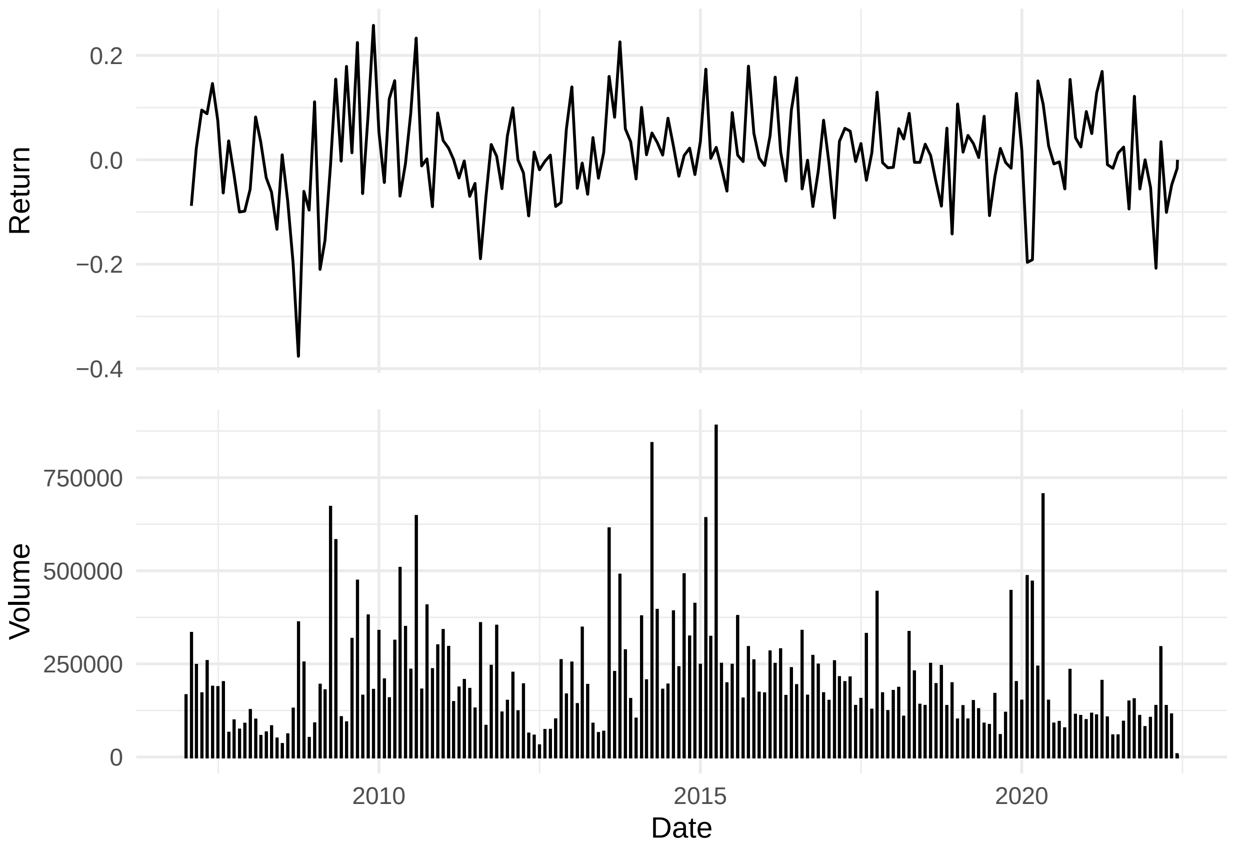
```r
library(data.table)
library(ggplot2)
library(quantmod)
library(grid)
library(tidyr)
library(forecast)

ponsse <- as.data.table(getSymbols("PON1V.HE", auto.assign = F, periodicity = "monthly"))
#ponsse <- as.data.table(getSymbols("^GSPC", from = as.Date("2014-11-30"), to = as.Date("2019-11-30")))
#ponsse <- as.data.table(GSPC)
colnames(ponsse) <- c("Date", "Open", "High", "Low", "Close", "Volume", "Adj")
ponsse[, Return := Adj/shift(Adj)-1]
ponsse <- ponsse[, .(Date, Volume, Adj, Return)]

ret <- ggplot(ponsse, aes(Date, Return)) + geom_line() + theme_minimal() +
  theme(axis.title.x = element_blank(), axis.text.x = element_blank())

vol <- ggplot(ponsse, aes(Date, Volume)) + geom_bar(stat = 'identity',
              color="black") + theme_minimal()

grid.newpage()
grid.draw(rbind(ggplotGrob(ret), ggplotGrob(vol), size = "last"))
```



```
# Probably something wrong with the data
ponsse[Return == 0, .N]
```

```
## [1] 3
```

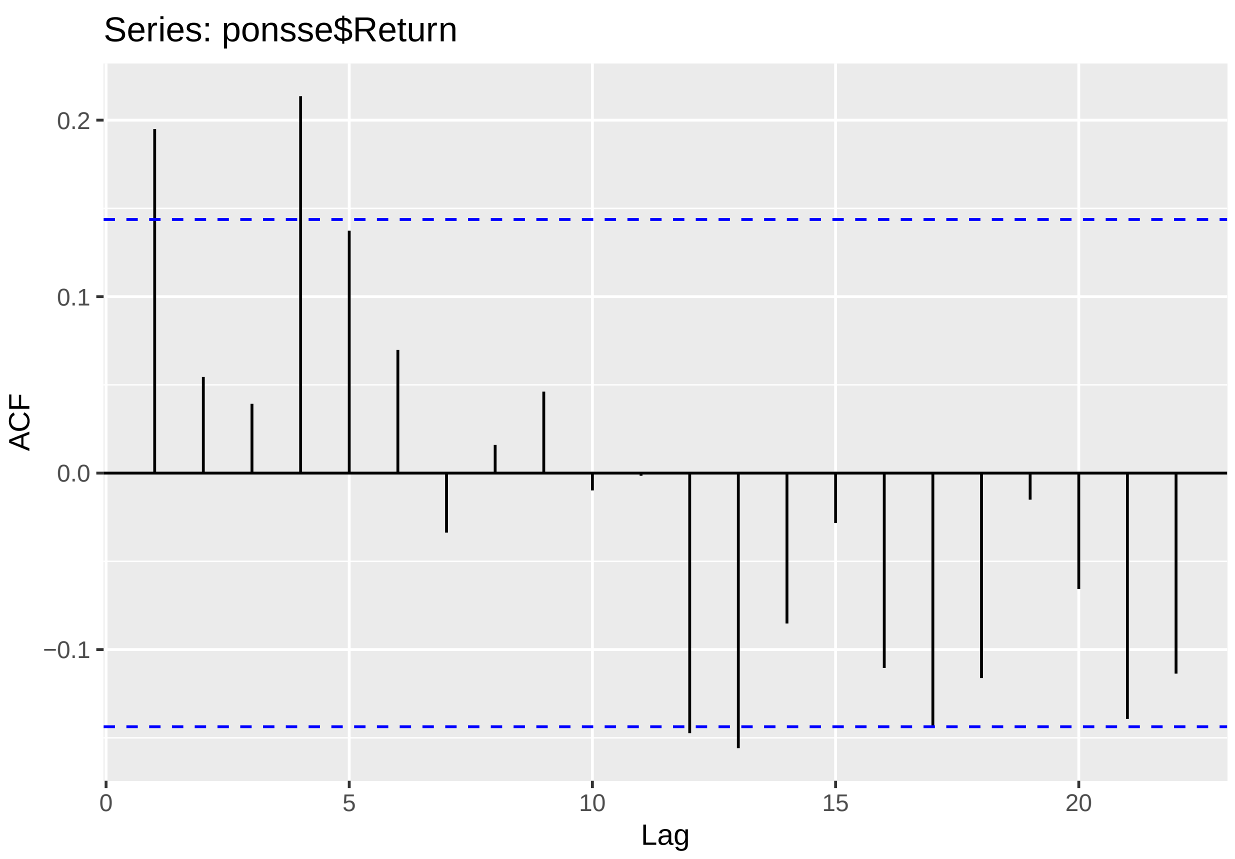
```
ponsse <- na.omit(ponsse)
```

Usually when we have made estimations about volatility of an asset we have used variance or standard deviation, i.e. average squared deviations from the mean (and square root of it). If we look at the above chart representing time series of monthly returns of our asset we can somewhat see that high price changes seem to follow high price changes and low price changes normally follow low price changes. So, historical volatility seems to affect future volatility. We use this hypothesis as basis of our volatility modelling.

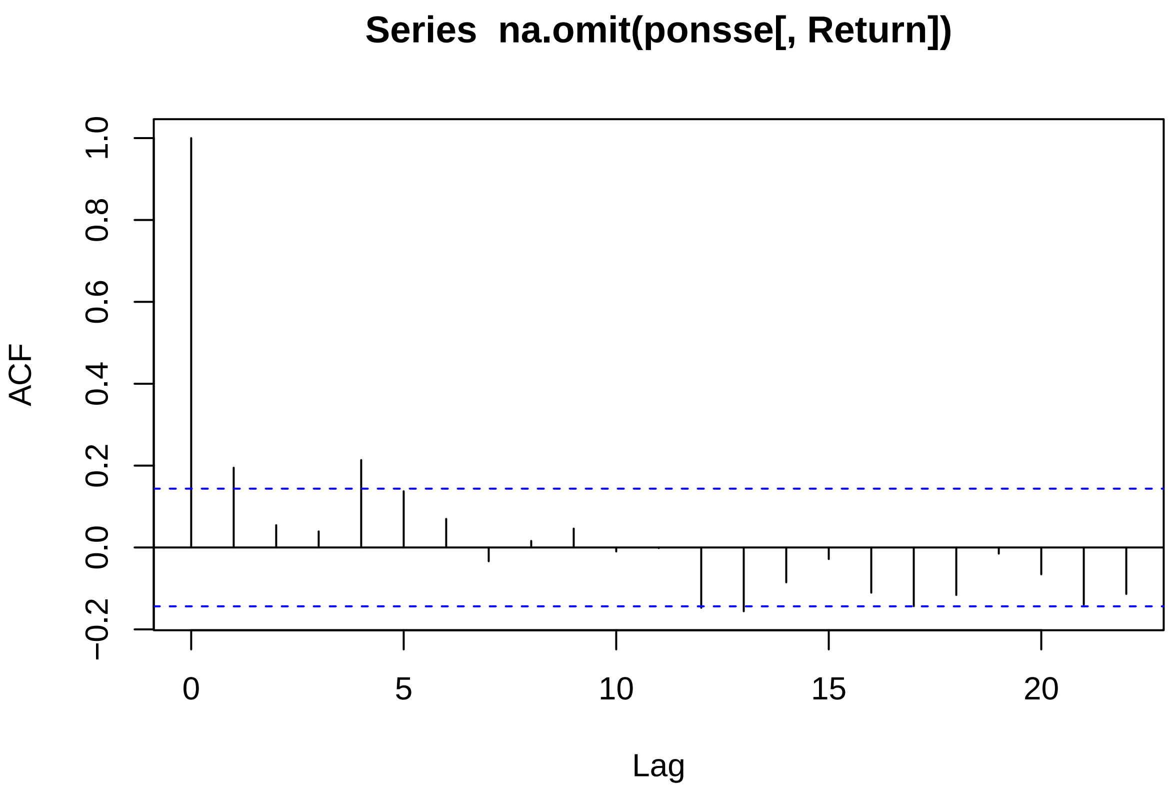
Number of lags

Next we probably want to know persistent the lag effect is. For example if previous months volatility affects this months volatility, how about two or three months prior volatility. Here autocorrelation plots come handy. In autocorrelation plot variables correlation with its lagged values is plotted on y-axis and the length of lag in x-axis. R also plot confidence intervals for the correlations telling whether they are statistically significant.

```
ggAcf(ponsse$Return)
```



```
acf(na.omit(ponsse[, Return])) # %>% autoplot
```



From the graph we can see that correlation with one month lagged (t_{-1}) volatility significant. Plot also shows correlation for four month lagged (t_{-1}) volatility. This doesn't seem as intuitive as the first one since the two months between show very little correlation with volatility of t_0. One possible explanation is quarterly reports and volatility associated with their publishing. Quarterly reports are usually reported in three month cycles which draws some doubts over our explanation. That's why I will use only the one month lagged term.

Other way to check if there are ARCH effects in the data, is to calculated residuals of each observation by substituting mean and squaring the residuals. Then one can simply regress time t residuals with time $t-1$ residuals.

```
ponsse2 <- ponsse[, .(Date, Return, Res = (Return - mean(Return)^2))]
summary(lm(data = ponsse2, Res ~ shift(Res)))
```

```
##
## Call:
## lm(formula = Res ~ shift(Res), data = ponsse2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.34664 -0.04594 -0.00448  0.04669  0.23193
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.008701   0.006561   1.326  0.18639
## shift(Res)  0.194948   0.072262   2.698  0.00763 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.08867 on 183 degrees of freedom
##   (1 observation deleted due to missingness)
## Multiple R-squared:  0.03825,    Adjusted R-squared:  0.03299
## F-statistic: 7.278 on 1 and 183 DF,  p-value: 0.007633
```

ARCH

Usually ARCH models are noted as ARCH(p), where p notes the number of lags used. Variance of stock returns is just squared mean of the deviations from the mean return. We can express each of these deviations by ϵ_t^2. In ARCH model these residuals are considered consisting of two parts, random variable z_t and time dependent standard deviation σ_t.

$$\epsilon_t = r_t - \bar{r}$$

$$\epsilon_t = \sigma_t + z_t$$

In ARHC model main idea is that volatility of stock depends on previous volatility. Then the volatility of stock at time t could be modeled as:

$$\sigma_t^2 = \alpha_0 + \alpha_1 \epsilon_{t-1}^2$$

Where α_0 is unconditional volatility and $\alpha_1 \epsilon_{t-1}^2$ describes effect of the last residual. What we are actually predicting is ϵ_t, residual in time t. We can estimate it with function:

$$\epsilon_t = \sqrt{\alpha_0 + \alpha_1 \epsilon_{t-1}^2} + z_t$$

Problem is that we don't know α_0 and α. One way to estimate them is to maximize log likelihood function. We have to calculate log likelihood value for each observation:

$$L(\mu, \omega, \alpha) = \frac{1}{\sqrt{2\pi\sigma_t}} e^{-\frac{\epsilon_t^2}{2\sigma_t^2}}$$

We can set initial values for variables μ, ω and α and maximize sum of logarithms of likelihood values. Note: also μ used to calculate residuals is variable in maximization.

```
library(tseries)
library(readxl)

mean <- ponsse[, mean(Return)]
sd <- ponsse[, sd(Return)]
var <- ponsse[, var(Return)]

options(scipen=9999)

# Returns list containing objective function value and constraint values
objective <- function(variables, dt) {
  dt[, Res := (Return - variables[1])^2]
  dt[, LRes := shift(Res)]
  dt[, Con.Var := variables[2] + variables[3]*LRes]
  dt$Con.Var[1] <- variables[2]
  dt[, Log.Like := log(1/sqrt(Con.Var*2*pi)*exp(-Res/(2*Con.Var)))]
  return (list(-(sum(dt$Log.Like)), c(variables[2:3], 1 - variables[3])))
}

# Penalty function. Gets bigger values when constraints are violated
alpha <- function(x, f, ponsse) {
  constrains <- f(x, ponsse)
  return (sum(min(constrains[[2]], 0))^2)
}

# Sum of objective function and penalty function values. Gives optimizer desire to obey constraints
penalized_function <-function(x, alpha, objective, ponsse) {
  return (objective(x, ponsse)[[1]] + r*alpha(x, objective, ponsse))
}

r <- 10
opt <- optim(c(mean, var, 0.001), (function(x) penalized_function(x, alpha, objective, ponsse)))
opt$par
```

```
## [1] 0.012830147 0.006640333 0.185888008
```

Like usual, we really don't have to code every step when using R. Library "tseries" has ready function that calculates ARCH estimates with one line of code.

```r
summary(garch(ponsse$Return, order = c(0, 1)))
```

```
##
##  ***** ESTIMATION WITH ANALYTICAL GRADIENT *****
##
##
##      I     INITIAL X(I)        D(I)
##
##      1     7.731941e-03     1.000e+00
##      2     5.000000e-02     1.000e+00
##
##     IT   NF      F         RELDF    PRELDF    RELDX   STPPAR   D*STEP   NPRELDF
##      0    1 -3.530e+02
##      1    6 -3.530e+02  1.64e-05  3.57e-05  9.9e-04  1.3e+06  1.0e-04  2.24e+01
##      2   12 -3.537e+02  1.95e-03  2.74e-03  3.6e-01  2.0e+00  5.6e-02  4.35e-01
##      3   13 -3.539e+02  7.24e-04  5.59e-04  1.8e-01  0.0e+00  4.8e-02  5.59e-04
##      4   14 -3.540e+02  1.25e-04  1.02e-04  7.3e-02  0.0e+00  2.4e-02  1.02e-04
##      5   15 -3.540e+02  1.07e-05  9.91e-06  2.5e-02  0.0e+00  9.2e-03  9.91e-06
##      6   16 -3.540e+02  2.86e-07  4.13e-07  2.9e-03  0.0e+00  1.1e-03  4.13e-07
##      7   17 -3.540e+02  3.77e-08  3.79e-08  1.1e-05  0.0e+00  4.5e-06  3.79e-08
##      8   18 -3.540e+02  8.64e-12  8.69e-12  1.2e-05  8.3e-01  4.5e-06  1.32e-11
##
##  ***** RELATIVE FUNCTION CONVERGENCE *****
##
##  FUNCTION    -3.539567e+02   RELDX        1.198e-05
##  FUNC. EVALS      18         GRAD. EVALS       9
##  PRELDF       8.694e-12      NPRELDF      1.321e-11
##
##      I      FINAL X(I)        D(I)          G(I)
##
##      1    6.762025e-03     1.000e+00    -2.410e-02
##      2    1.883646e-01     1.000e+00     3.286e-04

##
## Call:
## garch(x = ponsse$Return, order = c(0, 1))
##
## Model:
## GARCH(0,1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -3.16428 -0.42445  0.09702  0.63526  2.84927
##
## Coefficient(s):
##     Estimate  Std. Error  t value             Pr(>|t|)
## a0  0.006762    0.000825    8.196 0.000000000000000222 ***
## a1  0.188365    0.076104    2.475               0.0133 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Diagnostic Tests:
##  Jarque Bera Test
```

```
##
## data:  Residuals
## X-squared = 5.1761, df = 2, p-value = 0.07517
##
##
##  Box-Ljung test
##
## data:  Squared.Residuals
## X-squared = 0.047856, df = 1, p-value = 0.8268
```