**GARCH Models**

GARCH models are the generalization of ARCH models

GARCH(m, r)

We now examine the GARCH model in more detail. Note that the “G” stands for “Generalized”. The model is

yt = σtεt

where

 = α0 + α1 + … + αm +  + … + 

and εt ~ independent (0,1)

The additional parameters help incorporate past variances. Note that E(yt) is 0.

Notes:

* This model helps to incorporate past volatilities (as measured by the variance term) that may affect the present. The overall hope is that a smaller number of parameters will be needed for a GARCH model than if an ARCH model was used instead. In other words, we want to have a parsimonious model.
* The ,…,  are all unobservable.
* We use a normality assumption when estimating the model, so εt ~ independent N(0,1).
* If r = 0, then GARCH(m,r) = ARCH(m).
* The model can be reparameterized and thought of as a ARMA model for . For example, with a GARCH(1,1),

 = α0 + α1 + ,

we obtain,

 = () + ( - )

= (α0 + α1 + ) + ( - )

= (α0 + α1 + ) + ( - ) + ( - )

= α0 + (α1+β1) - β1( - ) + ( - )

= α0 + (α1+β1) - β1νt-1 + νt

where νt =  -  =  - =  plays the role of “wt” in a regular ARMA(1,1) model.

* The constraints on the parameters are: α0 > 0, αi ≥ 0, βi ≥ 0, and . Note that if m < r, then the extra αi’s in the sum are 0; vice versa for r < m and the βj’s.
* Integrated GARCH or IGARCH model

In a GARCH(1,1) model, it can be found that at times α1 + β1 = 1. For this case,

 = α0 + (α1+β1) - β1νt-1 + νt

= α0 +  - β1νt-1 + νt

⇒  -  = α0 - β1νt-1 + νt

⇒ (1-B) = α0 - β1νt-1 + νt

An interpretation of the IGARCH(1,1) model is that the volatility is persistent. This is because it can be shown that the past volatility (variances) has an effect on all future volatilities. For more information, see Chan’s and Pena, Tiao, and Tsay’s textbooks.

* Due to yt being squared and how it is included in the model, GARCH (including ARCH) models affect volatility (variability) in returns the same way for both positive and negative returns. This can be unrealistic due to how the stock market tends to react to
“good” and “bad” news. Hull’s finance textbook says that

The volatility of an equity’s price tends to be inversely related to the price so that a negative un-1 (yt-1) has a bigger effect on σn (σt) than the same positive un-1 (yt-1).

* Determining the value of r in the model is not straightforward. Pena, Tiao, and Tsay’s textbook says

The identification of GARCH models in practice is not simple. Only lower-order GARCH models are used in most applications.

What should you do then? One possibility is to try a few different models and see which model gives satisfactory residuals. From these models, choose the one with the smallest number of parameters.

Example: Monthly returns of value-weighted S&P 500 Index from 1926 to 1991 (SP500.R, sp500.txt)

This is an example from Pena, Tiao, and Tsay’s textbook. The authors say that an ARCH(9) model would be needed if the GARCH part was not used.

Below are my initial examinations of the data:

> sp500 <- read.table(file = "sp500.txt", header = FALSE,

 col.names = "x", sep = "")

> head(sp500)

 x

1 0.0225

2 -0.0440

3 -0.0591

4 0.0227

5 0.0077

6 0.0432

x <- sp500$x



> par(mfrow = c(2,2))

> acf(x = x, type = "correlation", lag.max = 20, xlab =

 "h", main = expression(paste("Estimated ACF for ",

 x[t])))

> pacf(x = x, lag.max = 20, xlab = "h", main =

 expression(paste("Estimated PACF for ", x[t])))

> acf(x = x^2, type = "correlation", lag.max = 20, xlab =

 "h", main = expression(paste("Estimated ACF for ",

 x[t]^2)))

> pacf(x = x^2, lag.max = 20, xlab = "h", main =

 expression(paste("Estimated PACF for ", x[t]^2)))

> par(mfrow = c(1,1))



The above plots show dependence in the xt and  series. I am not sure if it is really appropriate to look at the  series plots yet because we have not tried to model the dependence in the xt series. However, because the authors look at the , I still constructed the plots here.

The authors suggest a MA(3) or AR(3) model would be appropriate for xt. They focus on an AR(3).

> mod.fit.ar3 <- arima(x = x, order = c(3, 0, 0),

 include.mean = TRUE)

> mod.fit.ar3

Call:

arima(x = x, order = c(3, 0, 0), include.mean = TRUE)

Coefficients:

 ar1 ar2 ar3 intercept

 0.0890 -0.0238 -0.1229 0.0062

s.e. 0.0353 0.0355 0.0353 0.0019

sigma^2 estimated as 0.00333: log likelihood = 1135.25, aic = -2260.5

> y <- as.numeric(mod.fit.ar3$residuals)

> par(mfrow = c(2,2))

> acf(x = y, type = "correlation", lag.max = 20, xlab =

 "h", main = expression(paste("Estimated ACF for ",

 y[t])))

> pacf(x = y, lag.max = 20, xlab = "h", main =

 expression(paste("Estimated PACF for ", y[t])))

> acf(x = y^2, type = "correlation", lag.max = 20, xlab =

 "h", main = expression(paste("Estimated ACF for ",

 y[t]^2)))

> pacf(x = y^2, lag.max = 20, xlab = "h", main =

 expression(paste("Estimated PACF for ", y[t]^2)))

> par(mfrow = c(1,1))



The estimated ARMA(3,0) model is

xt = 0.0066 + 0.0890xt-1 - 0.0238xt-2 - 0.1229xt-3 + wt.

where (1-0.0890+0.0238+0.1229)×0.0062 = 0.0066.

Below is code for an ARMA(3,0) and GARCH(1,1) model suggested by the authors.

> library(fGarch)

> mod.fit.garch <- garchFit(formula = ~ arma(3,0) +

 garch(1, 1), data = x)

Series Initialization:

 ARMA Model: arma

 Formula Mean: ~ arma(3, 0)

 GARCH Model: garch

 Formula Variance: ~ garch(1, 1)

<EDITED>

> summary(mod.fit.garch)

Title:

 GARCH Modelling

Call:

 garchFit(formula = ~arma(3, 0) + garch(1, 1), data = x)

Mean and Variance Equation:

 data ~ arma(3, 0) + garch(1, 1)

<environment: 0x0000000015da8348>

 [data = x]

Conditional Distribution:

 norm

Coefficient(s):

 mu ar1 ar2 ar3 omega alpha1

 7.7077e-03 3.1968e-02 -3.0261e-02 -1.0649e-02 7.9746e-05 1.2425e-01

 beta1

 8.5302e-01

Std. Errors:

 based on Hessian

Error Analysis:

 Estimate Std. Error t value Pr(>|t|)

mu 7.708e-03 1.607e-03 4.798 1.61e-06 \*\*\*

ar1 3.197e-02 3.837e-02 0.833 0.40473

ar2 -3.026e-02 3.841e-02 -0.788 0.43076

ar3 -1.065e-02 3.756e-02 -0.284 0.77677

omega 7.975e-05 2.810e-05 2.838 0.00454 \*\*

alpha1 1.242e-01 2.247e-02 5.529 3.22e-08 \*\*\*

beta1 8.530e-01 2.183e-02 39.075 < 2e-16 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Log Likelihood:

 1272.179 normalized: 1.606287

Standardised Residuals Tests:

 Statistic p-Value

 Jarque-Bera Test R Chi^2 73.04842 1.110223e-16

 Shapiro-Wilk Test R W 0.9857969 5.961994e-07

 Ljung-Box Test R Q(10) 11.56744 0.315048

 Ljung-Box Test R Q(15) 17.78747 0.2740039

 Ljung-Box Test R Q(20) 24.11916 0.2372256

 Ljung-Box Test R^2 Q(10) 10.31614 0.4132089

 Ljung-Box Test R^2 Q(15) 14.22819 0.5082978

 Ljung-Box Test R^2 Q(20) 16.79404 0.6663038

 LM Arch Test R TR^2 13.34305 0.3446075

Information Criterion Statistics:

 AIC BIC SIC HQIC

-3.194897 -3.153581 -3.195051 -3.179018

> plot(mod.fit.garch)







The estimated GARCH model is:

 = 7.975×10-5 + 0.1242 + 0.8530

Notice all of the parameters are significantly different from 0. Because we re-estimated the ARMA(3,0) too, we can re-examine the significance of the corresponding parameters. Similar to the authors, I think the ARMA(3,0) model may not be necessary! Dropping the ARMA(3,0) model as in Pena, Tiao, and Tsay. I re-fit a GARCH model. However, note that non-normality indicated by the QQ-plot above. This was not mentioned in the book. Furthermore, this could adversely affect inferences as well.

Below is my corresponding code and output without the ARMA(3,0) part.

> mod.fit.garch <- garchFit(formula = ~ garch(1, 1), data =

 x)

Series Initialization:

 ARMA Model: arma

 Formula Mean: ~ arma(0, 0)

 GARCH Model: garch

 Formula Variance: ~ garch(1, 1)

<EDITED>

> summary(mod.fit.garch)

Title:

 GARCH Modelling

Call:

 garchFit(formula = ~ garch(1, 1), data = x)

Mean and Variance Equation:

 data ~ garch(1, 1)

<environment: 0x0000000010686418>

 [data = x]

Conditional Distribution:

 norm

Coefficient(s):

 mu omega alpha1 beta1

7.4497e-03 8.0615e-05 1.2198e-01 8.5436e-01

Std. Errors:

 based on Hessian

Error Analysis:

 Estimate Std. Error t value Pr(>|t|)

mu 7.450e-03 1.538e-03 4.845 1.27e-06 \*\*\*

omega 8.061e-05 2.833e-05 2.845 0.00444 \*\*

alpha1 1.220e-01 2.202e-02 5.540 3.02e-08 \*\*\*

beta1 8.544e-01 2.175e-02 39.276 < 2e-16 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Log Likelihood:

 1269.455 normalized: 1.602848

Standardised Residuals Tests:

 Statistic p-Value

 Jarque-Bera Test R Chi^2 80.32111 0

 Shapiro-Wilk Test R W 0.9850517 3.141228e-07

 Ljung-Box Test R Q(10) 11.2205 0.340599

 Ljung-Box Test R Q(15) 17.99703 0.262822

 Ljung-Box Test R Q(20) 24.29896 0.2295768

 Ljung-Box Test R^2 Q(10) 9.920157 0.4475259

 Ljung-Box Test R^2 Q(15) 14.21124 0.509572

 Ljung-Box Test R^2 Q(20) 16.75081 0.6690903

 LM Arch Test R TR^2 13.04872 0.3655092

Information Criterion Statistics:

 AIC BIC SIC HQIC

-3.195594 -3.171985 -3.195645 -3.186520

The estimated model is  = 8.061×10-5 + 0.1220 + 0.8544, where μ is estimated to be 7.45×10-3. A QQ-plot for the standardized residuals still indicates a normality problem.

What can you do about the normality problem?

* Assume a different distribution for εt. See the cond.dist argument of garchFit() for a list. Note that the documentation for what these distributions represent is not good.
* Use “robust” standard errors. The cond.dist = "QMLE" argument value produces these types of standard errors using a sandwich estimate of the covariance matrix. The documentation for this is still somewhat poor. In most other statistical applications, sandwich estimators produce consistent estimates of the covariance matrix even when the distributional assumptions are incorrectly specified.

The authors also suggest fitting an IGARCH model to the data due to  being close to 1. Unfortunately, garchFit() can fit these types of models. Code is included in my program for the ugarchfit() function of the rugarch package to fit the model. The estimated model is  = 0.0074 + 0.1430 + 0.8570 with xt = 0.0074 + wt where wt~ind. N(0,). The intercept parameter is estimated simultaneously with the other parameters. Notice  = 1 as required by the IGARCH(1,1) model, The authors state that “this model seems hard to justify for an excess return series.” This statement is in reference to how past volatility affects all future volatilities for this model (i.e., 1950 volatility affect 1990s volatility). Overall, I did not see improvement with this model over the others.

Final comments

* There are MANY other types of GARCH models. In fact, I would not be surprised if there are whole classes on the subject at other universities, perhaps in finance departments.
* One of these other GARCH models is an EGARCH (where the “E” stands for exponential) model which helps to account for potential asymmetry between positive/negative returns. Remember that ARCH and GARCH treated them symmetrically. In other words, we did not differentiate between positive or negative returns in the model since yt was always squared. The ugarchfit() function can fit these types of models (use model = "eGARCH" in the model specifications).
* An asymmetric power GARCH model can account for the asymmetry as well. Shumway and Stoffer’s textbook describe this model.