11 | GARCH Models

The focus of modelling stationary processes with ARIMA models has been on analyzing the mean behavior of a time series. A drawback of linear stationary models is their failure to account for changing volatility: The width of the 1-step-ahead forecast intervals remains constant even as new data become available, unless the parameters of the model are changed. To see this, suppose that $\{Y_t\}$ is stationary time series with the representation,

$$Y_t = \sum_{j=0}^{\infty} \psi_j \varepsilon_{t-j}, \qquad \varepsilon_t \sim i.i.d.(0, \sigma^2) \tag{11.1}$$

the width of the forecast interval is proportional to the square root of the one-step forecast error variance,

$$\text{var}(Y_{n+1} - \hat{Y}_{n+1}) = \text{var}(\varepsilon_t) = \sigma_\varepsilon^2 ,$$

which is a constant. On the other hand, actual time series such as economics and financial series often show sudden bursts of high volatility. For example, if a recent innovation was strongly negative (indicating a crash, etc.), a period of high volatility will often follow. A clear example of this is provided by the daily returns for NASDAQ Index from Sep 1st to Nov 30'th, 1987 in Figure **??**. The volatility increases markedly after the crash, and stays high for quite some time. Nevertheless, the forecast intervals based on a given ARMA model assuming independent white noise would have exactly the

same width after the crashes before. This would destroy the validity of post-crash forecast intervals. It would have equally devastating consequences for any methodology which requires a good assessment of current volatility, such as the Black-Scholes method of options pricing.

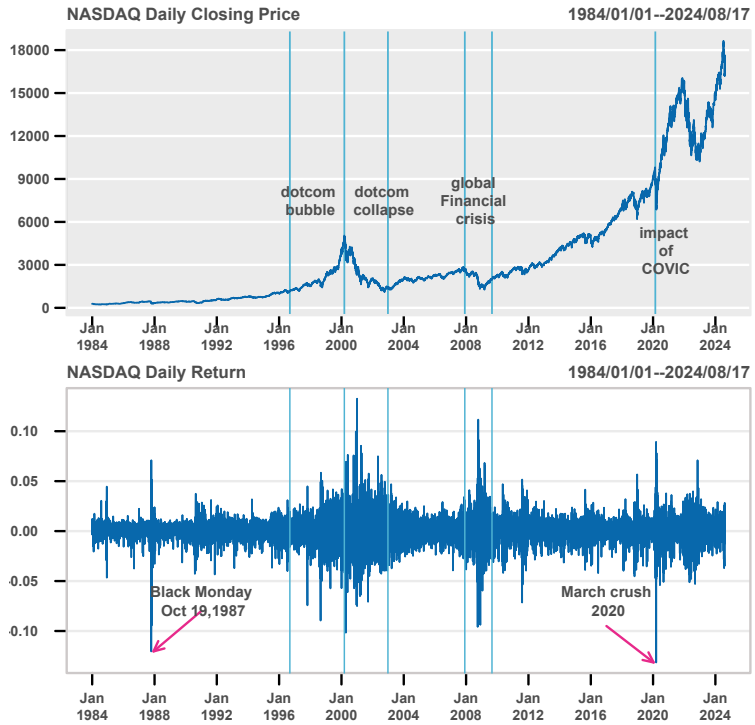

Figure 11.1: Daily adjusted closing prices and daily log returns of NASDAQ composite, starting from Jan 1, 1984 to Aug 17, 2024.

Furthermore, for financial series such as stock returns, highly volatile periods tend to be clustered together, an indication that there is a dependence of sudden bursts of variability in a return on the series own past. In the ARCH (Engle, 1982) model, the forecast intervals

are able to widen immediately to account for sudden changes in volatility, without changing the parameters of the model. Because of this feature, ARCH (and other related) models have become a very important element in the analysis of economic and financial time series.

Nonforecastable Errors

The errors ε_t in any time series model are assumed to be white noise process. Two types of white noise are commonly assumed, *i.i.d* and uncorrelated ones, which are of strong and weak restrictions respectively. Here, we will categorize white noise in terms of their forecastabilitiy.

Linearity A stationary time series $\{Y_t\}$ is *linear* if it has an MA(∞) representation in *i.i.d.* white noise as in (11.1). If $\{\varepsilon_t\}$ are *i.i.d.* , the best forecast of $\varepsilon_{n+h}, h \geq 1$,

$$E(\varepsilon_{n+h}|\varepsilon_n, \varepsilon_{n-1}, \ldots) = E(\varepsilon_{n+h}) = 0\,.$$

Thus, ε_{n+h} is not forecastable by any methods.

Nonlinear processes A nonlinear model is defined by exclusion as any model that is linear. If a white noise process $\{\varepsilon_t\}$ are merely uncorrelated, then ε_t is not linearly forecastable, but it may be possible to obtain a good forecast of e_{n+1} using a nonlinear function of $\varepsilon_n, \varepsilon_{n-1}, \ldots$. A wide variety of series may be written in the MA(∞) form with respect to uncorrelated white noise. A nonlinear white

noise example is given by,

$$\varepsilon_t = e_t + \beta e_{t-1} e_{t-2}, \qquad e_t \sim i.i.d\,(0, \sigma_e^2).$$

The nonlinear features thus can be divided into two categories – implicit and explicit. In the former case, we retain the general ARMA framework and choose the distribution of the white noise appropriately so that the resulting process exhibits a specified nonlinear feature. Although the form of the models have representation $\sum_j \psi_j \varepsilon_{t-j}$, the conditional expectations on their lagged values may well be nonlinear. The explicit nonlinear models typically express a random variable as a nonlinear function of its lagged values. For example,$Y_t = \alpha Y_{t-1}^2 + \varepsilon_t$.

Martingales and Martingale Differences One important class of stationary processes is the class that the best predictors and the best linear predictors are identical. If the best one-step predictor

$$E\,(\varepsilon_{n+1}|\mathscr{F}_n) = E\,(\varepsilon_{n+1}|\varepsilon_n, \varepsilon_{n-1}, \ldots) = 0, \tag{11.2}$$

we say $\{\varepsilon_t\}$ is not forecastable, either linearly or nonlinearly, so that the best possible forecast of ε_{n+1} is zero. Any process $\{\varepsilon_t\}$ satisfying (11.2) is said to be a *Martingale difference*. So for a stationary time series, *i.e.*, satisfying (11.1), the best possible predictor is the same as the best linear predictor only when the innovations $\{\varepsilon_t\}$ are a Martingale difference.

To understand the reason for this terminology, we have to define a *Martingale*. The process $\{Y_t\}$ is said to be a Martingale if for all n,

and for all lead time $h > 0$,

$$E\,(Y_{n+h}|\mathscr{F}_n) = Y_n, \tag{11.3}$$

where $\mathscr{F}_n$ represents the information available up to time n. Equation (11.3) says that the best possible forecast of Y_{n+h} given the available information is simply the most recent observation, Y_n. The "change", $Y_{n+h} - Y_n$ is not predictable, because, by (11.3), the best forecast of this change is

$$E\,[(Y_{n+h} - Y_n)\,|\mathscr{F}_n] = E\,[Y_{n+h}|\mathscr{F}_n] - Y_n = 0.$$

We see from the above that if $\{Y_t\}$ is a Martingale, then the first difference $\varepsilon_t = Y_t - Y_{t-1}$ is a Martingale difference. The values of any Martingale difference series $\{\varepsilon_t\}$ must be uncorrelated, and must have zero mean.

Is there a series whose first differences are a Martingale difference but are not independent? A concrete example is provided by the series $Y_t = Y_{t-1} + \varepsilon_t$, where $\{\varepsilon_t\}$ obeys an ARCH model.

ARCH Processes and Conditional Volatility

The acronym ARCH stands for **A**uto**R**egressive **C**onditional **H**eteroscedasticity. The term "heteroscedasticity" refers to changing volatility (i.e., variance). But it is not the variance itself which changes with time according to an ARCH model; rather, it is the conditional variance which changes, in a specific way, depending on the available data. The conditional variance quantifies our uncertainty about

the future observation, given everything we have seen so far. This is of more practical interest to the forecaster than the volatility of the series considered as a whole.

To provide a context for ARCH models, let's first consider the conditional aspects of the linear AR (1) model $Y_t = \phi Y_{t-1} + \varepsilon_t$, where the $\{\varepsilon_t\}$ are independent white noise. The best predictor for Y_t from Y_{t-1} is the conditional mean $E(Y_t|Y_{t-1}) = \phi Y_{t-1}$. The success of the AR (1) model for forecasting purposes arises from the fact that this conditional mean is allowed depend on the available data, and evolve with time. The conditional variance, however, is simply

$$\text{var}(Y_t|Y_{t-1}) = \text{Var}(\varepsilon_t) = \sigma_\varepsilon^2,$$

which remains constant regardless of the given data. The novelty of the ARCH model is that it allows the conditional variance to depend on the data.

The ARCH(1) process

The simplest ARCH model, is the ARCH(1) model,

$$\begin{aligned} a_t &= \varepsilon_t \sigma_t \\ \sigma_t^2 &= \omega + \alpha a_{t-1}^2, \end{aligned} \tag{11.4}$$

where $\{\varepsilon_t\}$ is standard Gaussian white noise, i.e., $\varepsilon_t \sim i.i.d\ N(0,1)$. We must impose some constraints on the model parameters to obtain desirable properties, say, $\omega > 0$ and $\alpha \geq 0$. Obviously, this is to guarantee that σ_t^2 is positive. The conditional distribution of a_t

given $\mathscr{F}_{t-1}$, the information available up to time $t-1$, is Gaussian,

$$a_t|\mathscr{F}_{t-1} \sim N\left(0, \sigma_t^2\right).$$

In addition, it is possible to write the ARCH(1) model as non-Gaussian AR(1) model in a_t^2 by adding $\eta_t = a_t^2 - \sigma_t^2$ to both sides of equations (11.4). We get

$$a_t^2 = \omega + \alpha a_{t-1}^2 + \eta_t, \tag{11.5}$$

and $\{\eta_t\}$ is zero mean white noise.

Basic Properties

We now discuss the properties of ARCH processes using ARCH(1) models for simplicity. All these properties can be generalized easily for ARCH(p) models which will be defined later.

The ARCH $\{a_t\}$ are a zero mean *white noise process* as long as $\alpha < 1$ (in addition to $\alpha \geq 0$). That is,

$$E(a_t) = 0\,, \quad \text{var}(a_t) = \omega/(1-\alpha)\,, \quad \text{Cov}(a_s, a_t) = 0\,, \quad s \neq t\,.$$

It is only the conditional volatility which change with time, not the overall volatility.

The ARCH process is *nonlinear*, because its conditional variance on the past observations is not a constant as a linear process would have. Since it is nonlinear, the distribution of the $\{a_t\}$ tends to be more long-tailed than normal. Thus, outliers may occur relatively often. It reflects the leptokurtosis which is often observed in practice. Moreover, once an outlier does occur, it will increase the condi-

tional volatility for some time to come. See Equation (11.4). Once again, this reflects a pattern often found in real data.

Figure 11.2 shows the plots of a simulated ARCH(1) process with $\omega = 1.5$ and $\alpha = .9$. We see the *nonnormal distribution with fat tails* from the histogram and QQ plot. The sample autocorrelations of $\{a_t\}$ resemble those of a white noise process, while the sample autocorrelations of $\{a_t^2\}$ are dependent.

Since $E(a_t|\mathcal{F}_{t-1}) = 0$, the process $\{a_t\}$ is a *Martingale difference*. The very best predictor of a_t based on the available information, linear or nonlinear, is simply the trivial predictor, namely the series mean, 0. In terms of point forecasting of the series itself, then, the ARCH errors offer no advantages over the linear ARMA models with $i.i.d.$ white noise. The advantage of the ARCH models lies in their ability to describe the time-varying stochastic conditional volatility, which can then be used to improve the reliability of interval forecasts and to help us in understanding the process.

Although the series $\{a_t\}$ itself is not forecastable, it is *possible to forecast the squared series* $\{a_t^2\}$ because

$$E(a_t^2|\mathcal{F}_{t-1}) = \sigma_t^2 = \omega + \alpha a_{t-1}^2. \tag{11.6}$$

This equation seems to agree with the AR(1) representation of a_t^2 in (11.5). However, further constraints need to be imposed to ensure that $\{a_t^2\}$ is indeed a stationary AR(1) process so that the techniques for ARMA models can be applied to $\{a_t^2\}$. If the variance of a_t is finite, $0 \le \alpha < 1$. The definition of a stationary AR(1) also requires

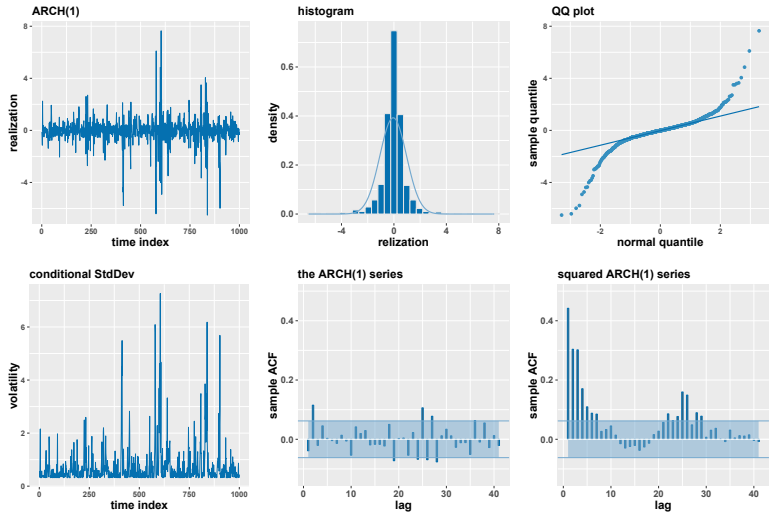

Figure 11.2: A realization of 1,000 from ARCH(1) model with $\alpha = .9$. The top panel are the time series plot, histogram and normal probability plot. The bottom panel are the time series of conditional standard deviations. The other two plots are the sample ACFs of the row and squared realizations.

$E\left(a_t^2\right)$ and $\text{var}\left(a_t^2\right)$ to be finite constant. A few algebraic steps give

$$E\left(a_t^2\right) = \text{var}\left(a_t\right) = \omega/\left(1-\alpha\right) ,$$
$$E\left(a_t^4\right) = \frac{3\omega^2}{(1-\alpha)^2}\frac{1-\alpha^2}{1-3\alpha^2} .$$

Thus if, in addition, $3\alpha^2 < 1$, the square of the process, $\left\{a_t^2\right\}$ follows a causal AR(1) model with ACF given by $\rho_{a^2}(h) = \alpha^h \ge 0$ for all $h > 0$. Note that (11.6) is the best forecast of a_t^2 and the white noise process $\{\eta_t\}$ is also a Martingale difference.

The ARCH(p) process

A process $\{a_t\}$ is ARCH (p) if the conditional distribution of a_t given the information available up to time $t-1$ is

$$a_t|\mathscr{F}_{t-1} \sim N\left(0,\sigma_t^2\right), \tag{11.7}$$

$$\sigma_t^2 = \omega + \sum_{i=1}^{p} \alpha_i a_{t-i}^2, \tag{11.8}$$

with $\omega > 0, \alpha_i \geq 0$ and $\sum_{i=1}^{p} \alpha_i < 1$.

Equation (11.14) says that the conditional distribution of a_t given $\mathscr{F}_{t-1}$ is normal, $N\left(0,\sigma_t^2\right)$. In other words, given the available information $\mathscr{F}_{t-1}$, the next observation has a normal distribution with a conditional mean $E\left(a_t|\mathscr{F}_{t-1}\right) = 0$, and a conditional variance of $\mathrm{Var}\left(a_t|\mathscr{F}_{t-1}\right) = \sigma_t$.

Equation (11.8) specifies the way in which the conditional variance σ_t^2 is determined by the available information. Note that σ_t^2 is defined in terms of squares of past innovations.

Graphicas for ARCH models

Because the ARCH (p) process a_t is white noise, we would typically expect to find that the ACF and PACF of the raw data are not significant at most lags. Examination of the ACF and PACF of the squared series, $\left\{a_t^2\right\}$ will often be more fruitful. Because the squared series obeys an AR(p) model, we would hope to find that the PACF of the squared series cuts off beyond lag p , while the ACF of the squared series "tails down".

Since any ARCH process is also a white noise process, its true ACF and PACF must be zero at all lags. Of course, the sample ACF and PACF will all differ from zero, due to sampling variation. We have been assuming that for any white noise process, the sample ACF and PACF will have a standard error of $1/\sqrt{n}$ which is used for the 95% confidence bands in the sample ACF and PACF plots. In fact , these reference lines are not necessarily valid unless we have independent white noise. For ARCH(1) processes, it can be shown that the variance of the sample ACF at lag h is

$$\mathrm{var}\{\hat{\rho}(h)\} \approx \frac{1}{n}\left\{1 + \frac{2\alpha^h}{1-3\alpha^2}\right\} .$$

Thus, the variance of $\hat{\rho}_h$ will be greater than $1/n$ in this case. For example, if $\alpha = .4$, then the variance of $\hat{\rho}_1$ is approximately $2.5/n$. If we naively compare the sample ACF with $\pm 1.96/\sqrt{n}$ as the case of no ARCH effects, we may face a problem. The same problem also appears in the Ljung-Box test for white noise, because its statistic which is the weighted sum of $\hat{\rho}_h^2$ relies on the sampling distributions of $\hat{\rho}_h, 1 \leq h \leq K$ being independent $N(0, 1/n)$ random variables under the null hypothesis.

On the other hand, using ACF and PACF of the squares to decide whether the raw series is independent is a justifiable procedure. If $\{\varepsilon_t\}$ is strict white noise and n is large, the sample ACF and PACF values for the squared series are approximately $i.i.d.\ N(0, 1/n)$. Thus any significant lags of the ACF or PACF of the squares can be taken as evidence of the raw series is not independent.

Instead of using graphical methods, which may be problematic, it is possible to use AIC or BIC to select the order p for the ARCH(p) model, but this requires that we estimate the ARCH parameters.

Eg 11.1. Daily returns of S&P 500 from Jan 1, 2008 to Nov 30, 2020, $n = 3251$.

```
head(Yn,2); tail(Yn,2);n = dim(Yn)[1];cat("\nsample size: n = ", n)

##                 GSPC
## 2008-01-03  0.000000
## 2008-01-04 -2.485797
##                 GSPC
## 2020-11-27  0.2394111
## 2020-11-30 -0.4606141
##
## sample size: n =  3251
```

Figure 11.3 shows that the ACF of returns is significant at a few lags but the magnitudes are relatively low in comparison to those of squared returns in both the ACF and PACF plots. The Ljung-Box tests have practically 0 values at all number of lags considered.

```
Ks = 2 +  3*(0:5); Y = cbind(Yn, Yn^2); pv = c()
for(i in Ks) pv = cbind(pv, apply(Y, 2, function(x) Box.test(x,i,"L")$p.val))
colnames(pv) = paste("K =", Ks); rownames(pv) = c("Yn", "Yn^2"); round(pv,4)

## P values of Ljung-Box Tests with different lags:
##       K = 2. K = 5  K = 8  K = 11 K = 14 K = 17
## Yn    0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
## Yn^2  0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
```

ARMA Models with ARCH Errors

Because an ARCH(p) process is Martingale difference and therefore nonforecastable, it is not usually used by itself to describe a time

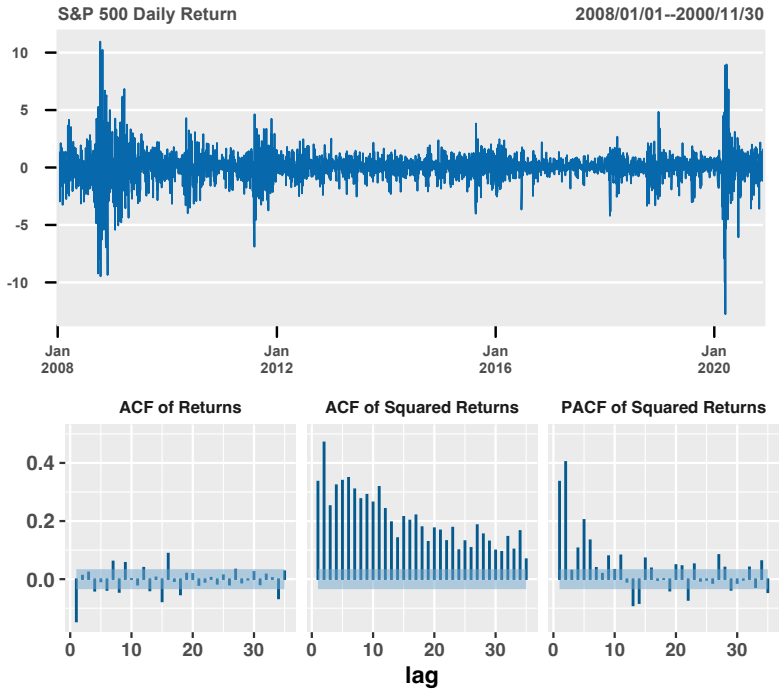

Figure 11.3: The top panel contains the daily log returns of S&P 500, 1/1/2008-11/30/2020. The bottom panel are the sample ACFs the sample ACF and Partial ACF of squared returns.

series when prediction is desired. In such cases, we can model our data $\{Y_t\}$ as an ARIMA(p_M, d, q_M) process where the innovations are ARCH(p). The difference equation of the model is the same as the regular ARIMA except the errors now are ARCHp). That is

$$Y_t = \sum_{j=1}^{p_M} \phi_j Y_{t-j} + \sum_{j=1}^{q_M} \theta_j a_{t-j} + a_t, \tag{11.9}$$

where $\{a_t\}$ is ARCH(p). Because $\{a_t\}$ is white noise, the model (11.9) is in many respects like an ordinary ARMA model which is

required the innovations simply being white noise. The ACF and PACF of the $\{Y_t\}$ given by (11.9) are the same as usual. In addition, the best possible predictor of Y_{n+m} based on $Y_n, Y_{n-1}, \ldots$ will be the best linear predictor because $\{a_t\}$ are a Martingale difference sequence. Perhaps the most notable difference with ARCH errors instead of independent errors is forecast intervals.

Suppose $\{Y_t\}$ is ARMA $(p_{\text{M}}, q_{\text{M}})$ with ARCH(p) errors $\{\varepsilon_t\}$ as in Models (11.9). Assume that all parameter values and model orders are known. The best forecast of Y_{n+1} based on $\mathscr{F}_n$ is

$$\hat{Y}^n_{n+1} = E\left[Y_{n+1}|\mathscr{F}_n\right] = \sum_{j=1}^{p_{\text{M}}} \phi_j Y_{n+1-j} + \sum_{j=1}^{q_{\text{M}}} \theta_j a_{n+1-j}. \qquad (11.10)$$

The one-step forecast error is $Y_{n+1} - \hat{Y}_{n+1} = a_{n+1}$. The conditional one step error variance is $\text{var}[a_{n+1}|\mathscr{F}_n] = \sigma^2_{n+1}$, which depends on the information only available at time n. We would like to construct a forecast interval for Y_{n+1} which has a *conditional* coverage rate of $1-\alpha$, given the observed information $\mathscr{F}_n$. Since $\{a_t\}$ is an ARCH process, we know from equation (11.14) that, conditionally on $\mathscr{F}_n$, the forecast error a_{n+1} is distributed as $N\left(0, \sigma^2_{n+1}\right)$. Thus a one-step forecast interval, with a coverage rate of $1-\alpha$, is given by

$$\hat{Y}^n_{n+1} \pm z_{\alpha/2}\sigma_{n+1} \qquad (11.11)$$

The interval can actually be constructed because σ^2_{n+1} depends on the information available at time n. The width of the one-step forecast intervals is $2z_{\alpha/2}\sigma_{n+1}$. This width will increase immediately in event of a sudden large fluctuation in the series.

Estimation of ARCH Models

Estimation of the parameter he parameter $\left(\omega, \alpha_1, \ldots, \alpha_p\right)'$ of the ARCH(p) model is typically accomplished by conditional MLE. This likelihood function is based on the conditional probability density function , which involves the joint density of $a_1, \ldots, a_p$,

$$L\left(\omega, \alpha_1, \ldots, \alpha_p | a_1, \ldots, a_p\right) = \prod_{t=p+1}^{n} f_{\omega,\alpha_1,\ldots,\alpha_p}\left(a_t|a_{t-1}\right)$$

where the density $f_{\omega,\alpha_1,\ldots,\alpha_p}\left(a_t|a_{t-1}\right)$ is the normal density specified in (11.14). Hence, the criterion function to be minimized, the negative logarithm of L is given by

$$\ell\left(\omega, \alpha_1, \ldots, \alpha_p\right) = \sum_{t=p+1}^{n} \log\left(\sigma^2_t\right) + \sum_{t=p+1}^{n} \left(\frac{a^2_t}{\sigma^2_t}\right).$$

The model can be selected model by either the AIC or BIC criteria. Apart from the normal distribution, some frequently used forms of $f(\cdot)$ are t-distribution (df > 2) and general error distribution.

The GARCH Models

The ARCH model has been extended in a number of directions. The most important of these is the extension to include "moving average" parts, namely the generalized ARCH (GARCH) model due to Bollerslev (1986). In empirical applications of the ARCH model, a relatively long lag (large order of p) is often required to have a reasonable fit to economic/financial time series. The GARCH models allow for a more flexible lag structure that can capture the con-

ditional heteroscedasticity in data in a more parsimonious way as compared to ARCH models.

The GARCH$(1,1)$ Process

The simplest but often very useful GARCH process is the GARCH(1,1) process. The GARCH(1,1) model is given by

$$\begin{aligned} a_t &= \varepsilon_t \sigma_t, \qquad \varepsilon_t \sim i.i.d.N(0,1), \\ \sigma_t^2 &= \omega + \alpha a_{t-1}^2 + \beta \sigma_{t-1}^2, \end{aligned} \tag{11.12}$$

with $\omega > 0, \alpha \geq 0,\ \beta \geq 0$. If $\beta = 0,\ \alpha \neq 0$, the model reduces to an ARCH(1). Analogous to the ARCH process, the conditional distribution of a_t given the information available up to time $t-1$ is

$$a_t | \mathscr{F}_{t-1} \sim N\left(0, \sigma_t^2\right).$$

The process $\{a_t\}$ is a stationary white noise if $\alpha + \beta < 1$ and the variance of a_t is

$$E\left(a_t^2\right) = \omega / (1 - \alpha - \beta).$$

Also, $\{a_t\}$ are a Martingale difference series, they are uncorrelated but not independent. The conditional variance σ_t^2 depends on not only past a_t^2 but also on past σ_t^2. The conditional variance σ_t^2 of a stationary GARCH(1,1) has an ARCH(∞) representation, that is the sum of infinite past a_t^2. From equation (11.12),

$$(1 - \beta B)\sigma_t^2 = \omega + \alpha a_{t-1}^2. \tag{11.13}$$

Applying the difference operator $(1-\beta B)^{-1}$ on both sides, we get

$$\sigma_t^2 = \sum_{j=0}^{\infty} \beta^j B^j \left(\omega + \alpha a_{t-1}^2\right) = \frac{\omega}{1-\beta} + \alpha \sum_{j=0}^{\infty} \beta^j a_{t-j-1}^2.$$

This is a marked difference from the ARCH(1) process in which the conditional variance depends only on a_{t-1}^2. This implies that the volatility cluster is more persistent for GARCH process than for ARCH processes. This also explains why a simple GARCH(1,1) model may provide a parsimonious representation for longer range dependence structure that can only accommodated by an ARCH(p) model with large p. The GARCH(1, 1) model has turned out to be tremendously successful in describing a wide variety of financial market data, it is regarded as the bench mark model by many econometricians.

The squared process of a GARCH(1,1) model entertains an interesting link to ARMA models. By letting $\eta_t = a_t^2 - \sigma_t^2$, we can write the squared process

$$a_t^2 = \omega + (\alpha + \beta) a_{t-1}^2 + \eta_t - \beta \eta_{t-1},$$

and $\{\eta_t\}$ is a white noise. It is easy to check that $\{\eta_t\}$ is also a Martingale difference series. Thus $\left\{a_t^2\right\}$ follows an ARMA(1,1) model. It can be shown that the fourth moment of a_t is

$$E\left(a_t^4\right) = \frac{3\omega^2(1+\alpha+\beta)}{(1-\alpha-\beta)(1-\beta^2-2\alpha\beta-3\alpha^2)}.$$

Thus under the additional condition $\beta^2 + 2\alpha\beta + 3\alpha^2 < 1$, $\left\{a_t^2\right\}$ is

a stationary and invertible ARMA(1,1) process. Its autocorrelation function is

$$\rho_{a^2,h} = \frac{\alpha\left(1-\beta^2-\alpha\beta\right)(\alpha+\beta)^{h-1}}{1-\beta^2-2\alpha\beta}, \quad h \geq 1.$$

The GARCH(p,q) Process

A white noise process $\{a_t\}$ is GARCH(p,q) if the conditional distribution of a_t given the information available up to time $t-1$ is

$$a_t|\mathscr{F}_{t-1} \sim N\left(0,\sigma_t^2\right), \tag{11.14}$$

$$\sigma_t^2 = \omega + \sum_{i=1}^{p}\alpha_i a_{t-i}^2 + \sum_{j=1}^{q}\beta_j\sigma_{t-j}^2, \tag{11.15}$$

with $\omega > 0, \alpha_i \geq 0,\ \beta_j \geq 0$ so that σ_t^2 is guaranteed to be positive. If $q=0$, we have the ARCH(p) process. In the GARCH(p,q) model, it is assumed that $q>0$ only if $p>0$. The necessary and sufficient condition for stationarity of $\{a_t\}$ is that $\sum_{i=1}^{p}\alpha_i+\sum_{j=1}^{q}\beta_j<1$. With this condition, $\{a_t\}$ are a Martingale difference hence white noise with zero mean and finite constant variance,

$$E\left(a_t^2\right) = \frac{\omega}{1-\sum_{i=1}^{q}\alpha_i-\sum_{j=1}^{p}\beta_j}\ .$$

For every stationary GARCH(p,q) process $\{a_t\}$, its conditional variance of σ_t^2 has an ARCH(∞) representation, i.e., the same of infinite past a_t^2. Furthermore, if $E\left(a_t^4\right)<\infty$, the squared process $\left\{a_t^2\right\}$

can be written as is process

$$a_t^2 = \omega + \sum_{i=1}^{p}\alpha_i a_{t-i}^2 + \sum_{j=1}^{q}\beta_j a_{t-i}^2 + \eta_t - \sum_{j=1}^{q}\beta_j\eta_{t-j}, \tag{11.16}$$

which is a causal and invertible ARMA(m,q), where $m=\max\{p,q\}$. Thus the sample autocorrelation and partial autocorrelation functions for the squared process can be helpful in identifying and checking time series behavior in the conditional variance equation of the GARCH form.

The conditional maximum likelihood estimation for the ARCH models can be modified for the GARCH models. The usual formulas of the AIC and BIC can then be computed.

ARMA models with GARCH errors

The ARCH error sequence $\{a_t\}$ in the ARMA model (11.9) can be extended to the larger class of GARCH white noise.

Testing for ARCH effect Classical tests based on the likelihood function have been derived for testing the null hypothesis of no ARCH effects, which can be formulated as

$$H_0: \alpha_1 = \cdots = \alpha_p \quad versus \quad H_A: \alpha_j \neq 0 \text{ for some } j.$$

These tests are the likelihood ratio test, the Wald statistic and the score test (also called Lagrange multiplier test). All three tests share the same asymptotic distribution under the null hypothesis, χ_q^2. The Lagrange multiplier test was advocated by Engle (1982) in the same paper that the ARCH model was introduced. Ljung-Box tests on the

squared series can also be applied as shown in Eg.11.1.

Model Diagnostics

After select a fitted model, we must perform appropriate model diagnostics. Like any model fit, diagnoistics are based on the residuals. For a given selected model, the "residuals" are defined as

$$\hat{\varepsilon}_t = a_t/\hat{\sigma}_t\,.$$

These residuals can be used for checking adequacy of the fitted ARCH model. We can inspect the sample ACF and PACF plots of squared residuals for possible correlation structure.

The tests for no ARCH effect can be applied on the residuals for testing the hypothesis of homoscedasticity against remaining conditional heteroscedasticity. If the fitting is perfectly adequate, the residuals should behave like Gaussian white noise according to equation (11.14). This can be checked by the normal probability plot of residuals. If the normality is violated, we may explore the possibility of fitting an GARCH model with heavy-tailed conditional distribution to this data.

Fitting ARMA + GARCH Models with R

The R package `rugarch` will be used for fitting GARCH models. It requires to specify both the ARMA and GARCH to be fit first. We will continue with the daily returns of S&P 500 of Eg. 11.1.

Eg 11.2. Fitting ARMA + GARCH to daily returns of S&P 500. The details of modeling conditional mean with an ARMA model are omit-

ted. We have selected AR(1) for the conditional mean model after carefully analyzing the data. Based on our discussion on pages 302-304 and ACF of returns in Figure 11.3, the white noise model can also be a candidate for our data.

```
ar1 = Arima(Yn, order = c(1,0,0)); ar1

## Series: Yn
## ARIMA(1,0,0) with non-zero mean
##
## Coefficients:
##           ar1    mean
##       -0.1456  0.0282
## s.e.   0.0174  0.0203
##
## sigma^2 = 1.752:  log likelihood = -5523.7
## AIC=11053.39   AICc=11053.4   BIC=11071.65

## Ljung-Box Tests on AR(1) residuals for DF 5--11
df = (5:11); pq = sum(ar1$arma[1:2]);pv = c()
for(i in df) pv[i] = Box.test(resid(ar1),i + pq, "L", pq)$p.val
pv = pv[is.na( pv)== F];names(pv) = df; round(pv,5)

##       5       6       7       8       9      10      11
## 0.05490 0.00343 0.00241 0.00014 0.00028 0.00044 0.00023
```

The `rugarch` package requires to specify a model to be fit with `ugarchspec()` then fit the model to data with `ugarchfit()`.

```
library(rugarch)
args(ugarchspec)
## function (variance.model = list(model = "sGARCH", garchOrder = c(1, 1),
##   submodel = NULL, external.regressors = NULL, variance.targeting = FALSE),
##   mean.model = list(armaOrder = c(1, 1), include.mean = TRUE,
##     archm = FALSE, archpow = 1, arfima = FALSE, external.regressors = NULL,
##     archex = FALSE), distribution.model = "norm", start.pars = list(),
##   fixed.pars = list(), ...)
args(ugarchfit)
## function (spec, data, out.sample = 0, solver = "solnp", ...)
```

The `variance.model` argument is for the GARCH model. There are many variants of GARCH, the one we have just discussed is `sGARCH`, the standard GARCH which is also the default model. The `garchOrder` argument gives the order of p and q of a GARCH. The distribution of ε_t which represents the conditional distribution of the GARCH process is specified in `distribution.model`. The default is normal distribution as defined in the original papers.

The `mean.model` argument is for the ARMA part. We will fit an AR(1) + GARCH(1,1) with default normal conditional distribution to the S&P 500 data to demonstrate `rugarch`'s `R` functions. The fit also provides useful return values and information for actual fits.

```
spec = ugarchspec(mean.model = list(armaOrder = c(1,0)),
    variance.model = list(garchOrder = c(1,1)))
fit = ugarchfit(data=Yn, spec=spec)
```

The returned object `fit` is an S4 object with a list of slots which are accessed with @ instead of $.

The `R` command `show(fit)` prints the summary report. We show only partial report with `showShort()` from my `R` code. The report is more than 2 pages.

```
showShort(fit)
```

```
## *---------------------------------*
## *          GARCH Model Fit        *
## *---------------------------------*
## GARCH Model : sGARCH(1,1)
## Mean Model : ARFIMA(1,0,0)
## Distribution : norm
```



```
##
## Optimal Parameters
## ------------------------------------
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.076341    0.012361   6.1757 0.000000
## ar1    -0.071843    0.019554  -3.6742 0.000239
## omega   0.027770    0.003897   7.1259 0.000000
## alpha1  0.165353    0.014937  11.0701 0.000000
## beta1   0.820572    0.013495  60.8064 0.000000
##
## LogLikelihood : -4441.774
##
## Information Criteria
## ------------------------------------
##  Akaike Bayes Shibata Hannan-Quinn
##  2.7356 2.745  2.7356        2.739
##
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                     0.9250  0.3362
## Lag[2*(p+q)+(p+q)-1][2]    0.9321  0.7799
## Lag[4*(p+q)+(p+q)-1][5]    2.0474  0.6976
## d.o.f=1
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                     0.4167  0.5186
## Lag[2*(p+q)+(p+q)-1][5]    1.8280  0.6596
## Lag[4*(p+q)+(p+q)-1][9]    2.6228  0.8192
## d.o.f=2
##
## Weighted ARCH LM Tests
## ------------------------------------
##             Statistic Shape Scale P-Value
## ARCH Lag[3]   0.05169 0.500 2.000  0.8202
## ARCH Lag[5]   1.62560 1.440 1.667  0.5599
## ARCH Lag[7]   1.84631 2.315 1.543  0.7498
##
## Elapsed time : 0.08213592
```

The list of slots of an S4 object and be shown with `SlotNames()`. There are two slots of the returned object `fit` from ugarchfit(), `fit@modle` and `fit@fit`. Both slots are S3 objects. The list of names in the slot can be shown by the command `names(fit@model)` and `names(fit@fit)`. The `@model` slot contains all the inputs include the data and spec. All the computational output from fitting the model to the data are in the `@fit` slot. We list those sub-object names that will be used for diagnostics in Table 11.1, they can also be retreived by the accompany R functions.

Table 11.1: Some R functions

R Function	Returned Object	What it is
`fitted(fit)`	`fit@fit$fitted.values`	$\hat{Y}_t, t = 1, \ldots, n$
`residuals(fit)`	`fit@fit$residuals`	$\hat{a}_t = Y_t - \hat{Y}_t$
`sigma(fit)`	`fit@fit$sigma`	$\hat{\sigma}_t, t = 1, \ldots, n$
`residuals(fit, stand = T)`	`fit@fit$z`	$\hat{\varepsilon}_t = \hat{a}_t / \hat{\sigma}_t$

In addition, the functions `likelihood(fit)` are `coef(fit)` are clear what their returned values are. The `infocriteria(fit)` function gives the four information criteria, AIC, BIC, Shibata and Hannan-Quinn . For example,

```
infocriteria(fit)

##
## Akaike       2.735634
## Bayes        2.744996
## Shibata      2.735630
## Hannan-Quinn 2.738988
```


There are 12 plots can be automatically generated. One can select a plot from the interactive menu in R console window.

```
> plot(fit)

 Make a plot selection (or 0 to exit):

 1:   Series with 2 Conditional SD Superimposed
 2:   Series with 1% VaR Limits
 3:   Conditional SD (vs |returns|)
 4:   ACF of Observations
 5:   ACF of Squared Observations
 6:   ACF of Absolute Observations
 7:   Cross Correlation
 8:   Empirical Density of Standardized Residuals
 9:   QQ-Plot of Standardized Residuals
10:   ACF of Standardized Residuals
11:   ACF of Squared Standardized Residuals
12:   News-Impact Curve

Selection:
```

Enter a number between 1 and 12 to Selection will generate the plot. We can always generate a plot by the command `plot(fit, which = 3)`. To generate all 12 plots, set `which = "all"`.

The weighed Ljung-Box statistic is the Ljung -Box statistic with a weight function which can potentially improve power of a test. The Ljung -Box tests for no remaining serial correlation are not significant. Both the Ljung -Box tests and LM tests for no remaining ARCH effects have high p-values, indicating the normalized sequence ε_t are uncorrelated.

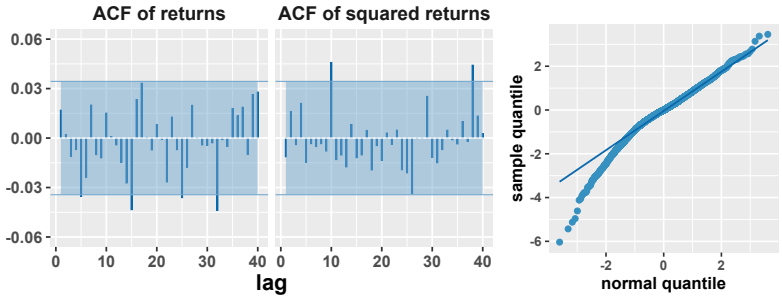

Figure 11.4: ACF of $\hat{\varepsilon}_t$ and $\hat{\varepsilon}_t^2$, and Normal probability plots of $\hat{\varepsilon}_t$ from fitting AR(1) + GARCH(1,1) with Normal conditional distribution.

Examining the residual plots (plots 8, 9, 10, 11) can be useful to further investigate the the model assumptions. We find that the conditional distribution of errors is deviated from the normal distribution in our model specification, see Figure 11.4.

This preliminary fit serves several purposes, (1) Recheck the ARMA coefficient redundancy; (2) decide the maximum p and q of GARCH to be considered; (3) obtain a proxy set of ε_t to find a suitable conditional distribution if the normal does not fit well as shown in Figure 11.4. Since the publication of ARCH and GARCH models, it has been well established that conditional normality assumption is often violated for financial data.

The AR coefficient in our model is significant. The tests for the fit are based on the standardized residuals $\hat{\varepsilon}_t$. The null hypotheses of zero correlation in ε_t and ε_t^2 are not rejected. The latter implies no remaining ARCH effects, which is also confirm by the Lagrange Multiplier tests. The GARCH(1,1) fits well, to be thorough, we can also check for order $p = 2$ or $q = 2$.

The conditional distribution of a_t is imperative in GARCH modeling as it is used to construct the prediction intervals and calculate VaR's. The distribution name is given in the `distribution.model` argument of `ugarchfit()` and it can be one of the eight distribution names, `"norm"`, `"snorm"`, `"std"`, `"sstd"`, `"ged"`, `"sged"`, `"nig"`, `"jsu"`. We will used $\hat{\varepsilon}_t$ the first fit to search for the best fit distribution. We will also find the best fit for the marginal distribution of a_t, which will be used to construct PI for the ARMA + ß.$i.d.$ white noise model later.

```
at = residuals(fit);  ## for fitting the marginal distribution
et = residuals(fit, stand = T);  ## for fitting the conditional distribution
```

The `distribution.model` argument in `ugarchfit()` can be one of the eight distribution names, `"norm"`, `"snorm"`, `"std"`, `"sstd"`, `"ged"`, `"sged"`, `"nig"`, `"jsu"`. Fitting these distributions to the 2 sets of residuals $\hat{a}_t$ and $\hat{\varepsilon}_t$ yields different minimum value models, the STD and NIG respectively.The BIC has the same value orders for distribution models, thus they are omitted.

```
cat("AIC of residuals from AR(1)"); sort(at.aic)

## AIC of residuals from AR(1)
##       std       nig      sstd     snorm       ged      sged       jsu
##  9896.042  9956.795 10573.275 11071.388 11071.388 11073.388 25349.880

cat("AIC of standardized residuals from AR(1) + GARCH(1,1)"); sort(et.aic)

## AIC of standardized residuals from AR(1) + GARCH(1,1)
##       nig       std     snorm       ged      sged      sstd       jsu
##  9042.018  9071.314  9223.640  9223.640  9225.640  9751.266 28712.131
```

Figure 11.5 shows the Q-Q plots of fitting the STD and NIG to both sets of residuals. These plots agree with the AICs. We also add the

fits to the original returns which, as expected, are almost identical of those Q-Q plots of the AR(1) residuals.

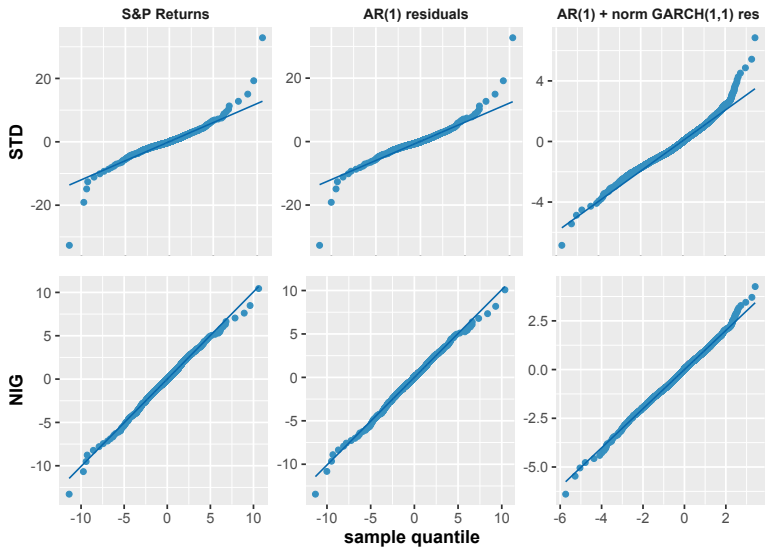

Figure 11.5: Sample quantiles versus standardized t and Normal Inverse Gaussian quantiles of the S&P 500 returns, residuals from fitting AR(1) and the standardized residuals from fitting AR(1) + GARCH(1,1) with Normal conditional distribution.

Contrary to the AIC/BIC selection, the t distribution does not fit $\hat{a}_t$ well. We will use NIG for both marginal and conditional distributions for a_t.

Eg 11.3. Selecting an ARMA + GARCH model. Continuation of Eg. 11.1 and Eg. 11.2. The diagnostic analysis of the first AR(1) + GARCH(1,1) fit shows no correlation or ARCH effect in ε_t but the normal conditional distribution is not suitable. We will change the conditional distribution by setting `distribution.model = "nig"`.

For a more thorough analysis, we will consider a few more GARCH models but only up to order of (2,2).

```
## Fitting AR(1) + GARCH(p,q) with NIG conditional distribution
ic = c(); ind = 0
fits = vector("list", 6)
for(p in 1:2){
    for(q in 0:2){
        ind = ind +1
        spec = ugarchspec(mean.model = list(armaOrder = c(1,0)),
                          variance.model = list(garchOrder = c(p,q)),
                          distribution.model = "nig")
        garch = ugarchfit(data=Yn, spec=spec)
        ic = cbind(ic, infocriteria(garch))
        fits[[ind]] = garch
        names(fits)[ind] = paste0("garch",p,q)
  }
}
colnames(ic) = names(fits); ic
```

```
##               garch10  garch11  garch12  garch20  garch21  garch22
## Akaike       2.875314 2.655423 2.655931 2.776346 2.652179 2.652623
## Bayes        2.886547 2.668529 2.670909 2.789452 2.667157 2.669473
## Shibata      2.875307 2.655414 2.655919 2.776337 2.652167 2.652607
## Hannan-Quinn 2.879338 2.660118 2.661296 2.781041 2.657545 2.658659
```

```
cat("Model Selected:");apply(ic,1,function(u) colnames(ic)[which.min(u)])
```

```
## Model Selected:
##       Akaike        Bayes      Shibata Hannan-Quinn
##    "garch21"    "garch21"    "garch21"    "garch21"
```

Both AIC and BIC choose GARCH(2,1) for the errors. The diagnostic plots in Figure 11.6 show very little difference between fits with GARCH(1,1) and GARCH(2,1). We will choose AR(1) + GARCH(1,1) with NIG conditional error distribution as our final model for the S&P 500 daily return series.

The coefficient estimates of the models are shown below. Note that

Figure 11.6: ACF of residuals and squared residuals, and the Q-Q plots of NIG versus sample quantiles. The top panel are based on the standardized residuals from AR(1) + GARCH(1,1) fit, the bottom panel are based on the standardized residuals from AR(1) + GARCH(2,1) fit.

the distribution of ε_t is assumed to have mean 0 and standard deviation 1, only the shape and skew estimates are listed.

```
showShort(fits$garch11)

## *---------------------------------*
## *          GARCH Model Fit        *
## *---------------------------------*
## GARCH Model : sGARCH(1,1)
## Mean Model : ARFIMA(1,0,0)
## Distribution : nig
##
## Optimal Parameters
## ------------------------------------
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.059766    0.011828   5.0530    0e+00
## ar1    -0.085637    0.017588  -4.8690    1e-06
```



```
## omega   0.017305    0.003866   4.4757    8e-06
## alpha1  0.149272    0.016339   9.1360    0e+00
## beta1   0.845267    0.014656  57.6748    0e+00
## skew   -0.237831    0.034524  -6.8889    0e+00
## shape   1.558982    0.236907   6.5806    0e+00
##
## LogLikelihood : -4309.391
##
## Information Criteria
## ------------------------------------
##  Akaike  Bayes Shibata Hannan-Quinn
##  2.6554 2.6685  2.6554       2.6601
##
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                      2.667 0.10247
## Lag[2*(p+q)+(p+q)-1][2]     2.671 0.06632
## Lag[4*(p+q)+(p+q)-1][5]     3.760 0.26624
## d.o.f=1
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                   0.002295  0.9618
## Lag[2*(p+q)+(p+q)-1][5]  1.765158  0.6748
## Lag[4*(p+q)+(p+q)-1][9]  2.789849  0.7931
## d.o.f=2
##
## Weighted ARCH LM Tests
## ------------------------------------
##             Statistic Shape Scale P-Value
## ARCH Lag[3]   0.01104 0.500 2.000  0.9163
## ARCH Lag[5]   1.68662 1.440 1.667  0.5447
## ARCH Lag[7]   2.06602 2.315 1.543  0.7033
```

The diagnostic tests of remaining correlation and ARCH effect are not significant, indicating the model fits well as Figure 11.6 also suggested.

Forecast with an ARMA + GARCH Models

Forecasting ARMA + GARCH processes is the same the regular ARMA models, but the prediction interval construction is very different from the ARMA with $i.i.d.$ white noise. The latter uses the constant estimates for the standard deviation of white noise, the former uses conditional standard deviation estimates of the white noise.

The h-Step-Ahead Forecasts

The one step forecast and interval is given by (11.10) and (11.11). The critical value $\pm z_{\alpha/2}$ should be replaced by that of the fitted distribution. The forecast requires both conditional mean and standard deviation $\hat{\sigma}_{n+1} = \sqrt{\hat{\sigma}^2_{n+1}}$ (instead of $\hat{\sigma}$), where

$$\hat{\sigma}^2_{n+1} = \hat{\omega} + \sum_{i=1}^{p} \hat{\alpha}_i \hat{a}^2_{n-i+1} + \sum_{j=1}^{q} \hat{\beta}_j \hat{\sigma}^2_{n-j+1}.$$

The h-step prediction $\hat{\sigma}^2_{n+h}$, $h \geq 2$ is obtained by replacing any future a^2_{n+i} and σ^2_{n+i} in the equation by the predicted values instead of fitted values. The h-step conditional standard deviation estimate $\hat{\sigma}_{n+h}$ based on the information up to time n converge to the $\sqrt{\hat{\gamma}_0}$, the unconditional standard deviation estimate as h increases.

For long lead times the prediction intervals for the two GARCH errors and $i.i.d$ white noise are similar because the forecast of conditional standard deviation will converge to the unconditional standard deviation. For shorter lead times, however, the prediction limits can be quite different.

Rolling Forecasts

For a fixed h, the rolling forecasts of h-step forecasts are computed by updating the most current information successively over a period time. The word "rolling" refers to the most distant past observation in the last forecast is also dropped as most of the models depend only on finite many past observations. It is a common approach among practitioners to fully utilized a time series model.

The rolling forecast is also an effective tool to check how well a model forecasts, by comparing the forecasted values with the actual observations. This is done by leaving the last, say, 200 observations out in estimating the model then comparing the rolling forecasts with these 200 observations. Forecasting ARIMA + GARCH model can be carried out using the `R` function `ugarchforecast()` from the `rugarhch` package.

Eg 11.4. Forecasting S&P 500 returns with AR(1) + GARCH(1,1) model. Continuation of Eg. 11.1 and Eg. 11.3. The S&P 500 daily return data starts from the beginning of 2008 and ends at November 15, 2024. We have used about the first 13 years 3,251 observations to model an ARMA + GARCH model and left 5 years 997 observations for validation.

```
n = dim(Yn)[1]; n0 = dim(Yt)[1]; n.fore = n0 - n
dates  = data.frame(data = c("Total data", "Sample", "Forecast"),
from = paste(time(Yt)[c(1,1,n + 1)]),to = paste(time(Yt)[c(n0, n, n0)]),
Size = c(n0,n,n.fore)); dates
```

```
##         data       from         to Size
## 1 Total data 2008-01-03 2024-11-15 4248
## 2     Sample 2008-01-03 2020-11-30 3251
## 3   Forecast 2020-12-01 2024-11-15  997
```

Before applying `ugarchforecast()` function, we need to re-apply `ugarchfit()` with the spec of chosen model to the full data. This will not change the parameter estimates, the purpose is to specify the training set and the validation set. This is done by setting the `data` argument in `ugarchfit()` being the full data and the `out.sample` argument the number of observations to be left out.

```
spec.11 = ugarchspec(mean.model = list(armaOrder = c(1,0)),
    variance.model = list(garchOrder = c(1,1)), distribution.model = "nig")
fit.11 = ugarchfit(data = Yt, spec = spec.11, out.sample = n.fore)
```

The `ugarchforecast()` function takes the returned object from `ugarchfit()` and produces forecasts based on the fitted model.

```
args(ugarchforecast)

## function (fitORspec, data = NULL, n.ahead = 10, n.roll = 0, out.sample = 0,
##     external.forecasts = list(mregfor = NULL, vregfor = NULL),
##     trunclag = 1000, ...)
```

h-Step-Ahead Forecasts If we are interested in forecast h-step-ahead forecasts, we only need to specify `n.head`. Applying `fitted()` and `sigma()` on the returned object will get the point forecast $\hat{Y}_{n+h}$ and forecast error standard deviations for $h = 1, \ldots, 5$.

```
fore = ugarchforecast(fit.11, n.ahead = 5)
cat("1-5 days ahead forecasts");fitted(fore)

## 1-5 days ahead forecasts
##     2020-11-30
## T+1 0.10432976
## T+2 0.05594971
## T+3 0.06009283
## T+4 0.05973802
## T+5 0.05976841
```



```
cat("1-5 days forecasting errors:")

## 1-5 days forecasting errors:

sigma(fore)

##     2020-11-30
## T+1  0.9214706
## T+2  0.9283193
## T+3  0.9350809
## T+4  0.9417575
## T+5  0.9483509
```

The h-step-ahead forecast intervals can be constructed with critical values from NIG. Since the NIG is not symmetric, both the lower and higher quantiles are required for calculating prediction intervals,

$$(\hat{Y}_{n+h} + q_{.025}\hat{\sigma}_{n+h},\ \hat{Y}_{n+h} + q_{.975}\hat{\sigma}_{n+h})$$

where $q_{.025}$ and is $q_{.975}$ are the 0.025 and 0.975 quantiles of the NIG. Recall that the ε_t is asummed to have mean 0 and standard deviation 1, which are the default of `rugarch`'s `qdist()` function, we only need to input the skew and shape estimates from the GARCH fit.

```
cri = qdist("nig", p = c(.025, .975), skew = coef(fit.11)["skew"],
            shape = coef(fit.11)["shape"])
cat("95% Prediction intervals:");cbind(fitted(fore) + cri[1]*sigma(fore),
fitted(fore) + cri[2]*sigma(fore))

## 95% Prediction intervals:
##     2020-11-30 2020-11-30
## T+1  -1.945873   1.795128
## T+2  -2.009491   1.759314
## T+3  -2.020392   1.775864
## T+4  -2.035601   1.787760
## T+5  -2.050241   1.799889
```

Rolling Forecasts The default number of rolling is set to 0 (`n.roll = 0`) and steps ahead 10 (`n.ahead = 10`). To obtain the intended rolling forecasts, we specify the settings in `ugarchforecast()` accordingly. For example, by setting `n.roll = 19` and `n.ahead = 5` will produce 5×20 dimensional forecasts and conditional standard deviation. These are the 1-, 2-, ..., 5-step-ahead forecasts forecasted at time n, $n+1, \ldots, n+19$.

We will compare the one-step-ahead forecasts along with their corresponding prediction intervals to the actual returns for dates from Dec 1, 2020 to Nov 15, 2024, a total of 997 dates. As before, the forecasts and forecast standard errors can be obtained by applying `fitted()` and `sigma()` on the object from `ugarchforecast()`.

```
fore = ugarchforecast(fit.11, n.roll = n.fore-1)
cat("Dimension of forecasts:", dim(fitted(fore)));  ## Default n.ahead = 10
```

```
## Dimension of forecasts: 10 997
```

```
fitted(fore)[1:2,1:5]; ## the 1- and 2-step ahead forecasts at 11-30 to12-04
```

```
##     2020-11-30  2020-12-01 2020-12-02 2020-12-03  2020-12-04
## T+1 0.10432976 -0.03109889 0.04955891 0.07023095 -0.01045459
## T+2 0.05594971  0.06754740 0.06064012 0.05886983  0.06577949
```

The default setting `n.ahead = 10`, we only need the 1-step ahead forecasts which are in the first row labelled `T+1`. The same dimension and labelling for the forecasting errors.

```
Ynh = Yt["2020-12-01::"]  # Obs starting from 2020-12-01
lo.s = fitted(fore)["T+1",] + cri[1]*sigma(fore)["T+1",] # lower bd of PI
up.s = fitted(fore)["T+1",] + cri[2]*sigma(fore)["T+1",] # upper bd of PI
cat("coverage rate of 95% PI:", mean((Ynh > lo.s)*(Ynh < up.s)))
```

```
## coverage rate of 95% PI: 0.9508526
```


Comparing with the actual observations and the forecasts, we find the coverage rate is excellent. The `R` function `rate()` I provided shows coverage rates for the nominal levels 95% and 90%.

```
cat("AR(1) + GARCH(1,1)"); rate(fore)
```

```
## AR(1) + GARCH(1,1)
##
##       One-Step Rolling Forecast
## ----------------------------------
##        coverage below PI beyond PI
## 95% PI   0.9509   0.0261    0.0231
## 90% PI   0.8977   0.0542    0.0481
```

The output also includes the rates below and beyond PIs. The empirical study shows asymmetry coverage in these prediction intervals. That is, there are more observations fall below the PIs than exceed the PIs. The asymmetry property is shown in the coverage rates.

We would like to compare to the forecast of the AR(1) + *i.i.d.* White Noise model. For a fair comparison, the white noise distribution is modeled and is set to be *i.i.d.* NIG from our data exploration earlier. The fit can be done using `rugarch`'s `arfimafit()` function.

```
spec.0 = arfimaspec(mean.model = list(armaOrder = c(1, 0)),
                    distribution.model = "nig")
fit.0 = arfimafit(data = Yt, spec = spec.0, out.sample = n.fore)
fore.0 = arfimaforecast(fit.0, n.roll = n.fore-1)
```

We also leave `n.fore = 997` for validation. The rolling forecasts of the ARMA + White Noise fit can be obtained by applying `rugarch`'s function `arfimaforecast()` to the `arfimafit()` returned object. The short version of summary output of `arfimafit()` can be obtained with `showShort0()` function from `GARCH_RFunction.R` file.

```
showShort0(fit.0)

## *----------------------------------*
## *          ARFIMA Model Fit        *
## *----------------------------------*
## Mean Model : ARFIMA(1,0,0)
## Distribution : nig
##
## Optimal Parameters
## ------------------------------------
##        Estimate  Std. Error  t value Pr(>|t|)
## mu     0.028250    0.021592   1.3084  0.19075
## ar1   -0.080333    0.014218  -5.6501  0.00000
## sigma  1.330018    0.043854  30.3282  0.00000
## skew  -0.165715    0.043906  -3.7743  0.00016
## shape  0.231334    0.024608   9.4007  0.00000
##
## LogLikelihood : -4791.638
##
## Information Criteria
## ------------------------------------
##  Akaike  Bayes Shibata Hannan-Quinn
##  2.9509 2.9602  2.9509       2.9542
##
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                         statistic   p-value
## Lag[1]                      14.32 1.539e-04
## Lag[2*(p+q)+(p+q)-1][2]     14.33 1.033e-14
## Lag[4*(p+q)+(p+q)-1][5]     17.42 4.268e-07
##
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                      279.5       0
## Lag[2*(p+q)+(p+q)-1][2]     640.0       0
## Lag[4*(p+q)+(p+q)-1][5]    1180.3       0
##
## ARCH LM Tests
## ------------------------------------
```



```
##              Statistic DoF P-Value
## ARCH Lag[2]      805.4   2       0
## ARCH Lag[5]      932.3   5       0
## ARCH Lag[10]    1001.7  10       0
```

Both the Ljung-Box test on the squared residuals and LM test for ARCH effects are significant as we expect. The computation of coverage rate of the one-step-ahead rolling forecast is similar. The R function `rate0()` is created for computing the coverage rates.

```
cat("AR(1) + i.i.d. NIG Noise");rate0(fore.0)

## AR(1) + i.i.d. NIG Noise
##
##       One-Step Rolling Forecast
## ----------------------------------
##        coverage below PI beyond PI
## 95% PI   0.9779   0.0130    0.0090
## 90% PI   0.9278   0.0341    0.0381
```

The coverage rates of the model with $i.i.d.$ noise are over the nominal rates. The difference can be seen in Figure 11.7. The grey band are the constant width 95% PIs based on the AR(1) + $i.i.d.$ NIG, the red dots denote those observations that are not covered by the PIs. We see the constant width PIs tend to be over-coverage during the low volatility periods and under-coverage in the high volatility periods. Notably, almost all misses concentrate on the first year of Russian-Ukraine War starting in February, 2022 when the volatility is high. Clustering in outliers is apparent.

The PIs based on the AR(1) + GARCH(1,1) are shown in light blue, with the dark blue dots indicating outliers, i.e., those not covered by

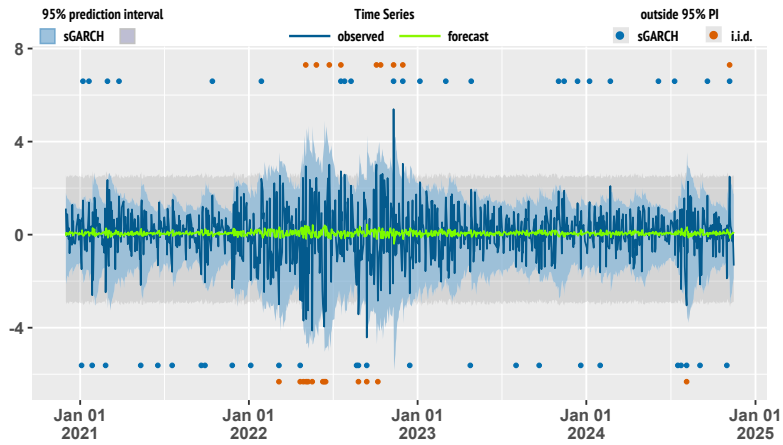

Figure 11.7: Rolling 1-step forecasts and 95% prediction intervals of S&P's daily returns from Dec 1, 2020 to Nov 15, 2024. The forecasts are based on AR(1) + *i.i.d.* NIG and AR(1) + GARCH(1,1) models.

the 95% PIs. The PIs based on the GARCH model adjust their widths according to the volatility persistence. The widths are shorter during the low volatility periods and wider during the high volatility periods. The observations not covered by the PIs are more evenly distributed over the entire period.

Estimating VaR and ES Using ARMA + GARCH Models

The parametric value at risk and expected shortfall discussed in Handout 9 treats asset returns as *i.i.d.* data. As we have learned, the *i.i.d.* assumption is not accurate. If the autocorrelation is weak as most of asset returns are, it may be ignored, but the volatility clustering is more crucial to the risk measure. By using ARMA + GARCH models, the VaR(α) and ES(α) can be adjusted to periods of high or low volatility.

Value at Risk Suppose that $F(\cdot)$ and $f(\cdot)$ are the CDF and PDF of the return distribution and S is the amount of investment, the $(1-\alpha)100\%$ VaR is

$$\text{VaR}(\alpha) = -S \times F^{-1}(\alpha)$$

where $F^{-1}(\alpha)$ is simply the α quantile of the distribution and is estimated by $F(\alpha\,|\,\hat{\boldsymbol{\theta}})$ with the MLE $\hat{\boldsymbol{\theta}}$ for the parameters. In stead of using the marginal distribution, we use the conditional distribution of the returns. If we have n returns, $R_1,\ldots,R_n$ and we need to estimate VaR and ES for the next return R_{n+1} conditioning on the current information. After fitting an ARMA + GARCH model, we would use the one-step-ahead forecasts $\hat{\mu}_{n+1|n}$ and $\hat{\sigma}_{n+1|n}$ combining the MLEs of the rest of parameters.

Eg 11.5. If we invest $S = \$10,000$ on the S&P 500 index, we can use the fitted model AR(1) + GARCH(1,1)|sGED from Eg. 11.3 to compute the one-day VaR. In this example, we will estimate the level 5% one-day VaR assuming today is Nov 30, 2020, Dec. 1, 2020, ..., and Dec. 4, 2020.

```
alpha = 0.05; S = 10000/100 ## remove %
est = coef(fit.11)[c("skew","shape")]
mu = fitted(fore)["T+1", 1:5]; sig = sigma(fore)["T+1", 1:5];
q = qdist("nig", p = alpha, mu = mu, sigma = sig,  skew = est[1],
          shape = est[2])
cat("1-day VaR with ARMA + GARCH(1,1) model:"); -S*q

## 1-day VaR with ARMA + GARCH(1,1) model:
## 2020-11-30 2020-12-01 2020-12-02 2020-12-03 2020-12-04
##   147.6491   164.8867   146.1244   133.8948   143.0787
```

From Figure 11.7, the volatility of the first week was low. The VaRs calculated with GARCH errors above are expected to be lower than those of AR(1) with i.i.d. Noise model calculated below.

```
mu.0 = fitted(fore.0)["T+1", 1:5];
est0 = coef(fit.0)[c("sigma", "skew", "shape")]
q.0 = qdist("nig", p = alpha, mu = mu.0,  sigma = est0[1],
            skew = est0[2], shape = est0[3])
cat("1-Day VaR with iid Noise:"); -S*q.0

## 1-Day VaR with iid Noise:
## 2020-11-30 2020-12-01 2020-12-02 2020-12-03 2020-12-04
##   193.2414   205.9455   198.3793   196.4401   204.0089
```

We also compare the one-day VaR estimates base on the AR(1) with GARCH error and with *i.i.d.* noise during the higher volatility period, say, next 5 trading days assuming today is 5/3/2022.

```
today = which(colnames(fitted(fore)) == "2022-05-03")
today = today:(today + 4)
#  ARMA + GARCH
mu = fitted(fore)["T+1", today]; sig = sigma(fore)["T+1", today]
q = qdist("nig", p = alpha, mu = mu, sigma = sig,  skew = est[1],
          shape = est[2])
#  ARMA with  Noise
mu.0 = fitted(fore.0)["T+1", today]
q.0 = qdist("nig", p = alpha, mu = mu.0, sigma = est0[1],
            skew = est0[2], shape = est0[3])
cat("1-Day VaR with ARMA + GARCH model:"); -S*q

## 1-Day VaR with ARMA + GARCH model:
## 2022-05-03 2022-05-04 2022-05-05 2022-05-06 2022-05-09
##   307.9672   364.1675   354.1658   354.8985   370.3307

cat("1-Day VaR with iid Noise:"); -S*q.0

## 1-Day VaR with iid Noise:
## 2022-05-03 2022-05-04 2022-05-05 2022-05-06 2022-05-09
##   200.8181   220.5797   167.7804   192.3704   170.7841
```


As expected, the VaR estimates based on the *i.i.d.* noise model are underestimated in the high volatile period. It shows that a model being able to be updated for volatility is crucial in measuring the investment risks.

Expected Shortfall The ES can be computed in a similar way if the conditional distribution is normal or t because the explicit formula are available. For other conditional distributions, we would either numerical evaluated the integration for the ES or simulate ES based on the distribution with the forecasts and estimates as parameters. The next routing demonstrates the simulated expected shortfalls for the same dates as the last calculation.

```
## Simulation for ES with AR(1) + GARCH(1,1) model
set.seed(11242024)
Tn = 100000; iter = 25; esf = c()  ## Tn:sample size each realization
system.time(  ## time the simulation
for(i in 1:length(q)){  ## each day
    x = c()
    for(j in 1:iter){
        rt = rdist("nig",Tn, mu[i], sig[i], skew = est[1], shape = est[2])
        x[j] = mean(rt[rt < q[i]])
    }
    esf[i] = mean(x)
})["elapsed"]

## elapsed
##   9.683

names(esf) = time(Ynh)[today-1]
cat("1-day Expected Shortfall with GARACH errors:"); -S*esf

## 1-day Expected Shortfall with GARACH errors:
## 2022-05-03 2022-05-04 2022-05-05 2022-05-06 2022-05-09
##   443.2748   513.8300   524.8938   514.8614   546.9101
```