Studies in Nonlinear Dynamics and Econometrics

Quarterly Journal
July 1998, Volume 3, Number 2
The MIT Press

Studies in Nonlinear Dynamics and Econometrics (ISSN 1081-1826) is a quarterly journal published electronically on the Internet by The MIT Press, Cambridge, Massachusetts, 02142. Subscriptions and address changes should be addressed to MIT Press Journals, Five Cambridge Center, Cambridge, MA 02142; tel.: (617) 253-2889; fax: (617) 577-1545; e-mail: journals-orders@mit.edu. Subscription rates are: Individuals $40.00, Institutions $130.00. Canadians add 7% GST. Prices subject to change without notice.

A Markov-Chain Sampling Algorithm for GARCH Models

Teruo Nakatsuma
Institute of Economic Research
Hitotsubashi University
Naka 2-1, Kunitachi
Tokyo 186-8603, Japan
cr00387@srv.cc.hit-u.ac.jp

Abstract. *This paper describes a GAUSS program of a Markov-chain sampling algorithm for GARCH models proposed by Nakatsuma (1998). This algorithm allows us to generate Monte Carlo samples of parameters in a GARCH model from their joint posterior distribution. The samples obtained by this algorithm are used for Bayesian analysis of the GARCH model. As numerical examples, GARCH models of simulated data and of weekly foreign exchange rate series are estimated and analyzed.*

Keywords. Bayesian inference; GARCH; Markov-chain Monte Carlo; Metropolis-Hastings algorithm

1 Overview

The ARCH/GARCH family of heteroskedastic time-series models (ARCH by Engle [1982], GARCH by Bollerslev [1986], among others) has gained popularity among academic researchers as well as financial practitioners; see the work of Bollerslev, Engle, and Nelson (1994) for more details on GARCH models. Shephard (1996) surveyed other stochastic volatility models as well as GARCH models, and Pagan (1996) and Campbell, Lo, and MacKinlay (1997) applied GARCH models to financial questions.

In most studies, the classical estimation methods such as the maximum-likelihood estimation (MLE) have been employed to estimate the GARCH model. However, the classical methods are difficult to use in numerical optimization of the objective function, which is not necessarily concave (or convex), particularly when we use a gradient-based optimization method.[1] Furthermore, constraints imposed on GARCH coefficients complicate statistical inference on the coefficients as well as the optimization procedures. As we consider later, conditions for stationarity and other time-series properties of a GARCH(1,1) process are expressed as complicated inequalities involving GARCH coefficients, and it is difficult to test them using the classical approach.

On the other hand, by using a Bayesian approach to the GARCH model we may avoid these problems. In the Bayesian approach, we evaluate the whole posterior distribution instead of the maximum of the likelihood function. Thus, computational difficulty in finding the maximum no longer presents an obstacle to modeling estimation or statistical inference. Constraints on GARCH coefficients are also easily handled in the Bayesian approach by using the truncated posterior distributions of the GARCH coefficients. Moreover, in the Bayesian approach, estimation of the probability of an inequality is straightforward.

In the Bayesian approach, we need to compute integrals of the posterior distribution in terms of nuisance parameters. For many models, including ARCH/GARCH models, an analytical solution of the integral is not available; thus we need to evaluate the integral numerically. One popular numerical integration method is the Monte Carlo type of integration method. Monte Carlo integration methods such as importance sampling and the Markov-chain Monte Carlo (MCMC) method have been applied to ARCH/GARCH models by Geweke (1989a, 1989b), Kleibergen and Van Dijk (1993), Müller and Pole (1995), and others. Nakatsuma (1998)

[1] This problem may be solved using a global optimization method such as simulated annealing.

developed an algorithm to generate Monte Carlo samples of parameters in the GARCH model from their joint posterior distribution. Since it is impossible to generate samples directly from the joint posterior distribution, Nakatsuma (1998) applied a Markov-chain sampling technique with the Metropolis-Hastings algorithm. The Metropolis-Hastings algorithm was first proposed by Metropolis et al. (1953), and was extended by Hastings (1970). Since this particular Monte Carlo method involves a Markov-chain sampling, it is called a Markov-chain Monte Carlo method. A full description of the MCMC method used here is given by Nakatsuma (1998). Nakatsuma's MCMC method is based on the ideas of Chib and Greenberg (1994) and Müller and Pole (1995).

Organization of this paper is as follows. In Section 2, we briefly explain our Markov-chain sampling algorithm for the GARCH model. In Section 3, we estimate GARCH models of simulated data and weekly Swiss franc/U.S. dollar foreign-exchange-rate series to show how this MCMC algorithm works. In Section 4, we describe how to use the GAUSS program. In the Appendix, we explain definitions of variables and procedures in the GAUSS program.

2 Explanation of the Markov-Chain Sampling Algorithm

In this section, we briefly explain the Markov-chain sampling algorithm for the GARCH model. A full description of the algorithm is given by Nakatsuma (1998). We consider the following GARCH model:

$$\begin{cases} y_t = x_t\gamma + u_t, & (t = 1, \ldots, n) \\ u_t = \sum_{j=1}^{p} \phi_j u_{t-j} + \epsilon_t + \sum_{j=1}^{q} \theta_j \epsilon_{t-j}, & \epsilon_t | \mathcal{F}_t \sim \mathrm{N}(0, \sigma_t^2), \\ \sigma_t^2 = \omega + \sum_{j=1}^{r} \alpha_j \epsilon_{t-j}^2 + \sum_{j=1}^{s} \beta_j \sigma_{t-j}^2, & \end{cases} \tag{1}$$

where y_t is a scalar of the dependent variable, x_t is a $1 \times k$ vector of the independent variables, γ is a $k \times 1$ vector of the regression coefficients, and $\mathcal{F}_t$ is an increasing sequence of σ-fields generated by $\{y_{t-1}, y_{t-2}, \ldots\}$; ϕ_j is the coefficient of the autoregressive (AR) process, and θ_j is the coefficient of the moving average (MA) process; ω, α_j, and β_j are the coefficients of the GARCH process. Let $\gamma \equiv [\gamma_1, \ldots, \gamma_k]'$, $\phi \equiv [\phi_1, \ldots, \phi_p]'$, $\theta \equiv [\theta_1, \ldots, \theta_q]'$, $\alpha \equiv [\omega, \alpha_1, \ldots, \alpha_r]'$, $\beta \equiv [\beta_1, \ldots, \beta_s]'$, $Y \equiv [y_1, \ldots, y_n]'$, $X \equiv [x_1', \ldots, x_n']'$, and

$$\Phi(L) \equiv \sum_{j=1}^{p} \phi_j L^j, \quad \Theta(L) \equiv \sum_{j=1}^{q} \theta_j L^j, \quad A(L) \equiv \sum_{j=1}^{r} \alpha_j L^j, \quad B(L) \equiv \sum_{j=1}^{s} \phi_j L^j, \tag{2}$$

where L is the lag operator. We impose the following constraints on parameters in the GARCH model:

C1: all roots of $1 - \Phi(L) = 0$ are outside the unit circle;

C2: all roots of $1 + \Theta(L) = 0$ are outside the unit circle;

C3: $\omega > 0$ and $\alpha_j > 0$ for $j = 1, \ldots, r$; and

C4: $\beta_j > 0$ for $j = 1, \ldots, s$.

C1 and C2 are related to stationarity and invertibility of the ARMA process. C3 and C4 are constraints to guarantee that the conditional variance is always positive.

Our goal is to develop an algorithm to generate Monte Carlo samples, which are used in Bayesian analysis of the GARCH model, from the posterior distribution. The posterior density function of the model $p(\delta|Y, X)$ is given as:

$$p(\delta|Y, X) = \frac{\ell(Y|X, \delta)p(\delta)}{\int \ell(Y|X, \delta)p(\delta)d\delta}, \tag{3}$$

where δ is the set of all parameters in the model, $\ell(\cdot)$ is the likelihood function, and $p(\cdot)$ is the prior. The likelihood function of the GARCH model is:

$$\ell(Y|X, \delta) = \prod_{t=1}^{n} \frac{1}{\sqrt{2\pi\sigma_t^2}} \exp\left(-\frac{\hat{\epsilon}_t^2}{2\sigma_t^2}\right), \tag{4}$$

where

$$\begin{cases} \hat{\epsilon}_0 = \epsilon_0, \qquad \hat{\epsilon}_t = 0 \quad (t < 0), \\ \hat{\epsilon}_t = y_t - x_t\gamma - \sum_{j=1}^{p} \phi_j (y_{t-j} - x_{t-j}\gamma) - \sum_{j=1}^{q} \theta_j \hat{\epsilon}_{t-j}, \end{cases} \tag{5}$$

and we estimate the presample error term ϵ_0 as one of the parameters. As the prior, we consider the following conjugate prior:

$$\begin{aligned} p(\epsilon_0, \gamma, \phi, \theta, \alpha, \beta) \;=\; & \mathrm{N}(\mu_{\epsilon_0}, \Sigma_{\epsilon_0}) \times \mathrm{N}(\mu_\gamma, \Sigma_\gamma), \\ & \times \mathrm{N}(\mu_\phi, \Sigma_\phi)\mathrm{I}_{\mathrm{C}1}(\phi) \times \mathrm{N}(\mu_\theta, \Sigma_\theta)\mathrm{I}_{\mathrm{C}2}(\theta), \\ & \times \mathrm{N}(\mu_\alpha, \Sigma_\alpha)\mathrm{I}_{\mathrm{C}3}(\alpha) \times \mathrm{N}(\mu_\beta, \Sigma_\beta)\mathrm{I}_{\mathrm{C}4}(\beta), \end{aligned} \tag{6}$$

where $\mathrm{I}_{\mathrm{C}*}(\cdot)$ $(* = 1, 2, 3, 4)$ is the indicator function that takes unity if the constraint is true; otherwise it is zero; and $\mathrm{N}(\cdot)\mathrm{I}_{\mathrm{C}*}(\cdot)$ means that the normal distribution is truncated at the boundary of the support of $\mathrm{I}_{\mathrm{C}*}(\cdot)$.

To construct a Markov-chain sampling algorithm, we consider the following *auxiliary ARMA models*:

AM1: regression model with an ARMA(p, q) error:

$$y_t = x_t\gamma + \sum_{j=1}^{p} \phi_j (y_{t-j} - x_{t-j}\gamma) + \epsilon_t + \sum_{j=1}^{q} \theta_j \epsilon_{t-j}, \qquad \epsilon_t \sim \mathrm{N}(0, \sigma_t^2), \tag{7}$$

AM2: ARMA(l,s) model of the squared errors ϵ_t^2:

$$\epsilon_t^2 = \omega + \sum_{j=1}^{l} (\alpha_j + \beta_j)\epsilon_{t-j}^2 + w_t - \sum_{j=1}^{s} \beta_j w_{t-j}, \qquad w_t \sim \mathrm{N}(0, 2\sigma_t^4), \tag{8}$$

where $l \equiv \max\{r, s\}$, $\alpha_j = 0$ for $j > r$, and $\beta_j = 0$ for $j > s$. Let $\delta_1 \equiv [\epsilon_0, \gamma', \phi', \theta']'$, and $\delta_2 \equiv [\alpha', \beta']'$. In our auxiliary ARMA model approach, we generate a set of parameters, δ_1, from their full conditional distributions in the auxiliary ARMA model AM1, and another set of parameters, δ_2, from their full conditional distributions in the auxiliary ARMA model AM2, alternately.

Obviously, AM1 is derived from the original GARCH model by assuming that each σ_t^2 is a known constant. The derivation of AM2, however, may need some explanation. As shown by Bollerslev (1986), a GARCH(r, s) process is expressed as an ARMA(l, s) process of:

$$\epsilon_t^2 = \omega + \sum_{j=1}^{l} (\alpha_j + \beta_j)\epsilon_{t-j}^2 + \tilde{w}_t - \sum_{j=1}^{s} \beta_j \tilde{w}_{t-j}, \tag{9}$$

where $\tilde{w}_t \equiv \epsilon_t^2 - \sigma_t^2$. Since $\tilde{w}_t = (\epsilon_t^2/\sigma_t^2 - 1)\sigma_t^2 = (\chi^2(1) - 1)\sigma_t^2$, the conditional mean of $\tilde{w}_t$ is $\mathrm{E}(\tilde{w}_t|\mathcal{F}_{t-1}) = 0$, and the conditional variance is $\mathrm{Var}(\tilde{w}_t|\mathcal{F}_{t-1}) = 2\sigma_t^4$. Since it is difficult to generate (α, β) directly from the full conditional distributions in Equation 9, we replace $\tilde{w}_t$ in Equation 9 with $w_t \sim \mathrm{N}(0, 2\sigma_t^2)$. Then we have the auxiliary ARMA model AM2 in Equation 8.

The outline of our Markov-chain sampling algorithm is as follows:

1. Generate ϵ_0, γ, ϕ, and θ from AM1, given $\{\sigma_t^2\}$ and the rest of the parameters.
2. Generate α and β from AM2, given $\{\epsilon_t^2\}$, $\{\sigma_t^2\}$, and the rest of the parameters.
3. Apply the Metropolis-Hastings algorithm after each parameter is generated.
4. Repeat steps 1–3 until the sequences become stable. In this sampling scheme, we update $\{\epsilon_t^2\}$ and $\{\sigma_t^2\}$ every time after corresponding parameters are updated.

The full conditional distribution of a parameter, say δ_j, in the corresponding auxiliary ARMA model is the normal distribution conditional on Y and X and the rest of the parameters, and it is given as the following form:

$$\delta_j | Y, X, \Sigma, \delta_{-j} \sim \mathrm{N}(\hat{\mu}_j, \hat{\Sigma}_j), \tag{10}$$

where

$$\hat{\mu}_j \equiv \hat{\Sigma}_j(X_j'\Sigma^{-1}Y_j + \Sigma_j^{-1}\mu_j), \qquad \hat{\Sigma}_j \equiv [X_j'\Sigma^{-1}X_j + \Sigma_j^{-1}]^{-1}. \tag{11}$$

Parameters μ_j and Σ_j are the hyper-parameters of Equation 6 corresponding to δ_j, and Σ is diag$\{\sigma_1^2, \ldots, \sigma_n^2\}$ for AM1 and diag$\{2\sigma_1^4, \ldots, 2\sigma_n^4\}$ for AM2. To construct Y_j and X_j, we use the following filters:

Filter 1:

$$z_t = x_t - \sum_{j=1}^{k} \kappa_j z_{t-j}, \tag{12}$$

Filter 2:

$$z_t = x_t - \sum_{j=1}^{b} \pi_j x_{t-j} - \sum_{j=1}^{k} \kappa_j z_{t-j}, \tag{13}$$

where κ_j and π_j are parameters in the GARCH model. These filters were originally proposed by Chib and Greenberg (1994). For example, for the regression coefficients γ, π_j is ϕ_j, and κ_j is θ_j. Then the t^{th} elements of Y_j ($y_{\gamma,t}$) and X_j ($x_{\gamma,t}$) are given by applying Filter 2:

$$y_{\gamma,t} = y_t - \sum_{j=1}^{p} \phi_j y_{t-j} - \sum_{j=1}^{q} \theta_j y_{\gamma,t-j}, \qquad x_{\gamma,t} = x_t - \sum_{j=1}^{p} \phi_j x_{t-j} - \sum_{j=1}^{q} \theta_j x_{\gamma,t-j}, \tag{14}$$

where $y_{\gamma,0} = \epsilon_0$, $y_t = y_{\gamma,t} = 0$ for all $t < 0$, and $x_t = x_{\gamma,t} = 0$ for all $t \leq 0$. From Equation 5, we can easily show that $\hat{\epsilon}_t = y_{\gamma,t} - x_{\gamma,t}\gamma$. For the other parameters, see the work by Nakatsuma (1998).

After each parameter is generated, we apply the Metropolis-Hastings algorithm. The Metropolis-Hastings algorithm[2] is given as:

$$\delta_j^{(i+1)} = \begin{cases} \hat{\delta}_j, & \text{with probability } \lambda, \\ \delta_j^{(i)}, & \text{with probability } 1-\lambda, \end{cases} \tag{15}$$

where $\hat{\delta}_j$ is a candidate of δ_j generated from Equation 10, $\delta_j^{(i)}$ is the i^{th} realization of δ_j in the Markov-chain sampling, and λ is the acceptance probability:

$$\lambda = \min\left\{ \frac{p(\hat{\delta}_j | Y, X, \delta_{-j}^{(i)})}{p(\delta_j^{(i)} | Y, X, \delta_{-j}^{(i)})} \frac{g(\delta_j^{(i)} | Y, X, \delta_{-j}^{(i)})}{g(\hat{\delta}_j | Y, X, \delta_{-j}^{(i)})}, \ 1 \right\}, \tag{16}$$

where $p(\cdot)$ is the posterior density and $g(\cdot)$ is the full conditional density of δ_j in the corresponding auxiliary ARMA model shown in Equation 10. In the literature, $g(\cdot)$ is called the *proposal density*. In the Metropolis-Hastings algorithm, constraints C1–C4 are easily handled. If a candidate does not satisfy a constraint, the acceptance probability is forced to be zero, and the next candidate is generated from the proposal distribution of Equation 10. If the candidate does satisfy the constraint, we advance to the Metropolis-Hastings algorithm. For example, we can generate samples satisfying constraints C3 or C4 by discarding any nonpositive candidates. For C1 and C2, we solve the polynomial of the lag operator, $1 - \Phi(L) = 0$ or $1 + \Theta(L) = 0$, and see if the modulos of the solutions are greater than unity. Of course, for a specific AR or MA process, we may use a closed-form solution. (For AR(1), $|\phi_1| < 1$ is the stationarity condition.)

Finally, we note how to compute the conditional variance $\{\sigma_t^2\}$ in our algorithm. The conditional variance can be computed by the third statement in Equation 1. In our algorithm, on the other hand, we use the fact $\sigma_t^2 = \epsilon_t^2 - w_t$, where w_t is the error term in AM2 (Equation 8). Then $\{w_t\}$ is computed by the following filter:

[2]For more information about the Metropolis-Hastings algorithm, see works by Tierney (1994), Chib and Greenberg (1995), and Roberts and Smith (1996) among others.

Filter 3:

$$w_t = \epsilon_t^2 - \omega - \alpha_t u_0 - \sum_{j=1}^{l}(\alpha_j + \beta_j)\epsilon_{t-j}^2 + \sum_{j=1}^{s}\beta_j w_{t-j}, \tag{17}$$

where $\alpha_t = 0$ for $t > r$, $w_t = 0$ for $t < 0$, and we assume $u_0 = \epsilon_0^2 - \omega$. Once we obtain $\{w_t\}$ by Equation 17, we have the conditional variance $\{\sigma_t^2\}$ by $\sigma_t^2 = \epsilon_t^2 - w_t$.

3 Numerical Examples

3.1 Simulated data

We estimate a GARCH model by our algorithm using simulated data. We consider the following GARCH model:

$$\begin{cases} y_t &= 1 + x_t + u_t, \qquad (t = 1, \ldots, 1{,}000) \\ u_t &= 0.8u_{t-1} + \epsilon_t - 0.5\epsilon_{t-1}, \qquad \epsilon_t|\mathcal{F}_t \sim \mathrm{N}(0, \sigma_t^2), \\ \sigma_t^2 &= 0.001 + 0.15\epsilon_{t-1}^2 + 0.85\sigma_{t-1}^2, \end{cases} \tag{18}$$

where x_t is generated from the uniform distribution between -0.5 and 0.5. Values of the parameters are $\gamma_1 = \gamma_2 = 1.0$, $\phi_1 = 0.8$, $\theta_1 = -0.5$, $\omega = 0.001$, $\alpha_1 = 0.15$, and $\beta_1 = 0.85$. We set the prior as:

$$\begin{aligned}[\epsilon_0, \gamma_1, \gamma_2, \phi_1, \theta_1, \omega, \alpha_1, \beta_1] \sim\ & \mathrm{N}(0,\ 10 \times I_8) \times \mathrm{I}(|\phi_1| < 1) \times \mathrm{I}(|\theta_1| < 1) \\ & \times \mathrm{I}(\omega > 0) \times \mathrm{I}(\alpha_1 > 0) \times \mathrm{I}(\beta_1 > 0), \end{aligned} \tag{19}$$

where I_8 is the 8×8 identity matrix.

The model of ϵ_t in Equation 18 is called an integrated GARCH (IGARCH) process by Engle and Bollerslev (1986), since $\alpha_1 + \beta_1 = 1$. The unconditional variance of the GARCH(1,1) process is given by $\mathrm{Var}(\epsilon_t) = \omega/(1 - \alpha_1 - \beta_1)$, but for the IGARCH(1,1) process, the finite unconditional variance does not exist. We may test whether a GARCH(1,1) process has the finite unconditional variance or not by estimating the posterior probability of $\Pr\{\alpha_1 + \beta_1 < 1\}$. We estimate this probability by a Monte Carlo method:

$$\Pr\{\alpha_1 + \beta_1 < 1\} \approx \frac{1}{m}\sum_{i=1}^{m}\mathrm{I}(\alpha_1^{(i)} + \beta_1^{(i)} < 1), \tag{20}$$

where $\alpha_1^{(i)}$ and $\beta_1^{(i)}$ are generated by the Markov-chain sampling, and $\mathrm{I}(\cdot)$ is the indicator function.

In the Markov-chain sampling, we first run the sampling algorithm 5,000 times as the "burn-in," and then repeat it 50,000 times. To avoid a strong serial correlation in the Markov-chain sampling, we only store samples every five runs; therefore, the sample size is 10,000. It takes 19978.54 s to run this program on a Dell Dimension XPS Pro200n with 200-MHz Pentium Pro and 64 MB RAM. The computing time may be reduced if we write the code in C/C++ or FORTRAN instead of GAUSS. The long computing time may be justifiable, because the MCMC method can solve problems that are often encountered using the classical approach. We also estimate the parameters in the GARCH model by the maximum-likelihood estimation. We use the simulated annealing by Goffe (1996) for the MLE. Table 1 lists the posterior statistics of the GARCH model.

In Table 1, the posterior means by the MCMC and the MLE of each parameter are similar, except for ϵ_0. The posterior mean of ϵ_0 is -0.317 for the MCMC, but is 1.81 for the importance sampling. The reason why the posterior mean and MLE of ϵ_0 are so different is explained by examining the shape of the marginal posterior pdf of ϵ_0 shown in Figure 1. The pdf in Figure 1 is estimated by the kernel smoothing with the Gaussian kernel. Obviously, the marginal posterior pdf of ϵ_0 is bimodal, and the posterior mean of this bimodal distribution seems to be around zero, which is attained by the MCMC. When we look at the graph carefully, we find that the right mode of the pdf is roughly equal to the MLE of ϵ_0. When the posterior distribution has more than one local maximum as in Figure 1, evaluating the posterior distribution only around the global maximum may give us inaccurate information about the distribution. This exercise demonstrates the advantage of the MCMC method in which we obtain Monte Carlo samples generated from the posterior distribution.

The posterior probability $\Pr\{\alpha_1 + \beta_1 < 1\}$ is shown in the last row of Table 1. This probability is about 30%. This is the result we expect, because the data is generated from the IGARCH model, which has no finite unconditional variance.

Table 1
Posterior statistics of the GARCH model (simulated data)

	MLE	MCMC	Acceptance[c]	SAC(1)[d]
γ_1	0.982	0.982	0.801	0.00765
	(0.0204)[a]	(0.0192)[b]		
γ_2	1.04	1.04	0.801	0.00769
	(0.0321)	(0.0279)		
ϕ_1	0.768	0.759	0.855	0.614
	(0.0393)	(0.0442)		
θ_1	−0.450	−0.437	0.351	0.717
	(0.0543)	(0.0629)		
ω	0.000723	0.000821	0.869	0.108
	(0.000421)	(0.000379)		
α_1	0.154	0.162	0.869	0.643
	(0.0276)	(0.0239)		
β_1	0.850	0.844	0.918	0.687
	(0.0220)	(0.0190)		
ϵ_0	1.81	−0.317	0.366	0.134
	(0.709)	(2.45)		
	$\Pr\{\alpha_1 + \beta_1 < 1\}$			0.296

[a] Standard error of the MLE.
[b] Posterior standard deviation.
[c] Acceptance rate in 55,000 runs.
[d] Sample autocorrelation at lag 1.

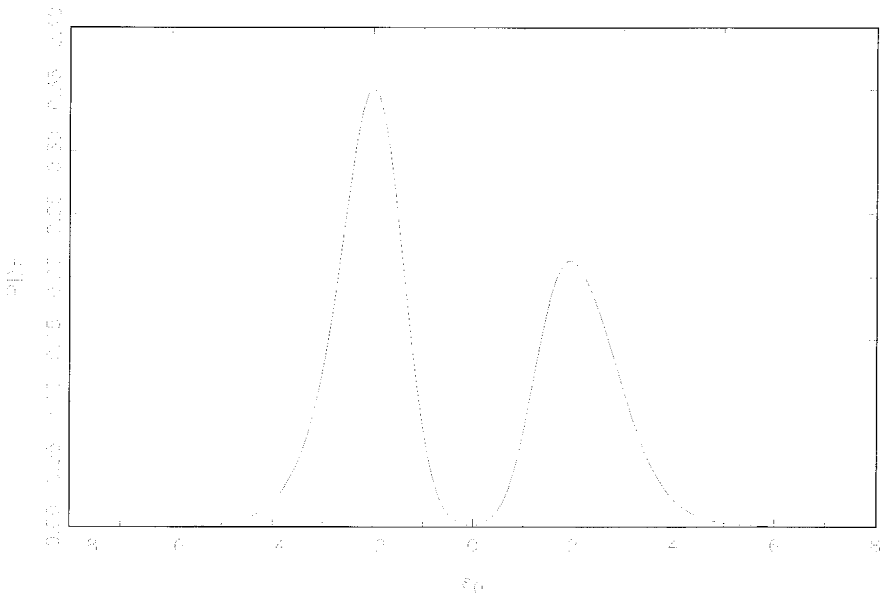

Figure 1
Marginal posterior pdf of ε_0

3.2 Weekly Swiss franc/U.S. dollar foreign-exchange-rate series

As an empirical example, we estimate a GARCH model of a weekly Swiss franc/U.S. dollar foreign-exchange-rate series. The model we estimate is as follows:

$$\begin{cases} y_t = \gamma_1 + u_t, \\ u_t = \phi_1 u_{t-1} + \epsilon_t, \qquad \epsilon_t | \mathcal{F}_t \sim \mathrm{N}(0, \sigma_t^2), \\ \sigma_t^2 = \omega + \alpha_1 \epsilon_{t-1}^2 + \beta_1 \sigma_{t-1}^2, \end{cases} \tag{21}$$

Table 2
Posterior statistics of the GARCH model (weekly Swiss franc/U.S. dollar exchange rate)

	Posterior Statistics	Acceptance Rate[f]	SAC(1)[g]
γ_1	−1.78	0.876	0.0213
	(0.192)[e]		
ϕ_1	1.00	0.981	0.00878
	(0.00142)		
ω	0.000119	0.281	0.532
	(6.64×10^5)		
α_1	0.134	0.281	0.644
	(0.0257)		
β_1	0.859	0.933	0.789
	(0.0269)		
ϵ_0	0.00256	0.910	0.281
	(0.205)		
	$\Pr\{\beta_1 + 2\alpha_1\beta_1 + 3\alpha_1 < 1\}$		0.000
	$\Pr\{\alpha_1 + \beta_1 < 1\}$		0.730
	$\Pr\{\mathrm{E}\sqrt{\beta_1 + \alpha_1 z^2} < 1\}$		0.918
	$\Pr\{\mathrm{E}\ln(\beta_1 + \alpha_1 z^2) < 0\}$		0.981

[e] Posterior standard deviation.
[f] Acceptance rate in 55,000 runs.
[g] Sample autocorrelation at lag 1.

where y_t is the Swiss franc/U.S. dollar exchange rate in a natural logarithm. We use Wednesday's closing values of the exchange rate if available; otherwise, we use the next trading day's closing values. The sample period is June 1973 to February 1987.

Since we found that the MA part of the error term was not supported in preliminary estimation, we omit it from the model. We also relax the stationarity constraint on the AR coefficient, since the exchange-rate series seems to have a unit root. Implementation of the MCMC algorithm is the same as in the previous example. It takes 11,486.59 s to run the GAUSS program on the same machine we used in the previous example. The posterior statistics of the parameters are shown in Table 2.

To demonstrate how useful the Bayesian approach is in analyzing GARCH models, we test the stationarity and other time-series properties of the estimated GARCH model. Bollerslev (1986) showed that a GARCH(1,1) process has the finite unconditional fourth moment if

$$\beta_1 + 2\alpha_1\beta_1 + 3\alpha_1 < 1, \tag{22}$$

and Nelson (1990) showed that the unconditional standard deviation of a GARCH(1,1) process is finite if

$$\mathrm{E}\sqrt{\beta_1 + \alpha_1 z_t^2} < 1, \tag{23}$$

where $z_t \equiv \epsilon_t/\sigma_t$. In our example, $z_t \sim \mathrm{N}(0, 1)$. Nelson (1990) also showed that the conditional variance of a GARCH(1,1) process is strictly stationary and ergodic if

$$\mathrm{E}\ln(\beta_1 + \alpha_1 z_t^2) < 0. \tag{24}$$

We estimate the posterior probability of Equations 22, 23, and 24, as well as $\Pr\{\alpha_1 + \beta_1 < 1\}$. The estimation procedure is the same as in the previous example, except that we estimate the expectation in Equations 23 and 24 by:

$$\mathrm{E}f(\alpha_1, \beta_1) \approx \frac{1}{m}\sum_{i=1}^{m} f(\alpha_1, \beta_1, z^{(i)}), \tag{25}$$

where $z^{(i)}$ $(i = 1, \ldots, m)$ is a random number generated from $\mathrm{N}(0, 1)$, and $f(\cdot)$ is $\sqrt{\beta_1 + \alpha_1 z^{(i)^2}}$ in Equation 23 or $\ln(\beta_1 + \alpha_1 z^{(i)^2})$ in Equation 24. We set $m = 10{,}000$. Equation 25 is computed for each pair of (α_1, β_1)

generated by our MCMC algorithm, and each posterior probability is estimated in the same fashion of Equation 20. Estimated posterior probabilities are listed in Table 2. The results suggest that the GARCH model of a Swiss franc/U.S. dollar exchange-rate series is unlikely to have the finite unconditional variance and fourth moment. The chance that the GARCH model will not have the finite unconditional standard deviation is about 8%, and it is likely that the conditional variance of the GARCH model is strictly stationary and ergodic.

4 How to Use the GAUSS Program

1. To run the GAUSS program, the constrained maximum-likelihood estimation library `cml` is required to solve the least-squares problem on the MA and GARCH coefficients. You may use a custom constrained-optimization procedure instead of `cml`.
2. Along with this code, the simulated time-series data used in the numerical example and the MLE of the parameters are included in GAUSS data files. The time-series data file is named `gar_dat1`, and the MLE file is `mle_cf1`. These data files are assumed under the directory `c : \research\mcmc\` in the code. You may change this as you wish.
3. The MLE is used as the starting value of the Markov-chain sampling. You may choose your own starting values.
4. When you run this program, some of the parameters may be stuck at certain values. This is checked by viewing a plot of Monte Carlo samples or the acceptance rate. A long flat line in the plot or a near-zero acceptance rate indicates the problem. If this happens, start the algorithm over with different starting values.
5. Generated Monte Carlo samples of the parameters are stored in a GAUSS data file named `mcmc1` under the directory `c : \research\mcmc\`. The posterior statistics are stored in `pstats1` under the same directory.

5 Concluding Remarks

This paper explains a GAUSS program of a Markov-chain sampling algorithm for GARCH models by Nakatsuma (1998). This algorithm enables us to generate parameters in a GARCH model from their joint posterior distribution. This paper also demonstrates the advantage of the Bayesian approach in testing stationarity and other time-series properties of the GARCH(1, 1) process, which is difficult when using the classical approach.

Appendix: Definitions of Variables and Functions/Procedures

Definitions of variables

n Sample size

k Dimension of the regression coefficient vector (γ)

p Dimension of the AR coefficient vector (ϕ)

q Dimension of the MA coefficient vector (θ)

r Dimension of the ARCH coefficient vector (α)

s Dimension of the GARCH coefficient vector (β)

m $\max\{p, q\}$

l $\max\{r, s\}$

mm Number of iterations in the "burn-in" phase

nn Number of iterations in the Markov-chain sampling

intvl Interval to store samples (= 5 by default)

y Regressand ($n \times 1$ vector)

x Regressors ($n \times k$ matrix)

u Residuals ($n \times 1$ vector) = $\{\epsilon_t\}$

v Squared residuals $\{\epsilon_t^2\}$ ($n \times 1$ vector)

h Conditional variance $\{\sigma_t^2\}$ ($n \times 1$ vector)

tau Vector of $\{2\sigma_t^4\}$

gam Vector of regression coefficients (γ)

phi Vector of AR coefficients (ϕ)

theta Vector of MA coefficients (θ)

omega Constant term in GARCH (ω)

alpha Vector of ARCH coefficients (α)

beta Vector of GARCH coefficients (β)

eps0 Presample error term = ϵ_0

eps02 Squared presample error term = ϵ_0^2

e0 $q \times 1$ vector (e0[q]=eps0, e0[i]=0 i<q)

e02 $s \times 1$ vector (e02[s]=eps02, e02[i]=0 i<s)

theta1 Least-squares estimate of theta

beta1 Least-squares estimate of beta

ab Alpha + beta

m*0 Mean of the prior = μ_j

iv*0 Inverse of the covariance matrix of the prior = Σ_j

y_* Regressand in the proposal distribution = Y_j

x_* Regressors in the proposal distribution = X_j

***temp** Candidate to the next realization in the Metropolis-Hastings algorithm

(* = gam, phi, the, alp, bet, eps)

p_f Log posterior density

p_f1 Previous log posterior density

p_p Ratio of the proposal density in logarithm min(exp($p\_f - p\_f1 - p\_p$), 1) is the acceptance probability λ in the Metropolis-Hastings algorithm

bd_the Bounds of theta used by `cml`

bd_bet Bounds of beta used by `cml`

gd_the Gradient of residuals for least-squares estimation of the MA coefficients

gd_bet Gradient of residuals for least-squares estimation of the GARCH coefficients; these gradients are used for linearization of the residuals (Nakatsuma 1998)

mcmc Matrix of generated samples (the sample is stored in rows)

accept Acceptance rate

pos_m Posterior mean

pos_sd Posterior standard deviation

acorr Autocorrelation at lag 1

pr_ucv Probability of existence of the unconditional variance $= \Pr\{\alpha_1 + \beta_1 < 1\}$

Definitions of functions and procedures

exprep( ) Alternative to exp( ) function to avoid overflows and under-flows

qdiff( ) Compute the quasi-difference;

this function computes $x_t - \sum_{j=1}^{k} \kappa_j x_{t-j}$ in Filter 2

lagm( ) Make a matrix of lagged variables:

the matrix of lagged variables made by this function is used in **qdiff( )** to compute the summation $\sum_{j=1}^{k} \kappa_j x_{t-j}$

filter1( ) Apply Filter 1 to a vector

filter1m( ) Apply Filter 1 to a matrix

filter2( ) Apply Filter 2 to a vector

filter2m( ) Apply Filter 2 to a matrix

filter3( ) Apply Filter 3 to a vector

cv_garch( ) Compute the conditional variance of the GARCH model

sum_coef( ) Return the sum of two vectors with different row size; this is used to compute $\alpha_j + \beta_j$ $(j = 1, \ldots, l)$ in AM2

grad( ) Compute the gradient

res_ma( ) Compute residuals for least-squares estimation of the MA coefficients

res_gar( ) Compute residuals for least-squares estimation of the GARCH coefficients

post_gar( ) Log posterior density of the GARCH model

draw_pro( ) Draw random numbers from the proposal distribution

pstats( ) Compute posterior statistics

References

Bollerslev, T. (1986). "Generalized autoregressive conditional heteroskedasticity." *Journal of Econometrics*, 31: 307–327.

Bollerslev, T., R. F. Engle, and D. B. Nelson (1994). "ARCH models." In R. F. Engle and D. L. McFadden (eds.), *Handbook of Econometrics*, vol. IV. Amsterdam: North-Holland, pp. 2959–3038.

Campbell, J. Y., A. W. Lo, and A. C. MacKinlay (1997). *The Econometrics of Financial Markets*. Princeton: Princeton University Press.

Chib, S., and E. Greenberg (1994). "Bayes inference in regression models with ARMA(p, q) errors." *Journal of Econometrics*, 64: 183–206.

Chib, S., and E. Greenberg (1995). "Understanding the Metropolis-Hastings algorithm."*American Statistician*, 49: 327–335.

Engle, R. F. (1982). "Autoregressive conditional heteroskedasticity with estimates of the variance of United Kingdom inflation." *Econometrica*, 50: 987–1008.

Engle, R. F., and T. Bollerslev (1986). "Modeling the persistence of conditional variances." *Econometric Reviews*, 5: 1–50.

Geweke, J. (1989a). "Bayesian inference in econometric models using Monte Carlo integration." *Econometrica*, 57: 1317–1339.

Geweke, J. (1989b). "Exact predictive densities for linear models with ARCH disturbances." *Journal of Econometrics*, 40: 63–86.

Goffe, W. L. (1996). "SIMANN: A global optimization algorithm using simulated annealing." *Studies in Nonlinear Dynamics and Econometrics*, 1: 169–176.

Hastings, W. K. (1970). "Monte Carlo sampling methods using Markov chains and their applications." *Biometrika*, 57: 97–109.

Kleibergen, F., and H. K. Van Dijk (1993). "Non-stationarity in GARCH models: A Bayesian analysis." *Journal of Applied Econometrics*, 8: S41–S61.

Metropolis, N., A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller (1953). "Equations of state calculations by fast computing machines." *Journal of Chemical Physics*, 21: 1087–1092.

Müller, P., and A. Pole (1995). "Monte Carlo posterior integration in GARCH models." Manuscript, Duke University.

Nakatsuma, T. (1998). "Bayesian analysis of ARMA-GARCH models: A Markov chain sampling approach." Mimeo, Rutgers University.

Nelson, D. B. (1990). "Stationarity and persistence in the GARCH(1,1) model." *Econometric Theory*, 6: 318–334.

Pagan, A. (1996). "The econometrics of financial markets." *Journal of Empirical Finance*, 3: 15–102.

Roberts, G. O., and A. F. M. Smith (1996). "Simple conditions for the convergence of the Gibbs sampler and Metropolis-Hastings algorithms." *Stochastic Processes and Their Applications*, 49: 207–216.

Shephard, N. (1996). "Statistical aspects of ARCH and stochastic volatility." In D. R. Cox, D. V. Hinkley, and O. E. Barndorff-Nielsen (eds.), *Time Series Models: In Econometrics, Finance and Other Fields*. London: Chapman & Hall, pp. 1–67.

Tierney, L. (1994). "Markov chains for exploring posterior distributions." *Annals of Statistics*, 22: 1701–1762.

Editorial Policy

The *SNDE* is formed in recognition that advances in statistics and dynamical systems theory may increase our understanding of economic and financial markets. The journal will seek both theoretical and applied papers that characterize and motivate nonlinear phenomena. Researchers will be encouraged to assist replication of empirical results by providing copies of data and programs online. Algorithms and rapid communications will also be published.

ISSN 1081-1826