Journal of Risk and Control, Vol. 10, No. 1, 2023, 1-14
ISSN: 2056-3701 (print version), 2056-371X (online)
https://doi.org/10.47260/jrc/1011
Scientific Press International Limited

The Usage of Markov Chain Monte Carlo (MCMC) Methods in Time-varying Volatility Models

Garefalakis Emmanouil[1], Giakoumatos Stefanos[2] and Rezitis Antonios[3]

Abstract

Markov Chain Monte Carlo (MCMC) techniques, in the context of Bayesian inference, constitute a practical and effective tool to produce samples from an arbitrary distribution. These algorithms are applied to calculate parameter values of predictive models of the phenomenon of varying volatility in data time series. For this purpose, 3 such research models of time-varying volatility are simulated in STAN a probabilistic programming language for statistical inference. The accuracy of these models' predictive function is confirmed by applying in data time series with known prior values. Moreover, Stan models' performance is illustrated by the real stock prices of two shares in the stock market of New York. Finally, an Information Criterion of the results is applied to each model as well, to evaluate their predictive ability, comparing and selecting the most effective one.

Keywords: MCMC, Time-series, Time-varying volatility models, STAN.

[1] Dept of Accounting and Finance, University of Peloponnese.
[2] Dept of Accounting and Finance, University of Peloponnese.
[3] Dept of Agricultural Economics and Development, Agricultural University of Athens.

Article Info: *Received:* February 20, 2023. *Revised*: March 15, 2023.
Published online: April 11, 2023.

1. Introduction

Time-varying volatility models have become one of the important areas of study in economics in recent years due to their ability to describe the clustering effect of the volatility exhibited by economic time-series (Hanzon, 2003). The description of volatility clustering is particularly important for financial institutions to manage portfolios or investments with as little risk as possible. These models can be univariate or multivariate (Wei, 2006).

For the prediction of the parameters of the above models, an effective tool is the Markov Chain Monte Carlo (MCMC) method. These algorithms, although proposed many decades ago, were not widely applied due to the computational complexity required in their calculations. In recent years, however, with the help of computers and statistical programming tools, these algorithms have been revived and are now being used to quickly and successfully predict the parameters of the study models of volatility.

In this study, an attempt is made to present and apply the MCMC algorithms for parameter prediction in the GARCH, Stochastic Volatility, and Unobserved ARCH volatility models. For their application the statistical package STAN is used, a programming tool where the parameters of the models are predicted by the Hamiltonian Monte Carlo method.

2. MCMC methods

Algorithms in this class, are derived from Monte Carlo methods but are sampled not from a random sample but from a Markovian chain. The sampling of the probability distribution in them is based on the construction of such a chain that has the same distribution as that of their equilibrium distribution. (Zhang, 2013).

MCMC methods generate a chain of values θ_1, θ_2, …. whose distribution approximates the a priori distribution. If we were to represent these values on a histogram, we would say that this is done by generating candidate values - points which if they approach the target distribution are accepted otherwise, they are rejected. A correct distribution is created by selecting those points which are close enough to the prior probability. The generation of the chain parameters θ^n , n=1, 2,… are generated by random number generators. Each new point θ_{n+1} must, according to the Markovian property, depend only on the previous point θ_n. (Atzberger, 2013).

The first algorithm of this class of algorithms is introduced in 1953 for Statistical Physics by Metropolis, Rosenbluth, A. Teller, and E. Teller (Metropolis et al, 1953). This algorithm is not considered efficient in the case where the target distribution we want to simulate does not resemble any known distribution or in the case where it is not easy to use. For this reason, Hastings (1970) proposed an improvement of the above known as the Metropolis-Hastings. The steps of this method steps are presented below.

Algorithm 1: Metropolis-Hastings

(i). Initialize by selecting a starting point θ_0

(ii). Select a new candidate point θ^{new} from a suitable proposed distribution $q(\theta^{new}|\theta^{old})$ which is based on the previous point in the chain and is not necessarily symmetric.

(iii). Calculate the acceptance rate r

$$r = \frac{\pi(\theta^{new}) * q(\theta^{old}|\theta^{new})}{\pi(\theta^{old}) * q(\theta^{new}|\theta^{old})}$$

(iv). Accept the point θ^{new} as θ with probability $\alpha(\theta^{old}, \theta^{new}) =$ min (1, r). If θ^{new} is not accepted then return to the previous point in the chain

Generally, his step can be done here by generating a random number u ε [0,1]. The new value of the chain θ^{old+1} is chosen from the relation:

$$\theta^{old+1} = \begin{cases} \theta^{new}, \alpha v\ u \leq \alpha(\theta^{old}, \theta^{new}) \\ \theta^{old}, \alpha v\ u > \alpha(\theta^{old}, \theta^{new}) \end{cases}$$

(v). Repeat step (ii) for a predetermined number of steps.

One of the best-known algorithms in this category is the Gibbs sampler, originally formulated by Besag (1974) and Geman and Geman (1984). For the operation of the algorithm the table of parameters is divided into elements. So we could say that: $\theta = (\theta_1, \theta_2, \ldots..\theta_d)$ where none of these elements θ_j need to be linear. We assume that the full conditional distributions of each of the points are known and we know how to simulate them. The basic idea behind Gibbs' method is to sequentially simulate each element θ_j from the conditional distributions (Chen & Edward, 2015) as shown below:

Algorithm 2: Gibbs sampler

$\theta^{cur} \leftarrow \theta^{(0)}$

For ι = 0, 1, …. N

For j = 0, 1, …. d repeat

Select new value j of element θ_j^{cur} from the conditional probability $f_{\Theta j|\Theta -j,\, Y}(\vartheta_j \mid \theta_{-j}^{cur}, y)$.

End

$\theta^{(\iota+1)} \leftarrow \theta^{(cur)}$

End

Since many times the above algorithms are inefficient and quite slow in the convergence of the target distribution *p(x)*, in recent years new methods have been proposed that aim at greater efficiency. One such algorithm is the Hybrid MC aka Hamiltonian MC since it adopts principles of natural systems dynamics instead of a probability distribution to suggest the next positions of a Markovian chain. (Duane, Kennedy, Pendleton , & Roweth, 1987). Hamiltonian equations describe the motion of an object in a time interval, but it is a continuous variable. If we want to represent this motion numerically, we have to digitize it. For this reason, many methods have been developed, with the best known of all being the Leap Frog method. In this algorithm, we start with an initial state $[x_0, m_0]$ and simulate the Hamiltonian dynamics for a short period using the Leap Frog method. The simulation produces two new variables for position and momentum, x* and m* respectively, which define a new position. The new position is accepted or rejected using the Metropolis acceptance criterion. Summing up the above the HMC method is presented as:

Algorithm 3: HMC

(i). Set t=0

(ii). Randomly set a starting position $x^0 \sim \pi^{(0)}$

(iii). Repeat until t = M

Set t=t+1

Choose a new initial momentum variable from the normal momentum distribution $m_0 \sim p(m)$

Run the Leap Frog method with initial values $[x_0, m_0]$ for L steps with step size δ to calculate the new proposed values x* και m*

Calculate the Metropolis probability of acceptance as:

$$\alpha = \min (1,\ e^{[\,U(x^*)+U(x_0)\,-K(m^*)+K(m_0)]}$$

Choose a random number u from the interval [0,1]

Accept the point with probability:

$$x^{(t)} = \begin{cases} x^*, \alpha\nu\ u \ \leq \ \alpha \\ x^{(t-1)}, \alpha\nu\ u > \ \alpha \end{cases}$$

End

3. Volatility Time Series Models

The methods of the previous section have proven to be a valuable tool for concluding sampling in many scientific fields. One of these scientific fields, where these algorithms are applied with great success, is that of econometrics, which are successfully used for modeling and forecasting in economic time series. In more detail, the phenomenon of heteroskedasticity occurs many times in time series, i.e. their values have different variation. In other words, the values may take quite large or small values at certain times. However, it has been observed that when indicators fluctuate, they remain at this level of change for some time before changing again, thus creating clusters of the same distribution (volatility clustering).

3.1 Models

The first model to study the instability phenomenon in detail is the autoregressive model with conditional heteroscedasticity (ARCH) formulated by Engle (1982). According to this, a model of order p known as ARCH(p) is given by the following formula:

$$y_t|\boldsymbol{a}, \boldsymbol{y}_{t-1} \sim N(0, \sigma_t^2)$$

$$\sigma_t^2 = a_0 + \sum_{i=1}^{p} a_i y_{t-i}^2 \quad (3.1.1)$$

where y_t is the time series at time t

The problem with the ARCH model is that for drawing safe inferences a large order of p is needed. This problem was solved by Bollerslev who proposed the Generalized Autoregressive model with Conditional Heteroskedastic GARCH (Bollerslev, 1986). Unlike the ARCH model in this model, the conditional distributions of the returns y_t and σ_t^2 depend not only on the prior y_t but also on the prior σ_t^2. A GARCH(p,q) model is given by the following formula:

$$y_t|\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{y}_{t-1} \sim N(0, \sigma_t^2)$$

$$\sigma_t^2 = a_0 + \sum_{i=1}^{p} a_i y_{t-i}^2 + \sum_{j=1}^{q} b_j \sigma_{t-j}^2 \quad (3.1.2)$$

An alternative class of models to the ARCH - GARCH models is that of the Stochastic Volatility (SV) models proposed by Taylor (1994). In this, the volatility is a random variable, unlike the previous models where the conditional variance is a deterministic function of the model parameters and past values. The SV model can be written as:

$$y_t \mid h_t \sim N(0, h_t)$$

$$h_t \mid a, d, \sigma_\eta^2 \sim LN(a + dlog(h_{t-1}), \sigma_h^2), \quad (3.1.3)$$

Finally, The Unobserved Arch model was introduced by Shephard (1996) . In it the parameters are observed with errors and can be written as:

$$y_t \mid x_t, \sigma^2 \sim N(x_t, \sigma^2)$$

$$x_t \mid x_{t-1}, a, \beta, x_0 \sim N(0, h_t)$$

$$h_t = a + \beta x_{t-1}^2 \quad (3.1.4)$$

3.2 Validating models in STAN

In order to better understand the operation of the above methods and to validate their correct operation, in this chapter, an attempt is made to simulate them in the probabilistic programming language STAN (Carpenter, Gelman, Hoffman, & Lee, 2017). STAN is one of the most recent statistical modeling tools that allows its user to calculate the probability p(x, θ) of a model in a programmatic way. For its sampling and sample selection, STAN uses the HMC algorithm as its MCMC method.

GARCH(1,1) example:

In order to study the operation of a GARCH(1,1) model we set in relation (3.1.2) p=1 and q=1 so that our model can be written as

$$\sigma_t^2 = a_o + a_1 y_{t-1}^2 + b_1 \sigma_{t-1}^2$$

We then create a random time series of 1500 time points in the programming language R. For this time series we randomly choose α_0=0.02, α_1=0.05, and β_1=0.9. This time series is given as input in our STAN model and a fit object is produced in the output providing the samples selected by the HMC method from the posterior distribution of the parameters declared in our model. The table below compares the initial values of the parameters we had set, with the average of the posterior values chosen in each iteration of the HMC algorithm. The posterior values of the parameters are quite close to our initial values.

Table 1: Comparison of initial and posterior GARCH values

Parameter	Initial Value	Average of posterior values
α_o	0,02	0,02
α_1	0,05	0,06
β_1	0,9	0,89

3.3 Stochastic Volatility Example

To validate the correctness of the Stan model for this category models, we chose to simulate the simple stochastic volatility model of Kim, Shephard, and Chib (1998). This model is:

$$y_t = \beta * e^{\frac{h_t}{2}}, \quad e_t \sim N(0,1)$$

$$h_t = \mu + \varphi * (h_{t-1} - \mu) + \sigma_n * n_t, \quad n_t \sim N(0,1)$$

We then chose to create a time series of 1500 moments with random parameters φ=0.95, σ=0.25, and μ=-10, and table 2 shows the initial values compared to the average of the posterior values of the object created by STAN.

Table 2: Comparison of initial and posterior values of Stochastic Volatility

Parameter	**Initial Value**	**Average of posterior values**
μ	-10	-9,97
φ	0,95	0,94
σ	0,25	0,27

Unobserved ARCH example:

For this example, we chose to simulate the one proposed by Giakoumatos (2004) where:

$$y_t = f_t + \sigma * e_t$$

$$f_t = u_t * \sqrt{h_t}$$

$$h_t = \alpha + \beta * f_{t-1}^2$$

with $u_t, e_t \sim N(0,1)$. We set randomly, $\alpha=0,1$, $\beta=0,8$ and $\sigma=0,2$.and the output object of STAN predicted the parameter values as in the following table:

Table 3: Comparison of initial and posterior values of Unobserved ARCH

Parameter	Initial Value	Average of posterior values
α	0,1	0,10
β	0,8	0,77
σ	0,2	0,22

Taking into account the results of the 3 tables we can easily understand that the simulated models in STAN managed to converge correctly and predict with high accuracy the values of the parameters we had initially set.

4. Illustration in real stock prices

After validating the correct functioning of GARCH, Stochastic Volatility and Unobserved ARCH, these models were fitted to the stock prices of 2 large companies of the New York Stock Exchange (Google and Apple) from January 2010 to February 2020. In both stocks the volatility clustering effect is evident and can be easily seen in figure 1 for google shares and in figure 2 for Apple shares.

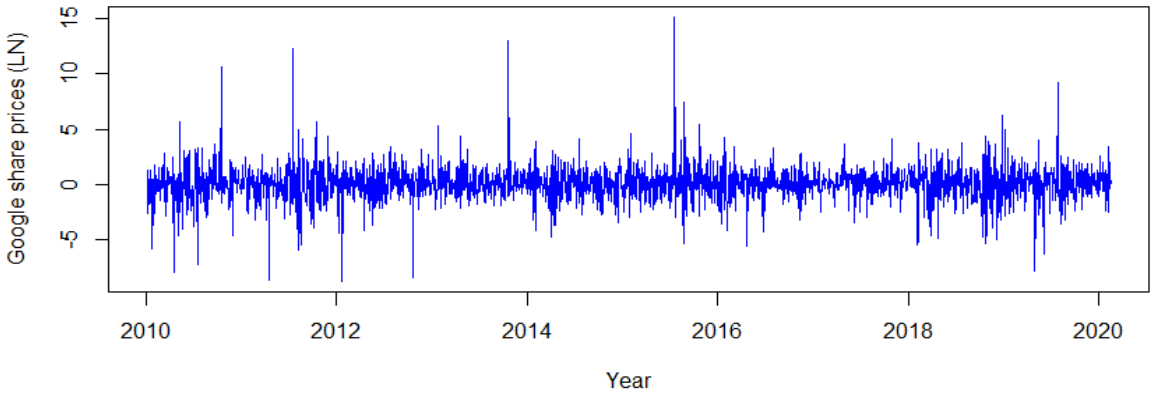

Figure 1: Google share log returns from 4/1/2010 to 2/14/2020

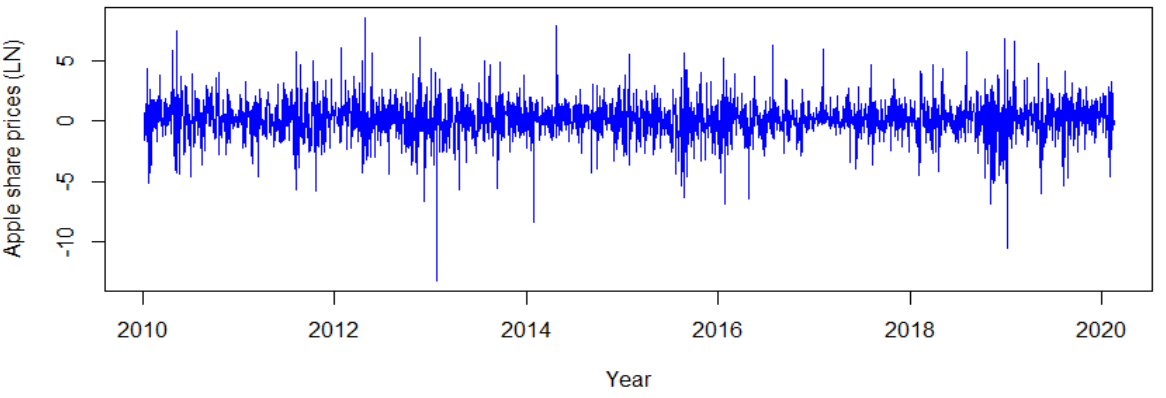

Figure 2: Apple share log returns from 4/1/2010 to 2/14/2020

The aim was to calculate the average values and variance of their parameters and to apply Information Criterion to assess the quality of these statistical models.
The statistical package STAN for model comparison uses the LOO library where the LOO and WAIC evaluation criteria are used (Vehtari, Gelman, & Gabry, 2017). The purpose of these criteria is to calculate for each model the expected log predictive density (ELPD) according to the following formula:

$$elpd = \sum_{i=1}^{N} \int p_t\,(\widetilde{y_i})\ \log(\widetilde{y_i}|y) d\widetilde{y_i}$$

4.1 Comparison of results

The log returns of the 2 stocks were given as input to Stan's GARCH model for 10000 iterations and the convergence of the chains and the parameters α_0, α_1, β_1 values can be seen in graphs 3 and 4.

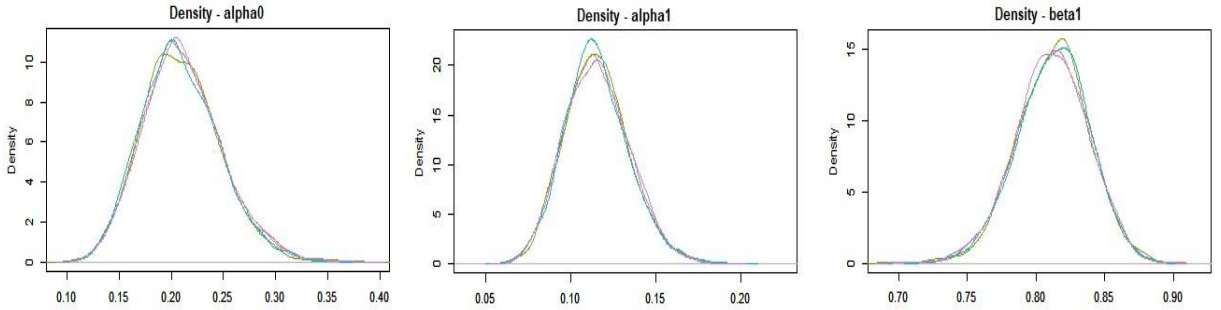

Figure 3: Density plot of GARCH model parameters for Apple shares

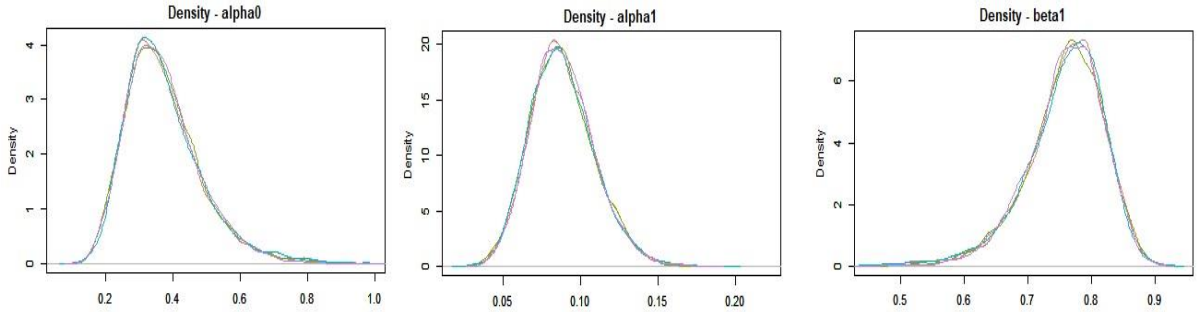

Figure 4: Density plot of GARCH model parameters for Google shares

In the same way we calculated the values of the parameters in the model Stochastic volatility. Figures 5 and 6 show the density plots parameters μ, φ, σ of each stock.

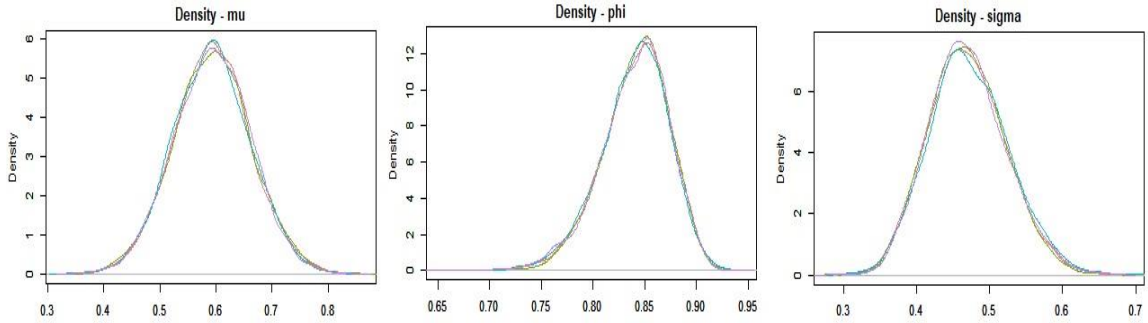

Figure 5: Density plot of SV model parameters for Apple shares

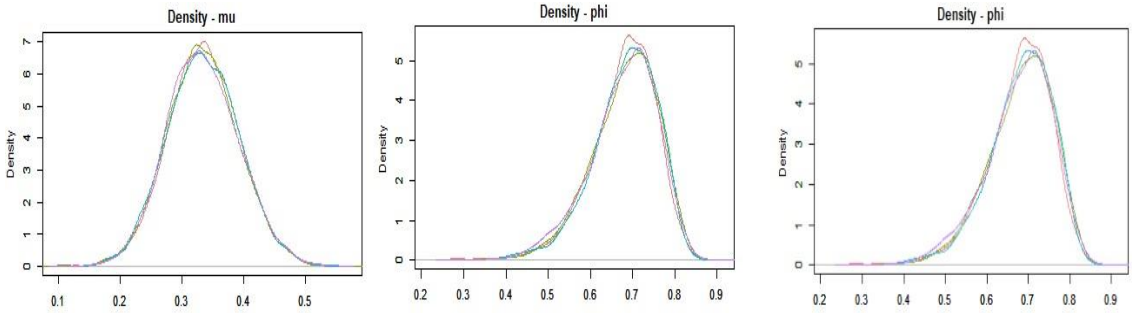

Figure 6: Density plot of SV model parameters for Google shares

Finally, Figures 7 and 8 show the density plots of the parameter values α, β, σ of the Unobserved ARCH model.

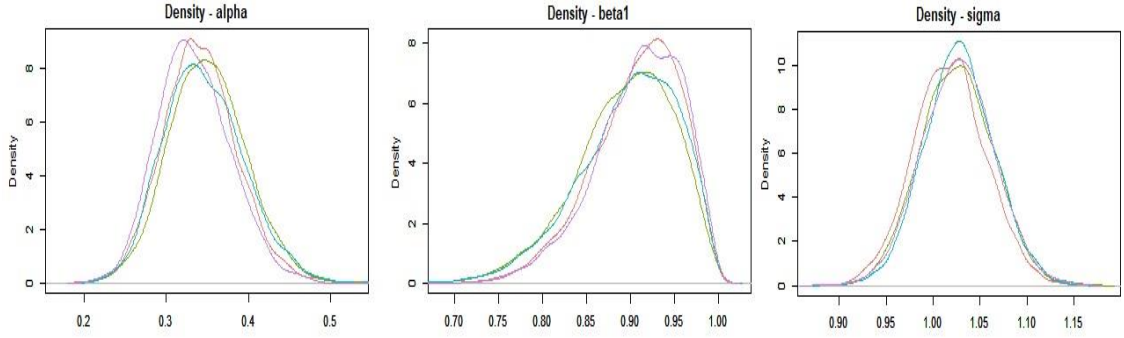

Figure 7: Density plot of uARCH model parameters for Apple shares

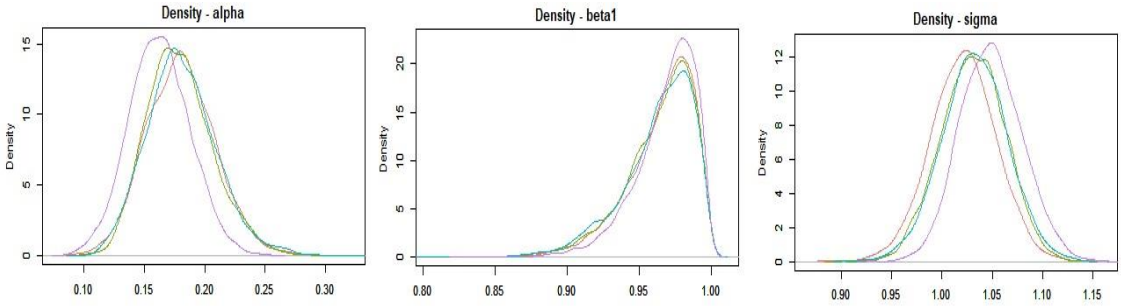

Figure 8: Density plot of uARCH model parameters for Google shares

The following table summarizes all the results of the 3 methods per share.

Table 4: Predicted parameters per share

	GARCH			**Stochastic Volatility**			**Unobserved Arch**		
	α_0	α_1	β_1	μ	φ	σ	α	β	σ
Apple	0,21	0,12	0,81	0,59	0,84	0,47	0,34	0,90	1,02
Google	0,37	0,09	0,76	0,33	0,68	0,68	0,18	0,96	1,03

The prices of both stocks seem to exhibit high persistence in price volatility phenomena, with Google stock being less prone to such phenomena. From the GARCH method it was calculated for Apple stock that $\alpha_1+\beta_1=0.93$ while for Google stock $\alpha_1+\beta_1=0.85$ which proves the above claim. A similar conclusion can be drawn by checking the variable φ of the Stochastic volatility method where it was calculated that $\varphi=0.84$ for Apple while $\varphi=0.68$ for Google. Finally, for the Unobserved Arch model, both stocks showed high values in the parameters demonstrating the high persistence of prices in volatility phenomena.

4.2 Information Criteria results

After the convergence of each model, STAN calculated the log predictive density, compared it to the values of the other models, and ranked each one in descending order from the one with the highest value to the one with the lowest. The results are shown in table 5:

Table 5: Predictive ability of models

	Apple		Google	
	elpd_diff	se_diff	elpd_diff	se_diff
Stochastic Volatility	0.00	0.00	0.00	0.00
Unobserved ARCH	-5.36	20.96	-78.70	27.99
GARCH (1,1)	-179.17	36.35	-331.96	68.65

In both stock cases the SV model was ranked first with the best ability to predict the parameter values. Model 2 with a small difference in the case of Apple but with a much larger difference for google was ranked 2nd while as expected GARCH's model was ranked last.

5. Conclusion and further research

In this paper, an attempt was first made to document the basic concept of the operation of a valuable statistical inference tool, Markov chains. For this purpose, the most important MCMC algorithms, the advantages of each of them and the necessary parameterizations required for their proper operation were presented.

One of the applications of these algorithms is to estimate the parameters of variance models that are capable of explaining the phenomenon of volatility clustering in economic time series data. Thus, 3 models that describe the later phenomenon were chosen to study and simulate in STAN. After establishing the correctness of the predictions of all three methods, they were tested on real data for 2 stock market shares.

In conclusion, we could say that in all cases the MCMC algorithms proved to be a powerful tool that was able to predict the values of the parameters of interest quickly and very accurately. At the same time, the STAN programming package was a very sophisticated high-level environment that provided the required libraries for modeling, validation, and visualization of the results. Furthermore, this work could be generalized to the use of MCMC algorithms for parameter value prediction in multivariate models.

References

[1] Atzberger, P. J. (2013). University of California, Santa Barbara. http://web.math.ucsb.edu/~atzberg/pmwiki_intranet/uploads/AtzbergerHomePage/Atzberger_MonteCarlo.pdf

[2] Besag, J. (1974). Spatial Interaction and the Statistical Analysis of Lattice Systems. Journal of the Royal Statistical Society. 36(2), 192-236.

[3] Bollerslev, T. (1986). Generalized autoregressive conditional heteroskedasticity. Journal of Econometrics, 31(3), 307-327.

[4] Carpenter, B., Gelman, A., Hoffman, M., & Lee, D. (2017). Stan: A probabilistic programming language. Journal of Statistical Software Articles, 1, 1-32.

[5] Chen, S.-H., & Edward, H. (2015). Behavior of the Gibbs Sampler When Conditional Distributions Are Potentially Incompatible. Journal of statistical computation and simulation, 85(16), 3266–3275.

[6] Duane, S., Kennedy, A. D., Pendleton, B. J., & Roweth, D. (1987). Hybrid Monte Carlo. Physics Letters, 216-222.

[7] Engle, R. (1982). Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of United Kingdom Inflation. Econometria, 50(4), 987-1007.

[8] Geman, S., & Geman, D. (1984). Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. IEEE Transactions on Pattern Analysis and Machine Intelligence, pp. 721-741.

[9] Giakoumatos, S. (2004). Auxiliary Variable Sampler for some time-varying volatility models. Athens: Τμήμα Λογιστικής Πανεπιστημίου Αθηνών.

[10] Hanzon, B. (2003). A short introduction to time-varying volatility in financial time series. IFAC Proceedings Volumes, 36(16), 217-220.

[11] Kim, S., Shephard, N., & Chibb, S. (1998). Stohastic Volatility: Likehood inference and Comparison with ARCH models. Review of Economic Studies (65), 361-393.

[12] Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A., & Teller, E. (1953, June). Equations of State Calculations by Fast Computing Machines. Journal of Chemical Physics, σσ. pages 1087–1092.

[13] Shephard, N. (1996). Statistical aspects of ARCH and stochastic volatility. Monographs on Statistics and Applied Probability, 65, 1-68.

[14] Taylor, S. J. (1994). Modelling stochastic volatility: A review and comparative study. Mathematical Finance, 4(2), 183-204.

[15] Vehtari, A., Gelman, A., & Gabry, J. (2017). Practical Bayesian model evaluation using leave-one-out cross-validation and WAIC. Statistics and Computing (27), 1413-1432.

[16] Wei, W. W. (2006). Time Series Analysis: Univariate and Multivariate Methods (2nd εκδ.). Addison Wesley Publishing Company.

[17] Zhang, Y. (2013). Markov Chain Monte Carlo (MCMC) Simulations. Encyclopedia of Systems Biology. New York: Springer.