12-2000

BUGS for a Bayesian analysis of stochastic volatility models

Renate Meyer

Jun YU
Singapore Management University, yujun@smu.edu.sg

Econometrics Journal (2000), volume **0**, pp. 1–17.

BUGS for a Bayesian Analysis of Stochastic Volatility Models

RENATE MEYER[†], JUN YU[‡]

[†]*Department of Statistics, University of Auckland, Private Bag 92019, Auckland, New Zealand.*
E-mail: meyer@stat.auckland.ac.nz
[‡]*Department of Economics, University of Auckland, Private Bag 92019, Auckland, New Zealand.*
E-mail: j.yu@auckland.ac.nz

Received: April 2000

Summary This paper reviews the general Bayesian approach to parameter estimation in stochastic volatility models with posterior computations performed by Gibbs sampling. The main purpose is to illustrate the ease with which the Bayesian stochastic volatility model can now be studied routinely via BUGS (Bayesian Inference Using Gibbs Sampling), a recently developed, user-friendly, and freely available software package. It is an ideal software tool for the exploratory phase of model building as any modifications of a model including changes of priors and sampling error distributions are readily realized with only minor changes of the code. BUGS automates the calculation of the full conditional posterior distributions using a model representation by directed acyclic graphs. It contains an expert system for choosing an efficient sampling method for each full conditional. Furthermore, software for convergence diagnostics and statistical summaries is available for the BUGS output. The BUGS implementation of a stochastic volatility model is illustrated using a time series of daily Pound/Dollar exchange rates.

Keywords: *Stochastic Volatility, Gibbs Sampler, BUGS, Heavy-tailed Distributions, Non-Gaussian Nonlinear Time Series Models.*

1. INTRODUCTION

The stochastic volatility (SV) model introduced by Tauchen and Pitts (1983) and Taylor (1986) is used to describe financial time series. It offers an alternative to the ARCH-type models of Engle (1982) and Bollerslev (1986) for the well documented time varying volatility exhibited in many financial time series. The SV model provides a more realistic and flexible modeling of financial time series than the ARCH-type models, since it essentially involves two noise processes, one for the observations, and one for the latent volatilities. The so called observation errors account for the variability due to measurement and sampling errors whereas the process errors assess variation in the underlying volatility dynamics (see, for example, Danielsson (1994), Geweke (1994), and Kim et al. (1998) for the comparative advantages of the SV model over the ARCH-type models).

Unfortunately, classical parameter estimation for SV models is difficult due to the intractable form of the likelihood function. Recently, a variety of frequentist estimation

methods have been proposed for the SV model, including Generalized Method of Moments (Melino and Turnbull, 1990), Quasi-Maximum Likelihood (Harvey et al., 1994), Efficient Method of Moments (Gallant et al., 1997), Simulated Maximum Likelihood (Danielsson, 1994), Maximum Likelihood Monte Carlo (Sandmann and Koopman, 1996), and direct Maximum Likelihood (Fridman and Harris, 1998). A Bayesian analysis of the SV model is complicated due to multidimensional integration problems involved in posterior calculations. These difficulties with posterior computations have been overcome, though, with the development of Markov Chain Monte Carlo (MCMC) techniques (Gilks et al. 1996) over the last two decades and the ready availability of computing power. MCMC procedures for the SV model have been suggested by Jacquier et al. (1994), Shephard and Pitt (1997), and Kim et al. (1998).

Among all of these methods, MCMC ranks as one of the best estimation tools. See, for example, Andersen et al. (1999) for a comparison of various methods in Monte Carlo studies. However, MCMC procedures are "computationally demanding and much harder to implement, using nonconventional software that is not widely available among researchers and practitioners in the field" (Fridman and Harris 1998, p. 285).

The purpose of this paper is to illustrate a Bayesian analysis of a SV model using Gibbs sampling. We report on significant progress made in facilitating the routine implementation which may have a revolutionary effect on modeling financial time series in everyday practice, even if only as a quick, preliminary step in an exploratory phase before employing more sophisticated and specialised time series software such as SsfPack (Koopman et al., 1999). This is achieved through BUGS, a recently developed software package (Spiegelhalter et al., 1996) that implements the Gibbs sampler. BUGS is available free of charge from

http://www.mrc-bsu.cam.ac.uk/bugs/welcome.shtml

for the operating systems UNIX, LINUX, and WINDOWS, among others. It comes with complete documentation and two example volumes. BUGS is not a package specially designed for econometric time series analysis like SsfPack but rather an all-purpose Bayesian software that does sampling-based posterior computations for a variety of statistical models such as random effects, generalized linear, proportional hazards, latent variable, and frailty models. We will show that state-space models (Harvey, 1990), in particular the SV model, is amenable to a Bayesian analysis via BUGS. The implementation of this SV model requires just 18 lines of code in its entirety and does not require the programmer to know the precise formulae for any prior density or likelihood. Its main strength lies in the ease with which any changes in the model, such as different autoregressive structures or polynomial state transitions, the choice of different prior distributions for the parameters, and the change from Gaussian to heavy-tailed observation error distributions, are accomplished. This is in contrast to tailored implementations of stochastic volatility models in low-level programming languages such as C, where any such change necessitates major reprogramming usually followed by even more time-consuming debugging. Consequently, we suggest that the BUGS software may permit the implementation phase of SV models to be relegated, thereby allowing more effort to be devoted to the challenging issues of model checking, sensitivity analysis, and prior specification.

The paper is organized as follows: In Section 2 we introduce a stochastic volatility model using a well-studied dataset. Section 3 puts the SV model into the framework of nonlinear state-space methodology and describes the Bayesian approach to parameter

estimation using Gibbs sampling. This model is then represented as a DAG in the fourth Section. Its implementation in BUGS is illustrated in Section 5 and a timing comparison is made to an implementation of the same model in SVPack, a freeware dynamic link library for the Ox programming language (Doornik, 1996) compiled using the Microsoft Visual C++ Compiler, available from the World Wide Web at ULR:

http://www.nuff.ox.ac.uk/users/shephard/ox/

Section 6 demonstrates that any modification of the model becomes a simple task in BUGS and is implemented in no time. We highlight the strength of the BUGS approach as well as its current limitations in the Discussion.

2. THE STOCHASTIC VOLATILITY MODEL

For illustrative and comparative purposes, we use a dataset that has been previously analyzed by Harvey et al. (1994) and more recently by Shephard and Pitt (1997) and Kim et al. (1998) using Gibbs sampling and by Durbin and Koopman (2000) using a ML as well as a Bayesian approach via importance sampling. The data consist of a time series of daily Pound/Dollar exchange rates $\{x_t\}$ from 01/10/81 to 28/6/85. The series of interest are the daily returns, $\{y_t\}$, given by the transformation $y_t = \log x_t - \log x_{t-1},\ t = 1, \ldots, n$. The SV model used for analyzing these data can be written in the form of a nonlinear state-space model (Harvey, 1990). A state-space model specifies the *conditional* distributions of the observations *given* unknown states, here the underlying latent volatilities, θ_t, and *given* the unknown observation error variance σ^2, in the observation equations:

$$y_t|\theta_t = \exp\left(\frac{1}{2}\theta_t\right) u_t, \quad u_t \overset{iid}{\sim} N(0,1),\ t = 1, \ldots, n. \tag{1}$$

The unknown states are assumed to follow a Markovian transition over time (therefore the state-space models are often referred to as "hidden Markov models") given by the state equations:

$$\theta_t|\theta_{t-1}, \mu, \phi, \tau^2 = \mu + \phi(\theta_{t-1} - \mu) + v_t, \quad v_t \overset{iid}{\sim} N(0, \tau^2),\ t = 1, \ldots, n, \tag{2}$$

with $\theta_0 \sim N(\mu, \tau^2)$. The state θ_t determines the amount of volatility on day t and the value of ϕ, $-1 < \phi < 1$, measures the autocorrelation present in the logged squared data. Thus ϕ can be interpreted as the persistence in the volatility, the constant scaling factor $\beta = \exp(\mu/2)$ as the instantaneous volatility, and τ as the volatility of log-volatilities (cf. Kim et al. 1998).

A full Bayesian model consists of the joint prior distribution of all unobservables, here the three parameters, μ, ϕ, τ^2, and the unknown states, $\theta_0, \theta_1, \ldots, \theta_n$, and the joint distribution of the observables, here the daily returns $y_1, \ldots, y_n$. Bayesian inference is then based on the posterior distribution of the unobservables given the data. In the sequel, we will denote the probability density function of a random variable θ by $p(\theta)$. By successive conditioning, the joint prior density is

$$p(\mu, \phi, \tau^2, \theta_0, \theta_1, \ldots, \theta_n) = p(\mu, \phi, \tau^2)p(\theta_0|\mu, \tau^2) \prod_{t=1}^{n} p(\theta_t|\theta_{t-1}, \mu, \phi, \tau^2). \tag{3}$$

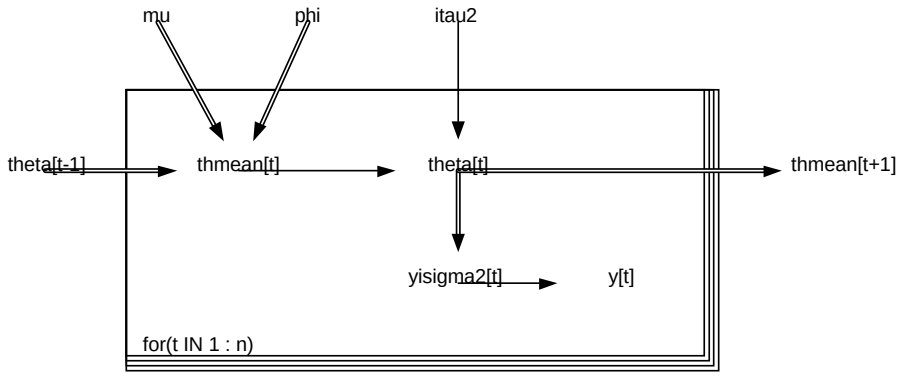

Figure 1. Representation of the stochastic volatility model as a directed acyclic graph (DAG).

We assume prior independence of the parameters μ, ϕ, and τ^2, and use the same priors as in Kim et al. (1998). We employ a slightly informative prior for μ, $\mu \sim N(0, 10)$. We set $\phi = 2\phi^* - 1$ and specify a Beta(α, β) prior for ϕ^* with $\alpha = 20$ and $\beta = 1.5$ which gives a prior mean for ϕ of 0.86. A conjugate inverse-gamma prior is chosen for τ^2, i.e. $\tau^2 \sim IG(2.5, 0.025)$ which gives a prior mean of 0.0167 and prior standard deviation of 0.0236. $p(\theta_t|\theta_{t-1}, \mu, \phi, \tau^2)$ is defined through the state equations (2). The likelihood $p(y_1, \ldots, y_n|\phi, \sigma^2, \tau^2, \theta_0, \ldots, \theta_n)$ is specified by the observation equations (1) and the conditional independence assumption:

$$p(y_1, \ldots, y_n|\mu, \phi, \tau^2, \theta_0, \ldots, \theta_n) = \prod_{t=1}^{n} p(y_t|\theta_t). \tag{4}$$

Then, by Bayes' theorem, the joint posterior distribution of the unobservables given the data is proportional to the prior times likelihood, i.e.

$$\begin{aligned} p(\mu, \phi, \tau^2, \theta_0, \ldots, \theta_n|y_1, \ldots, y_n) \quad \propto \quad & p(\mu)p(\phi)p(\tau^2)p(\theta_0|\mu, \tau^2) \textstyle\prod_{t=1}^{n} p(\theta_t|\theta_{t-1}, \mu, \phi, \tau^2) \times \\ & \textstyle\prod_{t=1}^{n} p(y_t|\theta_t). \end{aligned} \tag{5}$$

3. SV MODEL REPRESENTATION AS DAG

A graphical representation of the full Bayesian SV model not only helps to focus on the essential model structure but can be used in the WinBUGS version to implement the model. For any day t, let us represent all unobservables, $\mu, \phi, \tau^2, \theta_t$, and observables, y_t, as ellipses. A way to express the conditional independence assumptions is by drawing solid arrows between nodes (see Figure 1). Hollow arrows go to deterministic nodes, which are logical functions of other nodes. The conditional mean of θ_t, thmean[t], is an example of a deterministic node as it is a function of the nodes μ, ϕ, and θ_{t-1}.

This renders a model representation as a directed acyclic graph (DAG) as all edges in the graph are directed and there are no cycles because of the conditional independence assumptions. Let V denote the set of all nodes in the graph. Direct predecessors of a node $v \in V$ are called "parents", direct offspring the "children". The solid arrows indicate that *given its parent nodes*, each node v is independent of all other nodes except descendants of v. For instance, if on day t we know the volatility on day $t-1$ and the values of the parameters μ, ϕ, and τ^2, then our belief about the volatility, θ_t, on day t is independent of the volatilities on previous days 1 to $t-2$ and the data of all other days except the current return y_t.

It is then easy to construct the joint probability distribution of all stochastic nodes using the graphical description of the conditional independence assumptions:

$$p(V) = \prod_{v \in V} p(v|\text{parents}(v)). \tag{6}$$

For our specific SV model, (6) is the graph-theoretical version of the right hand side of (5). In this way, the DAG (Figure 1) assists in constructing the full Bayesian model. For further reading on conditional independence graphs and graphical chain models the interested reader is referred to Wermuth and Lauritzen (1990).

4. BAYESIAN INFERENCE USING BUGS

Let V_u denote the subset of unobservable nodes, and V_o the subset of observable nodes. Once $p(V)$ has been obtained from (6), a general technical difficulty encountered in any application of Bayesian inference is calculating the high-dimensional integral necessary to find the normalization constant in the posterior distribution of the unobservables given the data:

$$p(V_u|V_o) = \frac{p(V_u, V_o)}{p(V_o)} = \frac{p(V)}{\int p(V_u, V_o) dV_u}. \tag{7}$$

In our specific example this would require an $(N+4)$-dimensional integration as we have to integrate over the unobservables $\mu, \phi, \tau^2, \theta_0, \ldots, \theta_n$. Calculating the marginal posterior distribution of any variable would require a subsequent $(N+3)$-dimensional integration. High-dimensional integration problems can be solved via Markov chain Monte Carlo as reviewed in Gilks et al. (1996). The Gibbs sampler, a special MCMC algorithm, generates a sample from the posterior (7) by iteratively sampling from each of the univariate full conditional posterior distributions. These univariate full conditional posterior distributions $p(v|V\backslash v)$, for $v \in V_u$, can be easily constructed from the joint posterior distribution $p(V)$ in (6) by picking out those terms that depend on v:

$$p(v|V\backslash v) \propto p(v|\text{parents}(v)) \prod_{v \in \text{parents}(w)} p(w|\text{parents}(w)). \tag{8}$$

This is facilitated by the graphical representation (Figure 1) as the full conditional posterior distribution of any node v depends only on its parents, children, and co-parents. For instance, if $v = \theta_t$, then the full conditional posterior distribution of θ_t, $p(\theta_t|\mu, \phi, \tau^2, \theta_0, \ldots, \theta_{t-1}, \theta_{t+1}, \theta_n, y_1, \ldots, y_N)$, is proportional to

$$p(\theta_t|\theta_{t-1}, \mu, \phi, \tau^2) \times p(\theta_{t+1}|\theta_t, \mu, \phi, \tau^2) \times p(y_t|\theta_t).$$

Here, the dependence of the deterministic node thmean[t] as logical function of θ_t, μ, ϕ, and τ^2 has been resolved. In this way, BUGS exploits the representation of the model as a DAG for constructing these full conditional posterior distributions for all unobservable nodes. Once this is accomplished, it uses certain sophisticated sampling methods to sample from these univariate densities. BUGS contains a small expert system for choosing the best sampling method. The first choice is to identify conjugacy, where the full conditional reduces analytically to a well-known distribution, and to sample accordingly. If the density is not conjugate but turns out to be log-concave, it employs the *adaptive rejection sampling* (ARS) algorithm (Gilks and Wild, 1992). If the density is not log-concave, BUGS uses a Metropolis-Hastings (MH) step. The MH algorithms differ across the various BUGS versions and platforms. The current UNIX version 0.6 uses the Griddy Gibbs sampler as developed by Ritter and Tanner (1992). More efficient MH implementations currently under development include slice sampling (Neal, 1997) for variables with a restricted range, and adaptive techniques (Gilks et al., 1998) for variables with unrestricted range. A first version has been released under WinBUGS, the BUGS version for the WINDOWS95 operating system.

5. MODEL IMPLEMENTATION IN BUGS

For a typical BUGS run using the UNIX version 0.6, four different files have to be specified:

1. The data are entered in a file with extension .dat, here called returns.dat.
2. Initial values for all unobservables needed to start the Gibbs sampler are in a .in file, here sv.in.
3. The file that contains the model descriptions has the .bug extension, here sv.bug.
4. The commands for running BUGS that control the number of sampled values, which parameters to monitor and so on, go into the sv.cmd file. In WinBUGS, these commands are options in various menus.

These four files are listed in the Appendix.

The data file can be either in rectangular format like returns.dat or in SPLUS format as in returns_S.dat.

The format of the initial value file follows that of the data file, here for instance sv.in in SPLUS format. If initial values for parameters are not given in the initial value file, then values will be generated by forward sampling from the prior distributions specified in the model.

The DAG representation of our model not only serves BUGS-internal purposes but can assist us in specifying our model in the BUGS language. The file containing the

model specification, sv.bug, consists of two sections: the declarations and the model description. The declaration part specifies the name of the model, which nodes are constants (rectangles) and which are stochastic (ellipses), and gives the name of the files containing the data and initial values. The model description forms a declarative representation of the fully Bayesian model. It is enclosed in curly brackets $\{\ldots\}$. This is a translation of the graphical model into BUGS syntax. Each statement consists of a relation of two kinds:

$\sim$ which means "is distributed as" and substitutes the solid arrows,

$<-$ which means "is to be replaced by" and substitutes the hollow arrows.

Quantities on the left of a $\sim$ are stochastic, those on the left of $<-$ are deterministic. In general, each quantity should appear once and only once on the left-hand-side of a statement. The order of the expressions within a pair of braces is irrelevant.

As mentioned before, WinBUGS has an option Doodle that allows the user to specify the model graphically by drawing a DAG. It uses a hyper-diagram approach to add extra information to the graph to give a complete model specification. DoodleBUGS then writes the corresponding model in BUGS syntax to a file.

Note that BUGS uses a nonstandard parametrization of distributions in terms of the precision (1/variance) instead of the variance. For example, a normal distribution denoted by dnorm(ν, ψ^2) has density function $f(x|\nu, \psi^2) = \frac{\psi}{\sqrt{2\pi}} \mathrm{e}^{-\frac{1}{2}(x-\nu)^2\psi^2}$. This is the reason for specifying the node itau2. BUGS does not permit an expression to be used as a parameter of a distribution, and hence we need the deterministic nodes thmean[t] and yisigma2[t].

Finally, the file sv.cmd compiles the BUGS commands in sv.bug, generates an initial 10000 iterations (the so called "burn-in" period), monitors every specified parameter for the next 100,000 iterations, stores every 20th value, and calculates summary statistics of the sampled values for each specified parameter, based on a final chain length of 5000. To make results comparable to those in Kim et al. (1998), we chose to monitor the nodes ϕ, τ, and β. Our preference is to submit these commands as a batch job by using the command

```
backbugs "sv.cmd"
```

with session output automatically directed to the bugs.log file. Alternatively, BUGS can be run interactively by using the command bugs.

BUGS generates 5 files after completion:

1. The file bugs.log contains the BUGS code (sv.bug) that was used, any error messages, the running time, and the requested summary statistics of the marginal posterior distribution of each parameter. The posterior summaries from this file are listed in the Appendix.
2. The file bugs.out contains 2 columns. The first column gives the iteration number, the second column the corresponding sampled value.
3. The file bugs.ind tells you which line of the bugs.out file corresponds to which monitored variable. Here, lines 1 to 5000 of bugs.out are samples from variable phi, lines 5001 to 10000 from variable tau and lines 10001 to 15000 from variable beta.

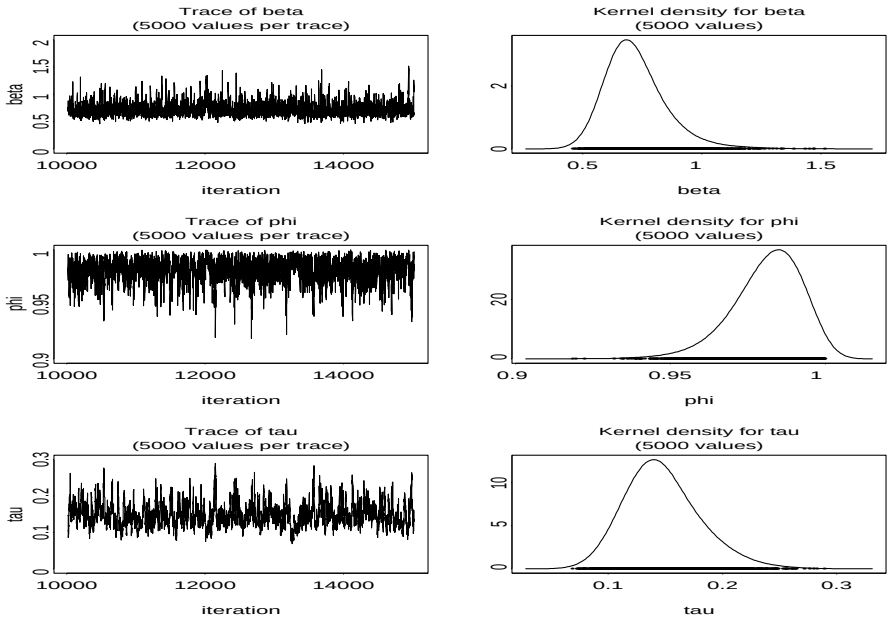

Figure 2. Trace and kernel density estimates of the marginal posterior distribution of model parameters.

4. The file bugs1.out contains the results of the stats command in a rectangular format suitable for reading into statistical packages for producing graphs, tables etc. The 10 columns contain the summary statistics: mean, sd, observed lower percentile being reported (default 2.5%), observed lower percentile, observed upper percentile being reported, observed upper percentile (default 97.5%), median, number of iterations, start iteration, finish iteration.
5. The file bugs1.ind contains the node names for the variables listed, and the corresponding row number in the bugs1.out file.

A menu-driven collection of SPLUS functions, CODA (Best et al. 1995), is available for analysing the output obtained by BUGS. Besides trace plots and convergence diagnostics based on Cowles and Carlin (1996), CODA calculates statistical summaries of the posterior distributions and kernel density estimates. A version of CODA that runs under the public domain software R can be downloaded from

http://www-fis.iarc.fr/codaversion.

Kernel estimates of the posterior densities of the parameters ϕ, τ, and β are shown in Figure 2. These compare with those given in Figure 2 of Kim et al. (1998) and Figure 2 of Shephard and Pitt (1997). Table 1 compares the posterior means and standard deviations of the parameters ϕ, τ, and β to the Bayesian estimates obtained using the SVPack

Table 1. Comparison of Bayesian estimates obtained from state-space model using BUGS with ML estimates obtained by Durbin and Koopman (2000) and with Bayesian estimates using SVPack as in Kim et al. (1998).

Variable	BUGS	SVPack	Durbin and Koopman
	posterior mean (SE)	posterior mean (SE)	ML estimate (SE)
ϕ	0.9807 (0.01075)	0.9773 (0.01060)	0.9731 (0.0131)
τ	0.1489 (0.03079)	0.1611 (0.03035)	0.1726 (0.03745)
β	0.7204 (0.1183)	0.6476 (0.10059)	0.6336 (0.06547)
time (sec)	6,171	198	

implementation of Kim et al. (1998) and to the ML estimates obtained by Durbin and Koopman (2000). Note that only approximate standard deviations are given, obtained by the delta method, from standard deviations of transformed parameters in Durbin and Koopman (2000). The results from SVPack are based on the same number of iterations, same priors, and starting values, and are basically identical to the ones quoted in Table 1 of Kim et al. (1998).

Extensive convergence diagnostics for this chain were calculated using the CODA software of Best et al. (1995). All parameters passed the Heidelberger and Welch stationarity and halfwidth tests. But Geweke's Z-scores for ϕ, τ, and β are still borderline with values close to ± 2 and the Raftery and Lewis convergence diagnostics (estimating the 2.5th percentile up to an accuracy of 0.02 with probability 0.9) suggest a larger sample size. This is reflected in high posterior autocorrelations as already noted by Kim et al. (1998) and Shephard and Pitt (1997). Note that Kim et al. (1998) based their results on a chain of length 1,000,000 after a burn-in period of 50,000.

The computing time to generate 100 iterations is 6.1 seconds using the LINUX version of BUGS on a Pentium III PC. True, this compares dismally with 0.18 seconds per 100 iterations for the SVPack implementation on the same computer. One should keep in mind, however, that the SVPack implementation used optimized C++ code that was dynamically linked to the Ox programming package. The all-purpose package BUGS is not meant to be a serious competitor to a specially tailored package for one particular model. Whereas BUGS uses a "black-box" simulation technique (ARS) to sample from a full conditional density, the SVPack implementation makes use of bounds for each full conditional density that results in efficient rejection sampling but requires a mathematical analysis of the full conditionals under consideration, as detailed in Kim et al. (1998). BUGS main strength, however, lies in its flexibility that allows the practitioner to experiment with different scenarios.

6. FLEXIBLE MODELLING

Any implementation of the Gibbs sampler requires the specification of each of the full conditional posterior densities and of a simulation technique to sample from them. Any change in the model such as a different prior distribution or different sampling distribution necessarily entails changes in those full conditional densities. As BUGS alleviates

from the tedious task of calculating the full conditionals and as it also chooses an efficient method to sample from them, one can experiment with different types of models without worrying about major reprogramming. Modifications of the model are straightforward to implement by changing just one or two lines in the code. To illustrate this major strength and flexibility of the BUGS software, we consider the following variations of the basic model implemented in the previous chapter:

- a sensitivity analysis, i.e. a change from the informative prior distributions for the model parameters μ, ϕ, and τ^2 to noninformative priors to check how sensitive the results are to prior specifications,
- fitting more complex models with additional parameters,
- and using heavy-tailed distributions instead of Gaussians for the observation errors.

Changing the prior distributions of the three model parameters comes down to changing three lines of code. BUGS only allows use of proper prior distributions but the noninformative distribution for a scale parameter like τ^2, $p(\tau^2) \propto 1/\tau^2$, is improper. Therefore, in BUGS one would use a gamma(0.001,0.001) prior for the inverse of τ^2 which is practically equivalent to $p(\tau^2) \propto \frac{1}{\tau^2}$. A vague Normal distribution with mean 0 and some low precision like 0.001 is used to substitute the noninformative distribution for a location parameter. The necessary changes in the BUGS code are:

```
mu ~ dnorm(0,0.001);
phistar ~ dunif(0,1);
itau2 ~ dgamma(0.001,0.001);
```

We observed only minor changes in the posterior distributions of the parameters (ϕ with posterior mean and standard deviation 0.9775 (0.01340), τ with 0.1646 (0.03708), and β with 0.7106 (0.1203)). This indicates that the statistical inference for this data is insensitive to misspecification of priors. Moreover, using flat priors implies equality of posterior mean and mode and should therefore yield estimates that are very close to the ML estimates (see Table 1) obtained by Durbin and Koopman (2000).

We fitted two models with one and two additional parameters, respectively, using the same priors for the common parameters μ, ϕ, and τ^2 as for the model in the previous section, now referred to as Model 1. A non-zero mean is added in the observation equation for

Model 2:

$$y_t|\theta_t = \alpha + \exp\left(\frac{1}{2}\theta_t\right) u_t, \quad u_t \overset{iid}{\sim} N(0,1),\ t = 1,\ldots,n,$$

$$\theta_t|\theta_{t-1},\mu,\phi,\tau^2 = \mu + \phi(\theta_{t-1}-\mu) + v_t, \quad v_t \overset{iid}{\sim} N(0,\tau^2),\ t = 1,\ldots,n.$$

and an AR(2) structure for the state transitions is specified in

Model 3:

$$y_t|\theta_t = \alpha + \exp\left(\frac{1}{2}\theta_t\right) u_t, \quad u_t \overset{iid}{\sim} N(0,1),\ t = 1,\ldots,n,$$

$$\theta_t|\theta_{t-1},\mu,\phi,\tau^2 = \mu + \phi(\theta_{t-1}-\mu) + \psi(\theta_{t-2}-\mu) + v_t, \quad v_t \overset{iid}{\sim} N(0,\tau^2),\ t = 1,\ldots,n.$$

Table 2. Comparison of posterior means and standard deviations (in brackets) obtained using BUGS for parameter estimation in Model 1–4.

Variable	Model 1	Model 2	Model 3	Model 4
ϕ	0.9807 (0.01075)	0.9795 (0.01117)	0.5491 (0.1960)	0.5355 (0.1774)
τ	0.1489 (0.03079)	0.1589 (0.03411)	0.1993 (0.04508)	0.1612 (0.03769)
β	0.7204 (0.1183)	0.7178 (0.1238)	0.7564 (0.1491)	0.7325 (0.1424)
α		-0.0631 (0.01863)	-0.06204 (0.01862)	-0.05733 (0.01846)
ψ			0.4311 (0.1944)	0.4512 (0.1766)
k				13.58 (4.279)
time (sec)	6,171	6,477	9,387	26,115

The implementation of Model 3 in BUGS, for instance, requires only adding α, ψ, and ψ^* to the parameter list, adding corresponding initial values in the vol.in file, changing the likelihood and state equation to:

```
y[t] ~ dnorm(alpha,yisigma2[t]);
thmean[t] <- mu + phi*(theta[t-1]-mu) + psi*(theta[t-2]-mu);
```

and adding priors for α and ψ:

```
alpha ~ dnorm(0,0.001);
psi <-2*psistar-1;
psistar ~ dunif(0,1);
```

In a further extension of Model 3, we consider a central Student t-distribution with unspecified degrees of freedom, k, for the observation errors. Observation and state equations for Model 4 are specified by:

Model 4:

$$y_t|\theta_t = \alpha + \exp\left(\frac{1}{2}\theta_t\right) u_t, \quad u_t \overset{iid}{\sim} t_k,\ t = 1, \ldots, n,$$

$$\theta_t|\theta_{t-1}, \mu, \phi, \tau^2 = \mu + \phi(\theta_{t-1} - \mu) + \psi(\theta_{t-2} - \mu) + v_t, \quad v_t \overset{iid}{\sim} N(0, \tau^2),\ t = 1, \ldots, n.$$

This is coded in BUGS as:

```
y[t] ~ dt(alpha,yisigma2[t],k);
k ~ dchisq(8)I(2,50);
```

We have to restrict the ranges of those nodes with non-logconcave full conditional distributions (such as k above) by specifying lower and upper bounds using the I(lower,upper) function for BUGS to be able to use the Metropolis-Hastings updating step.

The posterior means and standard deviations of the parameters for Models 1-4 and the total computing time needed for 100,000 iterations after a burn-in of 10,000 are given

in Table 2. The Metropolis-Hastings updating slows down the Gibbs sampler for Model 4. Interestingly, the parameter ψ is estimated to be positive and quite large with its upper interval bigger than zero and the posterior mean of k in Model 4 suggests that the observation errors are indeed non-Gaussian.

7. DISCUSSION

The great flexibility of implementation and the variety of alternative prior specification combine to create an enormous set of competing models. Model checking, sensitivity analysis (e.g. Gelman et al., 1996), and MCMC methods for calculating Bayes factors (Han and Carlin, 2000) will of course be vital tools, and model averaging may be a useful approach (e.g. Raftery, 1996). The issue of prior specification is receiving considerable research attention within the statistical community (e.g. Kass and Wasserman, 1996). These efforts will establish guidelines for prior specification. Nonetheless, there will never be an "absolute" regarding the choice of priors, and many modelers therefore see the need for priors as the Achilles heel of Bayesian methodology. However, it is also its strength, the Bayesian approach being the only coherent statistical methodology for updating knowledge using the information contained in data (De Finetti, 1974). As it enables the posterior from one analysis to be used as an induced prior in a subsequent analysis, one can build and exploit an accumulated base of knowledge by "standing on the shoulders of giants".

There is only a very moderate learning curve to ascend before one can write a first BUGS program. However, anyone who is comfortable with model formulae and has an intuitive grasp of fundamental statistical concepts should be capable of understanding, using, and modifying BUGS code that has already been written for a particular SV model. For a Bayesian analysis, the required statistical knowledge includes familiarity with the Bayesian paradigm and an appreciation for the numerically intensive methods used for posterior computations.

We noted a strong dependency of the mixing behaviour of the chains on the specification of bounds for each parameter with non-logconcave full conditional posterior. The tighter those bounds the faster the convergence due to the Griddy Gibbs sampler used in the implementation of the MH step that is necessary to sample from non-logconcave full conditional posteriors, as e.g. from the full conditional of k. As demonstrated by Carter and Kohn (1994) and Shephard and Pitt (1997), a multi-move sampler, i.e. a Gibbs sampler that updates the whole state vector at once instead of one state at a time, can be more efficient. But the multi-move and block samplers are more difficult to implement, requiring specialized code in a low-level programming language such as C++ (Shephard and Pitt, 1997). Writing and debugging might take anything from several days to a few weeks. A subsequent modification of the program, perhaps an extension of the model, choice of different priors, or an application to a different dataset, might well take several hours. The gain in efficiency is therefore largely outweighed by the ease of implementation in BUGS and the feasibility of running large chains, nowadays, on fast computers. The possibility of implementing multi-move or block samplers in BUGS still needs to be investigated.

BUGS offers the opportunity of easily robustifying the stochastic volatility model by using non-Gaussian error distributions with longer than normal tails. This requires

changing just one line of code. Instead of the normal distribution for the states, one could assume Student-t-distributions with low or high degrees of freedom, i.e. $u_t \sim t(0, \tau^2, df)$. Similarly, a heavy-tailed distribution might be more appropriate for the observations to allow for crude measurement errors. Following the terminology of Fahrmeir and Kunstler (1998), we call these *robust* state-space models as they allow for additive and innovation outliers by choosing a heavy-tailed distribution for the observation and process errors, respectively. For models with heavy-tailed observation error distribution, the filters and smoothers are robust against additive outliers. State-space models with heavy-tailed distributions for the states are robust against innovation outliers and have proven useful for fitting time series with change points (Fahrmeir and Kunstler, 1998). The benefits of fitting such robust state-space models to financial time series are being explored (using mixtures of normals, e.g. by Fridman and Harris (1998), Jacquier et al. (1999), Mahieu and Schotman (1998), Steel (1998)) and this remains an important topic for further research.

As typical for the "standard" SV models, with Gaussian errors and linear state transition, the full conditional distributions of all states $\theta_0, \theta_1, \ldots, \theta_n$ are lognormal and therefore sampling from these full conditionals can be very fast using adaptive rejection sampling. However, any nonlinearity in the state equations such as a polynomial for instance, i.e. $\theta_t = \phi_0 + \phi_1\theta_{t-1} + \phi_2\theta_{t-1}^2$, will result in non-logconcave full conditionals for *all* states, thereby requiring "Metropolis-Hastings-within-Gibbs" sampling. This makes the Gibbs sampler prohibitively slow for typical financial time series of about 1000 time points. With more efficient MH algorithms like those based on slice sampling and adaptive techniques that are currently being developed and implemented, this might only be a transient curb and overcome with future enhancements of the BUGS software.

ACKNOWLEDGEMENTS

This work was supported by the Royal Society of New Zealand Marsden Fund and the University of Auckland Research Committee. Furthermore, we would like to thank Neil Shephard for very helpful comments that led to significant improvements over a first draft of this paper.

REFERENCES

Andersen, T., H. Chung, and B. Sorensen. (1999). Efficient method of moments estimation of a stochastic volatility model: A Monte Carlo study. *Journal of Econometrics* 91, 61–87.

Best, N.G., M.K. Cowles, and S.K. Vines (1995). *CODA manual version 0.30.* MRC Biostatistics Unit, New York.

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

Carter C.K. and R. Kohn (1994). On Gibbs sampling for state space models. *Biometrika* 81, 541–553.

Cowles, M.K. and B.P. Carlin (1996). Markov chain Monte Carlo convergence diagnostics: a comparative review. J. Am. Stat. Assoc. 91, 883–905.

Danielsson, J. (1994). Stochastic volatility in asset prices: Estimation with simulated maximum likelihood. *Journal of Econometrics* 64, 375–400.

De Finetti, B. (1974). *Theory of probability; a critical introductory treatment, vol. 1.* Wiley, London.

Doornik, J.A. (1996). *Ox: Object oriented matrix programming, 1.10.* Chapman and Hall, London.

Durbin, J. and S.J. Koopman (2000). Time series analysis of non-Gaussian observations based on state space models from both classical and bayesian perspectives (with discussion). *Journal of the Royal Statistical Society Series B* 62, 3–56.

Engle, R.F. (1982). Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation. *Econometrica* 50, 987–1007.

Fahrmeir, L., and R. Kunstler (1998). Penalized likelihood smoothing in robust state space models. Discussion Paper 111. http://www.stat.uni-muenchen.de/sfb386/

Fridman, M. and L. Harris (1998). A maximum likelihood approach for non-Gaussian stochastic volatility models. *Journal of Business and Economic Statistics* 16, 284–291.

Gallant, A.R., D. Hsie, and G. Tauchen (1997). Estimation of stochastic volatility models with diagnostics. *Journal of Econometrics* 81, 159–192.

Gelman, A., J.B. Carlin, H.S. Stern, and D.B. Rubin (1996). *Bayesian Data Analysis.* Chapman and Hall, London.

Geweke, J. (1994). Bayesian comparison of econometric models. Working Paper, Federal Reserve Bank of Minneapolis, Minnesota.

Gilks, W.R. and P. Wild (1992). Adaptive rejection sampling for gibbs sampling. *Applied Statistics* 41, 337–48.

Gilks, W.R., A. Thomas, and D.J. Spiegelhalter (1994). A language and program for complex bayesian modelling. *The Statistician* 43, 169–178.

Gilks, W.R., S. Richardson, and D.J. Spiegelhalter (1996). *Markov Chain Monte Carlo in Practice*. Chapman & Hall, London.

Gilks, A., G.O. Robert, and S.K. Sahu (1998). Adaptive Markov Chain Monte Carlo through regeneration. *J. Amer. Statist. Assoc.* 93, 1045–1054.

Han, C. and B.P. Carlin (2000). MCMC methods for computing Bayes factors: a comparative review. MCMC preprint server http://www.statslab.cam.ac.uk/ mcmc/.

Harvey, A. (1990). *Forecasting, Structural Time Series Models and the Kalman Filter.* Cambridge University Press, New York.

Harvey, A.C., E. Ruiz, and N. Shephard (1994). Multivariate stochastic variance models. *Review of Economic Studies* 61, 247–264.

Jacquier, E, N.G. Polson, and P.E. Rossi (1994). Bayesian analysis of stochastic volatility models. *Journal of Business and Economic Statistics* 12, 371–389.

Jacquier, E, N.G. Polson, and P.E. Rossi (1995). Models and prior distributions for multivariate stochastic volatility. Unpublished Paper, Graduate School of Business, University of Chicago.

Jacquier, E, N.G. Polson, and P.E. Rossi (1999). Stochastic volatility: univariate and multivariate extensions. .

Kass, R.E. and L. Wasserman (1996). The selection of prior distributions by formal rules. *J. Amer. Stat. Assoc.* 91, 1343–1370.

Kim S., N. Shephard, and S. Chib (1998). Stochastic volatility: likelihood inference and comparison with ARCH models. *Review of Economic Studies* 65, 361–393.

Koopman, S.J., N. Shephard, and J.A. Doornik (1999). Statistical algorithms for models in state space using SsfPack 2.2. *Econometrics Journal* 2, 107-160.

Mahieu, R.J. and P.C. Schotman (1998). An empirical application of stochastic volatility models. *Journal of Applied Econometrics* 13, 333–360.

Melino, A. and S.M. Turnbull (1990). Pricing foreign currency options with stochastic volatility. *Journal of Econometrics* 45, 239–265.

Neal R.M. (1997). Markov chain monte carlo methods based on 'slicing' the density function. Technical Report No. 9722. Department of Statistics, University of Toronto.

Raftery, A.E. (1996). Hypothesis testing and model selection. In Gilks, W.R., S. Richardson, and D.J. Spiegelhalter, editors. *Markov Chain Monte Carlo in Practice*, pages 163–187. Chapman and Hall, London.

Ritter, C., and M.A. Tanner (1992). Facilitating the Gibbs sampler: the Gibbs stopper and the griddy-Gibbs sampler. *Journal of the Royal Statistical Society, Series B,* 59, 291–317.

Sandmann, G. and S.J. Koopman (1998). Estimation of stochastic volatility models via Monte Carlo maximum likelihood. *Journal of Econometrics* 87, 271–301.

Shephard, N. and M.K. Pitt (1997). Likelihood analysis of non-Gaussian measurement time series. *Biometrika*. 84 653–667.

Spiegelhalter, D.J., A. Thomas, N.G. Best, and W.R. Gilks (1996). *BUGS 0.5, Bayesian inference using Gibbs sampling. Manual (version ii)*. MRC Biostatistics Unit, Cambridge, UK.

Steel, M.F.J. (1998). Bayesian analysis of stochastic volatility models with flexible tails. *Econometrics Reviews* 17, 109–143.

Tauchen, G. and M. Pitts (1983). The price variability-volume relationship on speculative markets. *Econometrica* 51, 485–505.

Taylor, S.J (1986). *Modelling Financial Time Series*. John Wiley, New York.

Wermuth, N. and S.L. Lauritzen (1990). On substantive research hypothesis, conditional independence graphs and graphical chain models (with discussion). *Journal of the Royal Statistical Society Series B* 38, 21–72.

Appendix

returns.dat

```
-0.355531620227711
1.42540904228202
-0.443939876958876
1.02565001671892
1.6775789504837
0.369004108745397
-0.941009269863735
-1.05188251290776
.
.
.
2.18840602683325
```

returns_S.dat

```
list(y = c(-0.355531620227711, 1.42540904228202,..., 2.18840602683325))
```

sv.in

```
0.975
0
50
```

sv.cmd

```
compile("sv.bug")
update(10000)
monitor(phi,20)
monitor(tau,20)
monitor(beta,20)
update(100000)
stats(phi)
stats(tau)
stats(beta)
q()
```

sv.bug:

```
model volatility;
const n=945;

var y[n], yisigma2[n], theta0, theta[n], thmean[n],
    mu, beta, phi, phistar, tau, itau2;

data  y  in "returns.dat";
inits phistar, mu, itau2 in "sv.in";

{
# likelihood: joint distribution of ys

for (t in 1:n) { yisigma2[t] <- 1/exp(theta[t]);
                 y[t] ~ dnorm(0,yisigma2[t]);
               }

# prior distributions

mu ~ dnorm(0,0.1);
phistar ~ dbeta(20,1.5);
itau2 ~ dgamma(2.5,0.025);
beta <- exp(mu/2);
phi  <- 2*phi-1;
tau  <- sqrt(1/itau2);

theta0 ~ dnorm(mu,itau2);
thmean[1] <- mu + phi*(theta0-mu);
theta[1]  ~ dnorm(thmean[1],itau2);
for (t in 2:n) { thmean[t] <- mu + phi*(theta[t-1]-mu);
                 theta[t]  ~ dnorm(thmean[t],itau2);
               }
}
```

bugs.log

```
Bugs>stats(phi)
                  mean        sd       2.5% :  97.5%  CI    median      sample
                9.807E-1   1.075E-2   9.553E-1   9.965E-1   9.825E-1     5000
Bugs>stats(tau)
                  mean        sd       2.5% :  97.5%  CI    median      sample
                1.489E-1   3.079E-2   9.502E-2   2.163E-1   1.458E-1     5000
Bugs>stats(beta)
                  mean        sd       2.5% :  97.5%  CI    median      sample
                7.204E-1   1.183E-1   5.643E-1   1.026E+0   6.958E-1     5000
```