11-2009

Automated Likelihood Based Inference for Stochastic Volatility Models

H. Skaug

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

Automated Likelihood Based Inference for Stochastic Volatility Models

Hans J. SKAUG , Jun YU

November 2009

Paper No. 15-2009

Automated Likelihood Based Inference for Stochastic Volatility Models[*]

Hans J. Skaug[†] Jun Yu[‡]

October 7, 2008

Abstract: In this paper the Laplace approximation is used to perform classical and Bayesian analyses of univariate and multivariate stochastic volatility (SV) models. We show that implementation of the Laplace approximation is greatly simplified by the use of a numerical technique known as automatic differentiation (AD). Several algorithms are proposed and compared with some existing methods using both simulated data and actual data in terms of computational, statistical and simulation efficiency. It is found that the new methods match the statistical efficiency of the existing classical methods and substantially reduce the simulation inefficiency in some existing Bayesian Markov chain Monte Carlo (MCMC) algorithms. Also proposed are simple methods for obtaining the filtered, smoothed and forecasted latent variable. The new methods are implemented using the software AD Model Builder, which with its latent variable module (ADMB-RE) facilitates the formulation and fitting of SV models. To illustrate the flexibility of the new algorithms, several univariate and multivariate SV models are fitted using exchange rate data.

JEL Classification: C13, C22, E43, G13
Keywords: Laplace approximation, Automatic differentiation, Simulated maximum likelihood, Importance sampling, Bayesian MCMC.

[*]We would like to thank Essie Maasoumi and a referee for help comments. Skaug thanks his hosts at Singapore Management University for support and hospitality in a visit that took place in January 2007. Yu gratefully acknowledges financial support from the Ministry of Education AcRF Tier 2 fund under Grant No. T206B4301-RS. The computer code used in the paper, both raw and compiled, can be found at http://www.mysmu.edu/faculty/yujun/research.html.

[†]Department of Mathematics, University of Bergen, Johannes Brunsgate 12, 5008 Bergen, Norway; email: Hans.Skaug@math.uib.no

[‡]School of Economics, Singapore Management University, 90 Stamford Road, Singapore 178903; email: yujun@smu.edu.sg. URL: http://www.mysmu.edu/faculty/yujun/.

1 Introduction

Parameter estimation and statistical inference of stochastic volatility (SV) models has recently received a great deal of attention in the econometrics literature. One reason is that the likelihood function is expressed by a high dimensional integral which cannot be solved analytically due to the presence of a latent process. As a result, likelihood-based inferences are computationally demanding. While in the earlier literature some statistically inefficient but numerically simple methods have been proposed (see for example, Harvey et al., 1994, Melino and Turnbull, 1990, Andersen et al., 1996), the more recent literature shifts the focus towards developing full likelihood-based methods to analyze SV models. The reason for this shift of the focus is partly because the simulation based methods can produce more efficient parameter estimation and partly because computing facilities have rapidly improved.[1]

Among various full likelihood-based methods, two of them seem to have received the most attention. The first is the classical inferential method by maximizing a simulated likelihood. Contributions along this line of research include Danielsson and Richard (1993), Danielsson (1994), Shephard and Pitt (1997), Durbin and Koopman (1997), Sandmann and Koopman (1998), Liesenfeld and Richard (2006), Richard and Zhang (2006) and Durham (2006). The idea there is to evaluate the likelihood function numerically by integrating out the latent volatility process via importance sampling techniques, followed by maximization of the approximate likelihood function. The second is the Bayesian Markov chain Monte Carlo (MCMC) method. Contributions along this line of research include Jacquier, et al. (1994), Kim, et al. (1998), Meyer and Yu (2000), Meyer, et al. (2003), and Liesenfeld and Richard (2006). The idea there is to draw a sequence of correlated samples from posterior densities of unknown parameters and, possibly, the latent variables.[2]

Both likelihood-based methods are simulation-based and often used are specialized techniques. Many of the algorithms have been implemented in low level programming languages such as C++ or FORTRAN in order to increase the computational speed. For example, the SVPack implementation of Kim, et al. (1998) is based on C++ while the MCMC algorithm of Jacquier et al. (1994) and the importance sampler of Durham (2006) were implemented using FORTRAN. While the implementation of these specially tailored packages is numerically effi-

[1]An even more recent literature has focused on measuring variance from intra-day data using the so-called realized variance. See McAleer and Medeiros (2008) for a survey on the rapidly moving literature. Of course, if the intra-data are not available, the realized variance approach becomes infeasible.

[2]Other full likelihood-based methods can be found in two survey papers, Broto and Ruiz (2004) and Asai, McAleer and Yu (2006).

cient, the required effort for writing and debugging a program is usually major. In addition, to use the methods suggested in Kim et al. (1998) and Jacquier et al. (1994) one has to work out each full conditional density; in order to use the method of Durham (2006), one has to find the analytical expressions for the first and second order derivatives of the joint probability distribution of return and volatility, which are required for the Laplace approximation. To overcome some of these difficulties Meyer and Yu (2000) advocated using the all-purpose Bayesian software, Bayesian Inference Using Gibbs Sampling (BUGS), to implement MCMC estimation of SV models. It was shown that BUGS provides a flexible environment to estimate SV models.[3] However, being a Bayesian software, BUGS is not able to provide classical frequentist inference. Moreover, by augmenting the latent volatility into the parameter space, BUGS produces slowly mixing chains for SV models as the latent volatility is typically highly persistent (Meyer and Yu, 2000). Therefore, to achieve a satisfactory precision for parameter estimates, a large number of iterations is needed, increasing the computational cost.

One purpose of this paper is to develop several algorithms to perform classical *and* Bayesian likelihood-based analyses of SV models using automatic differentiation (AD), combined with the Laplace approximation. A caveat is to approximate the distribution of volatilities, conditional on returns, by a multivariate normal distribution. Based on the Laplace approximation, we also develop new methods for obtaining the filtered, smoother, and forecasted latent variable. We then demonstrate the ease by which univariate and multivariate SV models can be studied routinely using the latent variable module ADMB-RE of the software package *AD Model Builder* (Fournier, 2001). By combining AD with the Laplace approximation ADMB-RE provides a unified statistical software environment for dealing with nonlinear statistical models with latent random variables.[4] AD is a technique for calculating exact numerical derivatives of functions defined as computer algorithms (Griewank and Corliss, 1991), and should not be confused with symbolic differentiation as performed by for instance MATHEMATICA and MAPLE. As both the Laplace approximation and numerical likelihood optimization require derivatives, ADMB-RE facilitates parameter estimation and is an ideal software to do classical and Bayesian likelihood-based inference for non-Gaussian and non-linear state-space models.

Our paper is closely related to Meyer et al. (2003) where the likelihood function of the basic SV model was approximated by the Laplace approximation via AD. The focus of their paper is

[3]Since then a number of studies have used WinBUGS, the version of BUGS for the WINDOWS operating system, to estimate various specifications of univariate and multivariate SV models. Examples include Berg, et al. (2004), Lancaster (2004), Selçuk (2004), Yu (2005), Yu and Meyer (2006).

[4]ADMB-RE is available from http://otter-rsch.com/admbre/admbre.html.

to develop an efficient MCMC algorithm, however. Moreover, our method for approximating the likelihood function is different from theirs. Meyer et al. uses a Kalman filter approach, where a sequence of one-dimensional Laplace approximations is used to perform the one-step updates of the Kalman filter. We apply a single multivariate Laplace approximation jointly. While it is trivial to generalize the single multivariate Laplace approximation in the multivariate SV models, the same argument does not seem to apply to the sequential univariate normal approximation. Our work is also related to recent work by Liesenfeld and Richard (2006) where efficient importance sampling was used to perform a classical analysis and a Bayesian analysis of SV models, respectively. There are two major differences between our algorithms and theirs. First, for the classical analysis, different importance sampling techniques are used. Second, for the Bayesian analysis, to increase the simulation efficiency, Liesenfeld and Richard (2006) proposed to sample the entire volatility process as a single block whereas we use an integration sampler and hence avoid a need to sample the latent process. Finally, our paper is related to Durham (2006) where the same Laplace approximation is used for the importance density. His simulated maximum likelihood algorithm is the same as ours. However, we also propose to maximize the Gaussian density directly and a Bayesian method. In addition, while he advocates symbolic calculations of differentiation, we propose AD.

The rest of the paper is organized as follows. Section 2 introduces the Laplace approximation and discusses how to use it to perform classical maximum likelihood (ML) estimation. Section 3 discusses how to use the Laplace approximation to perform a Bayesian analysis. Section 4 explains how the Laplace approximation facilitates smoothing, predicting and filtering of the latent variable. Section 5 describes AD and the software ADMB-RE. In Section 6 we examine the relative performance of ADMB-RE using both simulated data and actual data. Section 7 demonstrates how more flexible univariate and multivariate SV models can be fitted under ADMB-RE, and Section 8 concludes. ADMB-RE code and file structure are described in more detail in two appendices.

2 Classical Likelihood-Based Methods

For illustrative purposes, we focus on the so-called basic SV model of Taylor (1982) which is defined by

$$\left\{ \begin{array}{lcl} X_t &=& \sigma_X e^{h_t/2}\epsilon_t,\ t = 1, \ldots, T, \\ h_{t+1} &=& \phi h_t + \sigma \eta_t,\ t = 1, \ldots, T-1, \end{array} \right. \tag{1}$$

where X_t is the return of an asset, $\epsilon_t \overset{iid}{\sim} N(0,1)$, $\eta_t \overset{iid}{\sim} N(0,1)$, $corr(\epsilon_t, \eta_t) = 0$, and $h_1 \sim N(0, \sigma^2/(1-\phi^2))$. The parameters of interest are $\theta = (\sigma_X, \phi, \sigma)'$. An alternative parametrization often used is $\alpha = 2(1-\phi)\ln\sigma_X$.

Let $\mathbf{X} = (X_1, \ldots, X_T)'$ and $\mathbf{h} = (h_1, \ldots, h_T)'$. The likelihood function of the basic SV model is given by

$$p(\mathbf{X};\theta) = \int p(\mathbf{X}, \mathbf{h};\theta) d\mathbf{h} = \int p(\mathbf{X}|\mathbf{h};\theta) p(\mathbf{h};\theta) d\mathbf{h}. \tag{2}$$

This is a high-dimensional integral which does not have a closed form expression due to the non-linear dependence of X_t on h_t.

To perform maximum likelihood (ML) estimation, one has to approximate the likelihood function. Following Skaug (2002), our first algorithm employs the Laplace approximation, that is, we match $p(\mathbf{X}, \mathbf{h};\theta)$ and a multivariate normal distribution for $\mathbf{h}$ as closely as possible (up to a constant proportion). More precisely, the Laplace approximation to the integral (2) is

$$p(\mathbf{X};\theta) \approx |\det(\Omega)|^{-1/2} p(\mathbf{X}, \mathbf{h}^*;\theta), \tag{3}$$

where

$$\mathbf{h}^* = \arg\max_{\mathbf{h}} \ \ln p(\mathbf{X}, \mathbf{h};\theta) \quad \text{and} \quad \Omega = \frac{\partial^2 \ln p(\mathbf{X}, \mathbf{h}^*;\theta)}{\partial\mathbf{h}\partial\mathbf{h}'}. \tag{4}$$

The Laplace approximation is exact when $p(\mathbf{X}, \mathbf{h};\theta)$ is Gaussian in $\mathbf{h}$. It typically works well when $p(\mathbf{h};\theta)$ is Gaussian or nearly Gaussian and $p(\mathbf{h};\theta)$ is more informative about $\mathbf{h}$ than $p(\mathbf{X}|\mathbf{h})$ is, in the sense of observed Fisher information. This is indeed the case for almost all the SV models used in practice. From an empirical viewpoint, the normality of $\mathbf{h}$ is documented as one of the stylized facts about volatility in Andersen et al. (2001). Theoretical results about the accuracy of the Laplace approximation, easily applicable in the present context, are lacking in the literature. It is worth noting that the accuracy of the approximation may vary with the parameter θ. It is interesting to investigate the accuracy of the approximation for problem at hand and this will be done in Section 4.

For SV models $\mathbf{h}^*$ does not have a closed form expression. To find $\mathbf{h}^*$, a numerical optimization method, such as Newton's method, is needed. While Ω may have a closed form expression, it is typically tedious and error prone to derive this expression by hand. Durham (2006) suggests using symbolic programs to calculate Ω. In this paper, we will evaluate Ω using AD.

In the case where the Laplace approximation does not work satisfactorily, one can improve the approximation by using importance sampling, which is a Monte Carlo approach to high-dimensional numerical integration. The likelihood can be written as

$$p(\mathbf{X};\theta) = \int p(\mathbf{X},\mathbf{h};\theta)d\mathbf{h} = \int \frac{p(\mathbf{X},\mathbf{h};\theta)}{q(\mathbf{h})}q(\mathbf{h})d\mathbf{h}, \tag{5}$$

where $q(\mathbf{h})$ is the importance density. The idea of our second algorithm is to draw samples $\mathbf{h}^{(1)},\ldots,\mathbf{h}^{(S)}$ from $q(\mathbf{h})$ so that we can approximate $p(\mathbf{X};\theta)$ by

$$\frac{1}{S}\sum_{i=1}^{S}\frac{p(\mathbf{X},\mathbf{h}^{(i)};\theta)}{q(\mathbf{h}^{(i)})}. \tag{6}$$

In general, $\mathbf{h}^{(i)}$ depends on $\mathbf{X}$ and θ. The law of large numbers ensures the convergence of (6) to $p(\mathbf{X};\theta)$ as $S\to\infty$.

For the method to be computationally efficient, we should match $p(\mathbf{X},\mathbf{h};\theta)$ and $q(\mathbf{h})$ as closely as possible while ensuring that it is still easy to sample from $q(\mathbf{h})$. To do that, following Shephard and Pitt (1997) and Durbin and Koopman (1997), we choose q to be the Laplace approximation to $p(\mathbf{X},\mathbf{h};\theta)$. That is $\mathbf{h}^{(s)}\sim N(\mathbf{h}^*,-\Omega^{-1})$ where $\mathbf{h}^*$ and Ω are from (4). The approach is termed simulated maximum likelihood (SML), and it should be noted that $q(\mathbf{h})$ depends on θ through $\mathbf{h}^*$ and Ω. Skaug and Fournier (2006) provide a formula for the gradient of the likelihood approximation (6) in the case that the importance density is the Laplace approximation.

While the importance sampling procedure used in the present paper is based on the Laplace approximation, other important sampling techniques exist in the literature. For example, the importance sampler developed by Richard and Zhang (2006), Liesenfeld and Richard (2006) is based on a particular factorization of the importance density. Lee and Koopman (2005) compared the performance of the two importance samplers in the context of univariate SV models and found the two methods to perform similarly well.

3 A Bayesian MCMC Method

The two algorithms discussed in Section 2 are the classical ML methods. We now discuss how the Laplace approximation can be employed to perform a Bayesian MCMC analysis. The goal of MCMC methods is to sample from posterior densities. There are a number of different ways in which Bayesian MCMC can be applied to SV models. The first one samples from the posterior $p(\theta|\mathbf{X})\propto p(\mathbf{X}|\theta)p(\theta)$. However, since the marginal likelihood $p(\mathbf{X}|\theta)$, given in terms of an integral, does not have a closed form expression, standard Bayesian analysis is not trivial. Alternatively, one can augment the parameter vector by the vector of latent variables and obtain the joint posterior $p(\theta, \mathbf{h}|\mathbf{X}) \propto p(\mathbf{X}, \theta|\mathbf{h})p(\mathbf{h}|\theta)p(\theta)$. Consequently, evaluation of $p(\mathbf{X}|\theta)$ becomes unnecessary. From the joint posterior, one can find the marginal distribution $p(\theta|\mathbf{X})$ to make inference about the model parameters and the marginal distribution $p(\mathbf{h}|\mathbf{X})$ to make inference about the log-volatilities. MCMC can be used to draw (correlated) samples from the high dimensional $(T + p)$ posterior density.

In the basic SV model, one way of sampling σ_X, ϕ, σ and $\mathbf{h}$ is to update each element of σ_X, ϕ, σ and each element in $\mathbf{h}$ one at a time (i.e. a single-mover). This so-called Gibbs sampling algorithm was suggested by Shephard (1993) and Jacquier et al. (1994). For SV models, the consecutive states are often highly dependent, rendering inefficient mixing and slow convergence of the Markov chain to the equilibrium distribution. To improve the simulation efficiency, Shephard and Pitt (1997), Kim et al. (1998) and Liesenfeld and Richard (2006) suggested MCMC methods which sample the vector $\mathbf{h}$ in a single block (i.e. a multi-mover).

In the present paper, we suggest an alternative Bayesian MCMC algorithm which achieves high simulation efficiency. The idea is as follows. First, an approximation to the likelihood function $p(\mathbf{X}|\theta)$ is obtained via the Laplace approximation (3), making augmentation of the parameter vector redundant. Second, we use the MH algorithm, developed by Metropolis et al. (1953) and Hastings (1970), to obtain a MCMC sample from the posterior of θ. To use the MH algorithm we first fit the model by maximizing the approximated marginal likelihood (3). Denote by $\hat{\theta}$ the resulting estimate of θ, and by $\hat{\Sigma}_{\hat{\theta}}$ its covariance matrix based on the observed Fisher information. The MH-proposal density is taken to be a multivariate normal, centered at the current parameter value, with covariance matrix $\hat{\Sigma}_{\hat{\theta}}$. Note further that for each value of θ proposed by the MH-algorithm the Laplace approximation is invoked via equation (3). While our algorithm is strongly related to that developed in Meyer et al. (2003), it differs in that Meyer et al. used a sequential Laplace approximation.

As the posterior sampling is done in a much lower dimensional space in the proposed integration sampler, it is natural to expect several advantages of our method relative to the MCMC algorithms that require sampling of the latent process. First, the integration sampler should have higher simulation efficiency. Second, while it is typically difficult to check the convergence of simulated chains of latent variables as the sample size T grows, it is trivial to do so in our integration sampler, regardless of the sample size.

A major advantage of the MCMC algorithms that sample the latent process is that they provide an integrated framework for parameter estimation and latent variable smoothing. Indeed, the smoothed latent variable $E(h_t|\mathbf{X})$, and the associated posterior variance, is a by-product of the MCMC output (Jacquier et al. 1994). There are two sources of variation contributing to the posterior uncertainty in $\mathbf{h}$: uncertainty in $\mathbf{h}$ given θ and uncertainty about θ. It is worth pointing out that MCMC, by providing samples from $p(\mathbf{h}|\mathbf{X})$, directly accounts for both sources of uncertainty.

4 Smoothing, Predicting and Filtering Latent Variables

While MCMC can trivially provide the smoothed estimate of unobserved volatility, a separate algorithm is needed for filtration. For example, Kim, Shephard and Chib (1998) appended a particle filter to the MCMC parameter estimates. Many other moment- and simulation-based approaches, also require a special purpose filtration procedure to estimate $\mathbf{h}$. For instance, Gallant and Tauchen (1998) propose a "reprojection" procedure to be applied after the EMM estimator has been obtained. Durham (2006) append a particle filter (Kitagawa, 1996 or Pitt and Shephard, 1999) to the ML estimates. In this section, we develop several algorithms for obtaining filtered, smoothed and forecasted estimate of the latent variable, all based on the Laplace approximation. A major advantage of the proposed algorithms is the ease of implementation, requiring little programming effort, and low computational cost. This is in the sharp contrast to the filtration methods available in the literature.

We first propose the algorithm for obtaining the smoothed estimate of the latent variable, with and without taking into account the two sources of error discussed above. Suppose $\hat{\theta}$ is the ML estimate of θ. A natural smoother of $\mathbf{h}$ is the empirical Bayes estimator

$$\hat{\mathbf{h}}^s = \arg\max_{\mathbf{h}} \ln p(\mathbf{X}, \mathbf{h}; \hat{\theta}). \tag{7}$$

That is, we use the mode of the conditional density $p(\mathbf{h}|\mathbf{X})$ to estimate $\mathbf{h}$, conditionally on $\mathbf{X}$.[5] This is because $\ln p(\mathbf{X}, \mathbf{h}) = \ln p(\mathbf{h}|\mathbf{X}) + \ln p(\mathbf{X})$, and the latter quantity does not depend on $\mathbf{h}$. It should be pointed out that the smoother (7) comes as a by-product of the Laplace approximation.

By ignoring errors in parameter estimation, we get the approximate covariance matrix of the smoothed estimate of the latent variables,

$$-\left[\frac{\partial^2 \ln p(\mathbf{X}, \hat{\mathbf{h}}^s; \hat{\theta})}{\partial \mathbf{h} \partial \mathbf{h}'}\right]^{-1}. \tag{8}$$

[5]Filtered and smoothed values are typically computed as the means of the corresponding conditional distributions. When the conditional distribution is symmetric, the mean and the mode are identical. A nice feature about the mode is that is robust to nonlinear transformations.

A variance formula[6] that takes the uncertainty in $\hat{\theta}$ into account is

$$-\left[\frac{\partial^2 \ln p(\mathbf{X}, \hat{\mathbf{h}}^s; \hat{\theta})}{\partial \mathbf{h} \partial \mathbf{h}'}\right]^{-1} + \frac{\partial \hat{\mathbf{h}}^{\mathbf{s}}}{\partial \theta} \Sigma_{\hat{\theta}} \left(\frac{\partial \hat{\mathbf{h}}^s}{\partial \theta}\right)', \tag{9}$$

where a formula for the Jacobian $\partial \hat{\mathbf{h}}^s / \partial \theta$ ($n \times q$ matrix) is given in Skaug and Fournier (2006). The formula (9) may be derived at in different ways. In a Bayesian framework where we think of both $\mathbf{h}$ and θ as random quantities, (9) arises as an approximation to the well known variance formula $\mathtt{Var}(\mathbf{h}) = E\left[\mathtt{Var}(\mathbf{h}|\theta)\right] + \mathtt{Var}\left[E(\mathbf{h}|\theta)\right]$.

The smoothed estimate of $\mathbf{h}$ can be used to generate model diagnostics. For example, an estimate of η_t is given as $(\hat{h}^s_{t+1} - \hat{\phi}\hat{h}^s_t)/\hat{\sigma}$. If the model specification is adequate, the estimated η_t should be approximately normal and be serially uncorrelated. Similarly, one can estimate ϵ_t by $y_t \exp(-0.5\hat{h}^s_t)/(\hat{\sigma}_X)$. Again, if the model specification is adequate, the empirical distribution of the estimated ϵ_t should be comparable to a normal distribution.

Prediction of future values of the latent variable can easily be done within this framework by extending the $\mathbf{h}$ vector with as many time periods as needed. If we want a K-step prediction we take $\mathbf{h} = (h_1, \ldots, h_T, h_{T+1}, \ldots, h_{T+K})'$. For the augmented $\mathbf{h}$ vector, the optimization problem (7) yields both the smoothed values $\hat{h}^s_1, \ldots, \hat{h}^s_T$ and the predicted values $\hat{h}^s_{T+1}, \ldots, \hat{h}^s_{T+K}$. Under the AR(1) structure for the latent variable, it is clear that $\hat{h}^s_{T+K} = \phi^K \hat{h}^s_T$, $K > 0$, but for more general latent processes an explicit expression may not exist.

It seems natural that θ is estimated using an un-augmented $\mathbf{h}$, and that prediction is conducted after the model has been fit. However, we shall argue that from a practical perspective it is convenient to use the augmented $\mathbf{h}$ also during the estimation phase. The reason is that no extra programming is required in order to obtain the prediction of $(h_{T+1}, \ldots, h_{T+K})$ and its covariance matrix, which is given by (9). As long as K is small compared to T the computational overhead is negligible. But we need to argue that augmenting $\mathbf{h}$ does not change the Laplace approximation (3) of the likelihood, and hence not the estimate of θ. This follows from general properties of the Laplace approximation. To see why, consider the following re-parametrization of the latent variables: $\mathbf{u} = (h_1, \ldots, h_T, \eta_T, \ldots, \eta_{T+K-1})'$, where $\eta_t = h_{t+1} - \phi h_t$ for $t = T, \ldots, T+K-1$. This is a linear transformation, and it can be proved that the Laplace approximation (3) is invariant under a one-to-one linear transformation of the latent variables. Further, $\mathbf{u_1} = (h_1, \ldots, h_T)'$ and $\mathbf{u}_2 = (\eta_T, \ldots, \eta_{T+K-1})'$ are independent, also when we condition on $\mathbf{X}$. For a conditional distribution with the property that

[6]We are grateful to D. Fournier for making us aware of this formula

$p(\mathbf{u}|\mathbf{X}) = p_1(\mathbf{u}_1|\mathbf{X})p_2(\mathbf{u}_2|\mathbf{X})$ we have a corresponding factorization of the Laplace approximation (3). But, since $p_2(\mathbf{u}_2|\mathbf{X}) = p(\eta_T, \ldots, \eta_{T+K-1})$ is Gaussian the Laplace approximation of $\int p_2(\mathbf{u}_2)d\mathbf{u}_2 = 1$ exactly, and it follows that inclusion of $\eta_T, \ldots, \eta_{T+K-1}$ does not affect the likelihood of θ.

Another important quantity is the filtered latent variable, that is the estimate of h_t conditional on $X_1, \ldots, X_t$, given the parameter value θ. Numerous nonlinear filtering algorithms have been proposed. In this paper, we propose an alternative filtered estimate of h_t based on the Laplace approximation. In particular, we define

$$(\hat{h}_1^f, \ldots, \hat{h}_t^f)' = \arg\max_{h_1,\ldots,h_t} \ln p(X_1, \ldots, X_t, h_1, \ldots, h_t; \hat{\theta}), \tag{10}$$

where $\hat{\theta}$ is the estimate of θ obtained from the entire sample $\{X_t\}_{t=1}^T$. The filtered estimate of h_t is $\hat{h}_t^f$. Obviously, it involves recursive Laplace approximations but is straightforward to program after the full-sample based estimation is complete. In Section 6 we will examine the accurate of the proposed filtration procedure with the particle filter of Kitagawa (1996).

5 Automatic Differentiation and ADMB-RE

The need for calculating derivatives arises in two places in the proposed algorithms. First, numerical solution of the optimization problem (4) is greatly simplified if the exact gradient of the objective function is available. Second, as noted above, the Hessian matrix (4) occurring in the Laplace approximation is typically too complicated to be evaluated by hand.

Automatic differentiation (AD) is a technique to numerically evaluate the derivatives of a function specified by a computer program (Griewank and Corliss, 1991). It exploits the fact that every computer program, no matter how complicated, executes a sequence of elementary arithmetic operations such as additions or elementary functions. By applying the chain rule repeatedly to these operations, derivatives of arbitrary order can in principle be computed automatically, and accurate to computer precision.

Two classical ways of evaluating derivatives are symbolic differentiation and the method of finite differences. While both AD and symbolic differentiation utilize the chain rule, it is important to emphasize differences between the two methods. The result of symbolic differentiation is a mathematical formula, while AD is aimed at producing numerical derivatives. Clearly, the mathematical derivative formula may be evaluated numerically, so there does not seem to be much to be gained. However, the real distinction is that AD may be applied to whole computer programs. Application of symbolic differentiation here is possible at a sub-expression level, but would involve quite a bit of manual labor. Two important drawbacks with finite differences are round-off errors in the discretization process, and cancelation. These problems multiply when calculating higher order derivatives. Automatic differentiation calculates exact (to machine precision) numerical derivatives and hence is not subject to the same problems.

Two different approaches to AD exist, the forward mode and the reverse mode. They differ in how the chain rule is used to propagate derivatives through the computation. The forward mode propagates derivatives of intermediate variables with respect to the independent variables. In contrast, the reverse mode of AD propagates derivatives of the final result with respect to all intermediate variables in the program. The program is first evaluated without derivative calculations, and subsequently derivatives are calculated in reverse order of the original program flow. During the latter phase the program must keep track of the values of all intermediate variables that impacts the final result, and thus the reverse algorithm may be very memory consuming (Griewank and Corliss, 1991).

AD Model Builder (ADMB) is a statistical modeling framework based on C++ that combines AD with a quasi-Newton function minimizer (Fournier 2001). An example of an ADMB program is given in Figure 1. When compiled and executed, the program will minimize the objective function (the negative log-likelihood) with respect to the independent parameters, i.e. those variables defined in the first part of the `PARAMETER_SECTION`. The use of AD to calculate the gradient of the objective function makes ADMB fast and numerically stable.

The latent variable module ADMB-RE, usually referred to as the random effects module, allows the Laplace approximation (3) to be calculated automatically, i.e. with the Hessian matrix in (4) calculated by automatic differentiation. The keyword `random_effects_vector` (see Figure 1) instructs the compiler that the vector $\mathbf{h}$ should be the target for the Laplace approximation. The objective function in the program must be taken to be $g(\mathbf{h}, \theta) = -\log\left[p(\mathbf{X}|\mathbf{h}, \theta)p(\mathbf{h}|\theta)\right]$. In order to facilitate numerical optimization of the log-likelihood based on (3), ADMB-RE calculates third order mixed derivatives of $g(\mathbf{h}, \theta)$ by automatic differentiation (Skaug and Fournier, 2006). Optionally, ADMB-RE employs the importance sampling approximation (6) with the density q chosen as explained at the end of Section 2. Appendix A provides some ADMB commands, the file structure and sample ADMB code for the basic SV model.

6 Performance of Proposed Algorithms

For the purposes of illustration and comparison, we first use a widely used dataset to fit the basic SV model. The dataset consists of 945 observations on daily pound/dollar exchange rate

```
DATA_SECTION
  init_int n                    // Length of time series
  init_vector X(1,n)            // Time series
PARAMETER_SECTION
  init_number phi
  init_number sigma
  init_number sigma_x
  random_effects_vector h(1,n)  // Vector of latent variables
  objective_function_value g
PROCEDURE_SECTION
  int i;                        // Local variables
  double log_2pi = 0.9189385;
  dvariable phi2 = square(phi);

  // Contribution from log[p(h_1)]
  g -= -0.5*log_2pi -log(sigma) + 0.5*log(1-phi2)
          - 0.5*square(h(1)/sigma)*(1-phi2);

  // Contribution from log[p(h_t|h_{t-1})]
  for (i=2;i<=n;i++)
    g -=  -0.5*log_2pi  -log(sigma)
            -.5*square((h(i)-phi*h(i-1))/sigma);

  // Contribution from log[p(X_t|h_t)]
  for (i=1;i<=n;i++)
  {
    dvariable sigma_X = sigma_x*exp(0.5*h(i));
    g -= -0.5*log_2pi -log(sigma_X) - 0.5*square(X(i)/sigma_X);
  }
```

Figure 1: AD Model Builder code for the basic SV model (1).

Method (Software)	Parameter	Estimate	SE	IACT	CPU(s)	Log-Lik
Classical Methods						
	ϕ	0.9743	0.0122		9.78	918.791
LA-ML	σ	0.1697	0.0363			
(ADMB-RE)	σ_X	0.6330	0.0688			
	ϕ	0.9748	0.0122		14.53	918.669
SML	σ	0.1687	0.0355			
(ADMB-RE)	σ_X	0.6337	0.0697			
	ϕ	0.9734	0.0120		21.60	917.458
SML	σ	0.1726	0.0360			
(MATLAB)	σ_X	0.6305	0.0694			
	ϕ	0.9743	0.0120		18.73	918.636
SML	σ	0.1724	0.0370			
(GAUSS)	σ_X	0.6300	0.0680			
Bayesian Methods						
	ϕ	0.9747	0.0134	10.3	67	
MCMC	σ	0.1717	0.0374	8.7		
(ADMB-RE)	σ_X	0.6526	0.1267	24.9		
	ϕ	0.9769	0.0109	133.7	3.2	
MCMC	σ	0.1611	0.0314	255.3		
(WinBUGS)	σ_X	0.6538	0.1360	10.8		

Table 1: Parameter estimates and standard errors for the basic SV models.

from 01/10/1981 to 28/06/1985. The same data were used in Harvey et al. (1994), Shephard and Pitt (1997), Meyer and Yu (2000), and Meyer et al. (2003).

Estimates and (asymptotic) standard errors of σ_X, ϕ and σ are given in panels 1, 2, 5 of Table 1, corresponding, respectively, to the Laplace approximation based maximum likelihood method in (3) (termed LA-ML) and the simulated maximum likelihood method in (6) (termed SML), and the integration MCMC sampler. For the integration sampler we use the independent flat priors for all three parameters and obtain 11,000 iterations with the first 1,000 discarded. Panel 3 reports the estimates and asymptotic standard errors of the simulated ML of Shephard and Pitt (1997) implemented in MATLAB. Panel 4 reports the estimates and asymptotic standard errors of the simulated ML of Richard and Zhang (2006) implemented in GAUSS. For all the simulated ML methods, we select $S = 64$. In the last panel we report the single-move MCMC estimates (posterior means) and standard errors (posterior standard errors) using the BUGS program of Meyer and Yu (2000) based on 110,000 iterations with the first 10,000 dis-

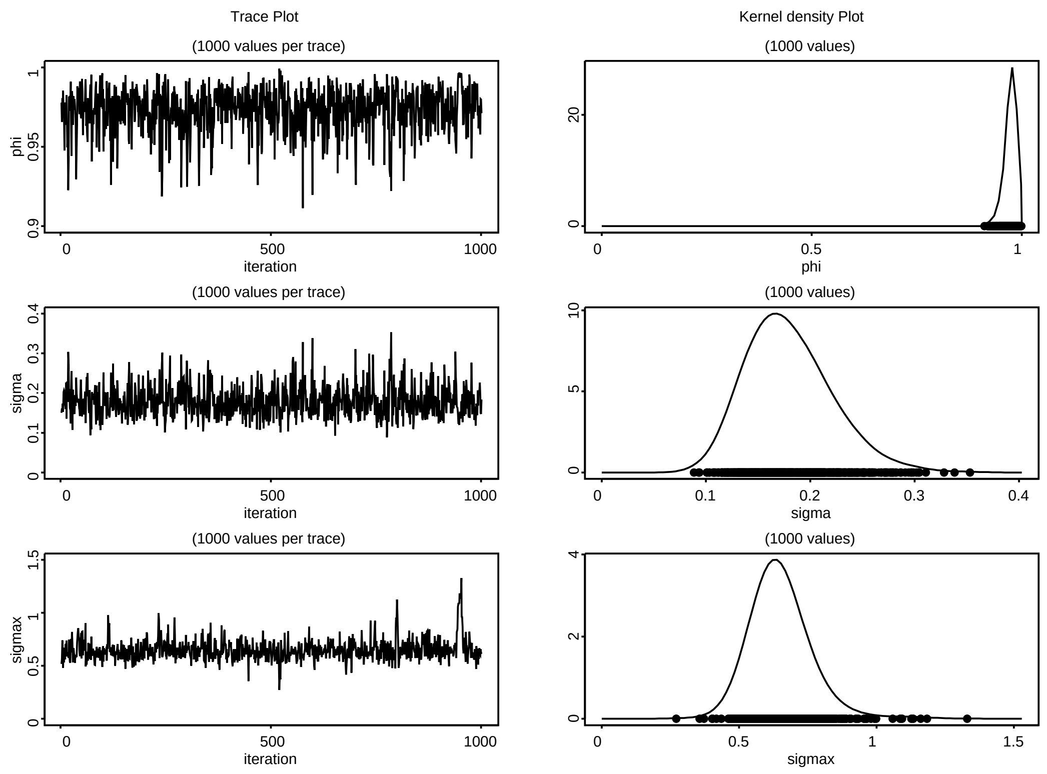

Figure 2: Laplace approximation based integration sampler for the Pound/Dollar exchange rate. Left panels show simulations against iteration. Right panels show marginal densities of posterior distributions.

carded. Also reported in Table 1 is the CPU time on a Pentium IV 3.2 GHz PC running on WINDOWS XP. For the classical methods, the reported CPU time is time-to-convergence of the optimizer, while for MCMC methods the CPU time is based on 100 iterations of the sampler. Finally, for the classical methods, we report the log-likelihood value. For the MCMC methods, we report the integrated autocorrelation times (IACT) of Sokal (1996) for measuring simulation inefficiency. Figure 2 shows every 10th iterations and the corresponding marginal densities for the Laplace-based MCMC sampler. This is compared to Figure 2 of Meyer and Yu (2000). All the chains reported have passed the Heidelberger and Welch stationarity and halfwidth tests.

It can be seen that the three sets of estimates from our algorithms are very close to each

	$T = 500$			$T = 2000$		
	α	ϕ	σ_X	α	ϕ	σ_X
True Value	-0.736	0.90	0.363	-0.736	0.90	0.363
Bias						
QML	-0.70	-0.09	0.09	-0.117	-0.02	0.20
GMM	0.12	0.02	-0.12	0.15	0.02	-0.08
EMM	-0.17	-0.02	0.02	-0.57	-0.07	-0.004
AML	-0.15	-0.02	0.02	-0.039	0.005	0.005
MCMC	-0.13	-0.02	-0.01	-0.026	-0.004	-0.004
N-ML	-0.13	-0.02	0.01	NA	NA	NA
MCL	0.14	0.0	0.01	NA	NA	NA
LA-ML	-0.248	-0.033	0.025	-0.058	-0.008	0.0018
SML	-0.130	-0.020	0.012	0.024	0.003	-0.016
RMSE						
QML	1.60	0.22	0.27	0.46	0.06	0.11
GMM	0.59	0.08	0.17	0.31	0.04	0.12
EMM	0.60	0.08	0.20	0.224	0.030	0.049
AML	0.42	0.06	0.08	0.173	0.023	0.043
MCMC	0.34	0.05	0.07	0.15	0.02	0.034
N-ML	0.43	0.05	0.08	NA	NA	NA
MCL	0.27	0.04	0.08	NA	NA	NA
LA-ML	0.632	0.085	0.099	0.195	0.026	0.043
SML	0.439	0.057	0.081	0.144	0.019	0.038

Table 2: Comparison of different estimation methods for the basic SV model.

other. Moreover, the estimates and standard errors obtained from our algorithms are very similar to those from the existing methods. Perhaps the most striking result is found in LA-ML, which provides almost identical estimates, standard errors and log-likelihood value to the other ML methods, indicating that the approximation error for the Laplace approximation is very small in this case. Since LA-ML is not simulation-based, not surprisingly it is the fastest to obtain, taking less than 10 seconds of CPU time.

While it is not fair to compare the computational time from different programming languages, the computational time is nonetheless indicative of computational cost. Among the four classical methods, the computational cost is similar. However, the programming effort in MATLAB and GAUSS is more much involved than that in ADMB-RE. For example, to implement SML of Shephard and Pitt (1997) in MATLAB, we had to find the first and second derivatives of the joint density of $\mathbf{X}$ and $\mathbf{h}$. An extension to the basic model requires even more serious effort. On the other hand, extensions to the basic model is trivial in ADMB-RE, as we will show in Section 7.

Between the two MCMC methods, our integration sampler is more time consuming than the single mover of BUGS (67 seconds versus 3.2 seconds). This is not surprising because the latent variable is marginalized via the Laplace approximation in our technique. However, the integration sampler substantially increases the simulation efficiency. In particular, IACT reduces to 10.3 from 133.7 for ϕ and 8.7 from 255.3 for σ. If we are concerned about the precision in the estimate of σ, for example, the precision achieved with 1 iteration from our integration sampler matches that achieved with $30(\approx 255.3/8.7)$ iterations from the single-mover. Obviously, obtaining 1 iteration from our integration sampler is faster than obtaining 30 iterations from the single-mover of BUGS. The better mixing in the integration sampler can be clearly seen from Figures 2, as opposed to that in Figure 2 in Meyer and Yu (2000).

To compare our multivariate marginal likelihood approximation (3) to the sequential univariate marginal likelihood approximation of Meyer et al. (2003) we include in the model the prior on $\theta = (\phi, \sigma, \sigma_X)'$ used by Meyer et al. The prior specifies independence for the components of θ, and that $\mu = 2\log(\sigma_X) \sim N(0, 10)$, $\phi^* = (\phi + 1)/2 \sim \text{Beta}(20, 1.5)$ and $\sigma^2 \sim \text{IG}(2.5, 0.025)$, where IG denotes an inverse gamma distribution. We then maximize $p(\mathbf{X}; \theta)p(\theta)$, where $p(\mathbf{X}; \theta)$ is given by (3) and $p(\theta)$ is the prior, to obtain estimates that are directly comparable to the posterior mode estimates of Meyer et al. (2003). These are $(\phi, \sigma, \sigma_X) = (0.9807, 0.1399, 0.6428)$ with our approximation, and $(\phi, \sigma, \sigma_X) = (0.9830, 0.1445, 0.7932)$ taken from Section 4 of Meyer et al. (2003). It is worth noting that our estimate of σ_X is much closer to that of the alternative methods reported in Table 1 than to that of Meyer et al. (2003). While not reported, we have found the IACTs for the two methods are comparable.

To investigate the statistical efficiency of the proposed algorithms, we simulate 500 and 2000 observations, respectively, from the basic SV model with $(\sigma, \phi, \alpha) = (0.363, 0.9, -0.736)$ where $\alpha = 2(1 - \phi)\ln(\sigma_X)$. The basic SV model is then fitted to each simulated sequence using LA-ML and SML (with $S = 128$). Each experiment is repeated 500 times to obtain the bias and the root mean square error (RMSE). The same parameters setting and Monte Carlo design have been adopted in numerous papers in the literature, for example, Jacquier et al. (1994), Andersen et al. (1999) and Bates (2006). Table 2 reports the results and appends to Andersen et al.'s Table 5 and Bates' Table 9.

First, our Laplace approximation based maximum likelihood method provides accurate estimates. In the case where the sample size is 500, it is clearly more efficient than QML. Also it

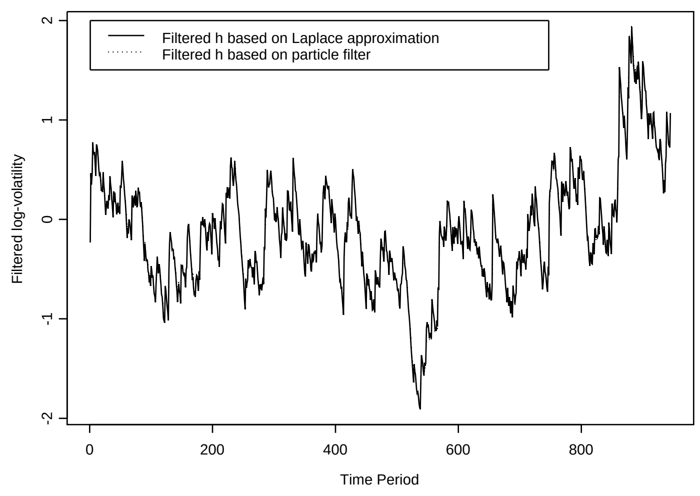

Figure 3: Filtered log-volatilities based on the Laplace approximation (10) for the basic SV model. The corresponding estimates based on the particle filter of Kitagawa (1996) are indistinguishably from those based on (10).

appears more efficient than GMM and EMM in terms of σ_X. In the case where the sample size is 2000, it outperform QML, GMM and EMM and performs nearly as well as AML. Second, our simulated maximum likelihood method stands out as one of the most efficient estimation procedures in both cases. When the sample size is 2000, it provides the most efficient estimate for α and ϕ. We also experiment with larger values for S and the results remain qualitatively unchanged.

In the final comparison, we obtain filtered time series values of $\mathbf{h}$ in the basic SV model using two methods, the Laplace approximation (10) and the particle filter of Kitagawa (1996). Figure 3 plots the two sequences. It is seen that the two filtered series are almost identical, suggesting that the Laplace approximation based filter works well.

7 Flexible Modeling

A major difficulty in most existing algorithms and computing softwares is that any change in the model specification entails substantial effort in careful algebraic derivations (such as finding the full-conditional distributions and finding the expressions of the first two derivatives) followed by

	Basic SV	SV with Leverage	SV-t	SV Skewed-t
		Laplace approximation		
ϕ	.974(.012)	.975(.013)	.979(.011)	.979(.011)
σ	.170(.036)	.168(.038)	.147(.037)	.151(.036)
σ_X	.633(.069)	.631(.070)	.613(.073)	.640(.075)
ν			22.73(18.14)	25.14(22.42)
λ				-.06
ρ		-.003(.026)		
CPU time	9.78	50.17	72.38	96.70
Log-lik	-918.79	-918.78	-918.05	-917.43
		Importance sampling ($N = 128$)		
ϕ	.974(.014)	.974(.014)	.978(.015)	.978(.011)
σ	.174(.037)	.173(.039)	.153(.038)	.157(.038)
σ_X	.631(.069)	.631(.070)	.613(.072)	.639(.075)
ν			24.25(20.97)	27.07(26.39)
λ				-.06(.05)
ρ		-.003(.026)		
CPU time	38.55	167.08	209.06	530.02
Log-lik	-918.40	-918.38	-917.75	-917.11

Table 3: Parameter estimates of two classical methods for the univariate SV models fitted to the Pound/Dollar exchange rates. The numbers in brackets are the asymptotic standard errors.

serious effort in coding and debugging. One exception is BUGS, as demonstrated in Meyer and Yu (2000) and Yu and Meyer (2006) in the context of SV models. As in BUGS, modification of the model is straightforward in ADMB-RE and usually only involves changing a few lines of code. To illustrate the strength and flexibility of ADMB-RE, we consider some variations over the basic SV model and one multivariate SV model. Appendix B provides details about the modifications of the ADMB-RE code.

First, we consider three univariate SV models, namely, SV-t model, SV-skewed-t model, and SV model with the leverage effect. All three models are fitted using the Pound/Dollar exchange rate series.

The so-called SV-t model is obtained by assuming that ϵ_t in the basic SV model (1) has a Student-t distribution with $\nu > 2$ degrees of freedom. As a further extension we allow ϵ_t to have a skewed Student-t distribution with density

$$f(t) = \begin{cases} bc\left(1 + \left(\frac{1}{\nu-2}\left(\frac{bt+a}{1-\lambda}\right)^2\right)\right)^{-(\nu+1)/2} & \text{if } t < -a/b, \\ bc\left(1 + \left(\frac{1}{\nu-2}\left(\frac{bt+a}{1+\lambda}\right)^2\right)\right)^{-(\nu+1)/2} & \text{if } t \geq -a/b, \end{cases} \tag{11}$$

where

$$a = 4\lambda c\left(\frac{\nu-2}{\nu-1}\right), \quad b = \sqrt{1 + 3\lambda^2 - a^2}, \quad c = \Gamma((\nu+1)/2)/(\sqrt{\pi(\nu-2)}\Gamma(\nu/2)).$$

Here, $\nu > 2$ still has the interpretation as being the degrees of freedom, while $-1 < \lambda < 1$ controls the degree of asymmetry. For $\lambda = 0$ we get the ordinary $t(\nu)$ distribution, and $\lambda < 0$ gives a negatively skewed distribution (i.e. the left tail is longer than the right tail); see Hansen (1994) for more details about the skewed t-distribution. Both ν and λ are parameters to be estimated, and are hence included in θ.

In the third variation of the basic SV model, we introduce a leverage effect into the basic SV model (1), by assuming that ϵ_t, η_t are iid $N(0,1)$ and $\text{corr}(\epsilon_t, \eta_t) = \rho$. A reparameterization of the model yields (Yu, 2005)

$$\begin{cases} X_t &= \frac{\sigma_X}{\rho\sigma}\exp(0.5h_t)(h_{t+1} - h_t) - \frac{\sigma_X}{\rho}\exp(0.5h_t)\sqrt{1-\rho^2}w_t, \\ h_{t+1} &= \phi h_t + \sigma\eta_t, \end{cases} \tag{12}$$

where η_t, w_t are iid $N(0,1)$ and $\text{corr}(\eta_t, w_t) = 0$.

Table 3 reports the estimates, asymptotic standard errors, CPU time and log-likelihood value for the three univariate models, obtained from the two ML methods.[7] For SML, we choose $S = 128$. The results for the basic SV model are reported for the purpose of comparison. First, it can be seen that all three models can be quickly estimated by LA-ML, with the resulting estimates and log-likelihood being almost identical to those generated by SML. The CPU time difference between these two methods increases with the complexity of estimated models. Second, both estimation procedures suggest that the model extensions do not improve the fit over the basic

[7]While the MCMC method is readily applicable, we only report the ML results to save space. We also calculated the Monte Carlo standard errors of the MLE estimates by repeating the estimation under different common random numbers. The Monte Carlo standard errors are small, indicating that the proposed method is reliable. To save space, we do not report the Monte Carlo standard errors.

	Σ			μ	Φ			Σ_η		
				Laplace approximation						
	1	-0.474	-0.24	-0.918	0.974	0	0	0.029	0	0
$\widehat{\theta}$	-0.474	1	0.377	-1.359	0	0.916	0	0	0.052	0
	-0.24	0.377	1	-1.938	0	0	0.913	0	0	0.161
	0	0.019	0.015	0.217	0.012	0	0	0.012	0	0
SE($\widehat{\theta}$)	0.019	0	0.016	0.101	0	0.033	0	0	0.023	0
	0.015	0.016	0	0.158	0	0	0.025	0	0	0.048
CPU time:	711.25	Log-lik:	-2183.12							
				Importance sampling ($N = 128$)						
	1	-0.474	-0.239	-0.921	0.974	0	0	0.03	0	0
$\widehat{\theta}$	-0.474	1	0.377	-1.354	0	0.923	0	0	0.046	0
	-0.239	0.377	1	-1.935	0	0	0.92	0	0	0.147
	0	0.019	0.015	0.217	0.012	0	0	0.013	0	0
SE($\widehat{\theta}$)	0.019	0	0.016	0.103	0	0.029	0	0	0.019	0
	0.015	0.016	0	0.162	0	0	0.023	0	0	0.041
CPU time:	4026.47	Log-lik:	-2184.10							

Table 4: Parameter estimates ($\widehat{\theta}$) and asymptotic standard errors (SE($\widehat{\theta}$)) for the multivariate SV model (13) fit with ADMB-RE.

model, judged by the marginal increase in the log-likelihood value. Moreover, in the SV-t model, the estimated degrees of freedom, ν, is very large, suggesting little evidence of fat-tails in ϵ_t. In the SV-skewed-t model, the estimated asymmetry parameter, λ, is very close to zero (-0.06) and statistically insignificant, suggesting no evidence of asymmetry in ϵ_t. In the leverage effect model, the leverage parameter is estimated to be very close to zero (-0.003) and statistically insignificant, suggesting no evidence of leverage effect.

To further illustrate the flexibility of our proposed algorithms and ADMB-RE, we fit a multivariate SV model to a dataset that has been used in the literature using LA-ML and SML with $S = 128$. It consists of 945 daily observations on three exchange rates from 01/10/1981 to 28/06/1985: Pound/Dollar, Deutschmark/Dollar and Yen/Dollar. The same data were used in Harvey et al. (1994), Liesenfeld and Richard (2006) and Jungbacker and Koopman (2006). As in BUGS, multivariate SV models can be estimated with a dozen lines on code in ADMB-RE. The multivariate model fitted here is taken from Harvey et al. (1994)

$$\begin{cases} \mathbf{X}_t &= \exp(0.5\mathtt{diag}(\mathbf{h}_t))\epsilon_t, \ \epsilon_t \sim N(0, \Sigma), \\ \mathbf{h}_{t+1} &= \mu + \Phi(\mathbf{h}_t - \mu) + \eta_t, \ \eta_t \sim N(0, \Sigma_\eta), \end{cases} \tag{13}$$

where $\mathbf{h}_t = (h_{1,t}, h_{2,t}, h_{3,t})'$, $\Sigma_\eta = \text{diag}\{\sigma^2_{\eta,1}, \sigma^2_{\eta,2}, \sigma^2_{\eta,3}\}$, $\mu = (\mu_1, \mu_2, \mu_3)'$, $\Phi = \text{diag}\{\phi_1, \phi_2, \phi_3\}$, and

$$\Sigma = \begin{pmatrix} 1 & \sigma_{12} & \sigma_{13} \\ \sigma_{12} & 1 & \sigma_{23} \\ \sigma_{13} & \sigma_{23} & 1 \end{pmatrix}.$$

Table 4 reports the estimates, asymptotic standard errors, CPU time and log-likelihood value in the multivariate SV model, obtained from the two ML methods. The two sets of parameter estimates are close to each other and appear reasonable. The estimated covariances in Σ are all statistically significantly different from zero, with the covariances between Pound and Deutschmark and between Pound and Yen being negative and the covariance between Deutschmark and Yen being positive. Indeed the importance of the correlations is also manifest in the log-likelihood value of the multivariate model which substantial improves the sum of the likelihood values obtained under the basic univariate SV model (not reported).

Note that the matrices Φ and Σ_η have been assumed to be diagonal, which means that the components of $\mathbf{h}_t$ are independent. From a techniqually point of view, there would be no problem to allow non-zero off-diagonal terms in Φ and Σ_η, although the identifiability of these extra parameters has not been checked.

8 Conclusion

This paper proposes to use AD and the Laplace approximation to provide the basis for numerical and simulated ML estimation and a Bayesian MCMC analysis for univariate and multivariate SV models. Also based on the Laplace approximation, we develop simple algorithms for obtaining the filtered, smoothed and forecasted latent variable. We show how the software ADMB-RE facilitates the automatic implementation of these algorithms. The proposed techniques are flexible, robust, computationally efficient, simulation-efficient, and statistically efficient. Applications of the proposed techniques using both simulated and actual data highlight these advantages. While the present paper only applies the approach to estimate SV models, the technique itself is quite general and can be applied in many other models with latent variables.

REFERENCES

Asai, M., M. McAleer, and J. Yu, (2006), Multivariate Stochastic Volatility: A Review. *Econometric Reviews*, 25, 145–175.

Andersen, T.G., T. Bollerslev, F.X. Diebold, and H. Ebens (2001), The distribution of realized stock return volatility, *Journal of Financial Economics*, 61, 43–76.

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.

Andersen, T. and B. Sorensen (1996). GMM estimation of a stochastic volatility model: A Monte Carlo study. *Journal of Business and Economic Statistics* 14, 329–352.

Bates, D. (2006). Maximum Likelihood Estimation of Latent Affine Processes. *Review Financial Studies* 19, 909–965.

Berg, A., R. Meyer, and J. Yu (2004). Deviance information criterion for comparing stochastic volatility models. *Journal of Business and Economic Statistics* 22, 107-120.

Broto, C. and E. Ruiz (2004). Estimation methods for stochastic volatility models: A survey. *Journal of Economic Surveys* 18, 613–649.

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

Danielsson, J. and J.F. Richard (1993). Accelerated Gaussian importance sampler with application to dynamic latent variable models. *Journal of Applied Econometrics* 8, s153–s173.

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.

Durham, G.S. (2006). Monte Carlo Methods for Estimating, Smoothing, and Filtering One and Two-Factor Stochastic Volatility Models. *Journal of Econometrics*, 133, 273-305.

Fournier, D. (2001). An introduction to AD MODEL BUILDER Version 6.0.2 for use in nonlinear modeling and statistics. Available from http://otter-rsch.com/admodel.htm.

Gallant, A.R. and G. Tauchen (1998). Reprojecting partially observed systems with application to interest rate diffusions. *Journal of the American Statistical Association* 93, 10-24.

Griewank, A. and G. Corliss (Eds.), (1991) *Automatic Differentiation of Algorithms*, SIAM, Philadelphia.

Hansen, B. E., (1994). Autoregressive Conditional Density Estimation. *International Economic Review* 35, 705-30.

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

Harvey, A.C. and N. Shephard (1996). The estimation of an asymmetric stochastic volatility model for asset returns. *Journal of Business and Economic Statistics* 14, 429–434.

Hastings, W.K. (1970) Monte Carlo Sampling Methods Using Markov Chains and Their Applications. *Biometrika*, 57, 97–109.

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.

Jungbacker, B. and S.J. Koopman (2006). Monte Carlo likelihood estimation for three multivariate stochastic volatility models. *Econometric Reviews*, 25, 385–408.

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

Kitagawa, G. (1996), Monte Carlo filter and smoother for Gaussian nonlinear state space models, *Journal of Computational and Graphical Statistics,* 5, 1–25.

Lancaster, T. (2004). *An Introduction to Modern Bayesian Econometrics.* Blackwell: Oxford.

Lee, K.M. and S.J. Koopman (2004), Estimating Stochastic Volatility Models: A Comparison of Two Importance Samplers, *Studies in Nonlinear Dynamics and Econometrics*, 8, 1–15.

Liesenfeld, R. and J.F. Richard (2006). Classical and Bayesian Analysis of Univariate and Multivariate Stochastic Volatility Models. *Econometric Reviews* 25, 335–360.

McAleer, M., and M. Medeiros, (2008), Realized Volatility: A Review. *Econometric Reviews*, 27, 10–45.

Metropolis, N., A. Rosenbluth, M. Rosenbluth, A. H. Teller and E. Teller (1953) Equation of State Calculations by Fast Computing Machines, *Journal of Chemical Physics*, 21, 1087-91.

Meyer, R., D.A. Fournier, and A. Berg (2003). Stochastic volatility: Bayesian computation using automatic differentiation and the extended Kalman filter. *Econometrics Journal* 6, 408–420.

Meyer, R., and J. Yu (2000). BUGS for a Bayesian analysis of stochastic volatility models. *Econometrics Journal* 3, 198–215.

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

Omori, Y., Chib, S., Shephard, N. and J. Nakajima (2006). Stochastic volatility with leverage: fast likelihood inference. *Journal of Econometrics*, forthcoming

Pitt, M., and N. Shephard (1999), Filtering via simulation: Auxiliary particle filter, *The Journal of the American Statistical Association*, 94, 590–599.

Richard, J.F. and W. Zhang (2006), Efficient High-Dimensional Importance, *Journal of Econometrics*, 141, 1385-1411.

Sandmann, G. and S.J. Koopman (1998). Maximum likelihood estimation of stochastic volatility models. *Journal of Econometrics* 63, 289–306.

Selçuk, F. (2004). Free float and stochastic volatility: The experience of a small open economy. *Physica A* 342, 693–700.

Shephard, N. (1993). Fitting nonlinear time series models with applications to stochastic variance models. *Journal of Applied Econometrics* 8, S135-52.

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

Skaug, H.J. (2002), Automatic differentiation to facilitate maximum likelihood estimation in nonlinear random effects models. *Journal of Computational and Graphical Statistics.* 11, 458–470.

Skaug, H.J. and D. Fournier (2006), Automatic Approximation of the Marginal Likelihood in non-Gaussian Hierarchical Models. *Computational Statistics and Data Analysis* 51: 699 - 709.

Taylor, S.J (1982). Financial returns modelled by the product of two stochastic processes — a study of the daily sugar prices 1961-75. In Anderson, O.D., Editor, *Time Series Analysis: Theory and Practice, 1*, 203–226. North-Holland, Amsterdam.

Yu, J. (2005) On leverage in a stochastic volatility model. *Journal of Econometrics* 127, 165–178.

Yu, J. and R. Meyer (2006) Multivariate Stochastic Volatility Models: Bayesian Estimation and Model Comparison. *Econometric Reviews*, 25, 361-384.

Appendix A: ADMB Commands, File Structure, Sample Code

ADMB exists for the WINDOWS and LINUX platforms as a set of C++ libraries and a preprocessor that converts the ADMB code (e.g. Figure 1) into C++. The C++ compiler is not part of ADMB, and must be installed separately. A model implemented in ADMB consists of three separate files that share the base name (here taken to be basicSV), but with different extensions:

- .tpl is known as the "template file", and specifies the likelihood function in terms of θ and $\mathbf{h}$ using a language that resembles C++. An example is given in Figure 1.
- .dat contains the data needed by the model, including dimensions of arrays.
- .pin contains initial values of θ used in the numerical optimization of the integrated likelihood.

The template file basicSV.tpl is compiled by the command

```
admb -re basicSV
```

into an executable file basicSV.exe (under WINDOWS) using a C++ compiler, where -re is a command line option invoking the latent variable module. When executed to perform classical estimation, basicSV.exe produces several output files. We here only discuss basicSV.std which contains point estimates and standard deviations for θ and $\mathbf{h}$ (smoothed values). In addition ADMB-RE will calculate the standard deviation of any user specified function $w(\theta, \mathbf{h})$, using the delta-method.

Figure 1 shows the template file which implements the basic SV model (1). Taking the logarithm (and changing the sign) of the likelihood function $p(\mathbf{X},\theta|\mathbf{h})p(\mathbf{h}|\theta)$ we obtain the objective function

$$-\left[\log\left\{p(h_1|\sigma,\phi)\right\}\right]-\left[\sum_{t=2}^{T}\log\left\{p(h_t|h_{t-1},\phi,\sigma)\right\}\right]-\left[\sum_{t=1}^{T}\log\left\{p(X_t|h_t,\sigma_X)\right\}\right]. \qquad (14)$$

The three terms in square brackets are specified separately in the template file. It should be noted that the Hessian matrix Ω is banded under this model, i.e. $\Omega_{ij}=0$, whenever $|i-j|>1$. This special structure can be exploited by ADMB-RE both in the manipulation of the matrix Ω and in the derivative calculations involved in the evaluation of Ω itself.[8]

A program compiled with ADMB is run from a command line prompt (DOS window), both under WINDOWS and LINUX. To do maximum likelihood estimation based on the Laplace approximation one gives the command:

```
  basicSV -ilmn 5
```

where the option -ilmn 5 invokes a limited-memory Newton method for the optimization problem (4). For high dimensional problems this optimization algorithm is more efficient than the quasi-Newton method used as the default in ADMB-RE. To perform the simulated maximum likelihood estimation, we use the command

```
  basicSV -ilmn 5 -is 64
```

where -is 64 specifies $S=64$ in the simulated maximum likelihood method.

As an integrated environment, ADMB-RE provides an option to perform Bayesian analysis in the way described in Section 3. To generate MCMC samples one uses the commands

```
  basicSV -ilmn 5 -mcmc 5000
  basicSV -mceval > mcmcout.txt
```

where -mcmc 5000 specifies that 5000 iterations of the sampler should be performed, and the second line re-directs the output from the integration sampler to the file mcmcout.txt. ADMB does not contain any functionality for post-processing the contents of mcmcout.txt. Usually, other statistical software such as the R function CODA is used for this purpose.

The quantities needed for the diagnostic tests suggested earlier should be calculated after the model has been fit. For instance, to calculate the estimates $(\hat{h}^s_{t+1}-\hat{\phi}\hat{h}^s_t)/\hat{\sigma}$ of η_t one would append the following code at the end of the program in Figure 1:

```
REPORT_SECTION
  dvar_vector eta(1,n+1);
  for (i=1;i<=n;i++)
    eta(i) = (h(i+1)-phi*h(i))/sigma;
  report << eta << endl;
```

[8]In order for ADMB-RE to recognize this structure, some directives must be added to the code in Figure 1.

which would result in the creation of a file basicSV.rep containing the estimated η values.

Appendix B: ADMB Code to Estimate Flexible SV Models

To estimate SV-t model, the only change is in the conditional distribution $X_t|h_t$. Hence, the only change we need to make in the code in Figure 1 is to introduce a new parameter nu in the PARAMETER_SECTION, and to replace the last for-loop with

```
for (i=1;i<=n;i++)
{
  dvariable sigma_X = sigma_X*exp(0.5*h(i));
  dvariable t = X(i)/sigma_X;
  g -= gammln(0.5*(nu+1.0)) - gammln(0.5*nu) - 0.5*log(nu) - 0.5723649429;
  g -= -log(sigma_X) - (nu+1.0)/2*log(1+square(t)/nu);
}
```

Here, gammln is the log-gamma function.

To estimate the SV model with the skewed t distribution, the challenge offered by this model relative to the ordinary Student-t model discussed above is the branch involved in the definition of the density (11). Straightforward coding in ADMB-RE would involve a statement like `if(t < -a/b)`. Unfortunately, since both a and b are functions of the independent parameter of the model, this may possibly yield a non-differentiable objective function, and cause trouble in the AD-scheme used by ADMB-RE. A more robust approach is to "smooth" the step function implicit in the if() statement, by employing a weighted sum of the two branches in (11). So, if log_g1 and log_g2 are variables holding the log-likelihood values of the two branches, we implement a logistic weighting scheme in ADMB-RE as follows:

```
dvariable w = exp(-20*(t+a/b));
w = w/(1.0 + w);
g -=  w*log_g1 + (1.0-w)*log_g2;
```

The choice of the scale factor 20 in the logistic function is experienced based, and the results we obtain are not very sensitive to the choice of the scale factor.

To estimate the leverage SV model, the first line in (12) is coded in ADMB-RE as:

```
g -=  -0.5*log_2pi - 0.5*log(1-square(rho))
      - 0.5*rho*(rho*square(X(i))*exp(h(i))/sigma_X)
      + rho*square((h(i+1)-phi*h(i))/sigma)
      - 2*X(i)*(h(i+1)-phi*h(i))*(1-square(rho))/(sigma*sigma_X*exp(0.5*h(i)))
      - 0.5*h(i) -log(sigma_X) - 0.5*square(X(i)/sigma_X)/exp(h(i));
```

In a recent article, Omori, Nakajima, Chib and Shephard (2006) developed an efficient MCMC algorithm to estimate the SV model with the leverage effect. If used in the Bayesian context, our method can be regarded as an alternative to theirs but does not need a mixture approximation.