Asia-Pacific Journal of Science and Technology: Volume: 23. Issue: 04. Article ID.: APST-23-04-04. Research Article

APST

Asia-Pacific Journal of Science and Technology

https://www.tci-thaijo.org/index.php/APST/index

Published by the Research and Technology Transfer Affairs Division,
Khon Kaen University, Thailand

Estimating parameters of a stochastic volatility model using the expectation-maximization algorithm coupled with a Gaussian particle filter

Tanit Malakorn[1, *], Thanapat Iamtan[2]

[1] Department of Electrical and Computer Engineering, Naresuan University, Phitsanulok, Thailand
[2] National Science and Technology Development Agency, Pathum Thani, Thailand
[*] Correspondent author: tanitm@nu.ac.th

Received 5 January 2018
Revised 26 April 2018
Accepted 15 August 2018

Abstract

In this paper, the expectation-maximization algorithm coupled with a Gaussian particle filter for maximum likelihood parameter estimation of a stochastic volatility model is investigated. Two data sets are provided for demonstration purposes: simulated data and daily foreign exchange rates data. Simulation studies illustrate that the parameter estimate trajectories are likely to converge to the true ones. When comparing the empirical results obtained from the conventional method and the proposed method, it can be seen that the parameter estimates from both methods are consistent with each other; however, the computational time is considerably reduced when using the method presented here.

Keywords: Maximum likelihood, Parameter estimation, Bootstrap filter, Daily exchange rates

1. Introduction

Option pricing theory has been the foundation of mathematical finance and financial engineering since Black and Scholes [1] published their groundbreaking paper in 1973. They formulated a partial differential equation, currently known as the Black-Scholes model or BS model for short, to evaluate the fair value of options over time. Five parameters are required in the BS model; the volatility σ, however, is the only parameter that cannot be directly observed from the market data and must be estimated. In finance, volatility is a statistic measure of fluctuations of return process; i.e., the higher the volatility is, the more fluctuations the asset price exhibits. In the beginning, most options were short-lived with maturities of a few months to a year. During the option's lifetime, the underlying asset prices gradually changed, and hence it seemed reasonable to assume the return volatility over the near future to be constant. Therefore, the historic volatility and the implied volatility are normally applied to the BS model for short term forecasting.

In recent years, most options in derivative markets are long-lived with maturities of 5 years or longer. The assumption of constant volatility is inconsistent with the characteristics of the real world financial data. Evidently, the returns of the underlying assets in financial markets have fluctuated constantly and randomly over time, and typically exhibit volatility clustering. Having a proper model for return volatility, one is able to investigate the behavior of asset's return and to forecast its corresponding volatility. Various models have been proposed in the econometric literature; among them, the so-called stochastic volatility (SV) models have attracted considerable attention since they can be regarded as a particular type of state space models corrupted by noises, which is commonly used in system & control communities. The first SV model was introduced by Taylor [2] in 1982 and its variants have drawn extensive studies thereafter.

Once the model is selected, it is customary to fit the model with the empirical data using parameter estimation methods. Several methods for parameter estimation have been increasingly developed in the time series literature; one of which is the expectation-maximization (EM) algorithm originally introduced by Dempster et al. [3] in 1977 for maximum likelihood (ML) estimation of stochastic state space models. Basically, the EM algorithm is

an iterative method consisting of two steps, namely the expectation step (E-step) and the maximization step (M-step). In the E-step, the algorithm computes the expectation of the likelihood of the complete data over the smoothing distribution using the current knowledge of the ML parameter estimates and the observed data. This step may require state filtering and smoothing with current parameter estimates to get filtered and smoothed state estimates. Then the M-step uses the smoothed state estimates to re-estimate the parameters by maximizing the lower bound of the marginal likelihood of the observed data so that the new likelihood function achieves the maximum. Due to its simplicity, the EM algorithm has tremendous applications in various fields such as applied mathematics, statistics, economics, robotics, and signal processing.

In state space models, state variables (sometimes called latent variables or hidden variables) are not directly observed but are rather inferred from a sequence of observations, or from a sequence output of the models. When the EM algorithm has been applied to such models, state variables over the smoothing distribution have to be estimated. For linear models with Gaussian distribution, the celebrated Kalman filter, also known as the linear quadratic estimator, and the Rauch-Tung-Striebel (RTS) smoother are commonly applied to estimate state variables in the E-step due to the simplicity of algorithms and ease of implementation acquired from having the closed-form solutions. Besides, the Kalman filter has long been regarded as an optimal state estimator in the sense that the mean squared error between the true state variables and the estimated ones is minimized as compared with other filters. When the linearity or the Gaussianity assumption is violated, the optimality no longer holds in general, see, e.g., [4–6]. The estimates obtained from the Kalman filter may lead to the quasi-maximum likelihood; however, it is still being employed successfully in a large number of applications, even though the models are nonlinear or contaminated by non-Gaussian noises.

Apart from the Kalman filter, a sequential Monte Carlo (SMC) method [7] introduced in the 1960's is an alternative approach to infer state variables via point mass representation of probability densities by performing importance sampling to get samples from the prediction distribution, computing normalized weights, and finally applying the re-sampling step in the filtering process. The re-sampling step is used for solving the degeneracy problem by eliminating samples with small (nearly zero) weights and regenerating a new set of samples with larger weights. Since samples are mostly called *particles*, the filter and smoother designed via the SMC method are commonly called the *particle filter* and the *particle smoother*, respectively. The simplest, and probably most widely used, type of particle filter is a bootstrap filter (BF) introduced by Gordon et al. [8] in 1993, where the state model is used as the importance distribution. Since then, BF has been successfully applied to a wide variety of interests, from natural sciences to applied sciences and engineering.

One major drawback of particle filters is that they are time-consuming due to the re-sampling step, which is a serial computation. To handle this problem, Kotecha and Djuric [9] proposed a new type of filter, called a Gaussian particle filter (GPF), in 2003. In their paper, the importance sampling is utilized to approximate the filtering distribution by the posterior mean and variance of a Gaussian distribution, rather than performing the re-sampling step. They also showed that the computational time in the filtering process can be reduced considerably; however, they did not take a further step to apply the GPF in the estimation problem. To the best of the authors' knowledge, this is the first attempt to apply the GPF in the EM algorithm for parameter estimation of SV models.

The main purpose of this paper is to investigate the performance of two different filters, namely the GPF and the BF, coupled with the EM algorithm for ML parameter estimation of the SV model. Two sets of experiments are performed in this study to provide empirical estimates: (1) the experiment with simulated data generated from the SV model, and (2) the experiment with real data collected from foreign exchange rates markets. The remainder of this paper is organized as follows.

The next section presents the SV model adopted from [10], where the volatility is recast as a nonlinear transformation of a state variable, followed by ML estimation and an EM algorithm. Next, particle methods for filtering and smoothing as well as Gaussian particle filters are briefly described. After that the experimental results are demonstrated, and conclusions of this work are provided. In order to facilitate readability, a list of acronyms as well as a list of mathematical symbols used in this paper are presented in Section 6.

2. Materials and methods

2.1 Stochastic volatility models

Let P_k denote the asset price at time k. The log return of P_k is then defined by $r_k \coloneqq \log\left(\frac{P_k}{P_{k-1}}\right)$, which admits the univariate SV model:

$$r_k = \sigma_k \varepsilon_k, \tag{1}$$

where ε_k is a white Gaussian noise process with zero mean and unit variance, and σ_k is a non-negative stochastic process representing the volatility of r_k. According to the empirical results, σ_k^2 is the log-normally distribution; i.e., there does exist a normal random variable x_k so that $x_k = \log \sigma_k^2$, and hence (1) becomes

$$r_k = \exp(x_k/2)\,\varepsilon_k. \tag{2}$$

Traditionally, the log volatility x_k is assumed to follow a first order autoregressive process, an AR(1) process, with Gaussian innovation noise:

$$x_k = \phi x_{k-1} + c + \omega_k, \tag{3}$$

where ϕ and c are constants and ω_k is a white Gaussian noise process with zero mean and variance Q. It is normally assumed that ω_k and ε_k are mutually independent. In addition, if $|\phi| < 1$, the above process is wide-sense stationary.

Rather than (2) and (3), one may introduce a scaling factor β in (2) to remove the constant term c from (3). Hence, the canonical SV model for the log return is given by

$$x_k = \phi x_{k-1} + \omega_k, \tag{4a}$$
$$r_k = \beta \exp(x_k/2)\,\varepsilon_k, \tag{4b}$$

with initial state x_0. This couple of equations is a particular type of stochastic nonlinear state space models, where x_k is acting as the state variable, r_k is the output of the model which is the return process in this case, ω_k and ε_k are respectively considered as the process noise and the measurement noise in the state space terminology, and $\theta = \{\phi, Q, \beta\}$ is the model parameters. Note that (4a) and (4b) are called the state model and the measurement model, respectively.

2.2 ML estimation

There are several approaches for parameter estimation proposed in the literature but the ML based approach is the most commonly employed. In statistics, a likelihood function denoted by $L(\theta)$ is a function of the parameters θ for given data. It is closely related to the probability density function as follows: Given a data sequence of length *N*, say $y_{1:N} = \{y_1, \dots, y_N\}$, the likelihood function is defined by

$$L(\theta) = p(y_{1:N}|\theta). \tag{5}$$

The ultimate goal of the ML estimation is to determine the best estimator, $\hat{\theta}$, which maximizes $L(\theta)$. It is worthwhile to note that the logarithmic function is monotonically increasing, and hence the value of $\hat{\theta}$ that maximizes $L(\theta)$ also maximizes $\log L(\theta)$. Thus, the ML estimation problem can be restated as to determine

$$\hat{\theta} = \arg\max_{\theta \in \Theta} \log L(\theta) = \arg\max_{\theta \in \Theta} \log p(y_{1:N}|\theta),$$

where Θ is the parameter space.

In the state space models as in (4), there are two processes running in parallel. The complete data acquired from the model consists of the state sequence $x_{0:N}$ and the measurement sequence $r_{1:N}$, and hence the likelihood (5) for the complete data becomes

$$L(\theta) = p(x_{0:N}, r_{1:N}|\theta). \tag{6}$$

Since the return volatility σ_k of financial assets, which in turn is a nonlinear function of the state variable x_k, cannot be directly observed from the market, the states must be integrated out from (6), which gives the marginal likelihood of the observed data

$$p(r_{1:N}|\theta) = \int p(x_{0:N}, r_{1:N}|\theta)\, dx_{0:N},$$

and the ML parameter estimation for state space models is to find

$$\hat{\theta} = \arg\max_{\theta \in \Theta} \log \int p(x_{0:N}, r_{1:N}|\theta)\, dx_{0:N}. \tag{7}$$

In general, the direct maximization of the marginal log-likelihood in (7) is analytically intractable due to the integration over all state variables $x_{0:N}$. Numerical methods, such as a grid search, a gradient-based search, a

Laplace approximation, and an EM algorithm, are broadly applied to approximately maximize the marginal log-likelihood.

2.3 EM algorithm

The EM algorithm is an iterative method for approximating the ML estimates of unknown parameters when the given data is incomplete in the sense that the state variables cannot be observed directly. Based on the fact that a lower bound for the marginal log-likelihood can be easily evaluated, the basic concept of the algorithm is to increase the bound by iteratively updating the parameter estimates, $\theta^{(\kappa)}$, and, as a consequence, increase the marginal log-likelihood. The major advantage of the EM algorithm is that the monotonic increasing property holds, i.e., in each consecutive iteration, $L\big(\theta^{(\kappa+1)}\big) \geq L\big(\theta^{(\kappa)}\big)$.

Assuming that the algorithm calculates the current parameter estimates, $\theta^{(\kappa)}$, it can be shown that (see, e.g., [11–14]),

$$\log p(r_{1:N}|\theta) \geq \log p\big(r_{1:N}|\theta^{(\kappa)}\big) + \ell\big(\theta, \theta^{(\kappa)}\big), \tag{8}$$

where

$$\ell\big(\theta, \theta^{(\kappa)}\big) = \int p\big(x_{0:N}|r_{1:N}, \theta^{(\kappa)}\big) \log \frac{p(x_{0:N}, r_{1:N}|\theta)}{p\big(x_{0:N}, r_{1:N}|\theta^{(\kappa)}\big)} dx_{0:N}. \tag{9}$$

Obviously, the lower bound of the marginal log-likelihood is the expression on the right hand side of (8). The next step is to find $\theta^{(\kappa+1)}$ in order to increase the marginal log-likelihood by maximizing the lower bound with respect to θ. Note that the terms which do not depend on θ can be treated as constants, and can be dropped from the maximization process. The parameter estimates in the next iteration is then given by

$$\theta^{(\kappa+1)} = \arg\max_{\theta \in \Theta} \mathcal{Q}\big(\theta, \theta^{(\kappa)}\big), \tag{10}$$

where

$$\begin{aligned} \mathcal{Q}\big(\theta, \theta^{(\kappa)}\big) &= \int p\big(x_{0:N}|r_{1:N}, \theta^{(\kappa)}\big) \log p(x_{0:N}, r_{1:N}|\theta)\, dx_{0:N}\,, \\ &= \mathbb{E}[\log p(x_{0:N}, r_{1:N}|\theta)]. \end{aligned} \tag{11}$$

In the discrete-time state space formalism, the state and the measurement sequences are normally assumed to follow the Markov property. Hence the log-likelihood of the complete data can be written as

$$\log p(x_{0:N}, r_{1:N}|\theta) = \log p(x_0|\theta) + \sum_{k=1}^{N} \log p(x_k|x_{k-1}, \theta) + \sum_{k=1}^{N} \log p(r_k|x_k, \theta),$$

and the $\mathcal{Q}$ function in (11) is given by the following expression:

$$\mathcal{Q}\big(\theta, \theta^{(\kappa)}\big) = \mathcal{Q}_0\big(\theta, \theta^{(\kappa)}\big) + \mathcal{Q}_x\big(\theta, \theta^{(\kappa)}\big) + \mathcal{Q}_y\big(\theta, \theta^{(\kappa)}\big), \tag{12}$$

where

$$\begin{aligned} \mathcal{Q}_0\big(\theta, \theta^{(\kappa)}\big) &= \int p\big(x_0|r_{1:N}, \theta^{(\kappa)}\big) \log p(x_0|\theta)\, dx_0, \\ &= \mathbb{E}[\log p(x_0|\theta)], \end{aligned} \tag{13}$$

$$\begin{aligned} \mathcal{Q}_x\big(\theta, \theta^{(\kappa)}\big) &= \sum_{k=1}^{N} \int p\big(x_k, x_{k-1}|r_{1:N}, \theta^{(\kappa)}\big) \log p(x_k|x_{k-1}, \theta)\, dx_k dx_{k-1}\,, \\ &= \textstyle\sum_{k=1}^{N} \mathbb{E}[\log p(x_k|x_{k-1}, \theta)], \end{aligned} \tag{14}$$

$$\begin{aligned} \mathcal{Q}_y\big(\theta, \theta^{(\kappa)}\big) &= \sum_{k=1}^{N} \int p\big(x_k|r_{1:N}, \theta^{(\kappa)}\big) \log p(r_k|x_k, \theta)\, dx_k, \\ &= \textstyle\sum_{k=1}^{N} \mathbb{E}[\log p(r_k|x_k, \theta)]. \end{aligned} \tag{15}$$

It is worth noting that the expectations in (13), (14), and (15) are over the smoothing distribution of the state estimates given the observed data, $r_{1:N}$, and the current values of the parameter estimates, $\theta^{(\kappa)}$. It turns out that

these expectations are much easier to calculate than the one in (11), which is the expectation over the full joint posterior distribution.

Taking into account the above discussion, the EM algorithm for discrete-time state space models can be concluded as follows:

Algorithm 1 (EM algorithm) Let $\theta^{(0)}$ be an initial guess of the model parameters. The algorithm iteratively generates a sequence of parameter estimates, $\theta^{(\kappa)}$, for $\kappa = 1,\ldots, N$ through the E-step and the M-step, where the E-step is to evaluate the $\mathcal{Q}$ function in (12) using (13), (14), and (15), and the M-step is to determine the new parameter estimates, $\theta^{(\kappa+1)}$, using (10). These two steps continue alternately until a convergence criterion is fulfilled.

2.4 Filters and smoothers

As mentioned previously, the state estimates over the smoothing distribution are required in the E-step to compute the $\mathcal{Q}$ function. This can be done by performing the smoothing process. The classic RTS smoother is an effective forward-backward algorithm for fixed interval smoothing to estimate states of linear models corrupted by Gaussian noises. The algorithm is based on two different Kalman filters; one of which propagates in the forward pass to get the filtered state estimates and the corresponding error covariance matrices, while the other propagates in the backward pass to produce the smoothed state estimates.

For nonlinear and/or non-Gaussian state space models, Kalman filters and RTS smoothers are no longer optimal in general. Based on a set of random particles with associated weights rather than a single estimate, particle filters and smoothers are commonly employed to approximate arbitrary distributions. Several design methods for particle smoothers have been proposed in the literature, and successfully applied to general state space models. The backward-simulation particle smoother (BSPS) introduced by Godsill et al. [15] in 2004 can be regarded as an analogue of the RTS smoother, where filtering results obtained from the forward pass are used to estimate smoothed states in the backward pass.

The aim of this section is to present the BSPS algorithm for computing smoothed state estimates of the SV model using the two types of filter: the BF and the GPF. For more details on the variation of the SMC method (particle filters and smoothers) and its applications, interested readers may consult [16–24] and references therein.

2.4.1 Bootstrap Filter: BF

The following presents the BF algorithm for the filtered state estimation of the SV model given in (4).

Algorithm 2 (BF algorithm) Let $x_k^{f(i)}$ and $w_k^{f(i)}$ denote, respectively, the i-th particle and its corresponding normalized weight at time step $k = 0,1,\ldots, N$ in the filtering process, where N is the length of the measurement sequence, or the observed data. The BF algorithm is the following.

(i) **Initialization**: Draw particles randomly from the prior Gaussian distribution with zero mean and unit variance, i.e.,

$$x_0^{f(i)} \sim p(x_0) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}x_0^2\right), \tag{16}$$

and set initial weights: $w_k^{f(i)} = 1/n_f$ for $i = 1,\ldots, n_f$, where n_f is the number of particles.

(ii) **Recursive Step**: For $k = 1,\ldots, N$,

Step 1: Predict new particles according to the state model in (4a)

$$x_0^{p(i)} \sim p\left(x_k | x_{k-1}^{f(i)}\right) = \frac{1}{\sqrt{2\pi Q}} \exp\left(-\frac{1}{2}\frac{\left(x_k - \phi x_{k-1}^{f(i)}\right)^2}{Q}\right). \tag{17}$$

Step 2: Calculate new weights from the measurement model in (4b)

$$\widetilde{w}_k^{f(i)} \propto p\left(r_k | x_k^{p(i)}\right) = \frac{1}{\beta\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left[x_k^{p(i)} + \frac{r_k^2}{\beta^2} e^{-x_k^{p(i)}}\right]\right), \tag{18}$$

and do normalization to get

$$w_k^{f(i)} = \frac{\tilde{w}_k^{f(i)}}{\sum_{i=1}^{n_f} \tilde{w}_k^{f(i)}}, \quad i = 1,\ldots, n_f. \tag{19}$$

Step 3: Update the particles by performing the re-sampling as follows: Consider $w_k^{f(i)}$ as a discrete probability of the re-sampling. Then, for $i = 1,\ldots, n_f$, draw new particles $x_k^{f(i)}$ according to,

$$P\left(x_k^{f(i)} = x_k^{p(i)}\right) = w_k^{f(i)}, \qquad j = 1,\ldots, n_f. \tag{20}$$

Step 4: Set $x_{0:k}^{f(i)} = \left\{x_{0:k-1}^{f(i)}, x_k^{f(i)}\right\}$, and $w_{0:k}^{f(i)} = \left\{w_{0:k-1}^{f(i)}, w_k^{f(i)}\right\}$.

2.4.2 Gaussian Particle Filter: GPF

The concept of the GPF algorithm for the filtered state estimation of the SV model given in (4) is as follows:

Algorithm 3 (GPF algorithm) This algorithm follows the same steps as the BF, except that Step 3 of Algorithm 2 is replaced by the following:

Step 3: Calculate the posterior mean and variance according to

$$\mu_k = \sum_{i=1}^{n_f} w_k^{f(i)} x_k^{f(i)}, \text{ and } \nu_k = \sum_{i=1}^{n_f} w_k^{f(i)} \left(x_k^{f(i)} - \mu_k\right)^2. \tag{21}$$

Then, for $i = 1,\ldots, n_f$, draw new particles $x_k^{f(i)}$ from Gaussian density:

$$x_k^{f(i)} \sim \frac{1}{\sqrt{2\pi\nu_k}} \exp\left(-\frac{1}{2}\frac{\left(x_k^{f(i)} - \mu_k\right)^2}{\nu_k}\right). \tag{22}$$

2.4.3 Backward Simulation Particle Smoother: BSPS

The following presents the BSPS algorithm for the smoothed state estimation of the SV model given in (4).

Algorithm 4 (BSPS algorithm) Let $x_k^{s(j)}$ denote the particle of the j-th smoothing trajectory at time step $k = 0, 1,\ldots, N$.

(i) **Initialization**: Collect $x_{0:N}^{f(i)}$ and $w_{0:N}^{f(i)}$, for $i = 1,\ldots, n_f$, from the filtering process, and select $x_N^{s(j)} = x_N^{f(i)}$ with probability $w_N^{f(i)}$.

(ii) **Recursive Step**: For $k = N-1,\ldots, 0$,

Step 1: For $i = 1,\ldots, n_f$, calculate new weights according to

$$w_k^{s(i)} \propto w_k^{f(i)} p\left(x_{k+1}^{s(j)} | x_k^{f(i)}\right) = \frac{w_k^{f(i)}}{\sqrt{2\pi Q}} \exp\left(-\frac{1}{2}\frac{\left(x_{k+1}^{s(j)} - \phi x_k^{f(i)}\right)^2}{Q}\right). \tag{23}$$

Step 2: Consider $w_k^{s(i)}$ as a discrete probability for particle selection and choose $x_k^{s(j)} = x_k^{f(i)}$ with probability $w_k^{s(i)}$.

Repeat (i) and (ii) for $j = 1,\ldots, n_s$, where n_s is the number of smoothing trajectories.

The smoothing trajectories, $x_{0:N}^{s(j)}$, along with the observed data, $r_{1:N}$, will be used to estimate the parameters in the EM algorithm (see Algorithm 1).

3. Results

In this section, two sets of experiments are performed to demonstrate the performance of the proposed method for parameter estimation of the SV model. The first set of experiments employs simulated data generated from the model in (4) to evaluate the accuracy of the algorithm by comparing the parameter estimates with the known

true parameters. The second set uses real data from financial markets. For validation purposes, the results of this study will then be compared with those obtained from the conventional method, i.e., the EM algorithm coupled with the BF.

For implementation of the EM algorithm, it is necessary to establish the Q function. In case of the SV model, the Q function in (12) becomes:

$$Q\left(\theta,\theta^{(\kappa)}\right)=-\frac{1}{n_s}\sum_{j=1}^{n_s}\frac{1}{2}\left[\left(x_0^{s(j)}\right)^2+\sum_{k=1}^{N}\frac{\left(x_k^{s(j)}-\phi x_{k-1}^{s(j)}\right)^2}{Q}+\sum_{k=1}^{N}\left(x_k^{s(j)}+\frac{r_k^2}{\beta^2}e^{-x_k^{s(j)}}\right)+N\log Q+2N\log\beta\right]+C$$

where $x_{0:N}^{s(j)}, j=1,\ldots,n_s$, is the smoothing trajectories of the smoothed state estimates acquired from the smoothing process (see previous section for more details), and C is the constant term.

By taking the partial derivative of $Q\left(\theta,\theta^{(\kappa)}\right)$ with respect to each parameter and setting them all to zero, the new parameter estimates for the next iteration $\theta^{(\kappa+1)}=\left\{\phi^{(\kappa+1)},Q^{(\kappa+1)},\beta^{(\kappa+1)}\right\}$ are calculated as follows:

$$\phi^{(\kappa+1)}=\sum_{j=1}^{n_s}\sum_{k=1}^{N}x_k^{s(j)}x_{k-1}^{s(j)}\Big/\sum_{j=1}^{n_s}\sum_{k=1}^{N}\left(x_{k-1}^{s(j)}\right)^2,$$

$$Q^{(\kappa+1)}=\frac{1}{n_s}\sum_{j=1}^{n_s}\left[\frac{1}{N}\sum_{k=1}^{N}\left(x_k^{s(j)}-\phi^{(\kappa+1)}x_{k-1}^{s(j)}\right)^2\right],$$

$$\beta^{(\kappa+1)}=\sqrt{\frac{1}{n_s}\sum_{j=1}^{n_s}\left[\frac{1}{N}\sum_{k=1}^{N}\left(r_k^2e^{-x_k^{s(j)}}\right)\right]}.$$

3.1 Simulated data

To illustrate the accuracy of the proposed method, the data set of length $N=500$ generated from the SV model in (4) with true parameters $\theta=\{\phi,Q,\beta\}=\{0.9,0.5,2.2\times10^{-3}\}$ is fed into Algorithm 1 for parameter estimation, with the use of Algorithm 3 and Algorithm 4 for filtering and smoothing processes, respectively. The initial guess, the number of particles and the number of smoothing trajectories are respectively set to $\theta^{(0)}=\{0.45,0.25,1.15\times10^{-3}\}$, $n_f=300$, and $n_s=150$.

Parameter estimates in each iteration are computed and plotted as shown in Figure 1, with 200 iterations. The solid line represents the estimated parameters, whereas the dotted line represents the true ones. Apparently, the trajectories of the parameter estimates converge to the true parameters with final estimates are $\hat{\phi}=0.9011\ (0.0362)$, $\hat{Q}=0.5228\ (0.1308)$, and $\hat{\beta}=2.18\times10^{-3}\ (0.004\times10^{-3})$, where the values in the parentheses indicate the standard variations of the estimates.

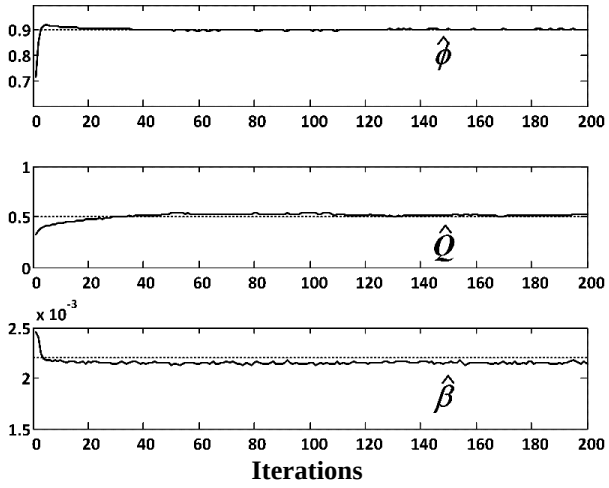

Figure 1 $\{\hat{\phi},\hat{Q},\hat{\beta}\}$ from the simulated data

As a standard procedure, the experiment is conducted repeatedly with different data sets. The averages of the final estimates are then taken into consideration. It turns out that such averages remain close to the true parameters. Thus, the simulation and the numerical results experimentally validate the accuracy and effectiveness of the proposed method.

3.2 Daily exchange rates

The aim of this example is two-fold: to estimate the SV model parameters from the daily exchange rates of US Dollar to Thai Baht (USD/THB) from April 10, 2014, through April 29, 2016, as a case study, and to validate the results by comparing with those obtained from the EM algorithm coupled with the BF. The time plot of the USD/THB daily exchange rates time series and their corresponding log returns are shown, respectively, in the upper and lower panels of Figure 2.

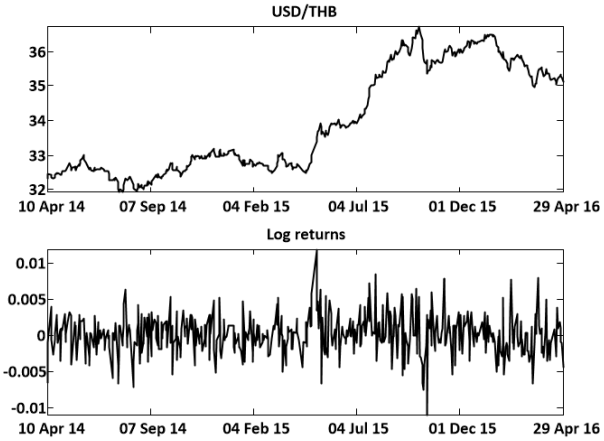

Figure 2 The USD/THB daily exchange rates and the corresponding log returns

In this experiment, the log returns are fed into two separate EM algorithms running in parallel; one is coupled with the BF (Algorithm 2), and the other is coupled with the GPF (Algorithm 3). The parameter estimate trajectories obtained via the two algorithms are depicted in Figure 3, with 200 iterations, and the final estimates, $\hat{\theta} = \{\hat{\phi}, \hat{Q}, \hat{\beta}\}$, as well as the corresponding standard deviations indicated in the parentheses are reported in Table 1.

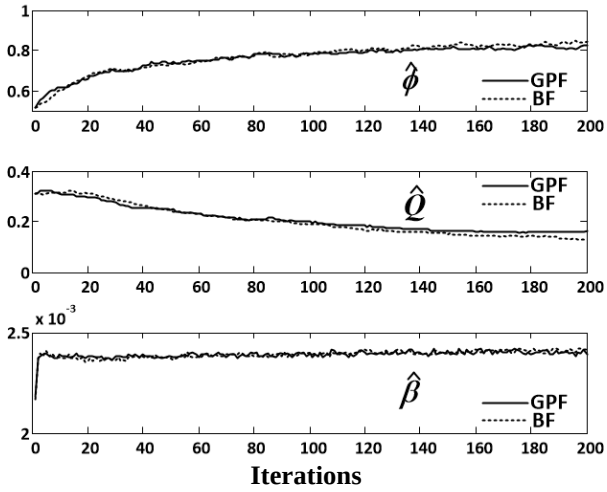

Figure 3 $\{\hat{\phi}, \hat{Q}, \hat{\beta}\}$ from the USD/THB daily exchange rates using the GPF (solid line), and the BF (dotted line)

Table 1 Final estimates for the USD/THB daily exchange rates

Estimates	GPF	BF
$\hat{\phi}$	0.8273 (0.0180)	0.8291 (0.0204)
$\hat{Q}$	0.1886 (0.0240)	0.1808 (0.0291)
$\hat{\beta}$	2.40×10^{-3} (0.002×10^{-3})	2.41×10^{-3} (0.002×10^{-3})

As a by-product of the EM algorithm, the volatility can be computed via the smoothed state estimates, x_k^s, as follows: $\sigma_k = \exp(x_k^s/2)$, where $x_k^s = \frac{1}{n_s}\sum_{j=1}^{n_s} x_k^{s(j)}$. The upper and lower panels of Figure 4 illustrate the volatility estimates using the two different filters: the GPF and the BF, respectively.

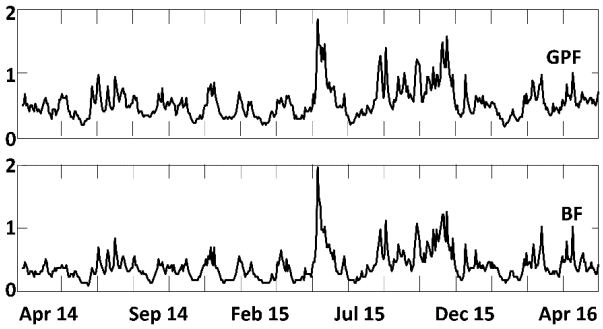

Figure 4 The simulated volatility

It is clear from Figure 3 that the estimate trajectories obtained via the two different filters are slightly different, and converge to the final values, which are sufficiently close to each other. However, when the computational time is taken into account, the GPF is preferable since it does not require the re-sampling step, which has computational complexity of $O(N)$, as in the case of the BF (see [9] for more details). Figure 5 illustrates the computational time in the filtering process between the GPF and the BF when varying the number of particles, n_f. Clearly, the computational time is considerably reduced when using the GPF. Note that all numerical simulations in this section are performed on a personal computer with 2.1 GHz Intel Core i3 processor with MATLAB.

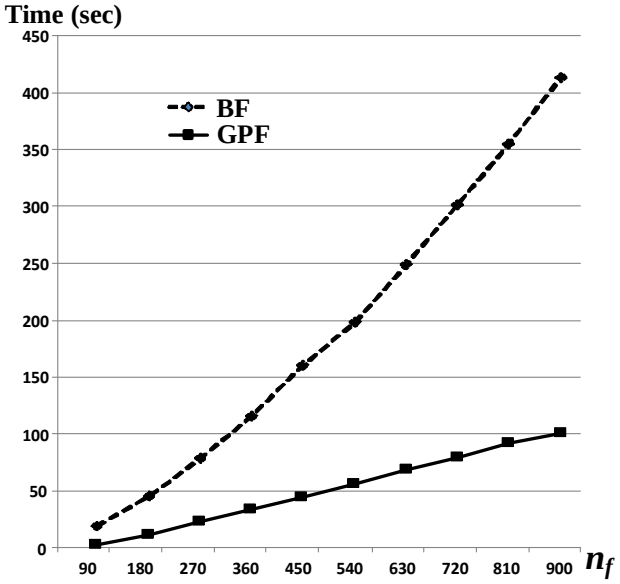

Figure 5 Computational time in the filtering process using the GPF (solid line), and the BF (dotted line)

4. Conclusions

The contribution of this work is to apply the EM algorithm coupled with a GPF to estimate the SV model parameters from given data. Two sets of experiments are conducted to demonstrate the effectiveness of the proposed method. In the first experiment, the data set is generated from the model and fed into the algorithm. The simulation and the numerical results show that the parameter estimate trajectories are likely to converge to the true parameters, with final estimates being sufficiently close to the true ones.

The data set in the second experiment is the log returns of the USD/THB daily exchange rates from April 10, 2014, through April 29, 2016. As a comparison, two EM algorithms run in parallel; one is coupled with the BF, and the other is coupled with the GPF. Evidently, the experimental results indicate that the parameter estimates obtained via the two different filters are consistent with each other; however, the computational time in the filtering process is reduced dramatically when using the GPF. Thus, it seems preferable to employ the EM algorithm coupled with a GPF for parameter estimation of the SV models, rather than that with the BF.

5. Acknowledgements

This work was fully supported by Naresuan University under Grant no. R2559C098. The authors are sincerely grateful to Kevin Roebl for his help in correcting the English.

6. Nomenclature

List of acronyms

BF	Bootstrap Filter
BS	Black-Scholes
BSPS	Backward-Simulation Particle Smoother
EM	Expectation-Maximization
GPF	Gaussian Particle Filter
ML	Maximum Likelihood
RTS	Rauch-Tung-Striebel
SMC	Sequential Monte Carlo
SV	Stochastic Volatility

List of mathematical symbols

k	Time step
$L(\theta)$	Likelihood function
N	Length of the measurement sequence, or observed data
n_f	Number of particles
n_s	Number of smoothing trajectories
P_k	Asset price at time step k
r_k	Log return of P_k, which is considered as the output of the SV model
$w_k^{f(i)}$	The i-th normalized weight at time step k in the filtering process
$w_k^{s(i)}$	Discrete probability of particle selection in the smoothing process
x_k	Log volatility, which is considered as the state variable of the SV model
$x_k^{f(i)}$	the i-th particle at time step k in the filtering process
$x_k^{s(i)}$	Particle of the j-th smoothing trajectory at time step k in the smoothing process
β	Scaling factor
μ_k	Posterior mean in the GPF
ν_k	Posterior variance in the GPF
σ_k	Volatility of the return r_k
ε_k	White Gaussian noise process with zero mean and unit variance, which is considered as the measurement noise
ω_k	White Gaussian noise process with zero mean and variance Q, which is considered as the process noise
$\theta = \{\phi, Q, \beta\}$	SV model parameters
$\hat{\theta} = \{\hat{\phi}, \hat{Q}, \hat{\beta}\}$	Estimated parameters

7. References

[1] Black, F., Scholes, M., 1973. The pricing of options and corporate liabilities. Journal of Political Economy 81, 637–654.

[2] Taylor, S.J., 1982. Financial returns modelled by the product of two stochastic processes, a study of daily sugar prices 1961–79. In: Anderson, O.D. (ed). Time series analysis: theory and practice, 203–226.

[3] Dempster, A.P., Laird, N.M., Rubin, D.B., 1977. Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society. Series B (Methodology) 39, 1–38.

[4] Simon, D., 2010. Kalman filtering with state constraints: a survey of linear and nonlinear algorithms. IET Control Theory & Application 4, 1303–1318.

[5] Crassidis, J.L., Markley, F.L., Cheng, Y., 2007. Survey of nonlinear attitude estimation methods. Journal of Guidance, Control, and Dynamic 30, 12–28.

[6] Voss, H.U., Timmer, J., Kurths, J., 2004. Nonlinear dynamical system identification from uncertain and indirect measurements. International Journal of Bifurcation and Chaos 14, 1905–1933.

[7] Doucet, A., de Freitas, N., Gordon, N., 2001. Sequential Monte Carlo Methods in Practice, 1st ed. Springer Verlag, New York.

[8] Gordon, N.J., Salmond, D.J., Smith, A.F.M., 1993. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. IEE Proceedings F - Radar and Signal Processing 140, 107–113.

[9] Kotecha, J.H., Djuric, P.M., 2003. Gaussian particle filtering. IEEE Transactions on Signal Processing 51, 2592–2601.

[10] Platanioti, K., McCoy, E.J., Stephens, D.A., 2005. A review of stochastic volatility: univariate and multivariate models, Imperial College, London.

[11] McLachlan, G.L., Krishnan, T., 2008. The EM Algorithm and Extensions, 2nd ed. John Wiley & Sons, New York.

[12] Brockwell, P.J., Davis, R.A., 2016. Introduction to Time Series and Forecasting, 3rd ed. Springer-Verlag, New York.

[13] Commandeur, J.J.F., Koopman, S.J., 2007. An Introduction to State Space Time Series Analysis, Oxford University Press, Oxford.

[14] Zia, A., Kirubarajan, T., Reilly, J.P., Yee, D., Punithakumar, K., Shirani, S., 2008. An EM algorithm for nonlinear state estimation with model uncertainties. IEEE Transactions on Signal Processing 56, 921–936.

[15] Godsill, S.J., Doucet, A., West, M., 2004. Monte Carlo smoothing for nonlinear time series. Journal of the American Statistical Association 99, 156–168.

[16] Sarkka, S., 2013. Bayesian filtering and smoothing, 1st ed. Cambridge University Press, Cambridge.

[17] Arulampalam, M.S., Maskell, S., Gordon, N., Clapp, T., 2002. A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. IEEE Transactions on Signal Processing 50, 174–188.

[18] Cappe, O., Moulines, E., Ryden, T., 2005. Inference in Hidden Markov Models, 1st ed. Springer-Verlag, New York.

[19] Kantas, N., Doucet, A., Singh, S.S., Maciejowski, J., Chopin, N., 2015. On particle methods for parameter estimation in state-space models. Statistical Science 30, 328–351.

[20] Xu, W., 2016. Estimation of dynamic panel data models with stochastic volatility using particle filters. Econometrics 4, 1–13.

[21] Ruiz, H.-C., Kappen, H.J., 2017. Particle smoothing for hidden diffusion processes: adaptive path integral smoother. IEEE Transactions on Signal Processing 65, 3191–3203.

[22] Finke, A., Singh, S.S., 2017. Approximate smoothing and parameter estimation in high-dimensional state-space models. IEEE Transactions on Signal Processing 65, 5982–5994.

[23] Wang, H., Nagayama, T., Su, D., 2017. Vehicle parameter identification through particle filter using bridge responses and estimated profile. Procedia Engineering 188, 64–71.

[24] Murakami, A., Shinmura, H., Ohno, S., Fujisawa, K., 2017. Model identification and parameter estimation of elastoplastic constitutive model by data assimilation using the particle filter. International Journal for Numerical and Analytical Methods in Geomechanics 42, 110–131.