Sequential Parameter Estimation in Stochastic Volatility Jump-Diffusion Models

Michael Johannes Nicholas Polson Jonathan Stroud*

August 12, 2003

Abstract

This paper considers the problem of sequential parameter and state estimation in stochastic volatility jump diffusion models. We describe the existing methods, the particle and practical filter, and then develop algorithms to apply these methods to the case of stochastic volatility models with jumps. We analyze the performance of both approaches using both simulated and S&P 500 index return data. On simulated data, we find that the algorithms are both effective in estimating jumps, volatility and parameters. On S&P 500 index data, the practical filter appears to outperform the particle filter.

*Johannes is at the Graduate School of Business, Columbia University, 3022 Broadway, NY, NY, 10027, mj335@columbia.edu. Polson is at the Graduate School of Business, University of Chicago, 1101 East 58th Street, Chicago IL 60637, ngp@gsbngp.uchicago.edu. Stroud is at The Wharton School, University of Pennsylvania, Philadelphia, PA 19104-6302, stroud@wharton.upenn.edu.

1 Introduction

Models incorporating time-varying volatility are essential for practical finance applications such as option pricing, portfolio choice, and risk management. Due to their importance, a large amount of recent research has addressed two related modeling issues: developing accurate time series models of asset returns and developing new methods for estimating these increasingly complicated models. Stochastic volatility models capture important empirical features such as mean-reversion in volatility and leverage effects and can also be extended to incorporate rare jump movements in returns (see, for example, Andersen, Benzoni, and Lund (2001) and Chernov, et al. (2003)). Despite the difficulties posed by these complicated models, there are now a wide-range of methods capable of estimating these models, including simulated and efficient methods of moments, simulated maximum likelihood, and Markov Chain Monte Carlo (MCMC).

In this paper we focus on a different aspect of the estimation problem: sequential estimation of parameters and states. This problem is essential for practical financial applications. For example, portfolio allocation and option pricing problems require that portfolio weights and option prices must be updated at a high frequency to reflect changes in volatility and the agent's views of the parameters. Despite its importance, the sequential problem has received little attention in literature in part due its difficulty.

In addition to the usual problems encountered when estimating models with stochastic volatility and jumps, sequential estimation adds an additional hurdle: computational cost. Most estimation approaches for models with stochastic volatility and jumps are computationally intensive and it is not practical to repeatedly apply these algorithms to perform sequential estimation. Thus, we require algorithms that can be applied sequentially on large data sets in a reasonable amount of computing time.

Recent research on sequential estimation focusses on alternative schemes for implementing sequential Bayesian inference. Bayesian methods are particularly attractive for sequential estimation for three reasons. First, the posterior distribution is, by definition,

the optimal filter. Second, an expansion of the posterior via Bayes rule provides a number of alternative ways of sampling the posterior, thus providing flexibility for alternative approaches to the problem. Third, Bayesian inference quantifies the uncertainty in the parameters and states which can be used for Bayesian decision makers and to construct finite sample confidence intervals.

In this paper, we focus on two related but different approaches for Bayesian sequential inference. The first, called practical filtering was developed in Johannes, Polson and Stroud (2001) and Stroud, Polson and Muller (2003) and uses an MCMC algorithm based on fixed-lag filtering. The practical filtering approach has been previously been applied in pure stochastic volatility models. The second approach, called particle filtering (e.g., Gordon, Salmond, and Smith (1993), Pitt and Shephard (1999)), has been applied in a wide range of setting for state variable filtering. Recently, Storvik (2002) extended the particle filtering approach to the case of sequential parameter estimation by relying on a low dimension set of sufficient statistics in the conditional posterior distribution and applied it in a number of simple cases. Stroud, Polson and Muller (2003) and Johannes, Polson and Stroud (2002) extended Storvik's approach to the case of stochastic volatility models.

In this paper, we further extend these methods to allow for jumps in returns and compare the relative merits of MCMC based and particle filtering based methods. Jumps add an additional challenge due to their rare nature. Since there are typically only a couple of jumps per year, it is especially difficult to learn about the parameters determining the jumps as most of the data provides little, if any, information about the jump process. We first perform some sampling experiments using simulated data to document the algorithms' efficiency and then apply the algorithms to real data using historical S&P 500 index returns. Real data examples are of particular interest as it provides insight regarding how the algorithms handle model misspecification, as the simple models we consider are likely misspecified.

The paper is outlined as follows. The next section discusses the sequential approach to estimation. Section 3 describes the particle and practical filter and provides the details of

the algorithms for the models we consider. Section 4 provides examples using simulated data and real data examples with S&P 500 returns. Section 5 concludes.

2 Estimation of Stochastic Volatility Jump Diffusions

The log stochastic volatility model is a benchmark for modeling stochastic volatility. However, recent research indicates that this model is not flexible enough to capture all of the empirical features of equity returns. For example, the model has a difficult time generating features such as the stock market crash of 1987. More formal evidence on shortcomings of pure stochastic volatility models are given in Andersen, Benzoni and Lund (2001), Eraker, Johannes and Polson (2003), and Chernov, Ghysels, Gallant and Tauchen (2003) document the importance of adding jumps in returns. There is also significant evidence from the index option pricing literature that jumps, in addition to stochastic volatility, are required to match observed prices (see, for example, Bakshi, Cao and Chen (1997), Bates (2000) and Pan (2002)).

We therefore add a discrete-time jump term in returns to the log stochastic volatility model

$$\begin{aligned} Y_{t+1} &= \sqrt{V_{t+1}}\varepsilon_{t+1} + J_{t+1}Z_{t+1} \\ \log(V_{t+1}) &= \alpha_v + \beta_v \log(V_t) + \sigma_v \eta_{t+1} \end{aligned}$$

where $P(J_t) = \lambda$, $Z_t \sim \mathcal{N}(\mu_z, \sigma_z^2)$, and ε_t and η_t are i.i.d. standard normal variables. Define $\Theta = (\lambda, \mu_z, \sigma_z, \alpha_v, \beta_v, \sigma_v)$ as the parameter vector, let $\psi = (\alpha_v, \beta_v)$ the volatility mean reversion parameters, and $X_t = \log(V_t)$ the log volatilities. We use the approach in Kim, et al. (1998) to approximate the distribution of $\log[(Y_t - J_t Z_t)^2]$ by a 7-component mixture of normals. The mixture indicators are I_t and (m_i^*, v_i^*, π_i^*) for $i = 1, \ldots, 7$ are the mixture parameters. We also define the collection of observations and state variables by $Y_{1,t} = (Y_1, \ldots, Y_t)$, $V_{1,t} = (V_1, \ldots, V_t)$, $X_{1,t} = (X_1, \ldots, X_t)$, $J_{1,t} = (J_1, \ldots, J_t)$, $Z_{1,t} = (Z_1, \ldots, Z_t)$, and $I_{1,t} = (I_1, \ldots, I_t)$.

Given a time series of observations, $Y_{1,T}$, the usual estimation problem is to estimate the parameters, Θ, and the unobserved states, $L_{1,T}$, from the observed data. In our case, the latent variables include the jump times, jump sizes and volatility states. In a Bayesian setting, this information is summarized by the posterior distribution, $p(\Theta, L_{1,T}|Y_{1,T})$. Samples from this distribution are usually obtained via MCMC methods by iteratively sampling from the complete conditional distributions, $p(L_{1,T}|\Theta, Y_{1,T})$ and $p(\Theta|L_{1,T}, Y_{1,T})$. From these samples, it is straightforward to obtain smoothed estimates of the parameters and states. For example, the posterior mean for the parameters or for volatility, is estimated as

$$E[\Theta|Y_{1,T}] \approx \frac{1}{G}\sum_{g=1}^{G}\Theta^{(g)}$$

and

$$E[V_t|Y_{1,T}] \approx \frac{1}{G}\sum_{g=1}^{G}V_t^{(g)}$$

where G is the number of samples generated in the MCMC algorithm, $\Theta^{(g)}$ is the parameter draw and $V_t^{(g)}$ is variance draw from the g^{th} iteration. The Ergodic Theorem for Markov Chains provides the limiting theorem justifying the Monte Carlo estimates.

It is important to recognize the smoothed nature of these estimators. When estimating volatility, the estimator uses the information embedded in the entire sample. Thus, for example, to estimate V_t, the posterior uses information in the entire data set. When volatility is persistent, it is clear that both future and past information is informative about V_t. However, for practical applications, researchers do not have the luxury of waiting to receive tomorrow's data to estimate today's volatility. They estimate the volatilities based on current information in a timely and efficient manner.

This sequential problem is solved by sequentially computing $p(\Theta, L_t|Y_{1,t})$ for $t = 1, ..., T$. This is the online estimation procedure and we stress that methods must be able to compute these distributions in practice and not only in theory. For example, in theory one could estimate this density as a marginal from $p(\Theta, L_{1,t}|Y_{1,t})$, which, in turn, can be computed by repeatedly applying standard MCMC algorithms. However, for large t, a MCMC algorithm,

efficiently programmed might take, for example, 5 minutes to compute. Repeating this hundreds or thousands of times for large daily data sets is clearly is not computationally feasible.

Both of the practical and particle filtering algorithms approximate the true density, $p\left(\Theta, L_t|Y_{1,t}\right)$. The particle filter approximates this density via a discretization whereby the distribution of (Θ, L_t) is approximated by a finite set of particles. The practical filter, on the other hand, approximates a conditional density in the MCMC algorithm, effectively limiting the influence that observations in the distant past can have regarding the current state.

Before discussing these algorithms in detail, we must specify the full conditional posterior distributions. The joint posterior for the states and parameters is

$$\begin{aligned} p(J_{1,t}, Z_{1,t}, X_{0,t}, \Theta|Y_{1,t}) &\propto p(Y_{1,t}|J_{1,t}, Z_{1,t}, X_{0,t})\ p(J_{1,t}|\Theta)\ p(Z_{1,t}|\Theta)\ p(X_{0,t}|\Theta)\ p(\Theta) \\ &= \prod_{\tau=1}^{t} p(Y_\tau|J_\tau, Z_\tau, X_\tau)\ p(J_\tau|\Theta)\ p(Z_\tau|\Theta)\ p(X_\tau|X_{\tau-1}, \Theta)\ p(\Theta). \end{aligned}$$

where $p\left(\Theta\right)$ is the prior distribution of the parameters. We assume the following conjugate priors for the parameters

$$\begin{aligned} \lambda &\sim Beta(S_0, F_0) \\ (\mu_z, \sigma_z^2) &\sim \mathcal{N}(\mu_z|m_0, k_0^{-1}\sigma_z^2)\ \mathcal{IG}(\sigma_z^2|a_0, b_0) \\ (\psi, \sigma_v^2) &\sim \mathcal{N}(\psi|\psi_0, \Psi_0^{-1}\sigma_v^2)\ \mathcal{IG}(\sigma_v^2|c_0, d_0). \end{aligned}$$

Given the conjugate priors, the complete parameter posterior conditionals are

$$\begin{aligned} p(\lambda|\ldots) &\propto p(J_{1,t}|\lambda)\ p(\lambda) = Beta(S_t, F_t) \\ p(\mu_z, \sigma_z^2|\ldots) &\propto p(Z_{1,t}|\mu_z, \sigma_z^2)\ p(\mu_z, \sigma_z^2) = \mathcal{N}(\mu_z|m_t, k_t^{-1}\sigma_z^2)\ \mathcal{IG}(\sigma_z^2|a_t, b_t) \\ p(\psi, \sigma_v^2|\ldots) &\propto p(X_{0,t}|\psi, \sigma_v^2)\ p(\psi, \sigma_v^2) = \mathcal{N}(\psi|\psi_t, \Psi_t^{-1}\sigma_v^2)\ \mathcal{IG}(\sigma_v^2|c_t, d_t) \end{aligned}$$

where for notational simplicity $p(y|\ldots)$ refers to the conditional distribution of y given all

other relevant variables. For the latent variables,

$$\begin{aligned}
p(J_t|\ldots) &\propto p(Y_t|J_t,Z_t,X_t)\; p(J_t|\lambda) \;=\; Bern(\lambda_t)\\
p(Z_t|\ldots) &\propto p(Y_t|J_t,Z_t,X_t)\; p(Z_t|\mu_z,\sigma_z^2) \;=\; \mathcal{N}(\mu_{z,t},\sigma_{z,t}^2)\\
p(X_{0,t}|\ldots) &\propto p(Y_{1,t}|J_{1,t},Z_{1,t},X_{1,t})\; p(X_{0,t}|\psi,\sigma_v^2) \;=\; \text{(Not recognizable)}\\
\tilde{p}(X_{0,t}|\ldots) &\propto \tilde{p}(Y_{1,t}|J_{1,t},Z_{1,t},X_{1,t},I_{1,t})\; p(X_{0,t}|\psi,\sigma_v^2) \;=\; \mathcal{N}(\mu_x,\Sigma_x)\ \text{(FFBS)}\\
\tilde{p}(I_t|\ldots) &\propto \tilde{p}(Y_t|J_t,Z_t,X_t,I_t)\; \tilde{p}(I_t) \;=\; Mult(\pi_{1,t}^*,\ldots,\pi_{7,t}^*).
\end{aligned}$$

Note that the distribution $p(X_{0,t}|...)$ is not a known distribution and FFBS refers to the forward-filtering, backward-sampling algorithm (see Johannes and Polson (2002) for a description of the details). For completeness, we now give analytic forms for the parameters in the conditional posteriors. Let $S = \sum_{\tau=1}^{t} J_\tau$ be the number of jumps, and $\hat{\psi} = (H^TH)^{-1}H^TX$, where $H = (H_1,\ldots,H_t)^T$ and $H_t = (1, X_{t-1})^T$ and $X = X_{1,t}$. We also denote by $Y_t^* = \log\left[(Y_t - J_tZ_t)^2\right]$ the transformed observation in the Kim et al. (1998) model. The updating recursions for the sufficient statistics are given below.

$$\begin{aligned}
\lambda_t &= \frac{\lambda\mathcal{N}(Y_t|\mu_z, V_t+\sigma_z^2)}{\lambda\mathcal{N}(Y_t|\mu_z, V_t+\sigma_z^2) + (1-\lambda)\mathcal{N}(Y_t|0,V_t)}\\
\mu_{z,t} &= \sigma_{z,t}^2\left((\sigma_z^2)^{-1}\mu_z + J_tY_tV_t^{-1}\right),\ \sigma_{z,t}^2 = \left((\sigma_z^2)^{-1} + J_tV_t^{-1}\right)^{-1}\\
Y_t^* &= X_t + m_{I_t}^* + \sqrt{v_{I_t}^*}\epsilon_t^*\\
X_{t+1} &= \alpha_v + \beta_vX_t + \sigma_v\eta_{t+1}\\
\pi_{t,i}^* &= \frac{\pi_i^*\,\mathcal{N}\left(Y_t^*|X_t+m_i^*, v_i^*\right)}{\sum_{j=1}^{7}\pi_j^*\,\mathcal{N}\left(Y_t^*|X_t+m_j^*, v_j^*\right)}\\
S_t &= S_0 + S,\ F_t = F_0 + t - S\\
m_t &= k_t^{-1}(k_0m_0 + \sum_{\tau=1}^{t} J_\tau Z_\tau),\ k_t = k_0 + S\\
a_t &= a_0 + S_t,\ b_t = b_0 + \sum_{\tau=1}^{t} J_\tau(Z_\tau - \bar{Z})^2\\
\psi_t &= \Psi_t^{-1}(\Psi_0\psi_0 + H\hat{\psi}),\ \Psi_t = \Psi_0 + HH^T\\
c_t &= c_0 + t,\ d_t = d_0 + \sum_{\tau=1}^{t}(X_\tau - H_\tau\hat{\psi})^2
\end{aligned}$$

The only complication in the conditional structure of the model is the conditional posterior for the log volatility states, X_t. As noted above, the conditional posterior, $p(X_{0,t}|\ldots)$, is not a known distribution. There are two ways to deal with this. First, following Jacquier, Polson and Rossi (1994) we could break this $t-$dimensional conditional into a set of $1-$dimensional distribution and perform single-state updating. The algorithm provides accurate inference, but convergence tends to be slow. For the practical filtering algorithm to run quickly, we must be able to quickly draw the states and generate an algorithm that converges rapidly. For this reason, we consider the approximation of Kim, et. al. (1998).

3 Approaches for Sequential Estimation

3.1 Particle Filtering

Consider a discrete time setting where we refer to L_t as the latent variables, Y_t the observed prices, and $Y_{0,t} = (Y_1, ..., Y_t)'$ as the vector of observed states up to time t. There are a number of densities associated with the filtering problem which we now define:

$$\begin{aligned} p\left(L_t|Y_{1,t}\right) &: \text{ filtering density} \\ p\left(L_{t+1}|Y_{0,t}\right) &: \text{ predictive density} \\ p\left(Y_t|L_t\right) &: \text{ likelihood} \\ p\left(L_{t+1}|L_t\right) &: \text{ state transition} \end{aligned}$$

Bayes rule links the predictive and filtering densities through the identity

$$p\left(L_{t+1}|Y_{1,t+1}\right) = \frac{p\left(Y_{t+1}|L_{t+1}\right) p\left(L_{t+1}|Y_{0,t}\right)}{p\left(Y_{t+1}|Y_{0,t}\right)}$$

where

$$p\left(L_{t+1}|Y_{1,t}\right) = \int p\left(L_{t+1}|L_t\right) p\left(L_t|Y_{1,t}\right) dL_t.$$

Particle filtering, also known as the bootstrap filter, was first formally introduced in Gordon, Salmond, and Smith (1993), who also discussed the problem of sequential parameter learning. We refer the reader to the edited volume by Doucet, de Freitas, and Gordon (2001) for a detailed discussion of the historical development of the particle filter, convergence theorems and potential improvements.

The key to particle filtering is an approximation of the (continuous) distribution of the random variable L_t conditional on $Y_{0,t}$ by a discrete probability distribution, that is, the distribution $L_t|Y_{0,t}$ is approximated by a set of particles, $\left\{L_t^{(i)}\right\}_{i=1}^{N}$ with probability $\pi_t^1, ..., \pi_t^N$. By assuming the distribution is approximated with particles, we can estimate the filtering and predictive densities via: (p^N refers to an estimated density)

$$p^N\left(L_t|Y_{1,t}\right) = \sum_{i=1}^{N} \delta_{L_t^{(i)}} \pi_t^i$$

$$p^N\left(L_{t+1}|Y_{0,t}\right) = \sum_{i=1}^{N} p\left(L_{t+1}|L_t^{(i)}\right) \pi_t^i,$$

where δ is the Dirac function. As the number N of particles increases, the accuracy of the discrete approximation to the continuous random variable improves. When combined with the conditional likelihood, the filtering density at time $t+1$ is defined via the recursion:

$$p^N\left(L_{t+1}|Y_{1,t+1}\right) \propto p\left(Y_{t+1}|L_{t+1}\right) \sum_{i=1}^{N} p\left(L_{t+1}|L_t^{(i)}\right) \pi_t^i.$$

As pointed out in Gordon, Salmond and Smith (1993), the particle filter only requires that the likelihood function, $p\left(Y_{t+1}|L_{t+1}\right)$, can be evaluated and the states can be sampled from their conditional distribution, $p\left(L_{t+1}|L_t\right)$. Given these mild requirements, the particle filter applies in an extremely broad class of models, including nearly all state space models of practical interest. The key to the particle filtering is to propagate particles with high importance weights and to develop an efficient algorithm for propagating particles forward from time t to time $t+1$. We use the sampling/importance resampling procedure of Smith and Gelfand (1992). In practice, this procedure can be improved for many applications using additional sampling methods such as those introduced in Carpenter, Clifford,

and Fearnhead (1999) and Pitt and Shephard (1999). We use the auxiliary particle filter approach of Pitt and Shephard (1999).

Storvik (2002) has developed an extension of the particle filter that applies to states and parameters in certain cases. The key assumption is that the marginal posterior distribution for the parameters, $p(\Theta|L_{1,t}, Y_{1,t})$, is analytically tractable and depends on the observed data and latent variables through a set of sufficient statistics which is straightforward to update. For example, in a jump model, conditional on the latent states, the jump intensity posterior depends only on total number of jumps, in this case a natural sufficient statistic.

If we denote $s_{t+1} = S(s_t, L_t, Y_t)$ as the sufficient statistic which can be computed using the previous sufficient statistic, s_t, as well as the previous prices and states, the particle filtering algorithm would consist of the following steps. First, assume a particle representation of the joint distribution, $(\Theta, L_t) \sim p(\Theta, L_t|Y_{1,t})$. Second, the algorithm then draws

$$\Theta \sim p(\Theta|s_t) \text{ and } L_{t+1} \sim p(L_{t+1}|L_t, \Theta)$$

and then finally re-weights (Θ, L_{t+1}) with weights proportional to the observation equation, $p(Y_{t+1}|L_{t+1}, \Theta)$.

Consider the following algorithm:

1. Initialization: given N initial particles representing the latent states, parameters and sufficient statistics, $\left(\Theta^{(g)}, L_t^{(g)}\right)$ and $\left(s_t^{(g)}\right)$, and let $\omega_t^{(g)}$ be the associated weights.
2. Sequential updating: for each re-sampled particle:
   (a) generate $\Theta^{(g)} \sim p\left(\Theta|s_t^{(g)}\right)$
   (b) generate $L_{t+1}^{(g)} \sim p\left(L_{t+1}|L_t^{(g)}, \Theta^{(g)}\right)$
   (c) update the sufficient statistics, $s_{t+1} = S\left(s_t^{(g)}, L_{t+1}^{(g)}, Y_{t+1}\right)$
   (d) Compute updated weights $w_{t+1}^i = w_t^i p(Y_t|L_{t+1}^i)$.

3. Resample the particles L_{t+1}^i with probabilities proportional to w_{t+1}^i.

In addition, we use the auxiliary particle filter of Pitt and Shephard (1998) between steps 1 and 2. In our experience, this approach is most helpful when there is some misspecification and the auxiliary particle filter prevents the algorithm from getting stuck.

Details of the particle filtering algorithm To apply particle filtering algorithm from above to the jump diffusion model, we need to specify the sufficient statistics which naturally arise in the conditional posteriors. The complete algorithm is given by:

1. For $i = 1, \ldots, N$: initialize $s_0^i = (S_0, F_0, m_0, k_0, a_0, b_0, \psi_0, \Psi_0, c_0, d_0)$ and generate $X_0^i \sim p(X_0)$.

2. For $t = 1, \ldots, T$ and $i = 1, \ldots, N$:

   (a) Generate $\lambda^i \sim p(\lambda | X_{0,t-1}^i, J_{0,t-1}^i, Z_{0,t-1}^i, Y_{1,t}) = p(\lambda | s_{t-1}^i)$

   (b) Generate $(\mu_z^i, \sigma_z^i) \sim p(\mu_z, \sigma_z | X_{0,t-1}^i, J_{0,t-1}^i, Z_{0,t-1}^i, Y_{1,t}) = p(\mu_z, \sigma_z | s_{t-1}^i)$

   (c) Generate $(\psi^i, \sigma_v^i) \sim p(\psi, \sigma_v | X_{0,t-1}^i, J_{0,t-1}^i, Z_{0,t-1}^i, Y_{1,t}) = p(\psi, \sigma_v | s_{t-1}^i)$

   (d) Generate $J_t^i \sim p(J_t | \lambda^i)$

   (e) Generate $Z_t^i \sim p(Z_t | \mu_z^i, \sigma_z^i)$

   (f) Generate $X_t^i \sim p(X_t | X_{t-1}^i, \psi^i, \sigma_v^i)$

   (g) Update sufficient statistics $s_t^i = s(s_{t-1}^i, J_t^i, Z_t^i, X_t^i)$

   (h) Update augmented particles $\tilde{X}_t^i = (J_t^i, Z_t^i, X_t^i, \Theta^i, s_t^i)$

   (i) Compute weights $w_t^i = w_{t-1}^i p(Y_t | J_t^i, Z_t^i, X_t^i)$.

3. Resample particles $\tilde{X}_t^i$ with probabilities proportional to w_t^i.

As mentioned above, between steps 1 and 2 we use an auxiliary step to "peak ahead" to improve the performance of the algorithm. All of these steps are straightforward given

that the parameter posteriors are recognizable distributions and the state transitions are easy to simulate.

3.2 Practical Filtering

To understand the practical filter, we first describe the generic MCMC algorithm and then discuss the development of the practical filter in the case of SVJ models. Consider the following MCMC algorithm: given $\Theta^{(g)}$ and $L_{1,t}^{(g)}$, draw

$$\Theta^{(g+1)} \sim p\left(\Theta | L_{1,t}^{(g)}, Y_{1,t}\right)$$
$$L_{1,t}^{(g+1)} \sim p\left(L_{1,t} | \Theta^{(g+1)}, Y_{1,t}\right)$$

the last step usually consists of separately drawing jump times, sizes and volatilities in blocks:

$$J_{1,T}^{(g+1)} \sim p\left(J_{1,t} | \Theta^{(g+1)}, Z_{1,t}^{(g)}, V_{1,t}^{(g)}, Y_{1,t}\right)$$
$$Z_{1,t}^{(g+1)} \sim p\left(Z_{1,t} | \Theta^{(g+1)}, V_{1,t}^{(g)}, J_{1,t}^{(g+1)}, Y_{1,t}\right)$$
$$V_{1,t}^{(g+1)} \sim p\left(V_{1,t} | \Theta^{(g+1)}, Z_{1,t}^{(g+1)}, J_{1,t}^{(g+1)}, Y_{1,t}\right).$$

For large G, these samples are draws from $p\left(\Theta, V_{1,t}, Z_{1,t}, J_{1,t} | Y_{1,t}\right)$.

The practical filter relies on the following decomposition of the joint distribution of parameters and states:

$$p\left(\Theta, L_t | Y_{1,t}\right) = \int p\left(\Theta, L_t | L_{1,t-k}, Y_{1,t}\right) p\left(L_{1,t-k} | Y_{1,t}\right) dX_{1,t-k}.$$

This decomposition shows that the filtering distribution is a mixture of the lag-filtering distribution, $p\left(L_{1,t-k} | Y_{1,t}\right)$.

This suggests the following approximate filtering algorithm:

1. Initialization: for $g = 1, ..., G$, set $\Theta^{(g)} = \Theta_0$ where Θ_0 are the initial values of the chain.

2. Burn-in (initial smoothing step): for $t = 1, ..., t_0$ and for $g = 1, ..., G$, simulate $(\Theta, L_{1,k}) \sim p(\Theta, L_{1,k}|Y_{1,t})$. Set $\left(\Theta^{(g)}, \widetilde{L}^{(g)}_{0,t-k}\right)$ equal to the last imputed $\left(\Theta, \widetilde{L}_{0,t-k}\right)$.

3. Sequential updating: for $t = t_0 + 1, ..., T$ and for $g = 1, ..., G$ generate

$$L_{t-k+1,t} \sim p\left(L_{t-k+1,t}|\Theta, \widetilde{L}^{(g)}_{0,t-k}, Y_{t-k+1,t}\right)$$
$$\Theta \sim p\left(\Theta|\widetilde{L}^{(g)}_{0,t-k}, L_{t-k+1,t}, Y_{1,t}\right)$$

and set $\left(\Theta, \widetilde{L}^{(g)}_{t-k+1}\right)$ equal to the last imputed (Θ, L_{t-k+1}) pair and leave $\widetilde{L}^{(g)}_{t-k}$ unchanged.

There are three separate issues that effect the efficiency and accuracy of the algorithm. First, in theory, as k increases, the algorithm will uncover the true density as the approximation disappears. However, the computational costs increase with k and therefore in principle one would prefer, if possible to choose a small k. Second, for each time step t, we need to make G draws from posterior and G must be sufficiently large so that we can safely assume that the algorithm has converged. Therefore, it is important to construct an efficient algorithm in the sense that it converges very quickly to its equilibrium distribution. Third, at each stage, it is helpful if the draws from the conditional posteriors are exact, that is, that the algorithm uses the Gibbs sampler.

Details of the algorithm For completeness, we now provide the details of the algorithm for the stochastic volatility jump-diffusion model given above:

1. For $g = 1, \ldots, G$, generate $(\Theta^{(g)}, X^{(g)}_{0,1}, J^{(g)}_1, Z^{(g)}_1) \sim p(\Theta, X_{0,1}, J_1, Z_1)$.

2. For $t = 1, \ldots, t_0$ and $g = 1, \ldots, G$

   (a) Set $\Theta^0 = \Theta^{(g)}$ and $(J^0_{1,t}, Z^0_{1,t}) = (0, 0)$.

   (b) For $i = 1, \ldots, I$:

i. Generate $X^i_{0,t} \sim p(X_{0,t}|J^{i-1}_{1,t}, Z^{i-1}_{1,t}, \Theta^{i-1}, Y_{1,t})$

ii. Generate $J^i_{1,t} \sim p(J_{1,t}|X^i_{0,t}, Z^{i-1}_{1,t}, \Theta^{i-1}, Y_{1,t})$

iii. Generate $Z^i_{1,t} \sim p(Z_{1,t}|X^i_{0,t}, J^i_{1,t}, \Theta^{i-1}, Y_{1,t})$

iv. Generate $\Theta^i \sim p(\Theta|X^i_{0,t}, J^i_{1,t}, Z^i_{1,t}, Y_{1,t})$

(c) Set $(\Theta^{(g)}, \tilde{X}^{(g)}_0) = (\Theta^I, X^I_0)$.

3. For $t = t_0 + 1, \ldots, T$:

(a) For $g = 1, \ldots, G$, set $\Theta^0 = \Theta^{(g)}$ and $(J^0_{t-k+1,t}, Z^0_{t-k+1,t}) = (0, 0)$.

(b) For $i = 1, \ldots, I$

i. Generate $X^i_{t-k+1,t} \sim p(X_{t-k+1,t}|\tilde{X}^{(g)}_{t-k}, J^{i-1}_{t-k+1,t}, Z^{i-1}_{t-k+1,t}, \Theta^{i-1}, Y_{t-k+1,t})$

ii. Generate $J^i_{t-k+1,t} \sim p(J_{t-k+1,t}|X^i_{t-k+1,t}, Z^{i-1}_{t-k+1,t}, \Theta^{i-1}, Y_{t-k+1,t})$

iii. Generate $Z^i_{t-k+1,t} \sim p(Z_{t-k+1,t}|X^i_{t-k+1,t}, J^i_{t-k+1,t}, \Theta^{i-1}, Y_{t-k+1,t})$

iv. Generate $\Theta^i \sim p(\Theta|\tilde{X}^{(g)}_{0,t-k}, X^i_{t-k+1,t}, J^i_{t-k+1,t}, Z^i_{t-k+1,t}, Y_{1,t})$

(c) Set $(\Theta^{(g)}, X^{(g)}_{t-k+1}) = (\Theta^I, X^I_{t-k+1})$.

4 Applications

We consider two applications of the algorithms: one with simulated data and one with S&P 500 index data.

4.1 Simulated data

We simulated 1000 daily observations using the following parameter values:

$$\begin{aligned} \text{Jump Process:} \quad & \lambda = 0.01, \mu_z = -0.04, \sigma_z = 0.05 \\ \text{Volatility Process} \quad : \quad & \alpha_v = 0, \beta_v = 0.98, \sigma_v = 0.1. \end{aligned}$$

These parameters are roughly consistent with observed equity return data, see, for example, Johannes, Kumar, and Polson (1998). As in Stroud, Polson and Muller (2002), we had difficulty learning the volatility of volatility parameter, and so we kept this parameter fixed throughout for both models.

Figures 1 and 2 display the sequential posterior summaries for the particle and practical filter, respectively. The same data were used for both filtering approaches. The top of each subplot indicates the state variable or parameter. For the volatility (annualized) and the parameters, the plots contain the posterior median and the (2.5,97.5)% posterior quantiles, while the plots for the jump times and sizes contain the true jump times or size (dots) and the posterior mean. In the case of jump times, it is posterior probability that a jump occurred.

A number of points emerge. First, both algorithms are able to successfully identify nearly all of the jumps. The only jump that was substantively missed was at data point 40 and was about -3.75%. Both algorithms identified a small jump, less than 1%, with 20% probability. As daily volatility was more than 1%, it is not difficult for the model to generate this move with negative shock to ε_t and a small jump. Effectively, the jump was too small for the algorithm to detect it. Moreover, the parameter posteriors for the jump process the algorithm were not very informative at this early stage in the algorithm and thus it was not able to identify the move as a jump. Both algorithms obtained similar estimates for the other jump times and sizes.

Second, the jump parameter posteriors appear to be collapsing nicely to their true values, for both algorithms. While we begin with relatively uninformative priors, for example, for the jump mean the (2.5,97.5)% confidence band is roughly (7,-11)%, it is evidence from the rapid revisions in jump parameter posteriors that the posterior quickly updates. For example, at approximately data point 250 a large jump, about -8%, arrived that was correctly identified by the algorithm. At the same time, the posterior means for the jump mean, the jump variance, and the jump intensity all decreased with the posterior variances falling also. Even though jumps are rare as the jump intensity is one percent, the sequential

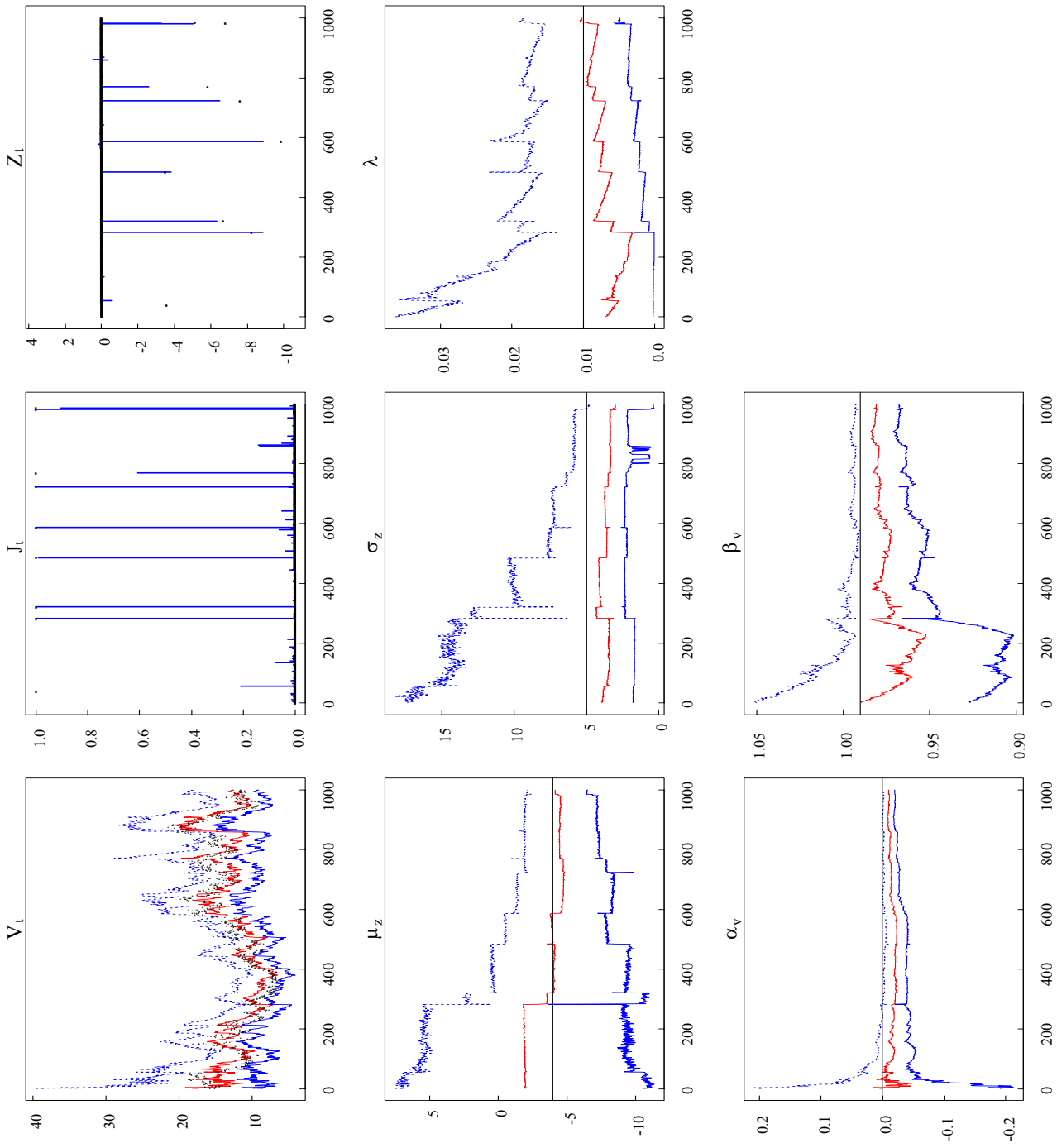

Figure 1: Sequential particle filtering estimates for 1000 simulated data points. The particle filter for N=25,000 particles. The algorithm took 8 minutes to run.

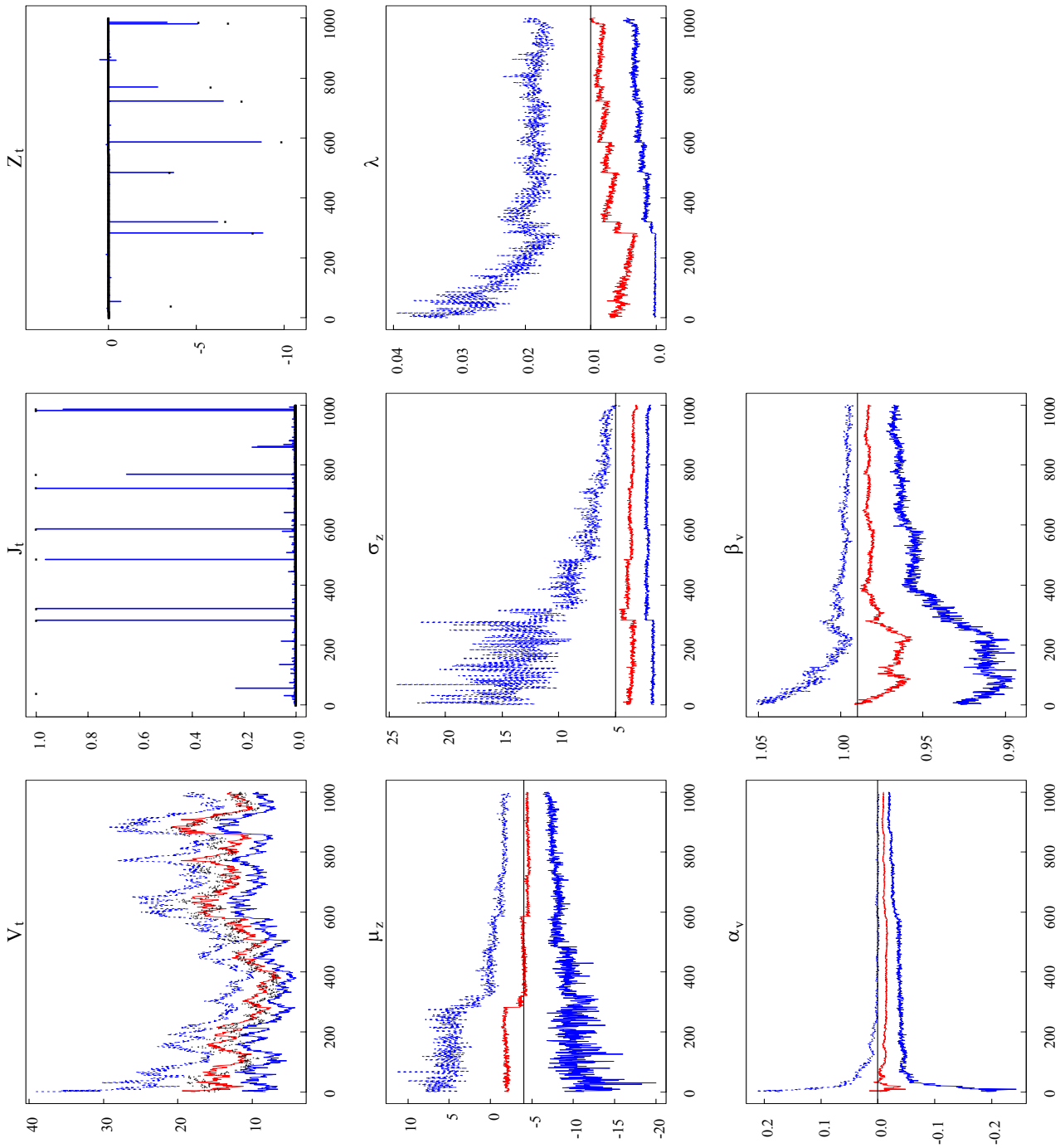

Figure 2: Sequential practical filter estimates for 1000 simulated data points. The practical filter was run with $G = 250$, $I = 10$, and $k = 25$. The algorithm took 8 minutes to run.

estimates are able to accurately estimate the parameters even with the small samples.

Third, the volatility parameter estimates are similar to those in Stroud, Polson and Muller (2003). The speed of mean reversion is accurately estimated and the estimates of α_v are slightly downward biased, as is common in the literature. Fourth, as indicated in the Figures, both algorithms are extremely computationally efficient as each algorithm takes about 8 minutes. In the case of the practical filter, we chose the combinations of $G = 250$, $I = 10$, and $k = 25$ so that the computing time was roughly equal in computing time to the particle filter. The computational efficiency of the algorithms imply that for practical applications, one could likely drastically increase N, G, I and k to obtain more accurate approximations to the posterior while still retaining computationally feasibly algorithms.

Finally, a general comparison of the two algorithms indicates that the particle filter posteriors are much smoother than those of the practical filter. Stroud, Polson and Muller (2003) found that the practical filter posteriors were more accurate when compared to the true posterior (as estimated by full sample MCMC) than those of the particle filter. It is very likely that the practical filter is more efficiently exploring the posterior distribution than the particle filter. A more detailed simulation design is required for more concrete conclusions regarding the relative merits of these algorithms on simulated data.

4.2 S&P 500

To analyze the performance of the algorithms using real data, we consider daily S&P 500 index returns from 1984-2000. As in the previous case, we set $\sigma_v = 0.10$ and learned the other parameters. The S&P data set offers an additional challenge as it is roughly four times as large as the simulated data. If there are degeneracies in the algorithms, we are likely to see them more clearly in the longer time series.

Figures 3 and 4 summarize the sequential estimation using particle and practical filter, respectively. Unlike the simulated data examples in the previous sections, the two algorithms now generate some fundamental differences. First, with regard to the Crash of

1987, the two algorithms generate different state variable estimates. On October 19 and 20, 1987, the particle filter estimates jump sizes of -12% and 9% while the practical filter estimates them to be -22% and +8% (the actual moves were -22 and 9%). In addition, the 97.5^{th} quantile for the volatility state peaked at about 50% for the particle filter but was over 60% for the practical filter. Since daily volatility was less than 2% and the jump contribution was only -12% using the particle filter, this implies that the model required more than a 5 standard deviation shock in ε_t to generate the Crash. While possible, it is highly unlikely and we conjecture that the particle filter was not able to simulate enough particles that generated large negative jump sizes. As pointed out by Pitt and Shephard (1999) particle filtering algorithms can have difficulties dealing with outliers.

Second, the parameter posteriors for μ_z, σ_z and λ are substantively different. For example, the posterior mean for μ_z with the practical filter is more negative and has fewer spikes that the corresponding one from the particle filter. Similarly, the posterior median for σ_z is higher for the practical filter. Most noticeable, the posterior confidence bands for λ are much wider with the practical filter. This is similar to some of the findings in Stroud, Polson and Muller (2003) who attribute it to a more accurate representation by the practical filter. Finally, the results are similar for the two approaches for α_v and β_v. In conclusion, a comparison of the two algorithms indicates they have important differences, and we conjecture that for the given parameters and simulation scheme (choice of G, N, etc.), that the practical filter more thoroughly samples from the posterior distribution.

5 Conclusions

This paper extends existing sequential algorithms to the case of stochastic volatility jump-diffusion models. We find that both practical and particle filtering provide accurate inference for simulated data, while the two approaches generate substantive differences for the S&P 500 data.

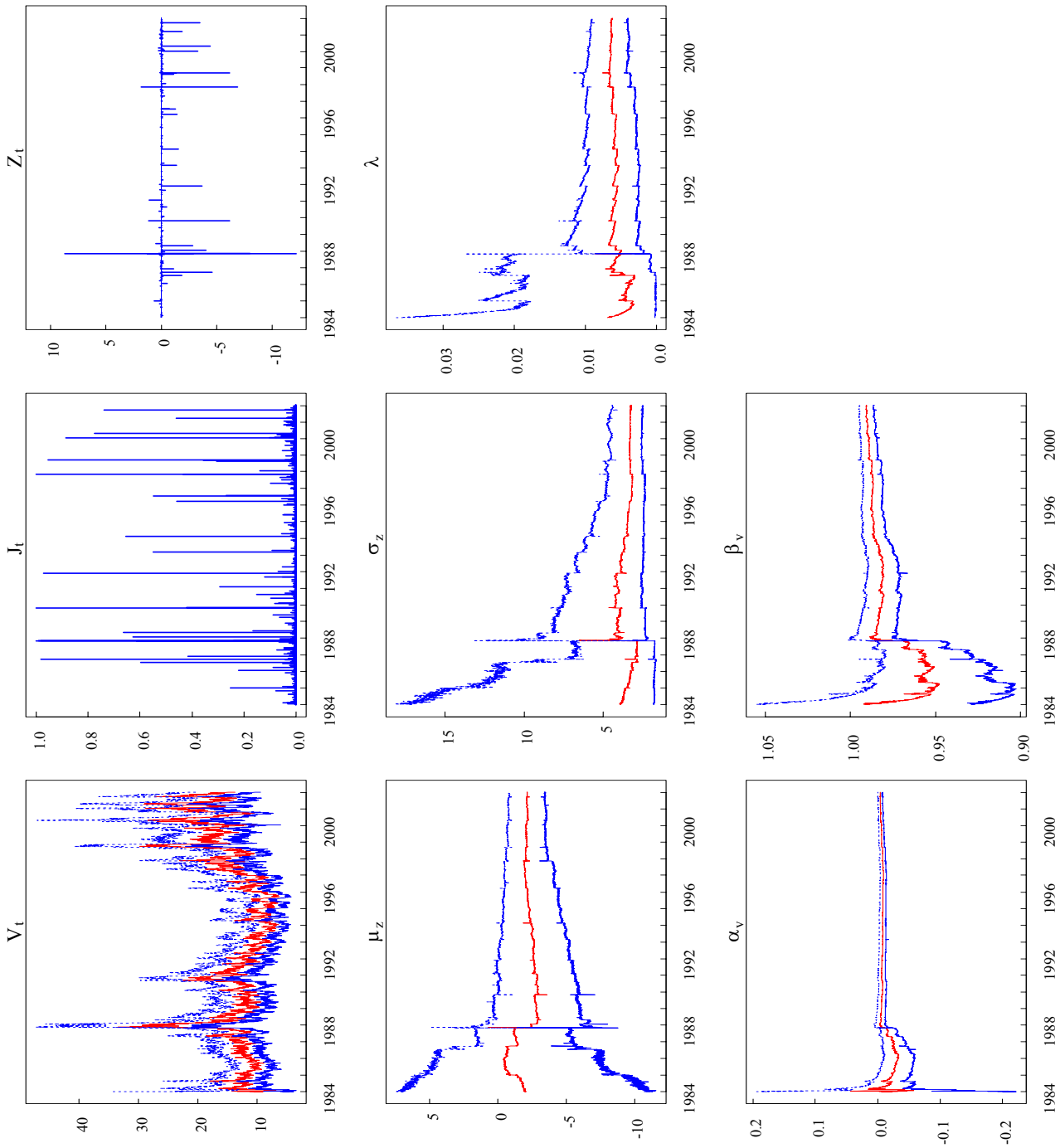

Figure 3: Sequential particle filtering estimates for S&P 500 index returns from 1984-2000. The particle filter was run with $N = 25,000$ particles. The algorithm took approximately 35 minutes to run.

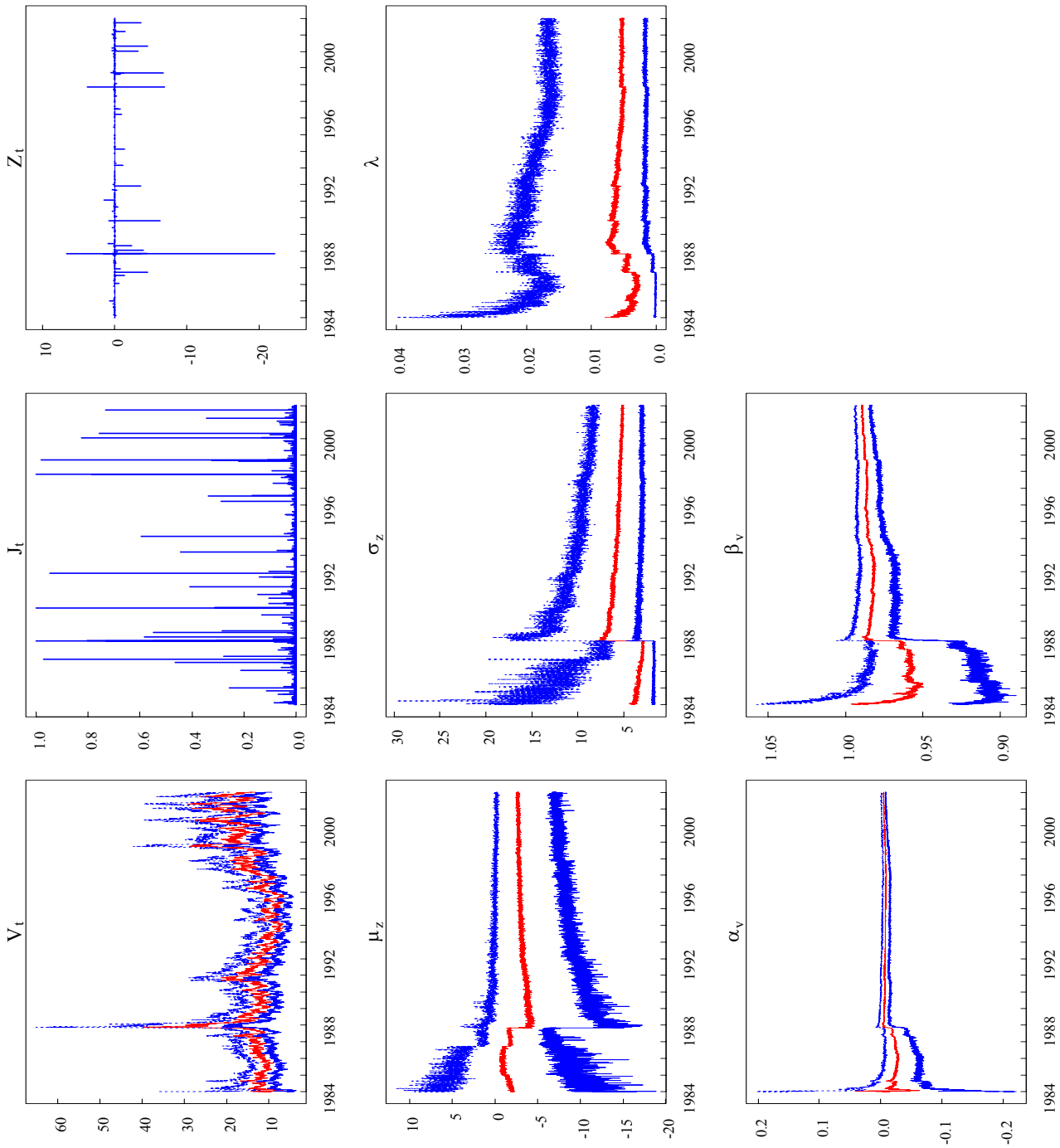

Figure 4: Sequential practical filter estimates for S&P 500 data from 1984-2000. The practical filter was run with $G = 250$, $I = 10$, and $k = 25$. The algorithm took approximately 37 minutes to run.

In the future, we plan a number of extensions. First, we plan more detailed simulation experiments to try and further identify differences in the algorithms. For example, do either of the algorithms rapidly degenerate at the size of the data set increases? How do the algorithms perform with other choices for the parameters? Given the computational efficiency of the algorithms, these simulations experiments are clearly feasible. Second, there are a number of potential extensions which are straightforward to perform. Like Johannes, Polson and Stroud (2002), we could consider continuous-time models and augment the state vector by filling in missing data. Also, it would be useful to consider more general models such those with square-root stochastic volatility and/or jumps in volatility.

Finally, as in Polson, Stroud, and Muller (2003), we found it difficult to learn certain parameters, namely σ_v. There are at least two potential causes of this problem. The marginal posterior for σ_v appears to have some long-memory properties, that is, data points far in the past have a strong influence on σ_v. This implies that the mixing assumption in the practical filter that observations past k-lags have little influence beyong the sufficient statistics may not be appropriate. Another potential cause could lie in the high posterior correlation between σ_v and β_v. One potential cause of this is the prior on β_v which is normally distributed and places positive probability that $\beta_v \geq 1$. It would be interesting to investigate the links between stationarity assumptions on β_v and estimating σ_v by imposing, for example, a normal prior truncated above at $\beta_v = 1$. Alternatively, there may different parameterizations or simulation steps that can be implemented to improve the algorithms performance with respect to σ_v.

References

Andersen, Torben, Luca Benzoni, and Jesper Lund, 2001, Towards an empirical foundation for continuous-time equity return models, *Journal of Finance* 57, 1239 - 1284.

Bakshi, Gurdip, Charles Cao, and Zhiwu Chen, 1997, Empirical performance of alternative option pricing models, *Journal of Finance* 52, 2003-2049.

Bates, David, 2000, Post-'87 Crash fears in S&P 500 futures options, *Journal of Econometrics* 94, 181-238.

Carpenter, J., , Peter Clifford., and Paul Fearnhead, 1999, An Improved Particle Filter for Nonlinear Problems. *IEE Proceedings – Radar, Sonar and Navigation*, 146, 2–7.

Chernov, Mikhail, Eric Ghysels, A. Ronald Gallant, and George Tauchen, (2003), Alternative models for stock price dynamics. Forthcoming, *Journal of Econometrics*.

Chib, Siddhartha, Federico Nardari and Neil Shephard, 2002, Markov Chain Monte Carlo methods for stochastic volatility models. *Journal of Econometrics*, 108, 281-316.

Doucet, Arnoud, Nando de Freitas, and Neil Gordon, 2001, *Sequential Monte Carlo Methods in Practice*, New York: Springer-Verlag, Series Statistics for Engineering and Information Science.

Eraker, Bjorn, Michael Johannes and Nicholas Polson, 2003, The impact of jumps in volatility and returns." *Journal of Finance 58*, 1269-1300.

Gordon, Neil, D. Salmond and Adrian Smith, 1993, Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEE Proceedings*, F-140, 107–113.

Jacquier, Eric, Nicholas Polson, and Peter Rossi, 1994, Bayesian analysis of Stochastic Volatility Models, (with discussion). *Journal of Business and Economic Statistics* 12, 371-417.

Jacquier, Eric, Nicholas Polson, and Peter Rossi, 2001, Models and Priors for Multivariate Stochastic Volatility, forthcoming, *Journal of Econometrics*.

Johannes, Michael, Nicholas Polson and Jonathan Stroud, 2001, Sequential Optimal Portfolio Performance: Market and Volatility Timing. Working paper, Columbia University.

Johannes, Michael, Nicholas Polson and Jonathan Stroud, 2002, Nonlinear Filtering of Stochastic Differential Equations with Jumps. Working paper, Columbia University.

Kim, Sangjoon, Neil Shephard, and Siddhartha Chib, (1998). Stochastic Volatility: Likelihood Inference and Comparison with ARCH Models, *Review of Economic Studies* 65, 361-93.

Pan, Jun, 2002, The jump-risk premia implicit in options: evidence from an integrated time-series study, *Journal of Financial Economics* 63, 3–50.

Pitt, Michael and Neil Shephard, 1999, Filtering via simulation: Auxiliary particle filter, *Journal of the American Statistical Association*, 590–599.

Smith, Adrian, and Alan Gelfand. (1992). Bayesian statistics without tears: a sampling-resampling perspective, *American Statistician* 46, 84-88.

Storvik, Geir, 2002, Particle filters in state space models with the presence of unknown static parameters, *IEEE Trans. on Signal Processing,* 50, 281–289.

Stroud, Jonathan, Nicholas Polson and Peter Müller, (2003), Practical Filtering for Stochastic Volatility Models, working paper.