Sequential Parameter Estimation in Stochastic Volatility Models with Jumps

Michael Johannes   Nicholas Polson   Jonathan Stroud*

January 11, 2006

Abstract

This paper analyzes the sequential learning problem for both parameters and states in a stochastic volatility model with jumps. We extend two existing algorithms, Storvik's (2002) particle filtering algorithm and Polson, Stroud and Muller's (2003) practical filtering algorithm, to incorporate jumps. We analyze the performance of these approaches using both simulated and S&P 500 index return data. We find that the particle filter provides more accurate sequential inference than the practical filtering approach. The differences are minor using simulated data, but greater using S&P 500 index data as the adapted particle filtering algorithm we use efficiently handles outliers. We analyze the implications of learning about jump parameters for option pricing and find that parameter learning generates important implications for option pricing.

*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@gsb.uchicago.edu. Stroud is at The Wharton School, University of Pennsylvania, Philadelphia, PA 19104-6302, stroud@wharton.upenn.edu.

1 Introduction

In this paper, we analyze the problem of sequential parameter learning and state estimation in a model incorporating both stochastic volatility and jumps. This sequential problem is fundamentally different from the usual inference procedures, whereby parameters are estimated based on the entire history of data. Here, we are interested in estimating parameters and state variables sequentially in real time as each new data point arrives. Standard estimation techniques do not apply because it is not computationally feasible to repeat traditional estimation algorithms, such as simulated method of moments or MCMC.

Due to these computational hurdles, the sequential problem has largely been overlooked by the recent literature, despite its central role in both practical applications and its importance for theoretical modeling. For practical finance applications, agents must estimate parameters and make forecasts in real-time. On the theoretical side, standard rational expectations asset pricing models assume that agents know all of the parameters and, conditional on these parameters, the agents price future payoffs. These models are silent regarding exactly how the agents learn about the parameter values and there is a large literature analyzing the equilibrium implications of parameter learning of economic agents (see, e.g., Townsend 1978 and 1983 or Bray and Kreps, 1987).

Recently, two new methods have been developed for sequential inference. The first approach, developed in Storvik (2002) and extended here, is based on the *particle filter* (Gordon, Salmond and Smith 1993 and its extensions in Pitt and Shephard, 1999). Here, the posterior density of the parameters and state variables is approximated by a discrete set of support points or particles. These particles are then sequentially updated as new data arrives. Particle methods are now the benchmark for nonlinear, non-Gaussian filtering problems. The second approach, called the *practical filter*, is based on rolling-window MCMC methods and was developed in Polson, Stroud, and Muller (2003) and Johannes, Polson, and Stroud (2004). Both approaches are computationally attractive and initial work in Storvik (2002) and Polson, Stroud and Muller (2003) shows that the approaches show promise for certain simple cases such as Gaussian models and log-stochastic volatility models, although there are some problems learning the volatility of volatility parameter in stochastic volatility models.

We focus on jump-diffusion models because these models play a central role in finance applications. There is ample evidence the jumps in prices are important in many markets and from disparate source.[1] The sequential problem is particularly interesting in these models as jumps are rare events, and it is difficult to 'disentangle' jumps from diffusive components (see Aït-Sahalia 2003). To learn about jump process parameters, the agent must first correctly identify the movement as a jump, and then update his/her views regarding the parameter values. The parameters controlling rare events are first-order important for asset pricing applications such as option pricing or credit risk modeling. Moreover, the dynamic process of learning about these parameters has important implications for asset pricing applications as shown recently in Collin-Dufresne, Goldstein, and Helwege (2003) and Benzoni, Collin-Dufresne and Goldstein (2005). Hansen and Sargent (2005) solve a related problem, whereby an agent sequentially learns about parameters and latent states, but here the agent is also concerned about potential model misspecification and therefore solves a robust control problem with sequential learning.

We provide four contributions to the literature: first, we extend existing sequential inference algorithms to incorporate jumps in prices, in addition to stochastic volatility; second, we analyze the performance of the algorithms using simulated data, to document their performance in a laboratory environment; third, we analyze the sequential parameter inference problem using S&P 500 data; and, finally, we investigate the option pricing

[1] See Bates (2000), Bakshi, Cao and Chen (1997), Pan (2002) and Eraker (2004) for evidence using option prices, Johannes, Kumar and Polson (1999), Andersen, Benzoni, and Lund (2001), Chernov, Ghysels, Gallant, and Tauchen (2003), Chib, Nardari and Shephard (2004), or Eraker, Johannes, and Polson (2003) for evidence based on the time series of returns. Andersen, Bollerslev, and Diebold (2005), Barndorff-Nielson and Shephard (2004) and Huang and Tauchen (2005) provide high-frequency evidence for the presence of jumps. These nonparametric methods are important as they are robust to the specification of the volatility process.

implications of the sequential jump parameter estimates.

Based on simulated data, we find that both algorithms are computationally feasible and can sequentially estimate the jump parameters in addition to the volatility parameters. For example, for a dataset of 1000 observations, the algorithms take less than 10 minutes. Jumps, while significantly complicating the observed distributions, pose no real problems for sequential estimation of the jump parameters despite their rare nature. Once a jump arrives, both algorithms update the posteriors for the jump intensity, jump mean and jump variance. Unlike Polson, Stroud, and Muller (2003) who find problems estimating the volatility of stochastic volatility for both the particle and practical filtering approaches, we do not find any problems estimating jump parameters. Thus, any problems they identify are likely specific to the log-stochastic volatility model, and not neccessarily general problem with the algorithms. Comparing across approaches, the practical filter perform marginally better, in that the posterior distribution of the parameters more efficiently adapts to the arrival of new information.

To understand how agents would sequentially learn parameters in a practical setting, we apply the algorithms to real data using historical S&P 500 index returns. Real data examples are of particular interest as they provide insights regarding how the algorithms handle model misspecification, as the simple models we consider are likely misspecified. In this realistic setting, the algorithms are again computationally attractive, providing sequential estimates of parameters and states using almost 20 years of daily index returns in 16 minutes.

Based on the S&P 500 sample, we find that there is substantial variation over time in the parameter posterior distributions. For example, the jump intensities and jump size volatility more than double over the sample and the mean jump sizes vary from about zero to less than minus 4 percent. The changes are not monotonic, especially in the jump parameters. Estimated jump parameters drastically change after events such as the Crash of 1987 and both algorithms quickly adapt to new information. The particle filter provides slightly more reliable inference (in a sense made precise below) than the practicle filter using real data.

Finally, we use our empirical results to analyze the implications of sequential parameter learning on option prices. As noted by Bates (1991), Rubinstein (1994), and others, there is a lot of time-variation in the implied volatility smile of index options. This could be due to time-varying jump-risk premia (Pan 2002 or Santa-Clara and Yan 2005) or other factors. When viewed through standard jump models such as Merton (1976), the data indicate that agent's views of the probability of large jumps change substantially over time, especially around the crash of 1987. Benzoni, Collin-Dufresne, and Goldstein (2005) embed this learning in an equilibrium consumption based model. Pan, Liu and Wang (2004) argue that there is substantial uncertainty over jump parameters and investors may robustly price options as a way of dealing with the uncertainty surrounding the jump parameters. Our sequential parameter estimates allow us to evaluate these issues from the perspective of a Bayesian investor who optimally learns about the parameters and states over time. We find that sequential learning has a major impact on option prices, with implied volatility smiles changing drastically over time. Thus, sequential learning alone can quantitatively explain some of the major moves of the implied volatility smile over time.

Our approach builds on a number of recent papers that develop methods for sequential parameter inference. If the parameters are known, the particle filter (Gordon, Salmond, and Smith 1993) and its extensions (Carpenter, Clifford and Fearnhead 1999 and Pitt and Shephard 1999) are well suited for estimating latent states in a wide-range of models. Johannes, Polson, and Stroud (2002) develop a sequential learning algorithm based on the practical filter (see Polson, Stroud and Muller 2003 for the details and Clapp and Godsill 2000 for a related particle filtering algorithm). Storvik (2002) uses particle filtering methods for sequential learning using sufficient statistics. Carpenter, Clifford, and Fearnhead (1999), Liu and West (2000), Kitagawa and Sato (2001), Marihno and Lopes (2002), and Doucet and Tadic (2003) provide related particle filtering results. Other approaches include Chopin (2002) for static models, and Gilks and Berziuni (2001) and Fearnhead (2002) who use particle filtering and MCMC methods together with a sufficient statistic structure for sequential parameter learning.

2 Estimating Stochastic Volatility Models with Jumps

Since the introduction of stochastic volatility models (Rosenberg (1972) and Taylor (1982)), a number of different estimation methods have been developed. Popular methods for estimating either discrete or continuous-time models include MCMC (Jacquier, Polson, and Rossi (1994), Kim, Shephard, and Chib (1998), Elerian, Shephard, and Chib (2001), Eraker (2001), Roberts and Stramer (2001), Eraker, Johannes, and Polson (2003), and Jacquier, Polson, and Rossi (2004)), simulated maximum likelihood (Danielsson (1994), Pedersen (1995), Brandt and Santa-Clara (2002), Durham and Gallant (2002), and Piazzesi (2004)), and simulated methods of moments (Duffie and Singleton (1993), Gallant and Tauchen (1996), Gallant, Hsieh, and Tauchen (1997), and Andersen, Benzoni, and Lund (2001), and Chernov, Ghysels, Gallant and Tauchen (2003)).

We consider sequential inference in the context the standard log-stochastic volatility model, augmented to include jumps in prices. In this model, if we let P_t denote the prices, $\sqrt{V_t}$, the volatility, and $Y_{t+1} = \log\left(P_{t+1}/P_t\right)$ the continuously-compounded returns, then the model is given by the difference equations

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

where $P(J_t = 1) = \lambda$, $Z_t \sim \mathcal{N}(\mu_z, \sigma_z^2)$, and ε_t and η_t are i.i.d. standard normal variables. The model without jumps ($J_t = 0$ for all t) is the benchmark stochastic volatility, but the aforementioned recent research indicates that the model without jumps is misspecified, at least for equity indices, as it cannot generate large negative movements. Similar evidence using related models and option prices is in Bakshi, Cao and Chen (1997), Bates (2000) and Pan (2002). For later use we 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 $X_t = \log\left(V_t\right)$ the log volatilities.

Given a time series of observations, $Y_{1,T} = (Y_1, .., Y_T)$, 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 volatility states, the jump times, and the jump sizes, thus $L_{1,T} = [J_{1,T}, Z_{1,T}, X_{1,T}]$. 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 and state variables as

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

and

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

where G is the number of samples generated in the MCMC algorithm, $\Theta^{(g)}$ is the g^{th} parameter draw and $L_t^{(g)}$ is g^{th} draw of the latent state vector.

It is important to recognize the smoothed nature of these estimators. When estimating volatility, for example, the estimator uses the information embedded in the entire sample. As volatility is persistent, it is clear that both future and past information is informative about V_t. For practical applications, however, researchers do not have the luxury of waiting to receive tomorrow's data to estimate today's volatility. They must estimate the volatility based only currently available information.

This sequential learning problem is solved by iteratively computing $p(\Theta, L_t|Y_{1,t})$ for $t = 1, ..., T$. This is the online or real-time 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 a couple of minutes to compute.

Repeating this thousands of times for large daily data sets is clearly not computationally feasible.

The two algorithms that we consider, the practical and particle filter, approximate the true posterior density, $p(\Theta, L_t|Y_{1,t})$. 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.

For the MCMC algorithm, it is important to use efficient sampling schemes for the latent state variables. Following Kim, Shephard, and Chib (1998) we approximate the model by assuming that the distribution of $\log\left[(Y_t - J_t Z_t)^2\right]$ is a mixture of normals. This redefines the model, however, the approximation error is typically small.[2] The mixture indicators are I_t and (m_i^*, v_i^*, π_i^*) for $i = 1, \ldots, 7$ are the mixture parameters. Define the collection of observations by $Y_{1,t} = (Y_1, \ldots, Y_t)$ and we collect the latent variables in similar vectors, $V_{1,t}$, $X_{1,t}$, $J_{1,t}$, $Z_{1,t}$, and $I_{1,t}$. Given this notation, the joint posterior for the states and parameters is

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

where $p(\Theta)$ is the prior distribution of the parameters. For reference, recall the model specification:

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

where $P(J_t = 1) = \lambda$, $Z_t \sim \mathcal{N}(\mu_z, \sigma_z^2)$, and ε_t and η_t are i.i.d. standard normal variables.

We assume the following conjugate priors for the parameters: $\lambda \sim Beta(S_0, F_0)$, $(\mu_z, \sigma_z^2) \sim \mathcal{N}(\mu_z|m_0, k_0^{-1}\sigma_z^2)\ \mathcal{IG}(\sigma_z^2|a_0, b_0)$, and $(\psi, \sigma_v^2) \sim \mathcal{N}(\psi|\psi_0, \Psi_0^{-1}\sigma_v^2)\ \mathcal{IG}(\sigma_v^2|c_0, d_0)$ where $\mathcal{IG}$ de-

[2] Omori, Chib, Shephard and Nakajima (2004) develop more accurate, higher order approximations, and extend the algorithm to incorporate a leverage effect.

notes the inverse Gamma distribution. Given the conjugate priors, the complete parameter posterior conditionals are

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

where for notational simplicity $p(y|\ldots)$ refers to the conditional distribution of y given all other relevant variables. For the latent variables,

$$
\begin{aligned}
p(J_t|\ldots) &\propto Ber(\lambda_t) \\
p(Z_t|\ldots) &\propto \mathcal{N}(\mu_{z,t}, \sigma_{z,t}^2) \\
p(X_{0,t}|\ldots) &\propto \text{(Not recognizable)} \\
p(I_t|\ldots) &\propto Mult(\pi_{1,t}^*, \ldots, \pi_{7,t}^*).
\end{aligned}
$$

The distribution $p(X_{0,t}|...)$ is not a known distribution. We approximate this distribution using the Kim, Shephard, and Chib (1998) approximation and sample from it using the forward-filtering, backward-sampling (FFBS) algorithm, see Johannes and Polson (2004) for a description of the details. The parameters indexing the state variable posteriors are

$$
\begin{aligned}
\lambda_t &= \frac{\lambda\mathcal{N}(Y_t|\mu_z, V_t + \sigma_z^2)}{\lambda\mathcal{N}(Y_t|\mu_z, V_t + \sigma_z^2) + (1-\lambda)\mathcal{N}(Y_t|0, V_t)} \\
\mu_{z,t} &= \sigma_{z,t}^2\left((\sigma_z^2)^{-1}\mu_z + J_t Y_t V_t^{-1}\right),\ \sigma_{z,t}^2 = \left((\sigma_z^2)^{-1} + J_t V_t^{-1}\right)^{-1} \\
Y_t^* &= X_t + m_{I_t}^* + \sqrt{v_{I_t}^*}\epsilon_t^* \\
\pi_{t,i}^* &= \frac{\pi_i^*\ \mathcal{N}\left(Y_t^*|X_t + m_i^*, v_i^*\right)}{\sum_{j=1}^{7}\pi_j^*\ \mathcal{N}\left(Y_t^*|X_t + m_j^*, v_j^*\right)}
\end{aligned}
$$

and the parameters indexing the parameter posteriors are

$$
\begin{aligned}
S_t &= S_0 + S,\ F_t = F_0 + t - S, \\
a_t &= a_0 + S_t/2,\ c_t = c_0 + t/2, \\
m_t &= k_t^{-1}(k_0 m_0 + \sum_{\tau=1}^{t} J_\tau Z_\tau),\ k_t = k_0 + S \\
b_t &= b_0 + \left(k_0 m_0^2 + \sum_{\tau=1}^{t} J_\tau Z_\tau^2 - k_t m_t^2 \right)/2 \\
\psi_t &= \Psi_t^{-1}(\Psi_0 \psi_0 + H^T X),\ \Psi_t = \Psi_0 + H^T H \\
d_t &= d_0 + \left(\psi_0^T \Psi_0 \psi_0 + X^T X - \psi_t^T \Psi_t \psi_t \right)/2
\end{aligned}
$$

where $S = \sum_{\tau=1}^{t} J_\tau$ and $H = (H_1, \ldots, H_t),^T$ $H_t = (1, X_{t-1})^T$, $X = X_{1,t}$, and $Y_t^* = \log\left[(Y_t - J_t Z_t)^2\right]$.

2.1 Particle Filtering

Particle filtering, also known as the bootstrap filter, was first 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. Although we do not describe it in detail, we implement a variant of the particle filter due to Pitt and Shephard (1999) known as the auxiliary particle filter (APF) and our description follows theirs closely.

There are a number of densities associated with the filtering problem: $p\left(L_t | Y_{1,t}\right)$ is the filtering density, $p\left(L_{t+1} | Y_{1,t}\right)$ is the predictive density, $p\left(Y_t | L_t\right)$ is the likelihood, and $p\left(L_{t+1} | L_t\right)$ is the state transition. Bayes rule links the predictive and filtering densities through the identity

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

where

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

The key to particle filtering is an approximation of the (continuous) distribution of the random variable L_t conditional on $Y_{1,t}$ by a discrete probability distribution, that is, the distribution $L_t|Y_{1,t}$ 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 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 $t+1$. 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 extend Storvik's (2002) particle filtering algorithm for estimating parameters and states by incorporating a look-ahead step via the use of auxiliary variables as in Pitt and Shephard (1999). The key assumption is that the conditional parameter posterior distribution, $p\left(\Theta|L_{1,t}, Y_{1,t}\right)$, is analytically tractable and depends on the observed data and latent variables only through a set of sufficient statistics which are 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 $s_{t+1} = S\left(s_t, L_{t+1}, Y_{t+1}\right)$ as the sufficient statistic which can be computed using the previous sufficient statistic, s_t, as well as the new prices and states, the particle filtering algorithm consists of the following steps. First, assume a particle representation of the joint distribution, $(\Theta, L_t) \sim p\left(\Theta, L_t|Y_{1,t}\right)$. Second, the algorithm then draws

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

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

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 $w_{t+1}^i = w_t^i \cdot p(Y_t|L_{t+1}^i)$.
3. Resample the particles $\left(\Theta^i, L_{t+1}^i\right)$ with probabilities proportional to w_{t+1}^i.

This naive algorithm performed extremely poorly on real data. Intuitively, the reason was that it did not generate enough tail draws for the state variables, in particular, the jump sizes. This is a common problem with naive particle filters and to correct this shortcoming, we use the auxiliary particle filter of Pitt and Shephard (1999) between steps 1 and 2. This, to a large extent, remedied any problems with outliers.

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. For completeness, we provide the entire algorithm:

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.

2.2 Practical Filtering stochastic volatility models with jumps

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

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

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

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

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:

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

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

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

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

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

There are three separate issues that effect the efficiency and accuracy of the algorithm. First, 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 therefore G must be sufficiently large. 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 rather than Metropolis-Hastings.

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 $(\Theta^{(g)}, X_{0,1}^{(g)}, J_1^{(g)}, Z_1^{(g)}) \sim p(\Theta, X_{0,1}, J_1, Z_1)$.
2. For $t = 1, \ldots, t_0$ and $g = 1, \ldots, G$
   (a) Set $\Theta^0 = \Theta^{(g)}$ and $(J_{1,t}^0, Z_{1,t}^0) = (0, 0)$.
   (b) For $i = 1, \ldots, I$:
      i. Generate $X_{0,t}^i \sim p(X_{0,t}|J_{1,t}^{i-1}, Z_{1,t}^{i-1}, \Theta^{i-1}, Y_{1,t})$
      ii. Generate $J_{1,t}^i \sim p(J_{1,t}|X_{0,t}^i, Z_{1,t}^{i-1}, \Theta^{i-1}, Y_{1,t})$
      iii. Generate $Z_{1,t}^i \sim p(Z_{1,t}|X_{0,t}^i, J_{1,t}^i, \Theta^{i-1}, Y_{1,t})$
      iv. Generate $\Theta^i \sim p(\Theta|X_{0,t}^i, J_{1,t}^i, Z_{1,t}^i, Y_{1,t})$
   (c) Set $(\Theta^{(g)}, \tilde{X}_0^{(g)}) = (\Theta^I, X_0^I)$.
3. For $t = t_0 + 1, \ldots, T$ and $g = 1, ..., G$

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

3 Applications

In this section, we compare the performance of these two algorithms using simulated data and S&P 500 index data from 1984 to 2000.

3.1 Simulated data

To analyze both of the algorithms' performance, we simulated 1000 daily observations from the log-stochastic volatility with jumps using the following parameter values:

$$\text{Jump Process: } \lambda = 0.01, \mu_z = -0.04, \sigma_z = 0.05$$
$$\text{Volatility Process: } \alpha_v = 0, \beta_v = 0.99, \sigma_v = 0.1.$$

These parameters are roughly consistent with observed equity return data, see, for example, Johannes, Kumar, and Polson (1999) and Andersen, Benzoni, and Lund (2002). This model will generate rare jumps (about 2 per year) with sizes that are consistent with large negative equity returns such as those in 1987 and 1997. Below, we will perform some sensitivity analysis below by varying λ to provide an understanding of how the algorithms perform as a function of critical parameters.

We calibrated the algorithms so that both take approximately the same amount of computing time. For the particle filter, we chose $N = 25,000$. In the case of the practical filter, we chose the combinations of $G = 250$, $I = 10$, and $k = 25$ so that the computing time (about 6 minutes) was roughly equal to that of the particle filter. The relative computational efficiency of the algorithms implies that in practice, one could likely drastically increase N, G, I and k to obtain more accurate approximations to the posterior while still retaining computationally feasibly algorithms.

Stroud, Polson and Muller (2004) document that both particle and practical filtering have difficulties sequentially learning the volatility of volatility parameter, σ_v. The problem they document is purely associated with the sequential problem, as many other authors have documented that a full MCMC smoothing approach can efficiently estimate this parameter (see Jacquier, Polson, and Rossi (1994) and Kim, Shephard, and Chib (1998)). The sequential problem has difficulties with this parameter for the following reason. As new data arrives, the posterior distribution naturally adapts. However, when very informative (e.g., tail) observations arrive, in principle the posterior on both the unknown parameter and the *entire past volatility path* should adapt rapidly. However, due to the sequential nature of the estimators, past volatility paths in the particle filter might not be representative once the parameters are updated. For example, if a very large shock occurs, driven by high volatility, the posterior on σ_v places higher weights on higher values. However, there may be very few high V_t particles to be updated, causing degeneracies in the particle filtering algorithm. This effect is also present, although to a somewhat lesser degree, in the practical filter. Here, the practical filter can update $k-$lags of volatility, and to a certain extent it will avoid or at least dampen the degeneracies. To abstract from this difficult and unresolved problem, we fix σ_v at its true value throughout. We have performed extensive sensitivity analysis to this parameter. Although not reported, none of our conclusions are affected by this assumption. One of our primary goals is to see if these degeneracies occur more generally in learning parameters that index other latent variables, in this case, jumps.

Throughout, we use the following prior parameters: $S_0 = 1$, $F_0 = 100$, $k_0 = 2$, $m_0 = -2$,

$a_0 = 2.25$, $b_0 = 25$, and $\Psi_0 = (0.001, 0.99)$. These priors are loose, in the sense that the prior variance is always large relative to the prior mean. For the jump parameters, the priors impose that investors assume jumps are rare, occurring about one percent of the time, but the prior standard deviation is also one percent, allowing for substantial uncertainty. The jump mean prior is minus two percent compared to minus four percent for the simulated data. Since our estimates are sequential, prior uncertainty can be evaluated at the beginning of the sample as the posterior distribution, with few observations, is essentially drawing from the prior distribution.

Figures 1 and 2 display the sequential posterior summaries for a representation simulation for the particle and practical filter, respectively. The label at the top of each subplot indicates which state variable or parameter posterior is being summarized. For the volatility (annualized) and the parameters, the plots contain the posterior median and the (2.5,97.5) percent posterior quantiles, while the plots for the jump times and sizes contain the true jump times or sizes (dots) and the posterior median. In the case of jump times, the plot provides the 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 percent. Both algorithms identified a small jump, less than 1 percent, with 20 percent probability. As daily volatility was more than 1 percent, it is not difficult for the model to generate this move with a two to three standard deviation negative shock to ε_t and/or a small jump. Essentially, the jump was too small for the algorithm to detect it. It is important to recognize that this is a signal to noise problem, and is shared by the smoothing problem (see Johannes, Kumar, and Polson (1999) for a related discussion). Moreover, the parameter posteriors for the jump process were not very informative at this early stage in the algorithm, and thus the algorithm was not able to identify the move as a jump. Near the end of the sample, there is a jump that both algorithms identify with high probability, although both algorithms underestimate the size. The degree of agreement between the practical and particle filter for estimating latent jump times and

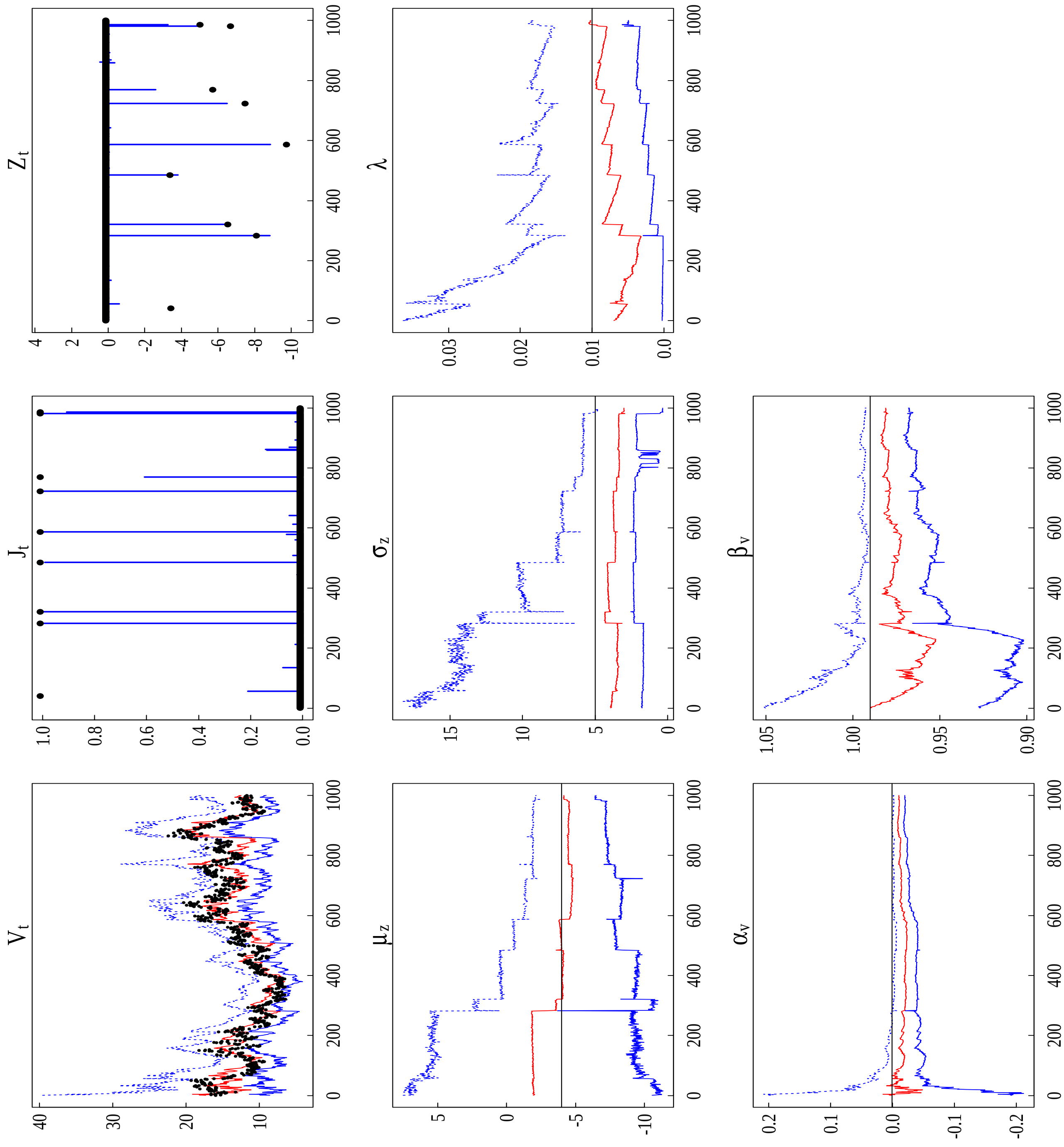

Figure 1: Sequential particle filtering estimates for 1000 simulated data points using the particle filtering algorithm. The particle filter was run using $N = 25,000$ particles. The algorithm took 6 minutes to run.

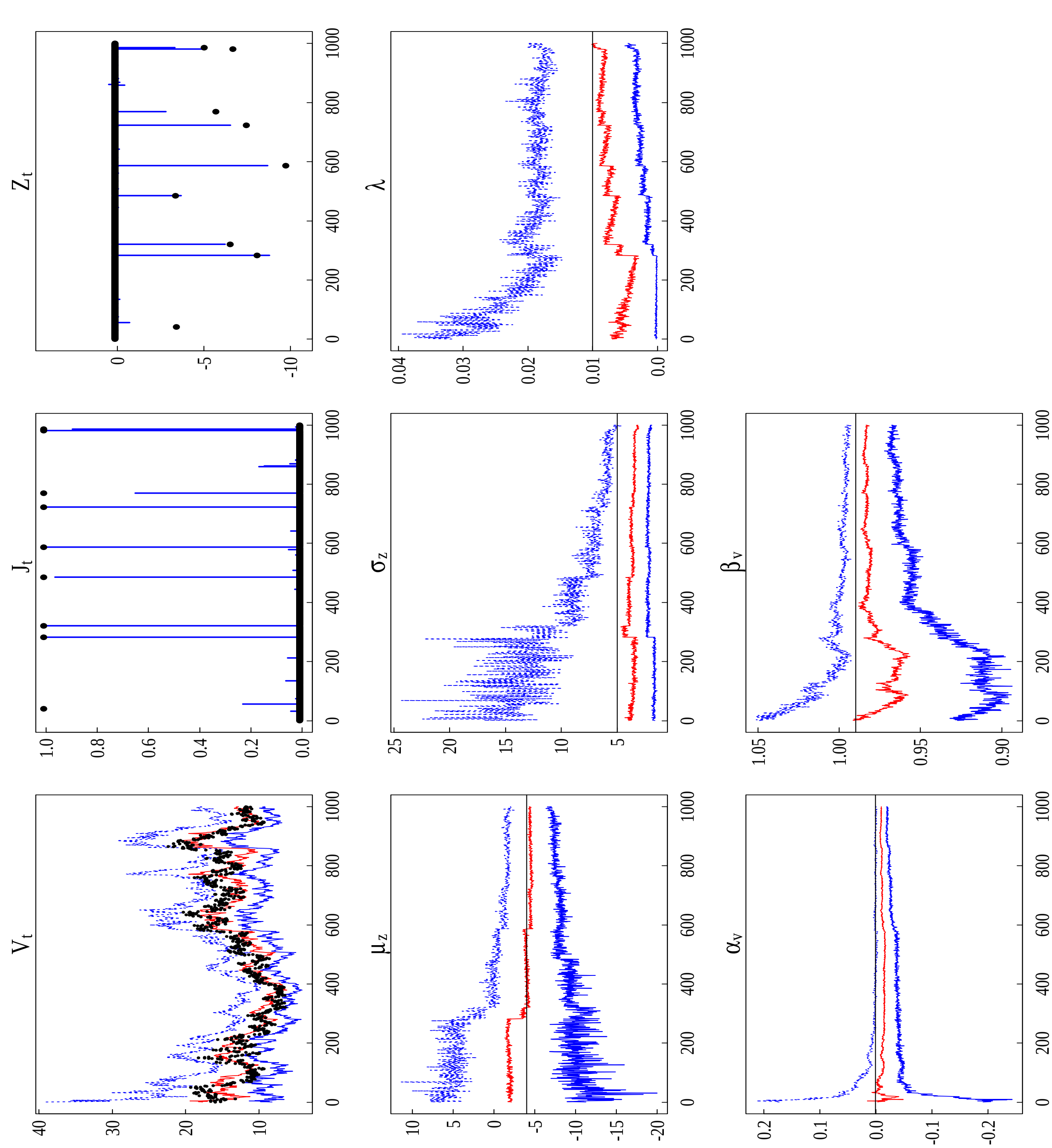

Figure 2: Sequential practical filter estimates for 1000 simulated data points. The practical filter was run with $G = 250$, $I = 10$, and $k = 25$. The algorithm took 6 minutes to run.

sizes is remarkable.

Second, the jump parameter posteriors appear to be collapsing nicely to their true values, for both algorithms. From both figures, it is clear that we assume relatively uninformative priors, for example, for the jump mean the (2.5, 97.5) percent confidence band is roughly (7, -11) percent. The figures show that the parameter posteriors rapidly update, moving toward the true parameter values. For example, at approximately data point 250 a large jump, about -8 percent, 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 algorithms are able to accurately estimate the parameters even with the relatively short time series.

Third, note that the estimates of the jump intensity increase sharply upon the arrival a jump, and then decrease over time, until the next jump arrives. This non-monotonicity is exactly what one would expect. To see this, consider the case of a continuous observation on a Poisson process, N_t, with constant intensity λ. The usual estimator of the intensity at time t, $\widehat{\lambda}_t$, is just

$$\widehat{\lambda}_t = E\left[\lambda | N_t\right] = \frac{N_t}{t}.$$

Since N_t is constant between jump times, the estimates of λ will decrease between jump times and increase discontinuously at a jump time. Regarding the volatility process parameters, α_v and β_v, are estimated reasonably well. The speed of mean reversion is accurately estimated and the estimates of α_v are slightly downward biased, as is common with likelihood based estimates of stochastic volatility mean reversion parameters.

Fourth, a comparison of the posteriors for the two algorithms reveals that the particle filter occasionally has "spikes." For example, for σ_z there is large spike in the 97.5 percent upper band around data point 250 which is quickly reversed. This is not a surprise as it is common for particle filtering algorithms to degenerate or impoverish, in the sense that a small number of particles receive very large weights. While potentially worrisome, the algorithm does recover quickly, as probability is more evenly distributed across the particles. There are also observed spikes in the posteriors for the other jump parameters, μ_z and λ. The spikes are not seen in the practical filtering algorithm. Stroud, Polson and Muller (2004) found similar results for a pure SV model.

In order to provide some sensitivity analysis, Figure 3 compares the performance of the particle and practical filter for the sequential learning of λ, for three different jump intensities, $\lambda = 0.01$, 0.05, and 0.10, again holding the computational time equal. Overall, the algorithms are able to correctly estimate the posterior mean, although the bands differ for the particle and practical. The left hand panels should that the posterior bands for λ using the particle filter again have some spikes. Some of these moves are transient and are quickly reversed, while others are rapid moves to a new region of the parameter space. Again, notice also that the practical filter has fewer spikes.

Given this, we further investigate how varying λ effects the parameter posteriors for β_v, μ_z and σ_z, using the practical filter in Figure 4. The top panel has $\lambda = 0.01$, the middle panel $\lambda = 0.05$, and the bottom panel $\lambda = 0.10$. This figure shows that the posterior estimates of β_v are relatively insensitive to variation λ. On the other hand, as λ increases, more jumps arrive and the posteriors for μ_z and σ_z more quickly converge to their true values. We have also performed extensive simulations documenting the sensitivity of the algorithm to variations in β_v, μ_z and σ_z. In general the results are similar and are therefore not reported. For example, the smaller the jump sizes (as measured by μ_z and/or σ_z), the more difficult it is to identify these parameters. This is not specific to sequential inference or our algorithms, but rather is a general property of Bayesian inference which occurs when the signal is reduced relative to the noise. We conclude from this that both algorithms are able to accurately sequentially learn the parameters and states. While the particle filter can have some spikes in the sequential posteriors, the impact on the overall efficacy of the algorithm is not substantial.

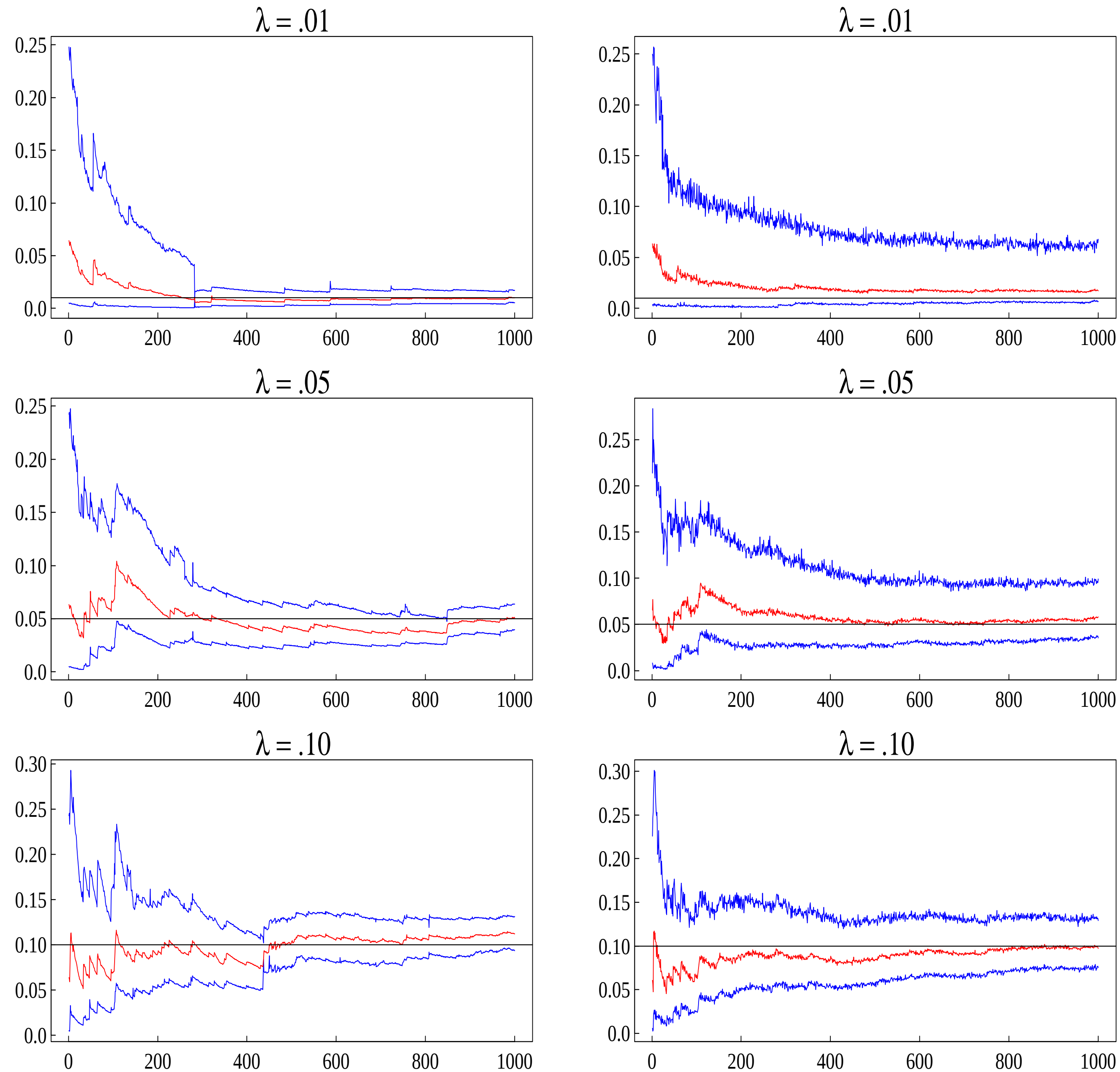

Figure 3: Sequential estimates for λ in the particle and practical filtering algorithm for three different jump intensities: $\lambda = 0.01$ (low intensity); $\lambda = 0.05$ (moderate intensity); and $\lambda = 0.10$ (high intensity).

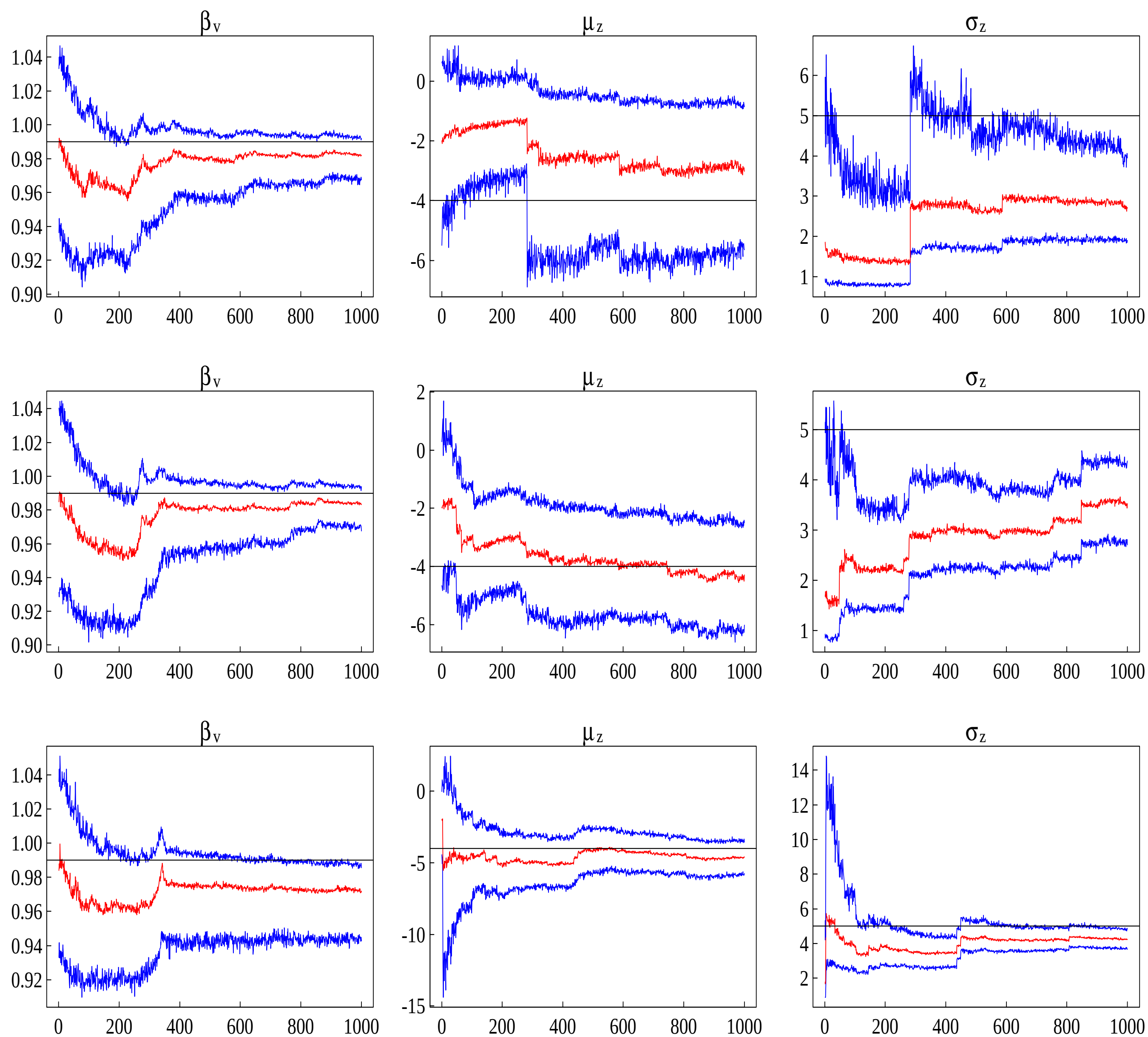

Figure 4: Sequential parameters estimates using the practical filter for β_v, μ_z, σ_z and λ for a low, moderate, and high jump intensity.

3.2 S&P 500

In this section, we consider sequential learning using daily S&P 500 index returns from January 1984 to January 2002. We are primarily interested in how investor's learn about the parameters and state variables of the jump process, but the S&P data set also offers an additional challenge for the algorithms 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. As in the previous case, we set $\sigma_v = 0.10$ and sequentially learned the other parameters.[3] For the particle filter, we used $N = 10,000$ and the same values for the practical filter reported earlier. Both algorithms took about 16 minutes of computing time.

Figures 5, 6, 7, 8, and 9 summarize the results: Figure 5 compares the latent state variable estimates; Figures 6 and 7 summarize the sequential parameter posteriors for the stochastic volatility and jump parameters, respectively; and Figures 8 and 9 analyze the crash of 1987 in detail, comparing the practical and particle filtering estimates with the true posteriors, as computed via full MCMC estimation. We will discuss the results in the order they are displayed in the figures.

Figure 5 compares the sequential estimates of the state variables for the two approaches. For the volatility state, we report (2.5, 50, 97.5) percent quantiles and for the jump times and sizes, we report the posterior medians. For all of the parameters, the state posteriors are similar. Focussing on the jump times, both algorithms identify the same jump size (about 22 percent) on October 19, 1987, the date of the stock market crash. It is somewhat surprising that both algorithms can identify this move as a jump, as it appears to be an outlier.[4] Prior to the crash, both algorithms (see Figure 7) estimated that jump sizes were

[3] The jump parameter estimates are not particularly sensitive to this parameters, although α_v and β_v are more sensitive to this parameter.

[4] Although not reported, an implementation of the particle filtering algorithm without using auxiliary variables resulted in substantially different jump estimates. The particle filter without extensions does not identify the crash as a large movement as a jump because when the algorithm propagates particles forward

relatively small ($\mu_z \approx -1.5$ and $\sigma_z = 3$). For both methods, however, there was substantial posterior uncertainty in these parameters so that simulations of jump sizes, taking into account parameter uncertainty, resulted in large negative draws for the jump sizes. For example, for μ_z and σ_z the confidence bands were roughly $(-5,\ 3)$ and $(2.5,\ 7)$ percent, respectively.

Figures 6 and 7 provide the sequential parameter estimates for the fixed parameters indexing the volatility and jump process. Figure 6 shows that the posteriors for (α_v, β_v) are quite similar, although there are some differences early in the time series. The quantile bands for the particle filter are much wider in the beginning of the sample and there are some spikes, similar to the pure simulation experiments, which are later reversed. Overall, there are not any substantial statistically significant differences in the sequential estimates for the volatility parameters.

The story is somewhat different for λ, μ_z and σ_z where there are more substantive differences. Again, the particle filter has a number of short-lived spikes and for λ, the upper 97.5^{th} quantile is much higher than for the practical filter. This is rapidly revised down after the Crash of 1987. Prior to the crash, there was effectively one statistically significant jump identified, so it is not surprising that the algorithms have different extreme quantiles given the substantial posterior uncertainty. Towards the end of the sample, the estimates of μ_z are much higher for the practical filter, with the upper quantile, well above zero, while the upper quantile is negative for the particle filter. For σ_z, the practical filter posterior is shifted well above the particle filter posterior, especially at the end of the sample, where the practical filter posterior is one to two percent higher than the particle filter posterior. By inspection, the practical filtering posterior for λ is somewhat below that of the particle filtering approach.

---

it simulates very few extremely large jumps. As pointed out by Pitt and Shephard (1999) particle filtering algorithms can have severe difficulties dealing with outliers. Due to the sequential nature of the particle filter, this misestimation has a residual affect in the algorithm as the sequential parameter estimates for the jump parameters were also substantially different.

Given the differences, the obvious question to ask is which one algorithm gets the right answer? To evaluate this, we compare the posterior distributions generated by full MCMC with those from the approximate filtering methods. That is, we characterize the "true" posterior using full-blown MCMC estimation and then compare these posteriors with those obtained from the practical and particle approaches. We perform two comparisons, in order to conserve space. First, we compare the marginal parameter and state posteriors after the Crash of 1987 and at the end of the sample. Figures 8 and 9 compare the posteriors after the Crash and Table 1 reports posterior summaries at the end of the sample.

In Figures 8 and 9, the posterior from full MCMC is given by the smooth line and the histogram gives the estimated posterior using practical or particle methods. A comparison of the left-hand panels reveals that while $p(Z_t|Y_t)$ are similar across methods, $p(V_t|Y_t)$ for the particle filter is more accurate than that of the practical filter. Similarly, the practical filter is clearly more accurate than the particle for $p(a_v|Y_t)$ and $p(\beta_v|Y_t)$. In the case of the practical filter, the posteriors for both of these parameters are substantially shifted to the right. For μ_z and σ_z, the posteriors are similar in location, but the particle filter generates more tail mass than the practical filter.

Finally, Table 1 compares the posterior means and standard deviations for all of the state variables and parameters. In every case, the particle filter provides more accurate inference, although the differences are often not significant in the sense that the approximate posterior means are close (relative to the standard deviation) to the true posterior means. The differences are greatest for the jump parameters. We conclude from this that on real data, where model misspecification is a concern, the particle filter performs better than the practical filter, for the same computing time. Again, it is important to recall that the key to the performance of the particle filtering algorithm is that the states were updated using the auxiliary particle filtering approach of Pitt and Shephard (1999). We next consider the impact of sequential learning on option prices.

Table 1: Full sample comparisons. This table compares the accuracy of the two sequential methods to full MCMC estimation. We summarize the differences in posterior distribution via the posterior mean and standard deviation.

	MCMC		Particle		Practical	
Parameter	Mean	Std. Dev	Mean	Std. Dev	Mean	Std. Dev
$p\left(\alpha_v \vert Y_T\right)$	-0.0035	0.0017	-0.0039	0.0017	-0.0039	0.0016
$p\left(\beta_v \vert Y_T\right)$	0.9905	0.0020	0.9905	0.0020	0.9881	0.0021
$p\left(\lambda \vert Y_T\right)$	0.0064	0.0027	0.0063	0.0012	0.0040	0.0009
$p\left(\mu_z \vert Y_T\right)$	-2.3399	1.3053	-2.1298	0.6737	-1.3049	1.4416
$p\left(\sigma_z \vert Y_T\right)$	4.2906	1.0294	3.3077	0.4902	6.5391	1.1087
$p(\log\left(V_T\right) \vert Y_T)$	-0.3393	0.3781	-0.2860	0.3871	-0.3765	0.3808
$p\left(J_T \vert Y_T\right)$	0.0011	0.0331	0.0015	0.0331	0.0000	0.0000
p$(J_T Z_T \vert Y_T)$	0.0002	0.0285	-0.0001	0.0325	0.0000	0.0000

3.3 Sequential learning and option prices

In this section, we focus on one of the asset pricing implications of our sequential learning results by quantifying how option prices change due to sequential parameter learning. As mentioned in the introduction, a number of authors have noticed that there was a dramatic increase the Black-Scholes implied volatility smiles after the crash, see, for example, Bates (1991), Rubinstein (1994), and others, there has been quite a bit of time-variation in the implied volatility smile of index options.

In terms of option pricing models, this suggests that in jump based models, the parameters indexing the jump distribution change over time. Bates (1991) addresses this issue by taking a Merton's (1976) jump-diffusion model and backing out option implied parameters. These parameters then provide "direct insights into the climate of expectations" of investors. Bates finds that there is substantial time variation in "crash" fears, as measured by these parameters both pre and post crash, although the largest movements were after the crash, naturally. Benzoni, Collin-Dufresne, and Goldstein (2005) embed this learning in an equilibrium consumption based model. In a related vein, Pan, Liu and Wang (2004) argue that investors may robustly price options, as a way of dealing with the substantial uncertainty surrounding the jump parameters. In this section, we quantify some of the option pricing implications, and relate them to existing findings in the literature.

To investigate the option pricing implications, we use the sequential parameters as inputs into Merton's option pricing model and then compute Black-Scholes implied volatility from a range of option strikes. We use the posterior medians of the parameters as inputs. In doing so, we abstract from the impact of learning on the stochastic volatility parameters. Since these parameters typically have a very minor impact on the volatility smile, they will have little impact on the results. All results are holding total volatility constant, which also mitigates any impact of stochastic volatility, and the options mature in two weeks. Since short-dated options display jump risks most clearly, they provide the relevant benchmark.

To understand the impact of the Crash of 1987, we report the posterior medians for the jump parameters before and after the Crash (October 19, 1987), as well as the values at the end of our sample. For the jump times, we report the annual estimate of the number of jumps and for μ_z and σ_z, the parameters are in percentages. For the jump parameters, the estimates for λ, μ_z, and σ_z for October 9, 1987, October 19, 1987 and December 30, 2001 are $\widehat{\lambda} = (1.05, 1.54, 1.34)$, $\widehat{\mu_z} = (-1.72, -4.35, -2.72)$, and $\widehat{\sigma_z} = (2.81, 7.12, 5.1)$. From this, it is clear the huge impact of the Crash: jump probabilities increased by about 50 percent, mean jump sizes fell dramatically, and the jump size volatility more than tripled.

To quantify the impact on implied volatility curves, Figure 10 provides Black-Scholes implied volatility smiles for prices computing from Merton's model using the above jump parameter estimates and constraining total volatility to be constant. The results indicate that the time-variation in parameters would generate drastic changes in implied volatility smiles. As measured by the slope of the implied volatility, the ratio of 5, 10, and 15 percent OTM implied volatility to ATM volatility changes from $(1.02, 1.14, 1.28)$ to $(1.28, 1.76, 2.09)$. In dollars terms, for example, for a $100 stock price, the 5 percent OTM option prices tripled from \$0.057 to \$0.175. The results also indicate that the slope of the implied volatility smile has moderated since the crash of 1987. Sequential jump parameter estimates indicate that investors now view jumps are less likely (as measured by estimates of λ) and smaller (as measured by $|\mu_z|$ and σ_z) at the end of the sample, than post-crash 1987.

Together, these results indicate that learning can have large and interesting implications for options pricing. Standard option pricing models assume that investors observe the true parameter values that index the stock price's evolution. Our results indicate that if investors learn about the parameters from past data, there is a substantial variation in these parameters, and, moreover, that this learning has a first order impact on option prices. This suggests that alternative option pricing approaches, such as those in Benzoni, Collin-Dufresne, and Goldstein (2005) provides a fruitful avenue for future research.

4 Conclusions

This paper extends existing sequential algorithms developed in Storvik (2002) and Johannes, Polson, and Stroud (2002) to the case of stochastic volatility models with jumps. We also extend Storvik's algorithm to incorporate an auxiliary particle filtering step. We find that both practical and particle filtering provide accurate inference for simulated data. On S&P 500 data, the algorithms generate some substantial differences, with the particle filter performing better. Both algorithms are computationally feasible as run times for even large datasets (2000 datapoints), are less than 20 minutes.

From an economic perspective, we find substantial variation in sequential parameter estimates, especially for jump parameters. Since jumps are rare, investors learn about the probability of a jump and the parameters indexing the parameters infrequently, resulting in major revisions in beliefs around jump events. We show that these revisions result in large changes in option prices, even if total volatility is held constant. Broadie, Chernov, and Johannes (2005) document that there is evidence for time-varying risk-neutral jump parameters based on option price data, and it would be interesting to compare the option based time-varying parameter estimates and the sequential parameter estimates from the time series.

In the future, we plan to further analyze the particle filtering algorithms, in order to better understand their shortcomings and to propose potential remedies. One potential remedy is to introduce alternative proposal densities for the parameters and the state variables, building on the work of Pitt and Shephard (1999). Second, we plan on analyzing the option pricing implications in greater detail. Our sequential estimates make strong predictions regarding how and when implied volatility smiles will change over time and it would be interesting to compare these implications, based solely on returns, to the actual variation based on index option data.

References

Aït-Sahalia, Y., 2004, Disentangling Diffusion from Jumps, *Journal of Financial Economics*, 74, 487-528.

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

Andersen, T.G., Bollerslev, T. and Diebold, F.X., 2005, Roughing It Up: Including Jump Components in the Measurement, Modeling and Forecasting of Return Volatility, working paper.

Bakshi, G., C. Cao, and Z. Chen, 1997, Empirical performance of alternative option pricing models, *Journal of Finance* 52, 2003-2049.

Barndorff-Nielson O. and N. Shephard, 2003, Econometrics of testing for jumps in financial economics using bipower variation, working paper, Nuffield College, Oxford University.

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

Benzoni, L., P. Collin-Dufresne, and R. Goldstein, 2005, Can standard preferences explain the prices of out-of-the-money S&P 500 put options, working paper.

Bray, M. and D. Kreps, 1987, Rational Learning and Rational Expectations, in W. Heller, R. Starr and D. Starrett, editors, Essays in Honor of Kenneth Arrow, Cambridge University Press, 597-625.

Bollerslev, T., 1986, Generalized autoregressive conditional heteroskedasticity, *Journal of Econometrics,* 51, 307-327.

Brandt, M. and P. Santa-Clara, 2002, Simulated likelihood estimation of diffusions with an applications to exchange rate dynamics in incomplete markets, *Journal of Financial Economics,* 63, 161-210.

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

Chernov, M., E. Ghysels, A.R. Gallant, and G. Tauchen, 2003, Alternative models for stock price dynamics. *Journal of Econometrics*, 116, 225-257.

Chib, S., F. Nardari and N. Shephard, 2004, Markov Chain Monte Carlo methods for stochastic volatility models. *Journal of Econometrics*, 108, 281-316.

Chopin, N., 2002, A Sequential Particle Filter method for static models. *Biometrika*, 89, 539-552.

Clapp, T. and S. Godsill, 2000, Fixed-lag smoothing using sequential importance sampling. In *Bayesian Statistics 6* (eds. Bernardo et al.). Oxford: Oxford University Press.

Collin-Dufresne, P, R. Goldstein, and J. Helwege, 2003, Is credit event risk priced? Modeling contagion via the updating of beliefs, working paper, UC Berkeley.

Danielsson, J., 1994, Stochastic volatility in asset prices: estimation with simulated maximum likelihood, *Journal of Econometrics*, 61, 375-400.

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

Doucet, A. and V. Tadic, 2003, Parameter Estimation in general state-space models using particle methods. *Annals of Inst. Stat. Math.*, 409-422.

Duffie, D. and K. Singleton, 1993, Simulated Moments Estimation of Markov Models of Asset Prices, *Econometrica*, 61, 929-97.52.

Durham, G. and A.R. Gallant, 2002, Numerical techniques for maximum likelihood estimation of continuous-time diffusion processes (with discussion), *Journal of Business and Economics Statistics, 20, 297-338.*

Elerian, O., S. Chib, and N. Shephard, 2001, Likelihood inference for discretely observed non-linear diffusions. *Econometrica*, 69, 959-993.

Engle, R.F., 1982, Autoregressive conditional heteroskedasticty with estimates of the variance of the United Kingdom inflation, *Econometrica*, 50, 987-1007.

Eraker, B., 2001, MCMC analysis of diffusion models with applications to finance. *Journal of Business and Economic Statistics*, 19-2, 177-191.

Eraker, B., 2004, Do Stock Prices and Volatility Jump? Reconciling Evidence from Spot and Option Prices, *Journal of Finance*, 59, 1367-1403

Eraker, B., M. Johannes and N.G. Polson, 2003, The impact of jumps in volatility and returns, *Journal of Finance 58*, 1269-1300.

Fearnhead, P., 2002 MCMC, sufficient statistics and particle filter. *Journal of Computational and Graphical Statistics*, 11, 848-862.

Gallant, A.R., D. Hsieh, and G. Tauchen, 1997, Estimation of stochastic volatility models with diagnostics, *Journal of Econometrics, 81, 159-192.*

Gallant, A.R. and G. Tauchen, 1996, Which moments to match, *Econometric Theory, 12, 657-681.*

Gilks, W. and Berziuni, C, 2001, Following a moving target - Monte Carlo inference for dynamic Bayesian models. *Journal of Royal Statistical Society*, B, 63, 127-146.

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

Hansen, L. and T. Sargent, 2005, Recursive Robust Estimation and Control without Commitment, working paper.

Huang, X. and G. Tauchen, 2005, The Relative Contribution of Jumps to Total Price Variance, forthcoming, *Journal of Financial Econometrics.*

Jacquier, E., N.G. Polson, and P. Rossi, 1994, Bayesian analysis of Stochastic Volatility Models, (with discussion). *Journal of Business and Economic Statistics* 12, 371-417.

Jacquier, E., N.G. Polson, and P. Rossi, 2004, Bayesian analysis for stochastic volatility with fat tails and correlation, *Journal of Econometrics*, forthcoming.

Johannes, M.,R. Kumar, and N.G. Polson, 1999, State-Dependent Jump Models: How do US Indices jump? Working paper, Columbia University.

Johannes, M. and N.G. Polson, 2004, MCMC methods for continuous-time methods financial econometrics, in Y. Ait-Sahalia and L.P. Hansen (Eds.), *The Handbook of Financial Econometrics*, North-Holland, forthcoming.

Johannes, M., N.G. Polson and J. Stroud, 2002, Sequential Optimal Portfolio Performance: Market and Volatility Timing. Working paper, Columbia University.

Johannes, M., N.G. Polson and J. Stroud, 2003, Nonlinear Filtering of Stochastic Differential Equations with Jumps. Working paper, Columbia University.

Kim, S., N. Shephard, and S. Chib., 1998, Stochastic volatility: likelihood inference and comparison with ARCH models, *Review of Economic Studies* 65, 361-93.

Kitagawa, G. and S. Sato, 2001, Monte Carlo smoothing and self-organizing state-space model, in *Sequential Monte Carlo Methods in Practice*, Doucet, A., N. de Freitas, and N. Gordon (eds.), New York: Springer-Verlag, Series Statistics for Engineering and Information Science.

Liu, J. and M. West, 2000, Combined parameter and state estimation in simulation-based filtering, In Sequential Monte Carlo Methods in Practice (eds.: A. Doucet et al.), Springer-Verlag.

Liu, J., J. Pan, and T. Wang, 2005, An equilibrium model of rare event premia and its implication for option smirks, *Review of Financial Studies* 18, 131-164.

Marinho, C. and Lopes, H., 2002, Simulation-based sequential analysis of Markov switching stochastic volatility models, working paper, University of Chicago.

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

Pedersen, A., 1995, A new approach to maximum likelihood estimation for stochastic differential equations on discrete observations, *Scandinavian Journal of Statistics,* 27, 55-71.

Piazzesi, M., 2004, Bond yields and the Federal Reserve, *Journal of Political Economy*, forthcoming.

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

Polson, N.G. and J. Stroud, 2003, Bayesian Inference for Derivative Prices, *Bayesian Statistics 7* (Bernardo et al., eds.), Oxford University Press, 641-650.

Polson, N., J. Stroud and P. Mueller, 2002, Practical filtering with sequential parameter learning. *Working paper*, University of Chicago.

Roberts, G.O. and O. Stramer, 2001, On inference for nonlinear diffusion models using the Hastings-Metropolis algorithms, *Biometrika* 88, 603-621.

Rosenberg, B., 1972, The behavior of random variables with nonstationary variance and the distribution of security prices, unpublished working paper, University of California, Berkeley.

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

Stroud, J., N. Polson and Peter Müller, 2004, Practical Filtering for Stochastic Volatility Models, working paper.

Taylor, S., 1982, Financial returns modelled by the product of two stochastic processes – a study of daily sugar prices 1961-1979, In O.D. Andersen (Ed.), *Time Series Analysis: Theory and Practice*, 1, 203-226, North-Holland.

Townsend, R. 1983, Forecasting the forecasts of others, *Journal of Political Economy*, 91, 546-588.

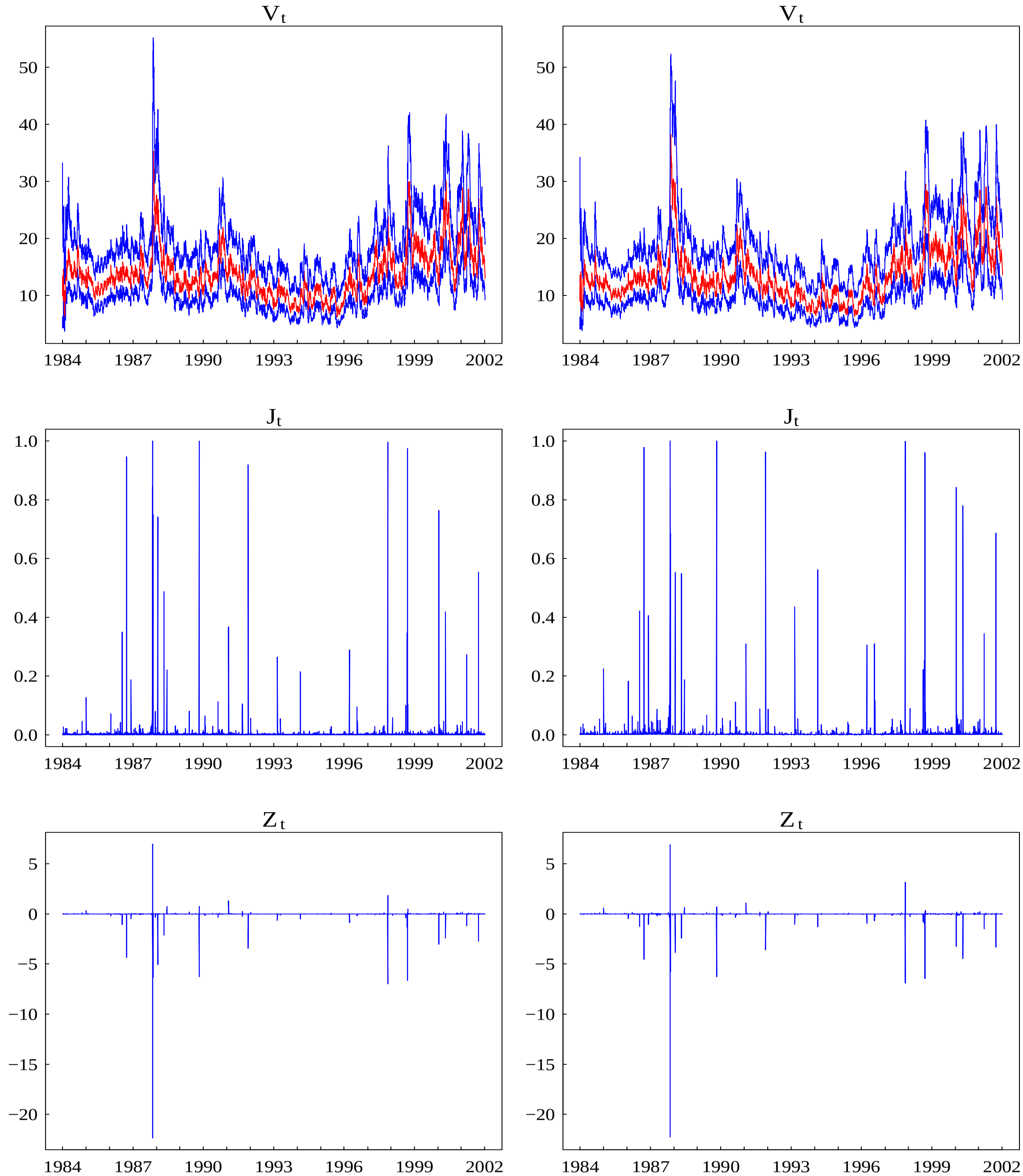

Figure 5: Filtered state variable estimates for the practical filter (left hand panels) and particle filter (right hand panels). For volatility, the figures display (5, 50, 95) percent confidence bands. For the jump times and sizes, we report posterior medians.

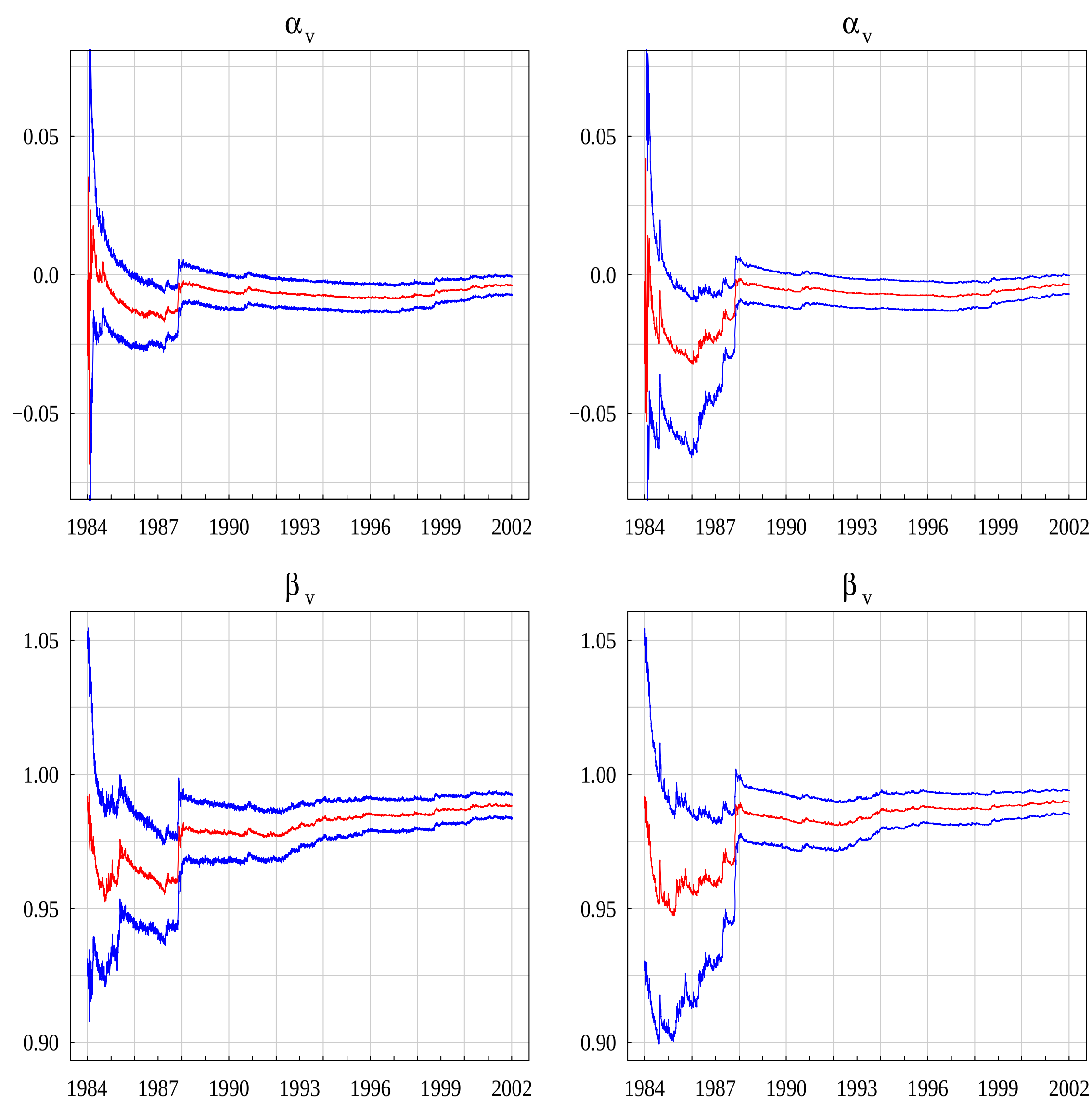

Figure 6: Sequential posterior summaries for the volatility parameters. The left (right) panels summarize the posterior (2.5,50, 97.5) percent quantiles for the practical (particle) filtering approach.

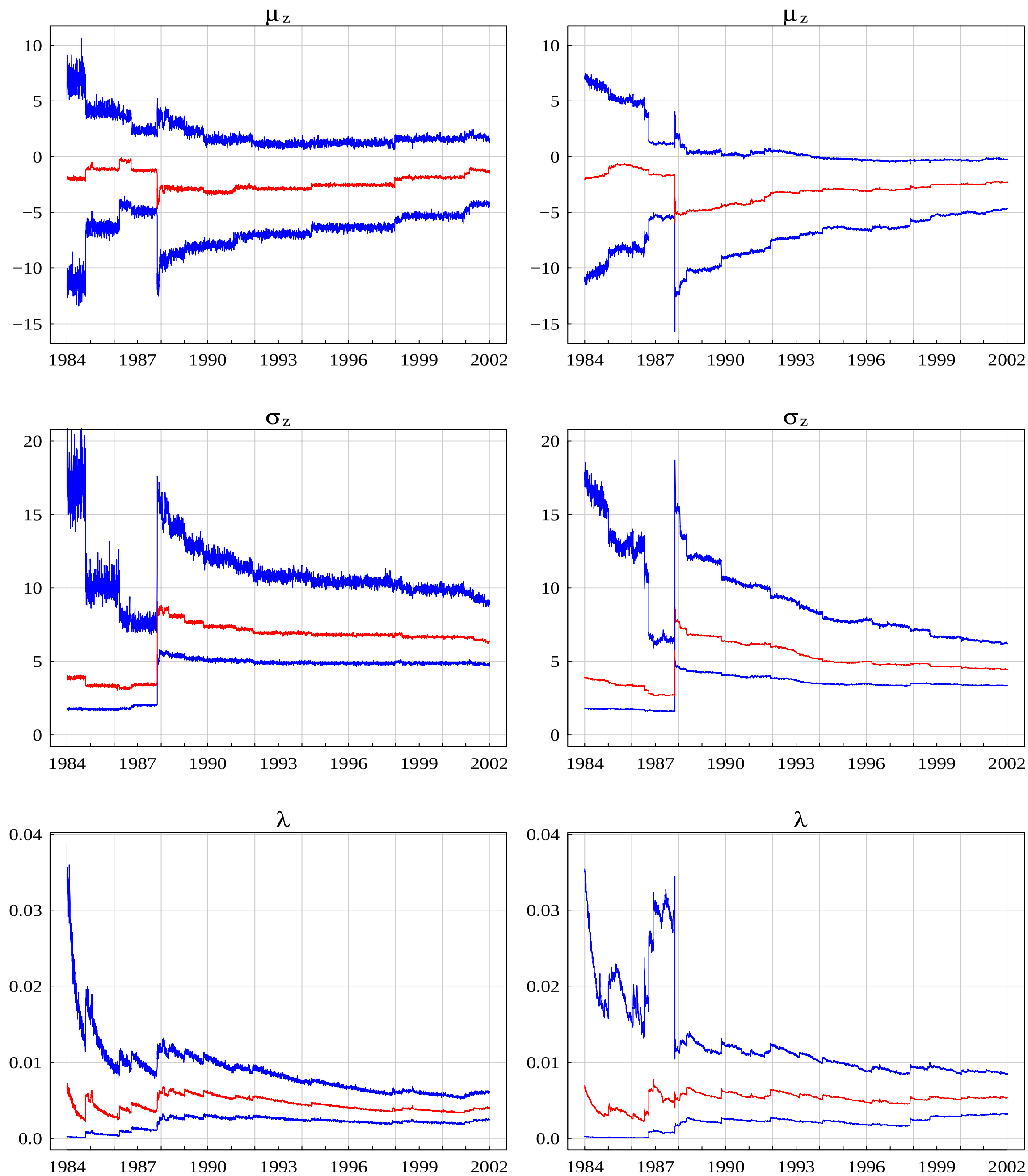

Figure 7: Sequential posterior summaries for the jump parameters. The left (right) panels summarize the posterior (2.5,50, 97.5) percent quantiles for the practical (particle) filtering approach.

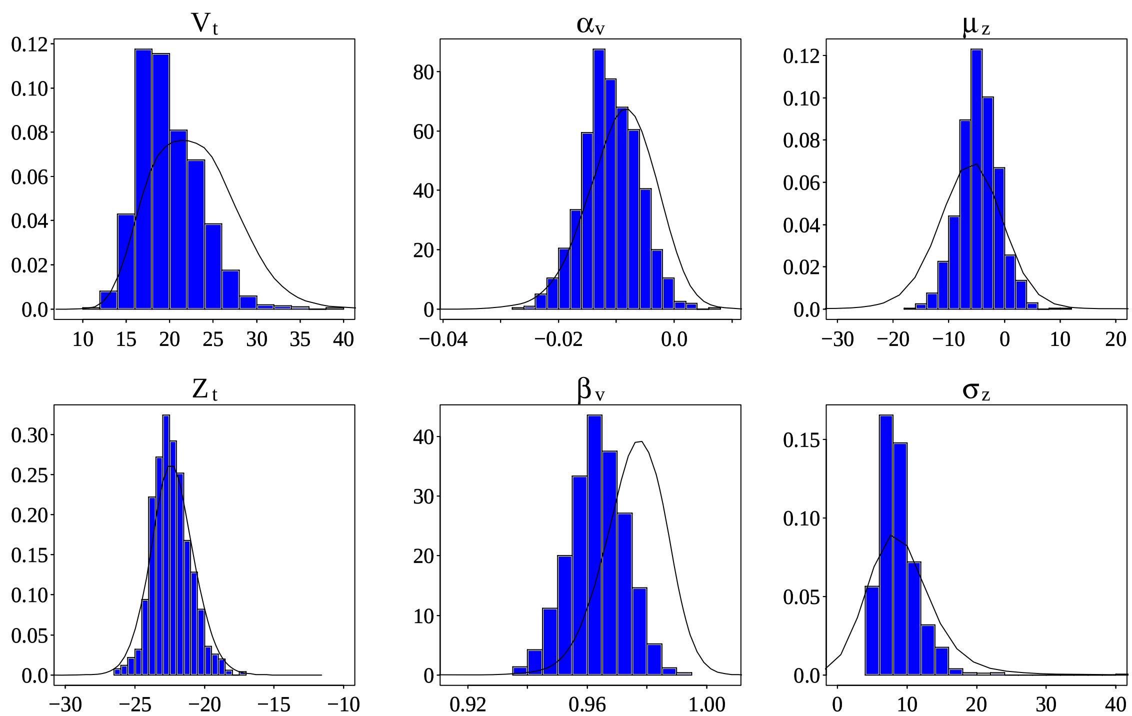

Figure 8: This figure compares the parameter and state posterior distributions for the practical filter (histogram) and full MCMC (smoothed density) for October 19, 1987. The x-axis is the value of the parameter or state variable.

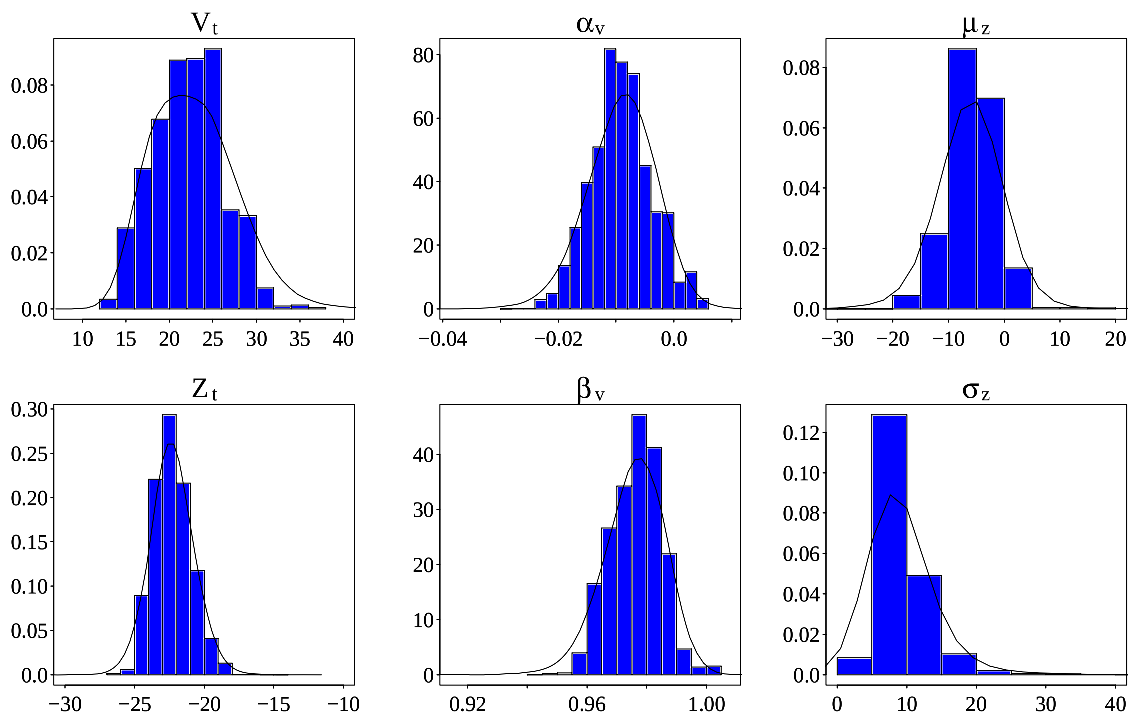

Figure 9: This figure compares the parameter and state posterior distributions for the particle filter (histogram) and full MCMC (smoothed density) for October 19, 1987. The x-axis is the value of the parameter or state variable.

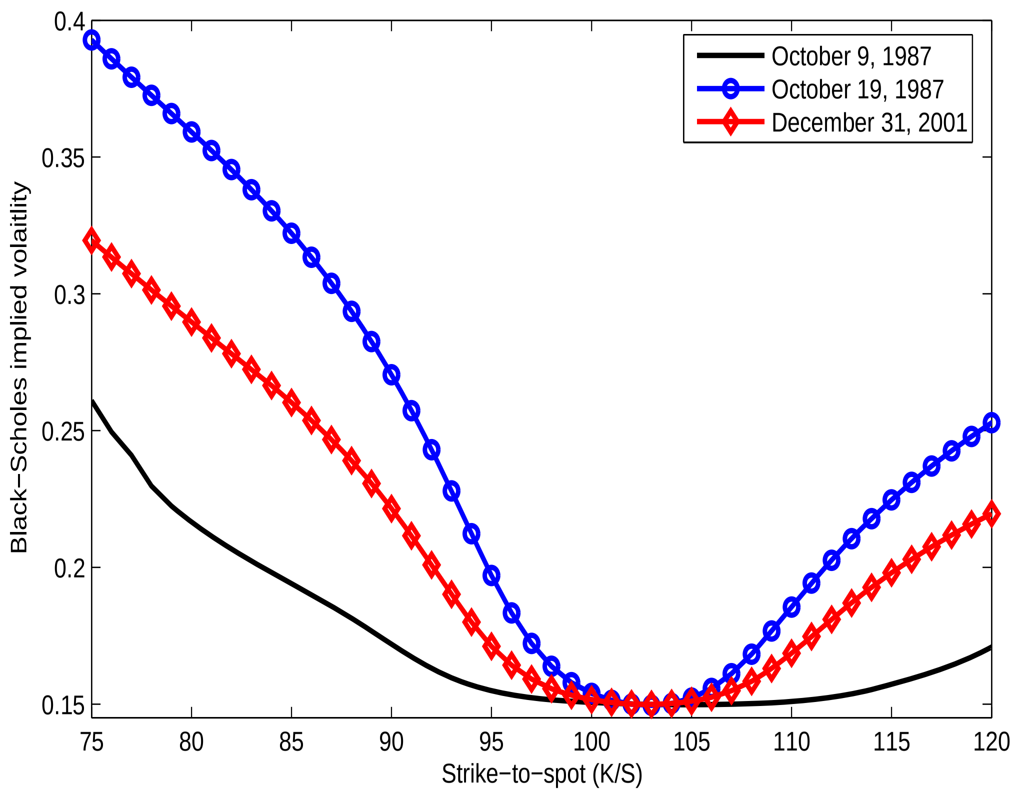

Figure 10: Black-Scholes implied volatility smiles for various dates. The model prices were computed using Merton's (1976) jump model, holding volatility constant, and using the sequential parameter estimates from the practical filter.