Exact Particle Filtering and Parameter Learning

Michael Johannes and Nicholas Polson*

First draft: April 2006
This draft: October 2006

Abstract

In this paper, we provide an exact particle filtering and parameter learning algorithm. Our approach exactly samples from a particle approximation to the joint posterior distribution of both parameters and latent states, thus avoiding the use of and the degeneracies inherent to sequential importance sampling. Exact particle filtering algorithms for pure state filtering are also provided. We illustrate the efficiency of our approach by sequentially learning parameters and filtering states in two models. First, we analyze a robust linear state space model with t-distributed errors in both the observation and state equation. Second, we analyze a log-stochastic volatility model. Using both simulated and actual stock index return data, we find that algorithm efficiently learns all of the parameters and states in both models.

*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, 5807 S. Woodlawn, Chicago IL 60637, ngp@gsb.uchicago.edu. We thank Seung Yae for valuable research assistance.

1 Introduction

Sequential parameter learning and state filtering is a central problem in the statistical analysis of state space models. State filtering has been extensively studied using the Kalman filter, analytical approximations, and particle filtering methods, however, these methods assume any static model parameters are known. In practice, parameters are typically unknown and filtered states are highly sensitive to parameter uncertainty. A complete solution to the sequential inference problem delivers not only filtered state variables, but also estimates of any unknown static model parameters.

This paper provides an exact particle filtering algorithm for sequentially filtering unobserved state variables, x_t, and learning unknown static parameters, θ, for wide class of models. Our algorithm generates exact samples from a particle approximation, $p^N(\theta, x_t|y^t)$, to the joint posterior distribution of parameters and states, $p(\theta, x_t|y^t)$, where N is the number of particles and $y^t = (y_1, ..., y_t)$ is the vector of observations up to time t. Our algorithm is "optimal" in the sense that we provide exact draws from the particle approximation to $p(\theta, x_t|y^t)$, thus avoiding the use of and the inherent degeneracies associated with importance sampling. The algorithm applies generally to nonlinear, non-Gaussian models assuming a conditional sufficient statistics structure for the parameter posteriors.

The algorithm relies on three main insights. First, we track a triple consisting of parameters, sufficient statistics, and states, denote by (θ, s_t, x_t), as in Storvik (2002) and Fearnhead (2002). Second, by tracking this triple, we can factorize the joint posterior density via

$$p\left(\theta, s_{t+1}, x_{t+1}|y^{t+1}\right) \propto p\left(\theta|s_{t+1}\right) p\left(s_{t+1}|x_{t+1}, y^{t+1}\right) p\left(x_{t+1}|y^{t+1}\right). \tag{1}$$

This representation suggests an approach of sampling the joint density via a marginalization procedure: update the states first via the filtering distribution, $p(x_{t+1}|y^{t+1})$, update the sufficient statistics, s_{t+1}, given the data and updated state, and finally drawing the parameters via $p(\theta|s_{t+1})$. Third, the key to operationalizing this factorization is generating draws from the particle approximation to $p(x_{t+1}|y^{t+1})$. We essentially follow this outline. To do this, we use an alternative representation to express $p^N(x_{t+1}|y^{t+1})$ as a mixture distribution that can be directly sampled. Given samples from $p^N(x_{t+1}|y^{t+1})$, updating the sufficient statistics and parameters is straightforward.

The key advantage to our algorithm is that it does not rely on sequential importance sampling (SIS). SIS methods are popular and have dominated previous attempts to imple-

ment particle-based sequential learning algorithms. Importance sampling, however, suffers from well known problems related to the compounding of approximation errors, which leads to sample impoverishment and weight degeneracies. Since our algorithm exactly samples from the particle distribution, it avoids the particle degeneracies of SIS algorithms.

To demonstrate the algorithm, we analyze in detail the class of models with linear observation and state evolutions and non-Gaussian errors. This class includes robust specifications such as models with t, stable, and discrete mixtures of normals errors, as well as dynamic discrete-choice models. In this class of models, the key to efficient inference is to represent the errors as a scale mixture of normals, to introduce an auxiliary latent scaling variable, and to use data augmentation. This scale mixture representation has been extensively used to analyze the state and parameter smoothing via MCMC methods (see, for example, Carlin and Polson (1991), Carlin, Polson, and Stoffer (1992), Carter and Kohn (1994, 1996), and Shephard (1994)) and for pure state filtering using standard particle filtering (Gordon, Salmond, and Smith (1992)) and extensions such as the auxiliary particle filter (Pitt and Shephard (1999)) and mixture Kalman filter (Chen and Liu (2000)).

Pure state filtering is special case of our algorithm if the static parameters are known. In this case, our general algorithm simplifies and generates an exact algorithm for particle-based state filtering. Again, this state filtering algorithm has the advantage that it does not resort to sequential importance sampling (SIS) methods. This algorithm provides an exact alternative to popular SIS algorithms that include the approach in Gordon, Salmond, and Smith (1993) and extended in Pitt and Shephard (1999) and Chen and Liu (2000).

We illustrate our approach using two models. The first is a model with a latent autoregressive state process controlling the mean and t-distributed observation and state equation errors, a robust version of the classic linear Gaussian state space model. In the case of pure filtering, models with t-distributed errors in either (but not both) the state or observation have been analyzed in depth using approximate filters; see, for example, Masreliez and Martin (1977), Meinhold and Singpurwalla (1987), West (1981), and Gordon and Smith (1993). We also analyze a log-stochastic volatility parameterized via a mixture of normals error term as in Kim, Shephard, and Chib (1998). In both cases, we show that the algorithm is able to accurately learn all of the parameters and state variables in both simulated and real data examples. We view our algorithms as simulation based robust extensions of the Kalman filter that handle both parameter and state learning in models with non-normalities.

To date, algorithms for parameter learning and state filtering have achieved varying degrees of success. Previous attempts include the particle filters in Liu and West (2001), Storvik (2002), Chopin (2002, 2005), Doucet, and Tadic (2003), Johansen, Doucet, and Davey (2006), Andrieu, Doucet, and Tadic (2006), and Johannes, Polson, and Stroud (2005, 2006), the pure MCMC approach of Polson, Stroud, and Muller (2006), and the hybrid approach of Berzuini, Best, Gilks, and Larizza (1997) and Del Moral, Doucet, and Jasra (2006). Most of these algorithms have limited scope or difficulties even in standard models. For example, Stroud, Polson, and Muller (2006) document that Storvik's algorithm has difficulties handling outliers in an autoregressive model, while their MCMC approach has difficulties estimating the volatility of volatility in a stochastic volatility model.

The rest of the paper is outlined as follows. Section 2 describes our general approach to understand our updating mechanism. We discuss in detail the simple case of state filtering and parameter learning in a linear Gaussian state space model and the special case of pure filtering. We introduce latent auxiliary variables to transform non-normal models into conditionally Gaussian models with a sufficient statistic structure. Section 3 provides examples of the methodology in the case of $t-$distributed errors and a stochastic volatility model using simulated and real data examples. Finally, Section 4 concludes.

2 State filtering and parameter learning

Consider a state space model specified via the observation equation, $p(y_t|x_t,\theta)$, state evolution, $p(x_{t+1}|x_t,\theta)$, initial state distribution, $p(x_0|\theta)$, and prior parameter distribution, $p(\theta)$. The sequential parameter learning and state filtering problem is characterized by the joint posterior distribution, $p(\theta,x_t|y^t)$, for each time t via analytical or simulation methods. The focus on $p(\theta,x_t|y^t)$ follows from the optimality properties of the posterior distribution for solving the filtering problem via $p(x_t|y^t)$ and learning problems via $p(\theta|y^t)$.

Sequential sampling from $p(\theta,x_t|y^t)$ is difficult due to the dimensionality of the posterior and the complicated functional relationships between the parameters, states, and data. MCMC methods have been developed to solve the smoothing problem, namely sampling from $p\left(\theta,x^T|y^T\right)$, but are too slow for the sequential problem, which requires on-line simulation based on a recursive or iterative structure. The classic example of recursive estimation is the Kalman filter in the case of linear Gaussian models with known parameters and most particle filtering algorithms utilize a recursive structure.

We use a particle filtering approach to characterize $p(\theta, x_t|y^t)$. Particle methods use a discrete representation of $p(\theta, x_t|y^t)$

$$p^N(\theta, x_t|y^t) = \frac{1}{N}\sum_{i=1}^{N} \delta_{(x_t,\theta)^{(i)}},$$

where N is the number of particles and $(x_t, \theta)^{(i)}$ denotes the particle vector. As in the case of pure state filtering, the particle approximation simplifies many of the hurdles that are inherent to sequential problems. Liu and West (2001), Chopin (2002), Storvik (2002), Andrieu, Doucet, and Tadic (2005), Johansen, Doucet, and Davey (2006), and Johannes, Polson, and Stroud (2005, 2006) all use particle methods for sequential parameter learning.

Given the particle approximation, the key problem is how to *jointly* propagate the parameter and state particles. This step is complicated because the state propagation depends on the parameters, and vice versa. To circumvent the codependence in a joint draw, it is common to use importance sampling. This, however, can lead to particle degeneracies, as the importance densities may not closely match the target densities. Degeneracies are also apparent in hybrid MCMC schemes due to the long range dependence between the parameters and state variables. One essential key to breaking this dependence is to track a vector of conditionally sufficient statistics, s_t, as in Storvik (2002) and Fearnhead (2002). We characterize $p(\theta, s_t, x_t|y^t)$ via a particle approximation and update the particles in three steps, in which each component is sequentially updated. As we now show this allows to generate an exact draw from $p^N(\theta, s_{t+1}, x_{t+1}|y^{t+1})$, given existing samples from $p^N(\theta, s_t, x_t|y^t)$.

2.1 General approach

Our approach begins by expressing the joint distribution $p\left(\theta, s_{t+1}, x_{t+1}|y^{t+1}\right)$ as

$$p\left(\theta, s_{t+1}, x_{t+1}|y^{t+1}\right) = p\left(\theta|s_{t+1}\right) p\left(s_{t+1}|x_{t+1}, y^{t+1}\right) p\left(x_{t+1}|y^{t+1}\right), \tag{2}$$

where s_{t+1} is a conditionally sufficient statistic defined by the recursion

$$s_{t+1} = \mathcal{S}\left(s_t, x_{t+1}, y_{t+1}\right).$$

The sufficient statistic is a functional relative to the random variables x_{t+1} and s_t, and y_{t+1} is observed. Viewed at this level, our algorithm uses the common mechanism of expressing a joint distribution as a product of conditional and marginal distributions. Our approach

essentially follows these steps, taking advantage of the mixture structure generated by a discrete particle approximation $p^N(\theta, s_t, x_t|y^t)$. We now discuss the mechanics of each step.

We first express $p(x_{t+1}|y^{t+1})$ relative to $p(x_t, \theta|y^t)$, via

$$p\left(x_{t+1}|y^{t+1}\right) = \int p(y_{t+1}|x_t, \theta)\, p(x_{t+1}|x_t, \theta, y_{t+1})\, dp\left(x_t, \theta|y^t\right). \tag{3}$$

This representation is somewhat nonstandard, and we discuss this issue further below in Section 2.2. Given a particle approximation, $p^N(x_t, \theta|y^t)$, to the previous period's posterior, this implies that $p^N(x_{t+1}|y^{t+1})$ is given by

$$p^N\left(x_{t+1}|y^{t+1}\right) = \int p(y_{t+1}|x_t, \theta)\, p(x_{t+1}|x_t, \theta, y_{t+1})\, dp^N\left(x_t, \theta|y^t\right) \tag{4}$$

$$= \sum_{i=1}^{N} w\left((x_t, \theta)^{(i)}\right) p\left(x_{t+1}|\,(x_t, \theta)^{(i)}, y_{t+1}\right), \tag{5}$$

where the weights, w, are given by

$$w(x_t, \theta) = \frac{p(y_{t+1}|x_t, \theta)}{\sum_{i=1}^{N} p(y_{t+1}|x_t, \theta)}.$$

The distribution $p^N(x_{t+1}|y^{t+1})$ is a discrete mixture distribution, where $w(x_t, \theta)$ are the mixing probabilities and $p(x_{t+1}|x_t, \theta, y_{t+1})$ is the conditional state distribution. Standard simulation methods can now be applied to sample from $p^N(x_{t+1}|y^{t+1})$ by first resampling the particle vector (θ, x_t, s_t):

$$(\theta, x_t, s_t)^{(i)} \sim Multi_N\left(\left\{w\left((x_t, \theta)^{(i)}\right)\right\}_{i=1}^{N}\right),$$

where $Multi_N$ denotes an N-component multinomial distribution. Note that resampling applies to the triple (θ, x_t, s_t), implying that $\left(\theta^{(i)}, s_t^{(i)}, x_{t+1}^{(i)}\right)$ is drawn from $p^N(\theta, s_t, x_{t+1}|y^{t+1})$. The multinomial draw selects which mixture components to simulate from, and given the mixture component, the states are simulated from $p\left(x_{t+1}|\,(x_t, \theta)^{(i)}, y_{t+1}\right)$.

To update s_{t+1}, we use the fact that the sufficient statistics are functionally related to the previous sufficient statistic (which was resampled), $x_{t+1}^{(i)}$, and y_{t+1},

$$s_{t+1}^{(i)} = \mathcal{S}\left(s_t^{(i)}, x_{t+1}^{(i)}, y_{t+1}\right).$$

Finally, given the sufficient statistic structure, the parameter posterior is assumed to be a recognized distribution, and therefore $\theta^{(i)} \sim p\left(\theta|s_{t+1}^{(i)}\right)$ propagates the parameters.

The exact particle filtering and parameter learning is given in the following four steps.

Algorithm: Exact state filtering and parameter learning

Step 1: Draw $(\theta, x_t, s_t)^{(i)} \sim Multi_N\left(\left\{w\left((x_t,\theta)^{(i)}\right)\right\}_{i=1}^{N}\right)$ for $i = 1, ..., N$

Step 2: Draw $x_{t+1}^{(i)} \sim p\left(x_{t+1}|\left(x_t,\theta\right)^{(i)}, y_{t+1}\right)$ for $i = 1, ..., N$

Step 3: Update sufficient statistics: $s_{t+1}^{(i)} = \mathcal{S}\left(s_t^{(i)}, x_{t+1}^{(i)}, y_{t+1}\right)$ for $i = 1, ..., N$

Step 4: Draw $\theta^{(i)} \sim p\left(\theta|s_{t+1}^{(i)}\right)$ for $i = 1, ..., N$.

From the representation in equation (2), the algorithm provides an exact draw from $p^N\left(\theta, x_{t+1}, s_{t+1}|y^{t+1}\right)$. Since there are fast algorithms to draw multinomial random variables (see Carpenter, Clifford, and Fearnhead (1998)), the algorithm is $O\left(N\right)$. For convergence proofs as N increases, see Doucet, Godsill, and West (2004) in the state filtering case and Hansen and Polson (2006) for the case with state filtering and parameter learning. As with any Monte Carlo procedure, the choice N will depend on the model, the dimension of the state and parameter vectors, and T. In particular, to mitigate the accumulation of approximation errors, increasing N with T is important for long datasets.

Discussion: The algorithm requires three steps: (1) A sufficient statistic structure for the parameters, (2) an ability to evaluate $p\left(y_{t+1}|x_t,\theta\right)$, and (3) an ability to sample from $p\left(x_{t+1}|x_t,\theta,y_{t+1}\right)$. In the next section, we use a linear Gaussian model as an example, as all of these distributions are known. Section 2.2 shows how to tailor the algorithm to models with discrete or continuous scale mixture of normal distributions errors. This modification introduces auxiliary variables indexing the mixture component in the error distributions, and generates a conditional sufficient statistic structure.

For nonlinear models, the only formal requirement is that there exists a conditional sufficient statistic structure. The distribution $p\left(y_{t+1}|x_t,\theta\right)$ can be computed in many models using, for example, accurate and efficient numerical integration schemes. Similarly, if $p\left(x_{t+1}|x_t,\theta,y_{t+1}\right)$ cannot be directly sampled, indirect methods such as rejection sampling or MCMC can be used, although the computationally efficiency of these methods will depend on the dimensionality of the distribution. In models for which these densities are not known, sequential importance sampling can be used to approximate $p\left(y_{t+1}|x_t,\theta\right)$ and

$p(x_{t+1}|x_t, \theta, y_{t+1})$. Johannes, Polson, and Stroud (2006) develop a general algorithm for this case, and provide an example using an inherently nonlinear model.

2.1.1 Example: AR(1) with noise

For a concrete example, consider the latent autoregressive, AR(1), with noise model:

$$\begin{aligned} y_{t+1} &= x_{t+1} + \sigma\varepsilon_{t+1}^{y} \\ x_{t+1} &= \alpha_x + \beta_x x_t + \sigma_x \varepsilon_{t+1}^{x}, \end{aligned}$$

where the shocks are independent standard normal random variables and $\theta = (\alpha_x, \beta_x, \sigma_x^2, \sigma^2)$. We assume an initial state distribution, $x_0 \sim \mathcal{N}(\mu_0, \sigma_0^2)$ and standard conjugate priors for the parameters: $\sigma^2 \sim \mathcal{IG}(a, A)$ and $p(\alpha_x, \beta_x|\sigma_x^2)\, p(\sigma_x^2) \sim \mathcal{NIG}(b, B)$, where $\mathcal{NIG}$ is the normal/inverse gamma distribution.

In order to implement our algorithm, we need the following quantities: the predictive likelihood, the updated state distribution, the sufficient statistics, and the parameter posterior. The predictive likelihood used in the initial resampling step is

$$p(y_{t+1}|x_t, \theta) \sim \mathcal{N}\left(\alpha_x + \beta_x x_t, \sigma^2 + \sigma_x^2\right),$$

which implies that

$$w\left((x_t, \theta)^{(i)}\right) \propto \frac{1}{\sqrt{(\sigma^2)^{(i)} + (\sigma_x^2)^{(i)}}} \exp\left(-\frac{1}{2}\frac{\left(y_{t+1} - \alpha_x^{(i)} - \beta_x^{(i)} x_t^{(i)}\right)^2}{(\sigma^2)^{(i)} + (\sigma_x^2)^{(i)}}\right).$$

The updated state distribution is

$$p(x_{t+1}|x_t, \theta, y_{t+1}) \propto p(y_{t+1}|x_{t+1}, \theta)\, p(x_{t+1}|x_t, \theta) \sim \mathcal{N}\left(\mu_{t+1}, \sigma_{t+1}^2\right),$$

where

$$\frac{\mu_{t+1}}{\sigma_{t+1}^2} = \frac{y_{t+1}}{\sigma^2} + \frac{\alpha_x + \beta_x x_t}{\sigma_x^2} \text{ and } \frac{1}{\sigma_{t+1}^2} = \frac{1}{\sigma^2} + \frac{1}{\sigma_x^2}.$$

The $p(x_{t+1}|x_t, \theta, y_{t+1})$ shows how sensitive the state updating is to the model parameters.

For the parameters and sufficient statistics, we re-write the state evolution as

$$x_{t+1} = Z_t'\beta + \sigma_x \varepsilon_{t+1}^{x}$$

where $Z_t = (1, x_t)$ and $\beta = (\alpha_x, \beta_x)'$. To update the parameters, we note that the posterior is given by

$$p(\theta|s_t) = p\left(\beta|\sigma_x^2, s_t\right) p\left(\sigma^2|s_t\right) p\left(\sigma_x^2|s_t\right),$$

and we can update first the volatilities and then the regression coefficients. The conditional posteriors are known and given by

$$\begin{aligned} p\left(\sigma^2|s_{t+1}\right) &\sim \mathcal{IG}\left(a_{t+1}, A_{t+1}\right) \\ p\left(\sigma_x^2|s_{t+1}\right) &\sim \mathcal{IG}\left(b_{t+1}, B_{t+1}\right), \\ p\left(\beta|\sigma_x^2, s_{t+1}\right) &\sim \mathcal{N}\left(c_{t+1}, \sigma_x^2 C_{t+1}^{-1}\right), \end{aligned}$$

where the vector of sufficient statistics, $s_{t+1} = (A_{t+1}, B_{t+1}, c_{t+1}, C_{t+1})$, is updated via the via the functional recursions

$$\begin{aligned} A_{t+1} &= (y_{t+1} - x_{t+1})^2 + A_t, \\ B_{t+1} &= B_t + c_t' C_t c_t + Z_{t+1}' Z_{t+1} - c_{t+1}' C_{t+1} c_{t+1}, \\ c_{t+1} &= C_{t+1}^{-1}\left(C_t c_t + x_{t+1} Z_{t+1}'\right), \text{ and} \\ C_{t+1} &= C_t + Z_{t+1} Z_{t+1}'. \end{aligned}$$

The hyperparameters are deterministic and given by $a_{t+1} = 1/2 + a_t$ and $b_{t+1} = 1/2 + b_t$.

The full algorithm consists of the following steps:

Algorithm: AR(1) model state filtering and parameter learning

Step 1: Draw $(\theta, s_t, x_t)^{(i)} \sim Multi_N\left[\left\{w\left((x_t, \theta)^{(i)}\right)\right\}_{i=1}^N\right]$

Step 2: Draw $x_{t+1}^{(i)} \sim p\left(x_{t+1}|x_t^{(i)}, \theta^{(i)}, y_{t+1}\right)$ for $i = 1, ..., N$

Step 3: Update $s_{t+1}^{(i)} = \mathcal{S}\left(s_t^{(i)}, x_{t+1}^{(i)}, y_{t+1}\right)$ for $i = 1, ..., N$

Step 4: Draw $(\sigma^2)^{(i)} \sim p\left(\sigma^2|s_{t+1}^{(i)}\right) \sim \mathcal{IG}\left(a_{t+1}^{(i)}, A_{t+1}^{(i)}\right)$,

$(\sigma_x^2)^{(i)} \sim p\left(\sigma_x^2|s_{t+1}^{(i)}\right) \sim \mathcal{IG}\left(b_{t+1}^{(i)}, B_{t+1}^{(i)}\right)$, and

$(\beta)^{(i)} \sim p\left(\beta|\left(\sigma_x^2\right)^{(i)}, s_{t+1}^{(i)}\right) \sim \mathcal{N}\left(c_{t+1}^{(i)}, \left(\sigma_x^2\right)^{(i)}\left(C_{t+1}^{-1}\right)^{(i)}\right)$ for $i = 1, ..., N$.

This algorithm essentially provides a simulation based extension to the Kalman filter that can also estimate the parameters. Johannes, Polson, and Stroud (2006) develop a similar algorithm using a slightly different interacting particle systems approach. Johannes and Polson (2006) provide extensions in multivariate extensions where the observed vector or states are multivariate. These multivariate Gaussian state space models are used extensively in modeling of macroeconomic time series.

2.1.2 State filtering

We utilize a somewhat nonstandard expression for $p\left(x_{t+1}|y^{t+1}\right)$ in updating the states. To understand the mechanics of this step and to contrast it with common particle filtering algorithms, we consider the simpler case of pure filtering. For the rest of this subsection, we assume the parameters are known and fixed at those true values.

The distribution $p\left(x_{t+1}, y_{t+1}|x_t\right)$ can be expressed in different ways. We express

$$p\left(x_{t+1}, y_{t+1}|x_t\right) = p\left(x_{t+1}|x_t, y_{t+1}\right) p\left(y_{t+1}|x_t\right), \tag{6}$$

which combines the predictive likelihood $p\left(y_{t+1}|x_t\right)$ and the conditional state posterior $p\left(x_{t+1}|x_t, y_{t+1}\right)$. This leads to the marginal distribution

$$p\left(x_{t+1}|y^{t+1}\right) = \int p\left(y_{t+1}|x_t\right) p\left(x_{t+1}|x_t, y_{t+1}\right) p\left(x_t|y^t\right) dx_t. \tag{7}$$

A particle approximation to $p\left(x_t|y^t\right)$ implies that

$$p^N\left(x_{t+1}|y^{t+1}\right) = \sum_{i=1}^{N} w\left(x_t^{(i)}\right) p\left(x_{t+1}|x_t^{(i)}, y_{t+1}\right), \tag{8}$$

where the mixing probabilities are given by

$$w\left(x_t^{(i)}\right) = \frac{p\left(y_{t+1}|x_t^{(i)}\right)}{\sum_{i=1}^{N} p\left(y_{t+1}|x_t^{(i)}\right)}. \tag{9}$$

It is important to note that the mixture probabilities are a function of x_t not x_{t+1}. This implies a two-step direct draw from $p^N\left(x_{t+1}|y^{t+1}\right)$.

The state filtering algorithm consists of the following steps:

Algorithm: Exact state filtering

Step 1: (Resample) Draw $x_t^{(i)} \sim Multi_N \left\{ w\left(x_t^{(1)}\right), ..., w\left(x_t^{(N)}\right) \right\}$

Step 2: (Propagate) Draw $x_{t+1}^{(i)} \sim p\left(x_{t+1}|x_t^{(i)}, y_{t+1}\right)$.

In contrast, the standard particle filtering approach expresses $p\left(y_{t+1}, x_{t+1}|x_t\right)$ as

$$p\left(x_{t+1}, y_{t+1}|x_t\right) \propto p\left(y_{t+1}|x_{t+1}\right) p\left(x_{t+1}|x_t\right) \tag{10}$$

and treats $p\left(x_{t+1}|y^{t+1}\right)$ as a marginal against $p\left(x_t|y^t\right)$:

$$p\left(x_{t+1}|y^{t+1}\right) = \int p\left(y_{t+1}|x_{t+1}\right) p\left(x_{t+1}|x_t\right) p\left(x_t|y^t\right) dx_t. \tag{11}$$

The particle approximation of $p\left(x_t|y^t\right)$ by $p^N\left(x_t|y^t\right)$ then implies that

$$p^N\left(x_{t+1}|y^{t+1}\right) = \sum_{i=1}^{N} w\left(x_{t+1}^{(i)}\right) p\left(x_{t+1}|x_t^{(i)}\right), \tag{12}$$

where

$$w\left(x_{t+1}^{(i)}\right) = \frac{p\left(y_{t+1}|x_{t+1}^{(i)}\right)}{\sum_{i=1}^{N} p\left(y_{t+1}|x_{t+1}^{(i)}\right)}$$

Sampling from this mixture distribution is difficult because the natural mixing probabilities depend on x_{t+1}, which has yet to be simulated. Instead of direct sampling, the common approach is to use importance sampling and the sampling-importance resampling (SIR) algorithm of Rubin (1988) or Smith and Gelfand (1992). This generates the classic SIR PF algorithm:

$$\begin{aligned} &\text{(Propagate) Draw } x_{t+1}^{(i)} \sim p\left(x_{t+1}|x_t^{(i)}\right) \text{ for } i = 1, ..., N \\ &\text{(Resample) Draw } x_{t+1}^{(i)} \sim Multi_N \left[\left\{ w\left(x_{t+1}^{(i)}\right) \right\}_{i=1}^{N} \right]. \end{aligned}$$

We use a multinomial resampling step, although other approaches are available (see Liu and Chen (1998) or Carpenter, Clifford, and Fearnhead (1999)). The classic PF algorithm suffers from a number of well-known problems as it blindly simulates states, even though y_{t+1} is observed, and relies on importance sampling. Importance sampling typically results in weight degeneracy or sample impoverishment.

Notice that our algorithm is in *exactly the opposite order* as the classical particle filter. First, the algorithm selects particles to propagate forward via their likelihood $p\left(y_{t+1}|x_t^{(i)}\right)$. This results in propagating high-likelihood particles multiple times and is key to an efficient algorithm. Second, the algorithm propagates states via $p\left(x_{t+1}|x_t^{(i)}, y_{t+1}\right)$, taking into account the new observation. The draws all have equal probability weights, so there is no need to track the weights.

Our algorithm is closely related to the optimal importance function algorithms derived in Doucet, et al. (2000). Their algorithm effectively reverses our Steps 1 and 2, by first simulating from $p\left(x_{t+1}|x_t^{(i)}, y_{t+1}\right)$ and then reweighting those draws. Like Doucet, et al. (2000), our algorithm requires that $p\left(y_{t+1}|x_t\right)$ is known and $p\left(x_{t+1}|x_t, y_{t+1}\right)$ can be simulated. However, our algorithm is not an importance sampling algorithm as it provides exact draws from the target distribution, $p^N\left(x_{t+1}|y^{t+1}\right)$.

2.2 Non-Gaussian models

In this section, we consider in detail the class of mixture models:

$$\begin{aligned} y_{t+1} &= x_{t+1} + \sigma\sqrt{\lambda_{t+1}}\varepsilon_{t+1} \\ x_{t+1} &= \alpha_x + \beta_x x_t + \sigma_x\sqrt{\omega_{t+1}}\varepsilon_{t+1}^x, \end{aligned}$$

where the specification of λ_{t+1} and ω_{t+1} determines the error distribution. For example, $\lambda_{t+1} \sim IG\left(\nu/2, \nu/2\right)$ generates a marginal distribution for observation errors that is $t-$distributed with $\nu-$degrees of freedom. The case of discrete mixtures is handled similarly. We also assume that there exist conditional sufficient statistics for the parameters, $p\left(\theta|s_t\right)$, where the recursions for the sufficient statistics are given by

$$s_{t+1} = \mathcal{S}\left(s_t, x_{t+1}, \omega_{t+1}, \lambda_{t+1}, y_{t+1}\right).$$

It is important to note that the parameter posteriors generally do not admit sufficient statistics unless we introduce the latent auxiliary variables.

This class of shocks has a long history in state space model. T-distributed errors in the observation equation were analyzed by Masreliez (1975) and Masreliez and Martin (1977), but they did not consider t-distribution state shocks. In the case of smoothing, this class of shocks is considered using MCMC methods by Carlin, Polson, and Stoffer (1992), Carter and Kohn (1994, 1996) and Shephard (1994). This implies that we allow for $t-$distributed

errors, stable errors, double exponential errors, and discrete mixture errors. This latter case includes the important class of log-stochastic volatility models using the representation of Kim, Shephard, and Chib (1998).

The algorithm outlined in Section 2.1 requires an analytical form for $p\left(y_{t+1}|x_t,\theta\right)$ and an ability to simulate from $p\left(\theta|s_{t+1}\right)$ and $p\left(x_{t+1}|x_t,\theta,y_{t+1}\right)$. For the mixture models, these densities are analytically known. However, the algorithm can be slightly modified to handle these non-Gaussian and non-linear components. The key is twofold: utilizing the fact that $p\left(y_{t+1}|x_{t+1},\lambda_{t+1},\theta\right)$ and $p\left(x_{t+1}|x_t,\lambda_{t+1},\theta\right)$ are Gaussian distributions and then a careful marginalization to sequentially update x_{t+1} and λ_{t+1}.

2.2.1 Main algorithm

The algorithm proceeds via an analog to (1). For notational parsimony, we will denote the latent variables by just λ_{t+1} from now on. The factorization is

$$p\left(\theta,s_{t+1},\lambda_{t+1},x_{t+1}|y^{t+1}\right)=p\left(\theta|s_{t+1}\right)p\left(s_{t+1}|x_{t+1},\lambda_{t+1}|y^{t+1}\right)p\left(x_{t+1}|\lambda_{t+1},y^{t+1}\right)p\left(\lambda_{t+1}|y^{t+1}\right),$$

by first updating λ_{t+1}, then x_{t+1}, then s_{t+1}, and finally θ. As in Section 2.1, to generate samples from the joint, we rely on the factorization and careful marginalization arguments.

Given existing particles, the first step is to propagate the mixture variables, λ_{t+1}. To do this, we generate draws from a higher dimensional distribution, and then obtain draws from $p\left(\lambda_{t+1}|y^{t+1}\right)$ as the marginal distribution. To do this, first note that

$$p\left(\lambda_{t+1},x_t,\theta|y^{t+1}\right)\propto p\left(y_{t+1}|\theta,\lambda_{t+1},x_t\right)p\left(\theta,\lambda_{t+1},x_t|y^t\right).$$

To sample from this joint distribution, we use the fact that

$$p\left(\lambda_{t+1},x_t,\theta|y^t\right)=p\left(\lambda_{t+1}\right)p\left(x_t,\theta|y^t\right),$$

as λ_{t+1} is conditionally independent. To sample this distribution, we first simulate $\lambda_{t+1}\sim p\left(\lambda_{t+1}\right)$ and augment the $\left(x_t,\theta\right)^{(i)}$ draw to obtain a joint draw $\left(\theta,\lambda_{t+1},x_t\right)^{(i)}$. Next, we sample from $p\left(\lambda_{t+1},x_t,\theta|y^{t+1}\right)$ by drawing from the discrete distribution

$$\left(\theta,\lambda_{t+1},x_t\right)^{(i)}\sim Mult_N\left\{\left\{w\left(\left(\theta,\lambda_{t+1},x_t\right)^{(i)}\right)\right\}_{i=1}^{N}\right\},$$

where the weights are given by

$$w\left(\left(\theta,\lambda_{t+1},x_t\right)^{(i)}\right)=\frac{p\left(y_{t+1}|\left(\theta,\lambda_{t+1},x_t\right)^{(i)}\right)}{\sum_{i=1}^{n}p\left(y_{t+1}|\left(\theta,\lambda_{t+1},x_t\right)^{(i)}\right)}.$$

Since $p\left(y_{t+1}|\lambda_{t+1}, x_t, \theta\right)$ is known for all of the models that we consider, this step is feasible.

To propagate the states, express

$$p\left(x_{t+1}|\lambda_{t+1}, y^{t+1}\right) = \int p\left(x_{t+1}|\theta, \lambda_{t+1}, x_t, y_{t+1}\right) p\left(\theta, x_t|y^{t+1}\right) d\left(\theta, x_t\right).$$

and note that we already have draws from the particle approximation to $p\left(x_t, \theta|y^{t+1}\right)$ as a marginal from $p\left(\lambda_{t+1}, x_t, \theta|y^{t+1}\right)$. Therefore, we can sample x_{t+1} via

$$x_{t+1}^{(i)} \sim p\left(x_{t+1}|\left(\theta, \lambda_{t+1}, x_t\right)^{(i)}, y_{t+1}\right),$$

since the distribution $p\left(x_{t+1}|\theta, \lambda_{t+1}, x_t, y_{t+1}\right)$ is known for all of the mixture models. Given the updated states, we update the sufficient statistics via

$$s_{t+1}^{(i)} = \mathcal{S}\left(s_t^{(i)}, x_{t+1}^{(i)}, \lambda_{t+1}^{(i)}, y_{t+1}\right),$$

and draw $\theta^{(i)} \sim p\left(\theta|s_{t+1}^{(i)}\right)$.

The full algorithm is given by the following steps.

Algorithm: Non-Gaussian sequential parameter learning and state filtering

Step 1: Draw $\lambda_{t+1}^{(i)} \sim p\left(\lambda_{t+1}\right)$ for $i = 1, ..., N$

Step 2: Draw $\left(\theta, s_t, \lambda_{t+1}, x_t\right)^{(i)} \sim Mult_N\left\{\left\{w\left(\left(\theta, \lambda_{t+1}, x_t\right)^{(i)}\right)\right\}_{i=1}^{N}\right\}$ for $i = 1, ..., N$

Step 3: Draw: $x_{t+1}^{(i)} \sim p\left(x_{t+1}|\left(\theta, \lambda_{t+1}, x_t\right)^{(i)}, y_{t+1}\right)$ for $i = 1, ..., N$

Step 4: Update: $s_{t+1}^{(i)} = \mathcal{S}\left(s_t^{(i)}, x_{t+1}^{(i)}, \lambda_{t+1}^{(i)}, y_{t+1}\right)$ for $i = 1, ..., N$

Step 5: Draw $\theta \sim p\left(\theta|s_{t+1}^{(i)}\right)$ for $i = 1, ..., N$.

This algorithm provides an exact sample from $p^N\left(\theta, s_{t+1}, \lambda_{t+1}, x_{t+1}|y^{t+1}\right)$.

After Step 3, an additional step can be introduced to update λ_{t+1} from $p\left(\lambda_{t+1}|\theta, x_{t+1}, y_{t+1}\right)$. This is effectively a one-step MCMC replenishment step. As the algorithm is already approximately sampling from the "equilibrium" distribution, the marginal for λ_{t+1}, an additional replenishment step for λ_{t+1} may help by introducing additional sample diversity.

2.2.2 Pure state filtering

If we assume the parameters are known and focus on the state filtering problem, we can adapt the algorithms from the previous section to provide exact particle filtering algorithms. Existing state filtering algorithms for these models rely on importance sampling methods either via the auxiliary particle filter of Pitt and Shephard (1999) or the mixture-Kalman filter of Chen and Liu (2000). Both of the algorithms above provide exact $O(N)$ algorithms for state filtering, and we briefly discuss these algorithms as they offer generic improvements on the existing literature.

There are two ways to factor the joint filtering densities:

$$p\left(x_{t+1}, \lambda_{t+1} | y^{t+1}\right) = p\left(x_{t+1} | \lambda_{t+1}, y^{t+1}\right) p\left(\lambda_{t+1} | y^{t+1}\right)$$

or

$$p\left(x_{t+1}, \lambda_{t+1} | y^{t+1}\right) = p\left(\lambda_{t+1} | x_{t+1}, y^{t+1}\right) p\left(x_{t+1} | y^{t+1}\right),$$

with the differences based on the order of auxiliary variable or state variable updates.

The first factorization leads to an initial draw from $p\left(\lambda_{t+1} | y^{t+1}\right)$. Since,

$$p\left(\lambda_{t+1}, x_t | y^{t+1}\right) \propto p\left(y_{t+1} | \lambda_{t+1}, x_t\right) p\left(\lambda_{t+1}, x_t | y^t\right)$$

and the latent auxiliary variables are i.i.d., we have that $p\left(\lambda_{t+1}, x_t | y^t\right) \propto p\left(\lambda_{t+1}\right) p\left(x_t | y^t\right)$. Therefore, to draw from

$$p\left(\lambda_{t+1} | y^{t+1}\right) = \int p\left(y_{t+1} | \lambda_{t+1}, x_t\right) p\left(\lambda_{t+1}\right) p\left(x_t | y^t\right) dx_t,$$

we can augment the existing particles $x_t^{(i)}$ from $p^N\left(x_t | y^t\right)$ with $\lambda_{t+1}^{(i)}$ draws, and resample with probabilities given by

$$w\left(\lambda_{t+1}, x_t\right)^{(i)} = \frac{p\left(y_{t+1} | \left(\lambda_{t+1}, x_t\right)^{(i)}\right)}{\sum_{i=1}^n p\left(y_{t+1} | \left(\lambda_{t+1}, x_t\right)^{(i)}\right)}.$$

Next, we update state variables via

$$p\left(x_{t+1} | \lambda_{t+1}, y^{t+1}\right) = \int p\left(x_{t+1} | x_t, \lambda_{t+1}, y_{t+1}\right) p\left(y_{t+1} | x_t, \lambda_{t+1}\right) p\left(x_t | y^t\right) dx_t$$

which implies that we can draw $x_{t+1} | \lambda_{t+1}^{(i)}$ using the resampled $\left(\lambda_{t+1}, x_t\right)^{(i)}$ and draw from

$$p\left(x_{t+1} | \left(\lambda_{t+1}, x_t\right)^{(i)}, y_{t+1}\right).$$

This generates an exact draw from $p^N\left(x_{t+1}, \lambda_{t+1}|y^{t+1}\right)$.

The second approach updates the state variables and then the latent auxiliary variables. To sample from $p^N\left(x_{t+1}|y^{t+1}\right)$, we use a slight modification of the filtering distribution,

$$p\left(x_{t+1}|y^{t+1}\right) = \int p\left(y_{t+1}|\lambda_{t+1}, x_t\right) p\left(x_{t+1}|x_t, \lambda_{t+1}, y_{t+1}\right) p\left(\lambda_{t+1}, x_t|y^t\right) d\left(\lambda_{t+1}, x_t\right).$$

Since λ_{t+1} is independent of y^t and x_t, we can simulate $\lambda_{t+1} \sim p\left(\lambda_{t+1}\right)$ and create an augmented particle vector $\left(x_t^{(i)}, \lambda_{t+1}^{(i)}\right)$. Given this particle approximation for (x_t, λ_t), we have that

$$p^N\left(x_{t+1}|y^{t+1}\right) = \sum_{i=1}^{N} w\left(x_t^{(i)}, \lambda_{t+1}^{(i)}\right) p\left(x_{t+1}|x_t^{(i)}, \lambda_{t+1}^{(i)}, y_{t+1}\right),$$

where

$$w\left(x_t^{(i)}, \lambda_{t+1}^{(i)}\right) = \frac{p\left(y_{t+1}|x_t^{(i)}, \lambda_{t+1}^{(i)}\right)}{\sum_{i=1}^{n} p\left(y_{t+1}|x_t^{(i)}, \lambda_{t+1}^{(i)}\right)}$$

This mixture distribution can again be exactly sampled. Updating the auxiliary variables is straightforward since

$$p\left(\lambda_{t+1}|x_{t+1}, y^{t+1}\right) \propto p\left(y_{t+1}|x_{t+1}, \lambda_{t+1}\right) p\left(\lambda_{t+1}\right)$$

is a known distribution for all of the mixture models under consideration.

3 Illustrative Examples

In this section, we provide the details of our sequential parameter learning and state filtering algorithms for the two models that we consider.

3.1 T-distributed errors

The first example assumes that the error distribution in both the observation and state equation are $t-$distributed with ν and ν^x degrees of freedom. We write the model in terms of the scale mixture representation:

$$\begin{aligned} y_{t+1} &= x_{t+1} + \sigma\sqrt{\lambda_t}\varepsilon_t \\ x_{t+1} &= \alpha_x + \beta_x x_t + \sigma_x\sqrt{\omega_{t+1}}\varepsilon_{t+1}^x \end{aligned}$$

where the auxiliary variables are independent and $\lambda_{t+1} \sim \mathcal{IG}(\nu/2, \nu/2)$ and $\omega_{t+1} \sim \mathcal{IG}(\nu^x/2, \nu^x/2)$. Conditional on λ_{t+1} and ω_{t+1}, the model is conditionally Gaussian.

Masreliez and Martin (1977) develop approximate robust state filters for models with $t-$distributed errors in either the state or observation equation, but not both. West (1981) and Gordon and Smith (1993) analyze the pure filtering problem. Storvik (2002) uses importance sampling to analyze sequential parameter learning and state filtering using importance sampling assuming the observation errors, but not state errors, are $t-$distributed. To our knowledge, this algorithm provides the first algorithm for parameter and state learning with t-errors in both equations.

Applying the general algorithm in Section 2.2.1, the distributions $p(y_{t+1}|\theta, \lambda_{t+1}, \omega_{t+1}, x_t)$ and $p(x_{t+1}|\theta, \lambda_{t+1}, \omega_{t+1}, x_t)$ are required to implement our algorithm. The first distribution, $p(y_{t+1}|\theta, \lambda_{t+1}, \omega_{t+1}, x_t)$, defines the weights which are given by

$$w\left((x_t, \theta)^{(i)}\right) \propto \frac{1}{\sqrt{\left[(\sigma^2)^{(i)} \lambda_{t+1}^{(i)}\right] + (\sigma_x^2)^{(i)}}} \exp\left(-\frac{1}{2} \frac{\left(y_{t+1} - \alpha_x^{(i)} - \beta_x^{(i)} x_t^{(i)}\right)^2}{(\sigma^2)^{(i)} \lambda_{t+1}^{(i)} + (\sigma_x^2)^{(i)}}\right).$$

The updated state distribution is

$$p(x_{t+1}|\theta, \lambda_{t+1}, \omega_{t+1}, x_t) \propto p(y_{t+1}|\lambda_{t+1}, x_{t+1}, \theta)\, p(x_{t+1}|\omega_{t+1}, x_t, \theta) \sim \mathcal{N}\left(\mu_{t+1}, \sigma_{t+1}^2\right),$$

where

$$\frac{\mu_{t+1}}{\sigma_{t+1}^2} = \frac{y_{t+1}}{\sigma^2 \lambda_{t+1}} + \frac{\alpha_x + \beta_x x_t}{\sigma_x^2 \omega_{t+1}} \text{ and } \frac{1}{\sigma_{t+1}^2} = \frac{1}{\sigma^2 \lambda_{t+1}} + \frac{1}{\sigma_x^2 \omega_{t+1}}.$$

For the parameter posteriors and sufficient statistics, we re-write the state equation as

$$x_{t+1} = Z_t' \beta + \sigma_x \sqrt{\omega_{t+1}} \epsilon_{t+1}$$

where $Z_t = (1, x_t)'$ and $\beta = (\alpha_x, \beta_x)'$. Given this parameterization, the sufficient statistic structure implies that $p(\sigma^2|s_{t+1}) \sim \mathcal{IG}(a_{t+1}, A_{t+1})$, $p\left(\sigma_x^2|s_{t+1}\right) \sim \mathcal{IG}(b_{t+1}, B_{t+1})$, and $p(\beta|\sigma_x^2, s_{t+1}) \sim \mathcal{N}\left(c_{t+1}, \sigma_x^2 C_{t+1}^{-1}\right)$. The hyperparameters are given by $a_t = \frac{1}{2} + a_{t-1}$,

$b_t = \frac{1}{2} + b_{t-1}$, and

$$
\begin{aligned}
A_{t+1} &= \frac{(y_{t+1} - x_{t+1})^2}{\lambda_{t+1}} + A_t \\
B_{t+1} &= B_t + c_t' C_t c_t + \frac{x_{t+1}^2}{\omega_{t+1}} - c_{t+1}' C_{t+1} c_{t+1} \\
c_{t+1} &= C_{t+1}^{-1} \left(C_t c_t + \frac{Z_{t+1}' x_{t+1}}{\omega_{t+1}} \right) \\
C_{t+1} &= C_t + \frac{Z_{t+1} Z_{t+1}'}{\omega_{t+1}},
\end{aligned}
$$

which defines the vector of sufficient statistics, $s_{t+1} = (A_{t+1}, B_{t+1}, c_{t+1}, C_{t+1})$, for sequential parameter learning.

The t-distributed error model requires the specification of the degrees of freedom parameter in the t-distributions. Here, we leave (ν, ν^x) as known parameters. It is not possible to add this parameter in the state vector, but one could compute their posterior distribution by discretizing the support.

3.2 SV errors

Consider next the log-stochastic volatility model, first analyzed in Jacquier, Polson, and Rossi (1994) and subsequently by many others:

$$
\begin{aligned}
y_t &= \exp\left(\frac{x_t}{2}\right) \varepsilon_t \\
x_t &= \alpha_x + \beta_x x_{t-1} + \sigma_v v_t
\end{aligned}
$$

where the errors are independent standard normal random variables. To estimate the model, we use the transformation approach of Kim, Shephard, and Chib (1998) and the 10-component mixture approximation developed in Omori, Chib and Shephard (2006).

The Kim, Shephard, and Chib (1998) transformation analyzes the logarithm of squared returns, $y_t^\star = \ln y_t^2$ and the $K = 10$-component normal mixture approximation we have a state space model of the form

$$
\begin{aligned}
y_t^\star &= x_t + \epsilon_t \\
x_t &= \alpha_x + \beta_x x_{t-1} + \sigma_v v_t
\end{aligned}
$$

where ϵ_t is a $log\left(\chi_1^2\right)$ which is approximated by a discrete mixture of normals with fixed weights, $\sum_{j=1}^{K} p_j Z_j$ where $Z_j \sim N(\mu_j, \sigma_j^2)$. The indicator variable I_t tracks the mixture components, with, for example, $I_t = j$ indicating a current state in mixture component j.

Our state filtering and sequential algorithm will track particles and sufficient statistics $\left(x_t^{(i)}, \theta^{(i)}, s_t^{(i)}\right)$. Here s_t are the usual sufficient statistics for estimating the parameters $\theta = (\alpha, \beta, \sigma_v)$. The sufficient statistics are conditionally on the indicator variables, and are of the same form as a standard AR(1) model, as the error distribution is known.

To implement the algorithm, first note that we can calculate the following conditional densities

$$p(y_{t+1}^{\star}|x_t, \theta) = \sum_{j=1}^{K} p_j N(\alpha + \beta x_t, \sigma_j^2 + \sigma_v^2)$$

That is the predictive density of the next observation given the current state is a mixture of normals. We use this to define weights $w(I_{t+1}, x_t, \theta)$ as

$$w\left(I_{t+1}, x_t, \theta\right) \propto \frac{1}{\sqrt{\sigma_{I_{t+1}}^2 + \sigma_v^2}} \exp\left(-\frac{1}{2} \frac{\left(y_{t+1}^{*} - \mu_{I_{t+1}} - \alpha_x - \beta_x x_t\right)^2}{\sigma_{I_{t+1}}^2 + \sigma_v^2}\right).$$

and we let $w^{\star}\left(I_{t+1}, x_t, \theta\right)$ denote the weights normalized to sum to one.

Notice that $p\left(x_t, \theta, I_{t+1}|y^t\right) = p\left(x_t, \theta|y^t\right) p(I_{t+1})$ as the mixture indicator are independent. Hence the next filtering distribution is given by

$$p(x_{t+1}|y^{t+1}) = \int_{x_t, \theta, I_{t+1}} w(I_{t+1}, x_t, \theta) p\left(x_{t+1}|(I_{t+1}, x_t, \theta)^{(i)}, y_{t+1}^{\star}\right)$$

Now we can compute the updated filtering distribution of the next log-volatility state and component indicator as follows

$$p\left(x_{t+1}, I_{t+1} = j|x_t, \theta, y_{t+1}^{\star}\right) \propto p\left(y_{t+1}^{\star}|x_{t+1}, I_{t+1} = j, \theta\right) p\left(x_{t+1}, I_{t+1} = j|x_t, \theta\right)$$

Therefore

$$p\left(x_{t+1}, I_{t+1} = j|x_t, \theta, y_{t+1}^{\star}\right) \propto p\left(y_{t+1}^{\star}|x_{t+1}, I_{t+1} = j, \theta\right) p\left(x_{t+1}, I_{t+1} = j|x_t, \theta\right) p(I_{t+1} = j)$$

Again by Bayes rule we can write this proportional to

$$p\left(x_{t+1}|I_{t+1} = j, \theta, y_{t+1}^{\star}\right) p\left(y_{t+1}^{\star}|x_t, \theta\right) p(I_{t+1} = j)$$

Using the definition of the predictive in terms of the weight function and the fact that $p(I_{t+1} = j) = p_j$ we obtain a density proportional to

$$\sum_{i=1}^{N} w_j^\star \left(x_t^{(i)}, \theta^{(i)}, I_{t+1}^{(i)} \right) p \left(x_{t+1} | I_{t+1}^{(i)}, x_t^{(i)}, \theta^{(i)}, y_{t+1}^\star \right)$$

Hence the next particle filtering distribution $p^N \left(x_{t+1} | \left(y^\star \right)^{t+1} \right)$ is a mixture of normals which can be sampled from directly.

The density is given by the Kalman filter recursion and is a conditional normal

$$p \left(x_{t+1} | (I_{t+1}, x_t, \theta)^{(i)}, y_{t+1}^\star \right) \sim N \left(\hat{x}_{t+1, I_{t+1}}, \hat{s}_{t+1, I_{t+1}}^2 \right)$$

where

$$\hat{x}_{t+1,j} = \frac{\sigma_v^2}{\sigma_v^2 + v_j^2} \left(y_{t+1}^\star - m_j \right) + \frac{v_j^2}{\sigma_v^2 + v_j^2} \left(\alpha + \beta x_t \right)$$

$$\hat{s}_{t+1,j}^{-2} = \sigma_v^{-2} + v_j^{-2}$$

Hence the next filtering distribution for $x_{t+1}^{(i)}$ is easy to sample from. The update sufficient statistics $s_{t+1}^{(i)}$. Then sample new $\theta | s_{t+1}^{(i)}$ draw.

Since the algorithm is slightly different from the ones above, we provide the details. The algorithm requires the following steps:

1. Draw $I_{t+1}^{(i)} \sim p(I_{t+1} | x_t, \theta) = p(I_{t+1}) = p_j$
2. Re-sample triples $(I_{t+1}^{(i)}, x_t^{(i)}, \theta^{(i)})$ with weights $w^\star \left(I_{t+1}, x_t, \theta \right)$
3. Draw $x_{t+1}^{(i)} \sim p \left(x_{t+1} | (I_{t+1}, x_t, \theta)^{(i)}, y_{t+1}^\star \right)$
4. Update sufficient statistics $s_{t+1}^{(i)} = S \left(s_t^{(i)}, I_{t+1}^{(i)}, x_{t+1}^{(i)}, y_{t+1} \right)$
5. Draw $\theta^{(i)} \sim p \left(\theta | s_{t+1}^{(i)} \right)$

Our approach uses be exact sampling from the particle approximation distribution. Other authors have done sequential and parameter learning but have approximate algorithms that also have difficulty in learning σ_v. Johannes, Polson and Stroud (2005) propose an alternative approach to the exact sampling scheme used here based on interacting particle systems using importance sampling. They also analyze the nonlinear model without using the mixture of errors transformation.

3.3 Numerical results

3.3.1 T-errors model: Simulated data

We simulate $T = 300$ observation from the $t-$distributed model with $\nu = \nu_x = 5$ and assuming $\alpha = 0$, $\beta = 0.9$, $\sigma = \sqrt{0.1} \approx 0.316$, and $\sigma_x = \sqrt{0.04} = 0.2$. These parameters generate a quite challenging inference problem because the state variables are quite volatile relative to the variance of the observations, and a volatility state variable is more difficult to estimate. We use the following priors hyperparameters: $\alpha_x|\sigma_x^2 \sim \mathcal{N}(0, 0.1\sigma_x^2)$, $\beta_x|\sigma_x^2 \sim \mathcal{N}(0.9, 2\sigma_x^2)$, $\sigma_x^2 \sim \mathcal{IG}(10, 0.36)$, and $\sigma^2 \sim \mathcal{IG}(10, 0.9)$. The algorithm was run with $N = 5000$ particles.

The results from a representative simulation are provided in Figures 1 to 3. Figure 1 shows the simulate sample path of y_t (top panel), the simulated sample path of x_t (bottom panel, thick line), and the 5%, 50%, and 95% posterior quantiles (bottom panel, thin lines). The true values are always within the posterior confidence bands. Interestingly, despite the fat-tails for the observation and state transition shocks, the algorithm does a good job of learning the state variables, although of course, the accuracy depends on the parameters.

Figures 2 and 3 summarize the parameter learning. For each parameter, Figure 2 provides 5, 50, and 95% posterior quantiles at each time point, as well as the true values (horizontal line). From this, we see that although the priors for each parameter are relatively loose, the algorithm is able to accurately learn all of the parameters. Due to the t-error specification, large observations do not have a major impact on the parameter estimates, consistent with bounded influence functions for models with $t-$errors.

Figure 3 provides summaries of the posterior distribution at time $t = 300$ via a histogram, a smoothed histogram, and the true parameter value (solid vertical line) for each of the parameters. Although some of the parameters are slightly biased, this is due to well known finite sample biases of likelihood based estimators. The posterior means are consistent MLE estimators computed using the true simulated state variables, indicating any biases are due to finite sample concerns.

3.3.2 T-errors: real data

We consider a real data example using daily Nasdaq 100 stock index returns from 1999 to August 2006 for a total of $T = 1953$ observations. The priors are given by $\alpha_x|\sigma_x^2 \sim \mathcal{N}(0, 0.1\sigma_x^2)$, $\beta_x|\sigma_x^2 \sim \mathcal{N}(0.7, 2\sigma_x^2)$, $\sigma_x^2 \sim \mathcal{IG}(20, 0.154)$, and $\sigma^2 \sim \mathcal{IG}(5, 10)$. Again, the

algorithm was run with $N = 5000$ particles. The results are in Figures (4) to (6).

The results provide a number of interesting findings. First, Figure (4) indicates that there is little evidence for a time-varying mean for Nasdaq 100 stock returns. This is not surprising because stock returns, and Nasdaq returns in particular, are quite noisy and past evidence indicates that it is difficult to identify mean predictability over short frequencies such as daily. Predictability, if it is present, is commonly seen over longer horizons such as quarterly or annually. The filtered quantiles in the bottom panel of Figure (4) indicate that there could be predictability, as the $(5, 95)\%$ bands are roughly -0.5% and 0.5%, but there is too much uncertainty to identify it.

Second, one source of the uncertainty, especially in the early parts of the sample, is the uncertainty over the parameters, which is shown in Figure (5). For each of the parameters, the priors are relatively loose. This generates substantial uncertainty for the early portion of the sample, and contributes to the highly uncertain filtered state distribution.

Third, a closer examination of Figure (5) shows that posterior for σ seems to be varying over time, as it increase in the early portion of the sample and decreases in the latter portion. This is capturing time-varying volatility as volatility declined in equity markets since the early part of 2003. This can be seen from the data in the upper panel of Figure (4) and will be clear in the stochastic volatility example below. It is important to note that this is not due to outliers, since we allow for fat-tailed t-errors in both the observation and state equation. This provides a useful diagnostic for slow time-variation: the fact that the posterior for σ appears to be varying over time indicates that the model is misspecified and a more general specification with stochastic volatility is warranted. Finally, Figure (6) shows the posterior distribution at time T, and shows that the posteriors are slightly non-normal, consistent with the findings in Jacquier, Polson, and Rossi (1994).

3.3.3 Simulated data: stochastic volatility model

We simulate $T = 300$ observations from the stochastic volatility model assuming $\alpha = -0.0084$, $\beta = 0.98$, $\sigma_x = \sqrt{0.04} = 0.2$. We use the following priors hyperparameters $\alpha_x|\sigma_x^2 \sim \mathcal{N}(0, \sigma_x^2/30)$, $\beta_x|\sigma_x^2 \sim \mathcal{N}(0.95, 0.1\sigma_x^2)$, and $\sigma_x^2 \sim \mathcal{IG}(8, 0.35)$. The algorithm was run using $N = 5000$ particles.

The results from a representative simulation are provided in Figures (7) to (9). The results are largely consistent with those given previously for the AR(1) with $t-$errors. The true values are always within the posterior confidence bands, and the algorithms are able to

learn the true parameter values. Of note is that despite the near-unit behavior of stochastic volatility, we are able to accurate estimate the persistence parameter.

3.3.4 Real data

We consider a real data example using daily Nasdaq 100 stock index returns from 1999 to August 2006. The priors used are given by $\alpha_x|\sigma_x^2 \sim \mathcal{N}(0, 0.1\sigma_x^2)$, $\beta_x|\sigma_x^2 \sim \mathcal{N}(0.7, \sigma_x^2/4)$, and $\sigma_x^2 \sim \mathcal{IG}(30, 0.725)$. The algorithm was run using $N = 5000$ particles. The results are given in Figures (10) to (12)

The previous results, in Figure (5), indicated the estimates of σ varied over time in the t-errors model. This is more easily seen in the bottom panel of Figure (10), which displays the posterior quantiles of daily volatility, $exp(x_t/2)$. Daily volatility was high and volatile in the 2000-2002 period, volatility declined almost monotonically in 2003-2006. This slow time-variation is exactly what the stochastic volatility models aims to capture.

Figure (11) shows the posterior quantiles over time, and provides some evidence of time-variation. In the early portion of the sample, volatility was higher than the latter portion. This feature is captured in the top panel of (11) by time-variation in the posterior for α_x, which controls the mean of log-volatility. The posterior means for α are much higher in 1999-2000, than in the latter years, although there is greater uncertainty in the early portion of the sample. It is interesting to note that the posteriors for β and σ_x vary less over time. Figure (12) displays the posterior at time T. Given the large sample, there is relatively little evidence for non-normality in the posteriors.

4 Conclusions

In this paper, we provide an exact sampling algorithm for performing sequential parameter learning and state filtering for nonlinear, non-Gaussian state space models. The implication of this is that we do not resort to importance sampling, and thus avoid the well known degeneracies associated with sequential importance sampling methods. Formally, the only assumption we require is that the parameter posterior admits a sufficient statistic structure. We analyze the class of linear non-Gaussian models in detail, and exact state filtering is a special case of our algorithm. Thus, we provide an exact sampling alternative to algorithms such as the auxiliary particle filter of Pitt and Shephard (1999) and mixture Kalman filter of Chen and Liu (2000) We provide both simulation and real data examples to document

the efficacy of the approach.

We are currently working on two extensions. First, in Johannes and Polson (2006), we examine sequential parameter learning and state filtering algorithms for multivariate Gaussian models, deriving the exact distributions required to implement the algorithms. Second, in Johannes, Polson, and Yae (2006), we consider the problem of robust filtering. Here, we adapt our algorithms to handle sequential parameter and state filtering via "robust" non-differentiable criterion functions such as least absolute deviations and quantiles. Our algorithm compare favorably with those in the existing literature.

5 References

Andrieu, C., A. Doucet, and V.B. Tadic, 2006, Online simulation-based methods for parameter estimation in non linear non Gaussian state-space models, *Proc. IEEE CDC*, forthcoming.

Berziuni, C., Best, N., Gilks, W. and Larizza, C. (1997). Dynamic conditional independence models and Markov Chain Monte Carlo methods. *Journal of American Statistical Association*, 92, 1403-1412.

Carlin, B. and N.G. Polson, 1992, Monte Carlo Bayesian Methods for Discrete Regression Models and Categorical Time Series. *Bayesian Statistics 4*, J.M. Bernardo, et al. (Eds.), Oxford, Oxford University Press, 577-586.

Carlin, B., N.G. Polson, and D. Stoffer, 1992, A Monte Carlo Approach to Nonnormal and Nonlinear State-Space Modeling, *Journal of the American Statistical Association*, 87, 493-500.

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

Carter, C.K., and R. Kohn, 1994, On Gibbs Sampling for State Space Models, *Biometrika*, 81, 541-553.

Carter, C.K., and R. Kohn, 1994, Markov chain Monte Carlo in conditionally Gaussian state space models, *Biometrika* 83, 589-601.

Chen, R. and J. Liu, 2000, Mixture Kalman filters, *Journal of Royal Statistical Society Series B.* 62, 493-508.

Chopin, N., 2002, A sequential particle filter method for static models. *Biometrika,* 89, 539-552.

Chopin, N., 2005, Inference and model choice for time-ordered hidden Markov models, working paper, University of Bristol.

Del Moral, P., Doucet, A. and Jasra, A., 2006, Sequential Monte Carlo Samplers, *Journal of Royal Statistical Society*, B, 68, 411-436.

Doucet, A, Godsill, S and Andrieu, C., 2000. On sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and Computing, 10,* 197-208.

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 Tadic, V., 2003, Parameter Estimation in general state-space models using particle methods. *Annals of Inst. Stat. Math.* 55, no. 2, pp. 409-422, 2003.

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

Godsill, S.J., Doucet, A. and West, M. (2004). Monte carlo Smoothing for Nonlinear Time Series. *Journal of American Statistical Association*, 99, 156-168.

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

Gordon, N. and A.F.M. Smith (1993). Approximate Non-Gaussian Bayesian Estimation and modal consistency. *Journal of Royal Statistical Society, B.*, 55, 913-918.

Hansen, L. and N.G. Polson (2006). Tractible Filtering. *Working paper*, University of Chicago.

Jacquier, E., N.G. Polson, and P. Rossi, 1994, Bayesian analysis of Stochastic Volatility Models, (with discussion). *Journal of Business and Economic Statistics* 12, 4.

Johannes, M., and Polson, N.G., 2006, Multivariate sequential parameter learning and state filtering, working paper, University of Chicago.

Johannes, M., Polson, N.G., and S. Yae, 2006, Robust sequential parameter learning and state filtering, working paper, University of Chicago.

Johannes, M., Polson, N.G. and Stroud, J.R., 2005, Sequential parameter estimation in stochastic volatility models with jumps. Working paper, University of Chicago.

Johannes, M., Polson, N.G. and Stroud, J.R., 2006, An interacting particle systems approach to sequential parameter and state learning, working paper, University of Chicago.

Johansen, A., A. Doucet and M. Davy, 2006, Maximum likelihood parameter estimation for latent variable models using sequential Monte Carlo, to appear *Proc. IEEE ICASSP*.

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

Kunsch, H., 2005, Recursive Monte carlo filters: Algorithms and theoretical analysis, *Annals of Statistics*, 33, 5, 1983-2021.

Liu, J. and Chen, R., 1995, Blind deconvolution via sequential imputations, *Journal of American Statistical Association.*, 89, 278-288.

Liu, J. and Chen, R., 1998, Sequential Monte Carlo Methods for Dynamical Systems. *Journal of the American Statistical Association*, 93, 1032-1044.

Liu, J. and M. West, 2001, Combined parameter and state estimation in simulation -based filtering, in *Sequential Monte Carlo Methods* in Practice, A. Doucet, N. de Freitas, and N. Gordon, Eds. New York: SpringerVerlag, 197-217.

Masreliez, C.J., 1975, Approximate non-Gaussian filtering with linear state and observation relations, *IEEE Transactions on Automatic Control 20,* 107-110.

Masreliez, C.J. and R.D. Martin, 1977, Robust Bayesian Estimation for the linear model and robustifying the Kalman filter, *IEEE Transactions on Automatic Control* 22*,* 361-371.

Meinhold, R.J. and Singpurwalla, N.D. (1987). Robustification of Kalman Filter Models. *Journal of American Statistical Association,* 84, 479-486.

Omori, Y., S. Chib, N. Shephard, and J. Nakajima, 2006, Stochastic Volatility with Leverage: Fast and Efficient Likelihood Inference, forthcoming *Journal of Econometrics*.

Pitt, M., and N. Shephard, 1999, Filtering via simulation: auxiliary particle fillters, *Journal of the American Statistical Association* 94, 590-599.

Polson, N.G, J. Stroud, and P. Mueller, 2006, Practical filtering with Sequential Parameter Learning. *Working paper*, University of Chicago.

Shephard, N. 1994, Partial non-Gaussian time series models, *Biometrika* 81, 115-31.

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

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

Stroud, Jonathan, Nicholas Polson and Peter Müller, (2004), Practical Filtering for Stochastic Volatility Models. In *State Space and Unobserved Components Models* (eds Harvey, A. et al), 236-247. Oxford University Press.

West, M. (1981) Robust Sequential Approximate Bayesian Estimation. *Journal of Royal Statistical Society, B.*, 43(2), 157-166.

West, M. and J. Harrison, 1997, Bayesian Forecasting and dynamic models, New York, Springer-Verlag.

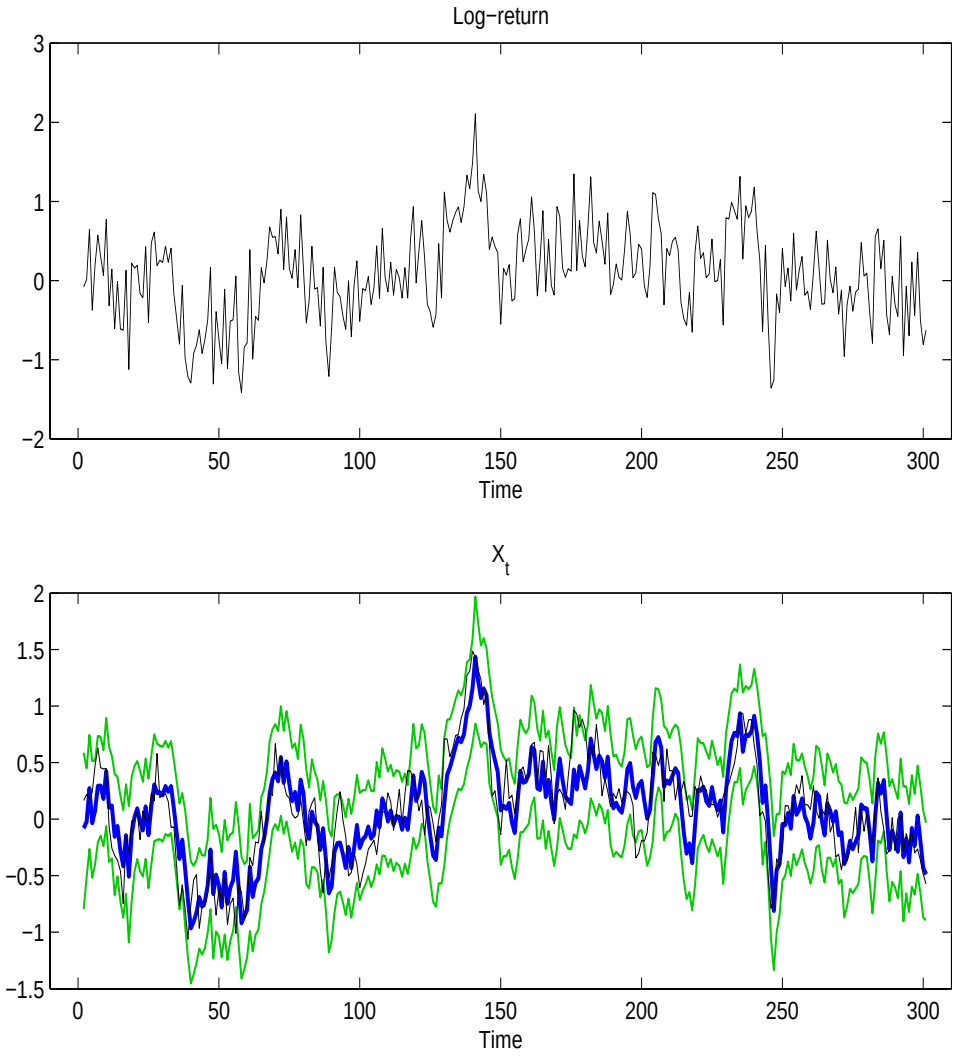

Figure 1: The top panel plots the observed time series, y_{t+1}, simulated from the t-distributed AR(1) model. The second panel plots the true simulated x_t series (thick line) as well as the $(5, 50, 95)$ posterior quantiles of $p\left(x_t|y^t\right)$.

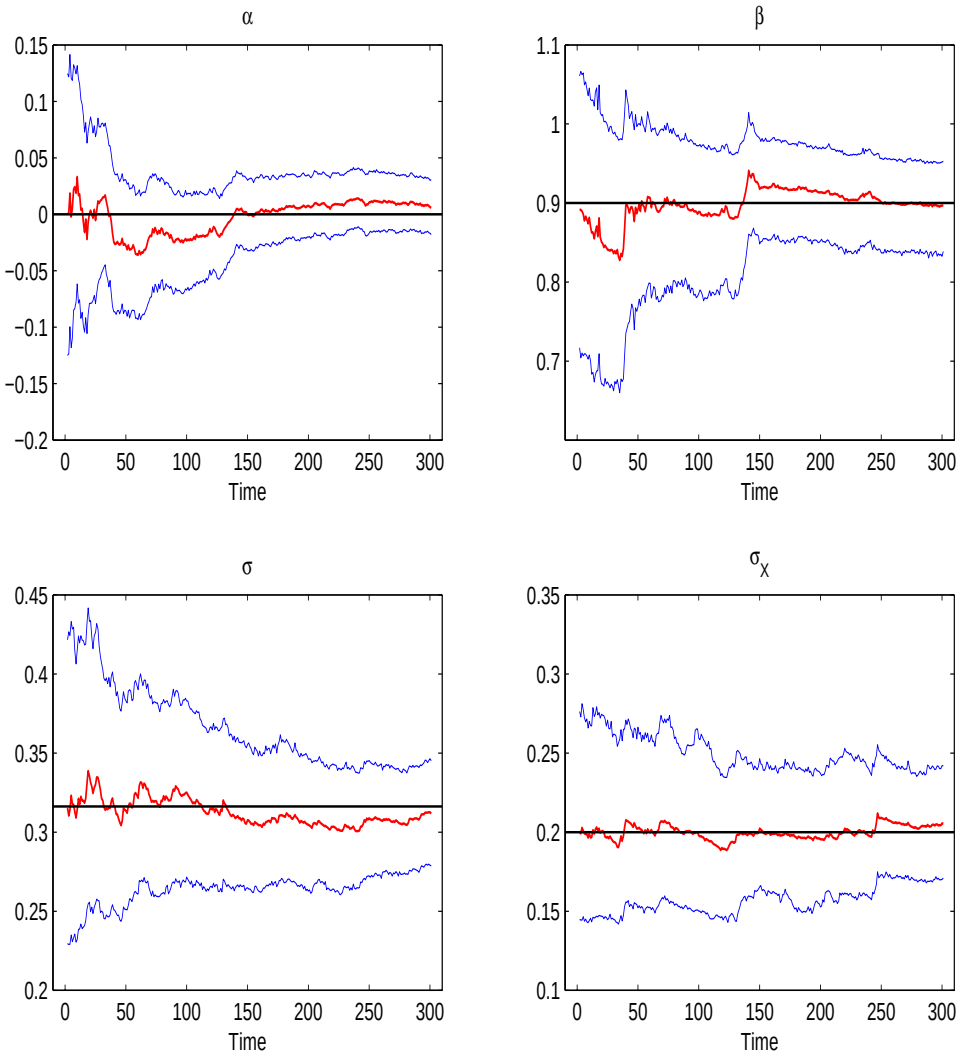

Figure 2: This figure displays sequential summarizes of the parameter posterior, $p(\theta|y^t)$. Each panel plots the $(5, 50, 95)$ posterior quantiles for the given parameter and also provides the true parameters used in simulation denoted by the horizontal line.

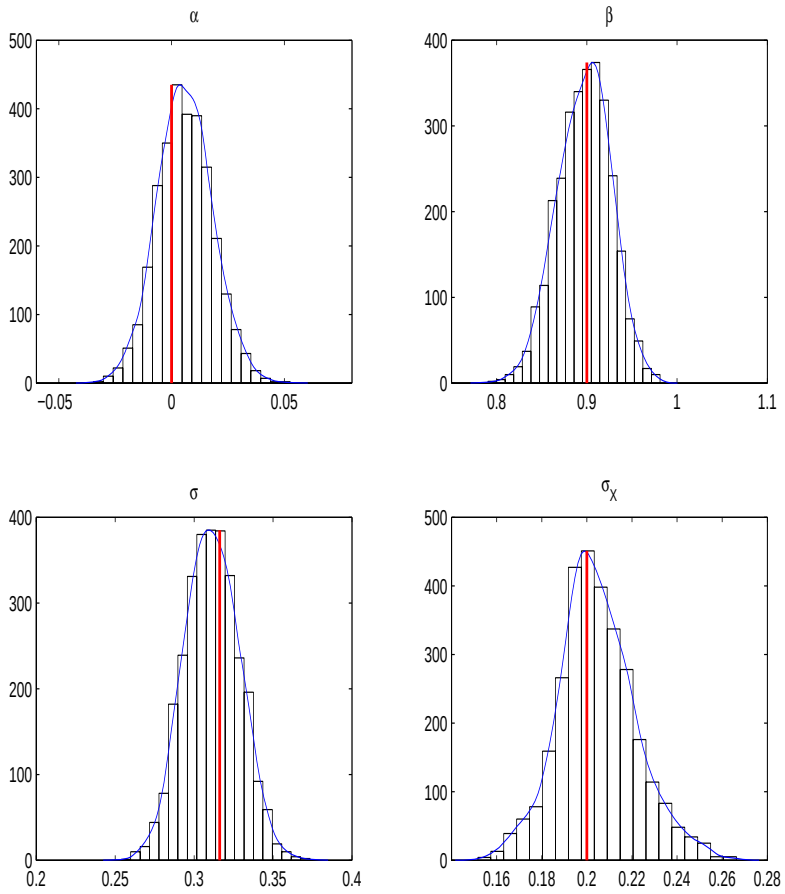

Figure 3: This figure summarizes the posterior distribution of the parameters at time $T = 300$. Each panel provides a histogram of the posterior, a smoothed estimate of the posterio, and the true parameter value that is denoted by a solid vertical line.

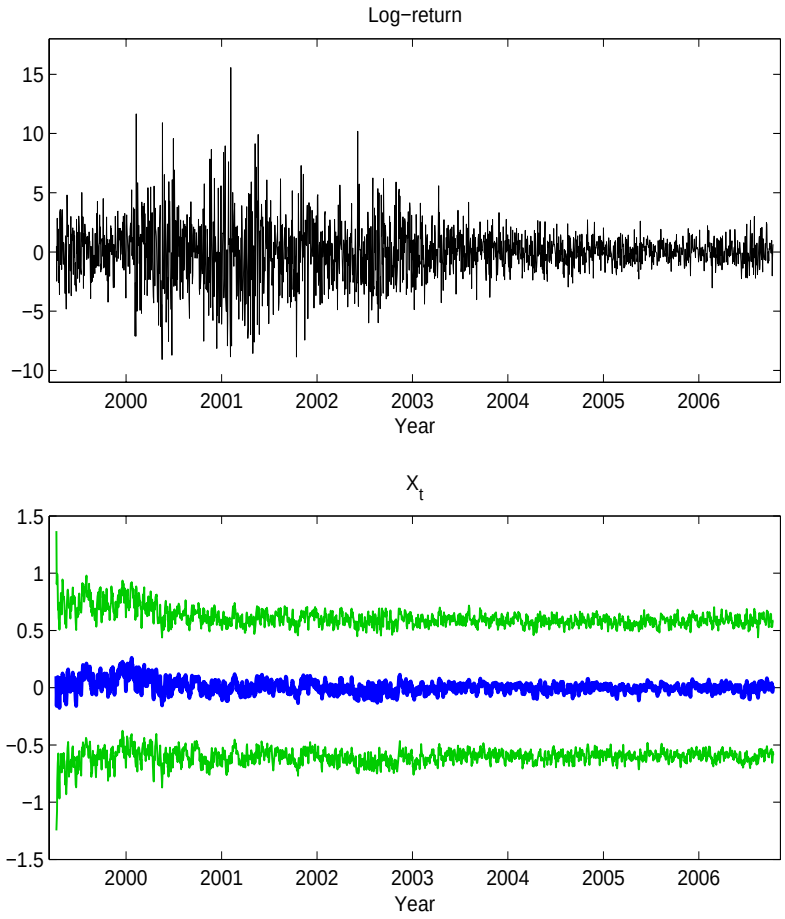

Figure 4: The top panel plots the observed time series, y_{t+1}, simulated from an autoregressive model with $t-$errors. The second panel plots the true simulated x_t series (thick line) as well as the $(5, 50, 95)$ posterior quantiles of $p\left(x_t|y^t\right)$.

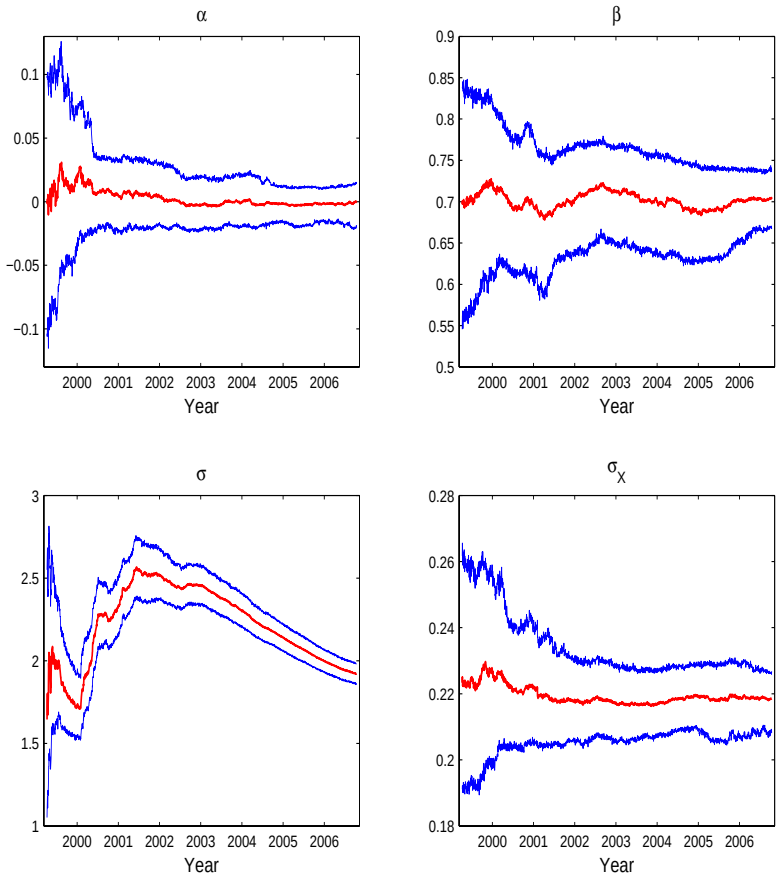

Figure 5: This figure displays sequential summarizes of the parameter posterior, $p(\theta|y^t)$. Each panel plots the $(5, 50, 95)$ posterior quantiles for the given parameter.

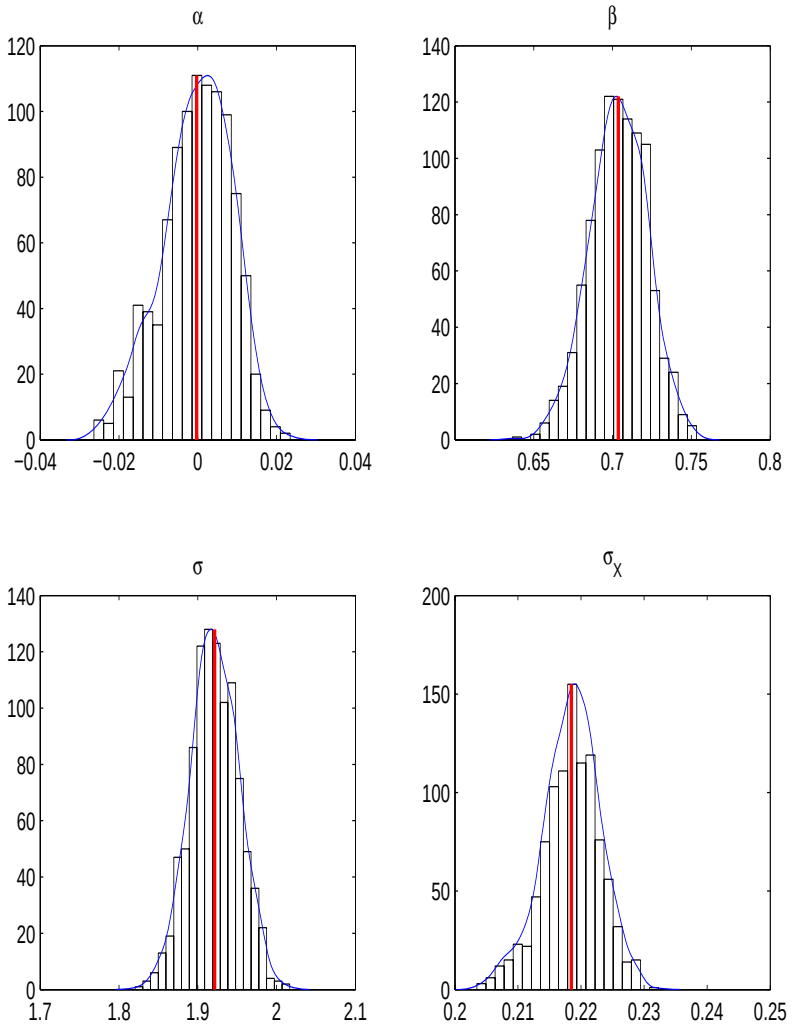

Figure 6: This figure summarizes the posterior distribution of the parameters at time $T = 1963$. Each panel provides a histogram of the posterior, a smoothed estimate of the posterior, and the posterior mean is indicated by a solid vertical line.

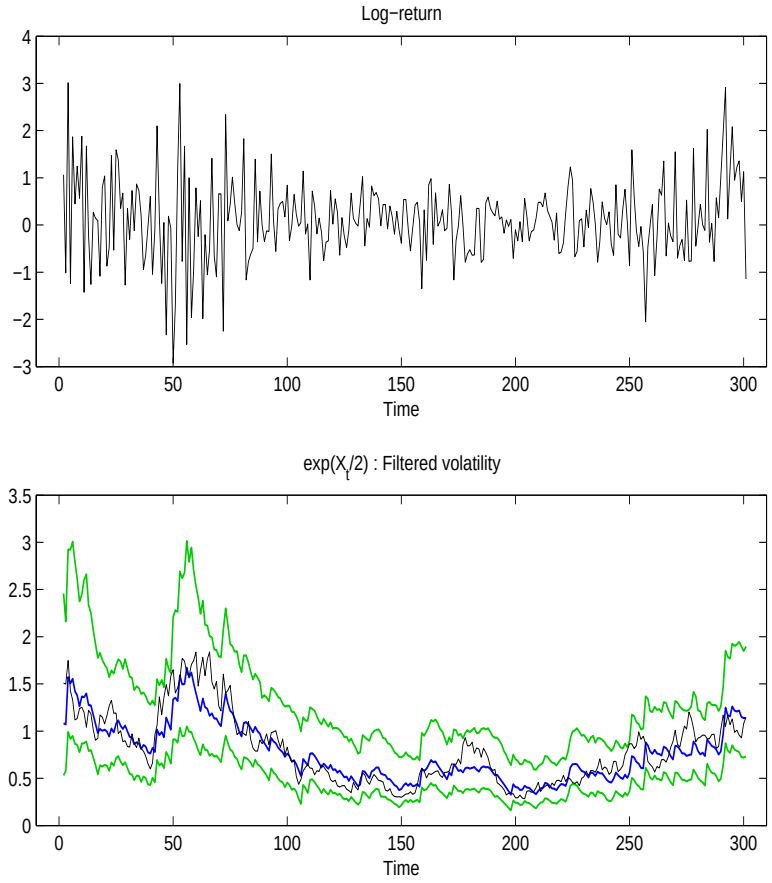

Figure 7: The top panel plots the observed time series, y_{t+1}, simulated from the stochastic volatility model. The second panel plots the true simulated x_t series (thick line) as well as the $(5, 50, 95)$ posterior quantiles of $p\left(x_t|y^t\right)$.

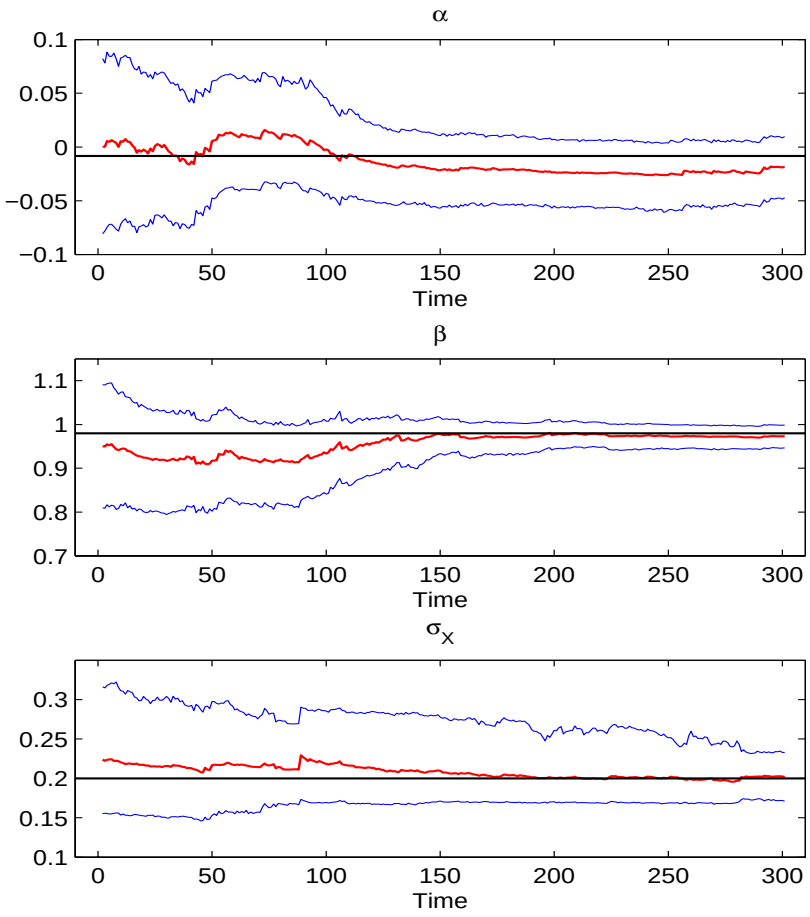

Figure 8: This figure displays sequential summarizes of the parameter posterior, $p(\theta|y^t)$. Each panel plots the $(5, 50, 95)$ posterior quantiles for the given parameter and also provides the true parameters used in simulation denoted by the horizontal line.

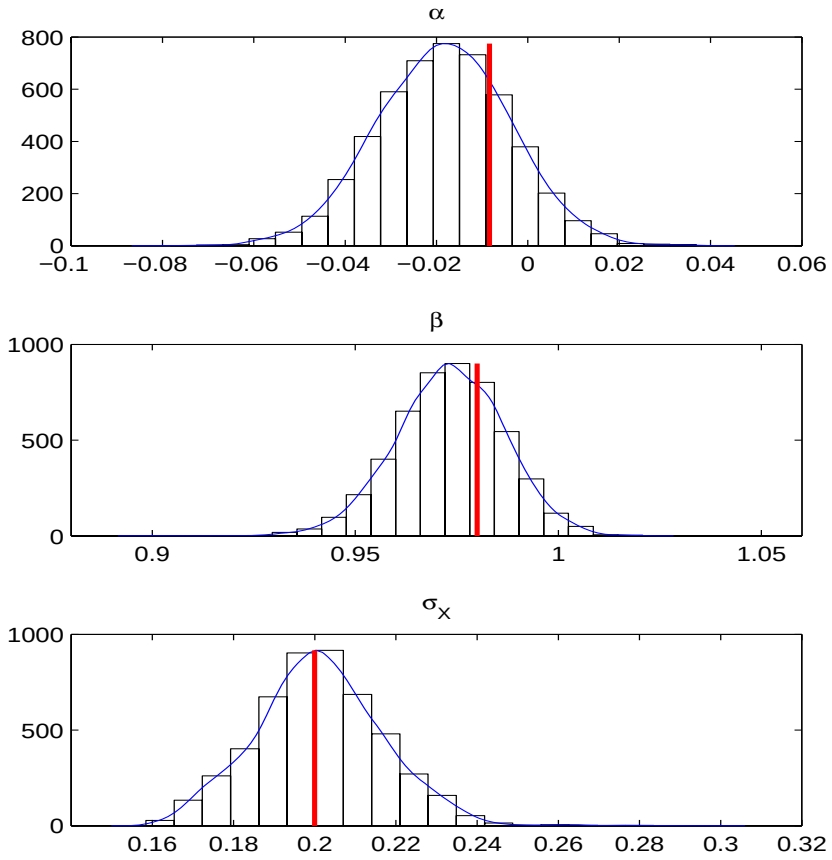

Figure 9: This figure summarizes the posterior distribution of the parameters at time $T = 300$. Each panel provides a histogram of the posterior, a smoothed estimate of the posterio, and the true parameter value that is denoted by a solid vertical line.

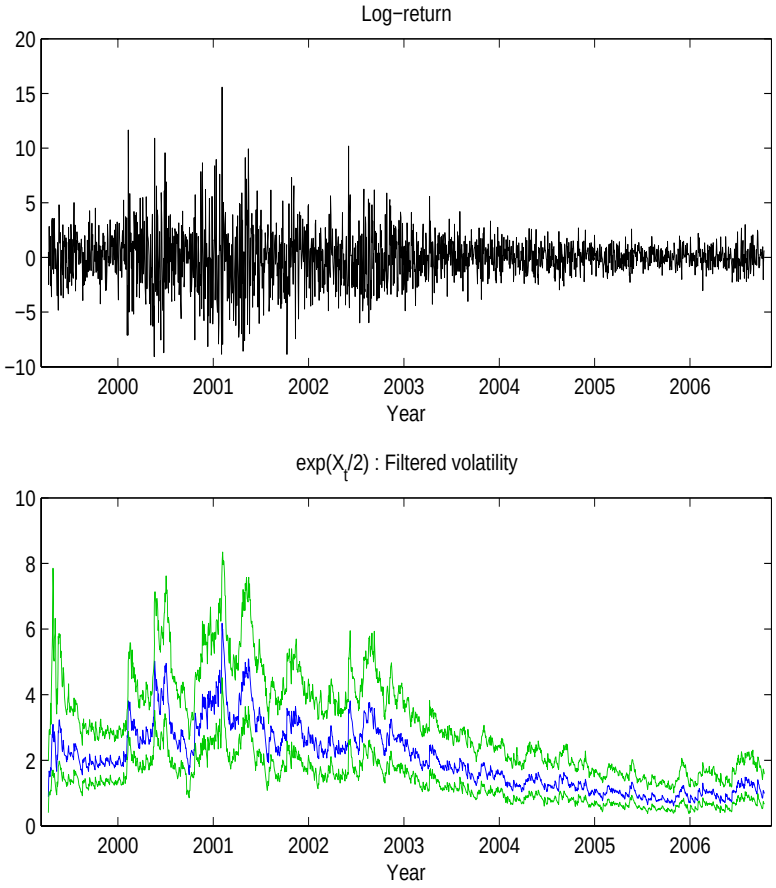

Figure 10: The top panel plots the observed time series, y_{t+1}, of Nasdaq 100 stock returns. The second panel plots the true simulated x_t series (thick line) as well as the $(5, 50, 95)$ posterior quantiles of $p\left(x_t|y^t\right)$.

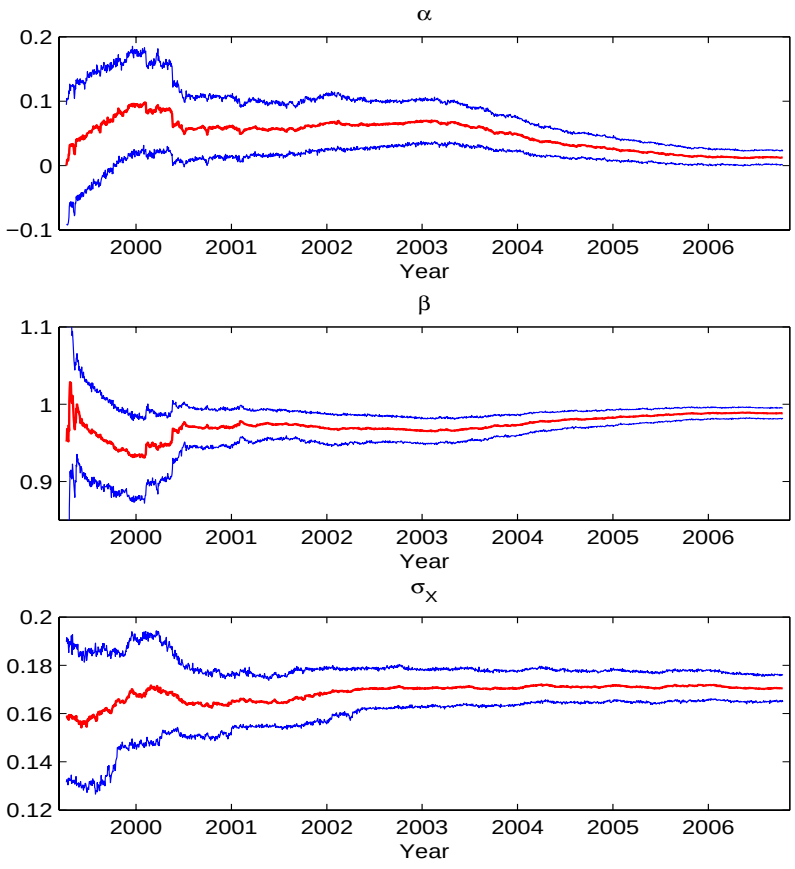

Figure 11: This figure displays sequential summarizes of the parameter posterior, $p(\theta|y^t)$. Each panel plots the $(5, 50, 95)$ posterior quantiles for the given parameter.

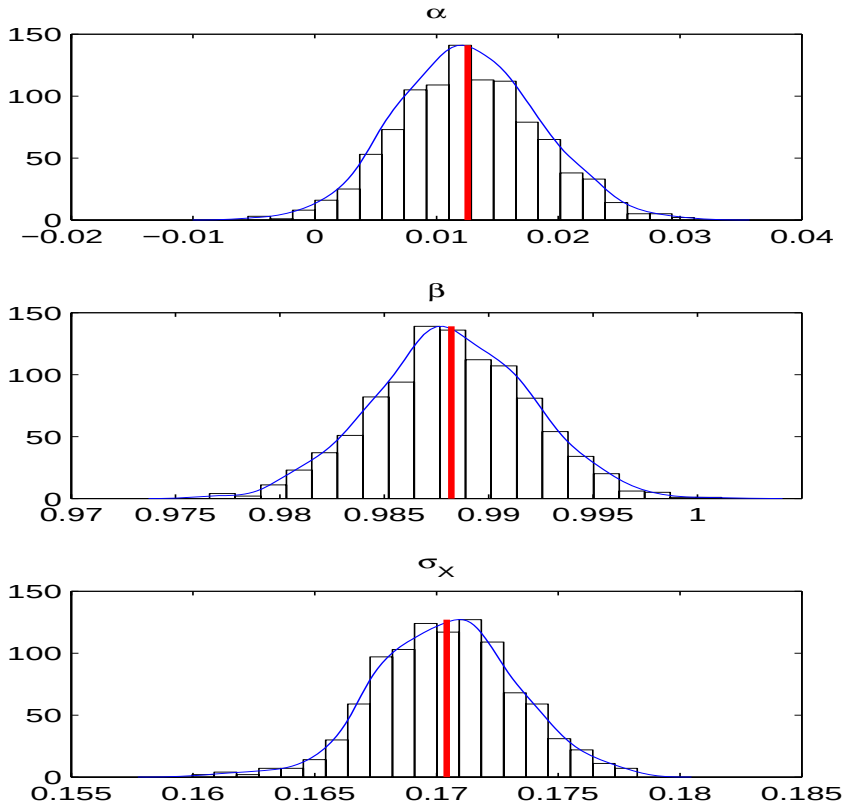

Figure 12: This figure summarizes the posterior distribution of the parameters at time $T = 1963$. Each panel provides a histogram of the posterior, a smoothed estimate of the posterior, and the posterior mean is indicated by a solid vertical line.