Sequential Monte Carlo samplers for Bayesian DSGE models

Drew Creal*

First version: February 8, 2007 Current version: March 27, 2007

Abstract

Dynamic stochastic general equilibrium models have become a popular tool in economics for both forecasting and macroeconomic policy analysis. A Bayesian approach typically uses Markov chain Monte Carlo (MCMC) algorithms to estimate the models as importance sampling (IS) algorithms have a difficult time in high-dimensional state spaces. In this paper, I develop improved IS algorithms for DSGE models using recent advances in Monte Carlo methods known as sequential Monte Carlo samplers. Sequential Monte Carlo samplers are a generalization of particle filtering designed for full simulation of all parameters from the posterior. I build two separate algorithms; one for batch estimation and a second which performs sequential Bayesian estimation. The algorithms are compared to MCMC and IS on real and simulated data. The batch algorithm is substantially more reliable than standard IS. The sequential Bayesian algorithm indicates that the posterior density changes substantially over time.

*PRELIMINARY. Comments are welcome. Author's contact: dcreal@feweb.vu.nl, Department of Econometrics and Operations Research, Vrije Universitiet, De Boelelaan 1105, 1081 HV Amsterdam, The Netherlands. I gratefully acknowledge the financial support of the Grover and Creta Ensley Fellowship while at the University of Washington. I would also like to thank Siem Jan Koopman and Eric Zivot for helpful discussions. All remaining errors are my own.

1 Introduction

Economics has recently seen an explosion in the popularity of estimating dynamic stochastic general equilibrium (DSGE) models. The reason for the increase is twofold. New frequentist and Bayesian estimators are allowing more complex models to be estimated. More importantly, the models are improving to the point where they are trusted to answer deeper structural questions for policy analysis. Recent work has even demonstrated their ability to compete with vector autoregressions in forecasting. Smets and Wouters (2005), Christiano *et al.* (2006), and Del Negro *et al.* (2007) are a few important contributions along these lines.

Lately, An and Schorfheide (2006) provided a review of Bayesian methods for estimating and comparing DSGE models, which included both Markov chain Monte Carlo (MCMC) and importance sampling (IS) algorithms. MCMC algorithms are definitely the preferred tool in the literature as IS algorithms do not work as effectively in higher dimensional spaces. My contribution in this paper is to expand the Bayesian toolkit from An and Schorfheide (2006) by building improved IS algorithms based upon a recently developed methodology known as sequential Monte Carlo (SMC) samplers. I design several algorithms and I compare their performance with MCMC and IS on both real and simulated data. One of the algorithms performs sequential Bayesian estimation, which is entirely new to the DSGE literature.

SMC samplers are a generalization of particle filtering to full simulation of all unknowns from a posterior distribution. Particle filters are algorithms originally designed for sequential state estimation or optimal filtering in nonlinear, non-Gaussian state space models. They were proposed by Gordon *et al.* (1993) and further developments of this methodology can be found in books by Doucet *et al.* (2001), Ristic *et al.* (2004), and Cappé *et al.* (2005). Particle filters were introduced into the econometrics literature by Kim *et al.* (1998) to study the latent volatility of asset prices and into the DSGE literature by Fernández-Villaverde and Rubio-Ramírez (2005 and 2007).

A recent contribution by Del Moral *et al.* (2006) (hereafter DDJ) has demonstrated how SMC algorithms can be applied more widely than originally thought, including the ability to estimate static parameters. Additional references in this field include Gilks and Berzuini (2001), Chopin (2002), Liang (2002), and Cappé *et al.* (2004). SMC samplers are an alternative to MCMC for posterior simulation, although in reality they will often incorporate Metropolis-Hastings within them. SMC samplers do not rely on the same convergence properties as MCMC for their validity.

And, they improve on some of the limitations of regular IS (discussed below) when it is applied to higher dimensional spaces.

I build two different SMC sampling algorithms in this paper. The first is based on the simulated tempering approach from DDJ (2006), which estimates parameters using all of the data collected in a batch. The SMC sampler I design requires little more coding than an MCMC algorithm. In addition, the algorithm has only a few tuning parameters. The second algorithm performs sequential Bayesian estimation in the spirit of Chopin (2002). That is, the algorithm adds an additional observation at each iteration and estimates the evolving posterior distribution through time. Paths of the parameters can then be considered as filtered estimates providing additional information on time-variation of parameters and the stability of the model. This type of algorithm is well beyond the capability of an MCMC algorithm.

In order to focus on the methodology, I borrow a small New Keynesian model from Rabanal and Rubio-Ramírez (2005) (hereafter RR). Based on real and simulated data, the SMC sampler using simulated tempering works well and may be preferable to MCMC in difficult settings. The importance weights at the end of this sampler are almost perfectly balanced, indicating that the draws are almost exactly from the posterior. Alternatively, the method can be used to establish the reliability of MCMC output. I also compare it to an IS algorithm, built using best practices. The IS algorithm does not work well. Using test statistics and diagnostics developed in Koopman and Shephard (2004), I show that the variance of the importance weights is likely not to exist. Estimates of the posterior from the sequential algorithm are close to the batch estimates. The sequential algorithm indicates that the posterior distribution of this DSGE model varies significantly over time.

It is important to recognize that the use of SMC methods in this paper is distinct from how other authors use them in the DSGE literature. Recent articles on likelihood-based inference for DSGE models by Fernández-Villaverde and Rubio-Ramírez (2005 and 2007) and An and Schorfheide (2006) have emphasized the importance of computing higher order nonlinear approximations. Particle filters are then used to approximate (and sometimes maximize) the log-likelihood function. Higher order approximations generally improve the identification of some parameters and in some cases allow parameters to be identified that are not under a first-order approximation. Although I consider only first-order approximations in this paper, the methods described here can be used for nonlinear DSGE models as these will most likely become the standard over time.

2 A basic New Keynesian model[1]

The model I consider in this paper is the EHL model from RR (2005). The EHL model is a standard New Keynesian model based on theoretical work by Erceg *et al.* (2000), who combined staggered wage contracts with sticky prices using the mechanism described in Calvo (1983). As a derivation of the model is described in RR (2005), their appendix, and their references, I highlight only the dynamic equations that describe equilibrium in order to provide an economic interpretation of the parameters.[2] These equations are the log-linear approximation of the first-order conditions and exogenous driving variables around the steady state. All variables are in log-deviations from their steady state values.

The model includes an Euler equation relating output growth to the real interest rate

$$y_t = E_t y_{t+1} - \sigma \left(r_t - E_t \Delta p_{t+1} + E_t g_{t+1} - g_t \right) \tag{1}$$

where y_t denotes output, r_t is the nominal interest rate, g_t is a shock to preferences, p_t is the price level, and σ is the elasticity of intertemporal substitution.

The production and real marginal cost of production functions are given by

$$y_t = a_t + (1 - \delta) n_t \qquad mc_t = w_t - p_t + n_t - y_t \tag{2}$$

where a_t is a technology shock, n_t are the number of hours worked, mc_t is real marginal cost, w_t is the nominal wage, and δ is the capital share of output. The marginal rate of substitution between consumption and hours worked is described by

$$mrs_t = \frac{1}{\sigma} y_t + \gamma n_t - g_t \tag{3}$$

where γ denotes the inverse elasticity of labor supply with respect to real wages.

Monetary authorities are assumed to follow a Taylor rule with interest rate smoothing

$$r_t = \rho_r r_{t-1} + (1 - \rho_r) \left[\gamma_\pi \Delta p_t + \gamma_y y_t \right] + z_t \tag{4}$$

[1] Please note that this section closely follows section 2 of RR (2005).

[2] The model is also well detailed in an excellent set of lecture notes (with code) on solving and estimating DSGE models by Fernández-Villaverde and Rubio-Ramírez.

The parameters γ_π and γ_y measure the monetary authority's responses to deviations of inflation and output from their equilibrium values. The degree of interest rate smoothing is given by ρ_r. The Taylor rule also includes an exogenous monetary shock z_t. Altogether, the exogenous shocks are given by

$$\begin{aligned} a_t &= \rho_a a_{t-1} + \varepsilon_t^a && (5) \\ g_t &= \rho_g g_{t-1} + \varepsilon_t^g && (6) \\ z_t &= \varepsilon_t^z && (7) \\ \lambda_t &= \varepsilon_t^\lambda && (8) \end{aligned}$$

where the ε_t^i are assumed to be *i.i.d.* normally distributed with variances σ_i^2.

The representative agent is assumed to follow Calvo (1983) price-setting which provides the New Keynesian Phillip's curve

$$\Delta p_t = \beta E_t \Delta p_{t+1} + \kappa_p \left(mc_t + \lambda_t \right) \tag{9}$$

This describes how prices are set by firms based upon their real marginal cost mc_t, expected future inflation, and the price mark-up shock λ_t. The parameter β measures the agent's rate of time preference. RR (2005) show that the parameter κ_p is equal to $\kappa_p = (1-\delta)(1-\theta_p\beta)(1-\theta_p) / \{\theta_p [1+\delta(\bar{\varepsilon}-1)]\}$ where $\bar{\varepsilon}$ is the steady-state value of ε.

The EHL model assumes that wages are also set by a Calvo (1983) mechanism resulting in the following process for their dynamics

$$\Delta w_t = \beta E_t \Delta w_{t+1} + \kappa_w \left(mrs_t - (w_t - p_t) \right) \tag{10}$$

where again RR (2005) show that the parameter κ_w is equal to $\kappa_w = (1-\theta_w)(1-\beta\theta_p) / \{\theta_w [1+\phi\gamma]\}$.

The model is finally closed by RR (2005) assuming that real wage growth, nominal wage growth, and inflation are related by the identity

$$w_t - p_t = w_{t-1} - p_{t-1} + \Delta w_t - \Delta p_t \tag{11}$$

In addition, note that the real wage must equal the marginal rate of substitution in equilibrium

$$w_t - p_t = mrs_t \tag{12}$$

The structural parameters of the model are now collected as

$$\Theta = \{\sigma, \theta_w, \theta_p, \beta, \phi, \gamma_\pi, \gamma_y, \rho_a, \rho_g, \rho_r, \delta, \bar{\varepsilon}, \sigma_a, \sigma_z, \sigma_\lambda, \sigma_g\}\,.$$

2.1 Data and prior distribution

My dataset consists of four series: real output, real wages, price inflation, and interest rates for the U.S. at a quarterly frequency from 1960:01 to 2006:03. This results in 187 observations. I chose the same series as RR (2005), which consists of output for the nonfarm business sector, compensation per hour for the nonfarm business sector, the corresponding implicit price deflator, and the federal funds rate respectively. All the series were obtained from the FRED database at the Federal Reserve Bank of St. Louis. The data were first demeaned and the real wage and output were detrended using a simple structural time series model described in appendix B.

The prior distributions are chosen to be the same as in RR (2005) in order to make comparison to their results straightforward. The priors are listed in Table 1 next to the posterior estimates. Practical considerations require that several parameters be set to constants as they are not identifiable from the likelihood or alternatively there is a tradeoff in identifiability with another parameter. Accordingly, I set these parameters ($\beta = 0.99$, $\delta = 0.36$, $\phi = 6$, $\bar{\varepsilon} = 6$) to the same values as RR (2005).

2.2 Bayesian estimation

Given equations (1-12) that describe equilibrium, the model may be solved for the reduced form using any of several popular solution techniques. This approximation may then be placed in state space form and the Kalman filter used to compute the quasi-likelihood; see, e.g. Durbin and Koopman (2001).

The most popular method for estimating Bayesian DSGE models are MCMC algorithms. Independence M-H and random-walk Metropolis algorithms are used to propose moves of all the structural parameters at once through the state space. Although the model can be placed in linear, Gaussian state space form, Gibbs samplers are precluded because solution methods that map the

structural parameters to the reduced form parameters are highly nonlinear and cannot be inverted.

The main concern with these algorithms is the inability to prove convergence of the Markov chain to the stationary distribution and to determine the reliability of ergodic averages calculated from the chain. M-H algorithms for DSGE models often mix poorly as draws may have potentially extreme correlation. Correlation can cause MCMC algorithms to sample a disproportionate number of draws in some regions of the posterior relative to the true posterior probability. Researchers cope with this problem in practice by skipping a large number of iterations from the algorithm when they compute their estimates. Draws far enough apart are hopefully close to being independent and the Markov chain will have had the opportunity to move to another area of the support. Nevertheless, it can be difficult to determine when this strategy is completely effective.

Consequently, it is important to explore alternative approaches so that applied researchers have another method for comparison. The recent success of particle filtering algorithms in nonlinear, non-Gaussian state space models has created a resurgence of interest in IS as a means of creating posterior simulators with the ability to adapt over time. Rather than viewing MCMC and IS as distinct from one another, researchers working in Monte Carlo methods recognized that potentially better algorithms could be built by combining both principles. Leading references in this field include Gilks and Berzuini (2001), Chopin (2002), Liang (2002), Cappé *et al.* (2004) and DDJ (2006).

Each paper describes how to draw from a posterior density using sequential proposals that gradually evolve and adapt to the posterior distribution based upon their past performance. In a seminal contribution, DDJ (2006) demonstrated how many of the existing algorithms in the literature fit within a common framework. Prior experience with MCMC does not get thrown away as SMC sampling algorithms can be created from existing MCMC algorithms. As these methods are fundamentally based on IS, the algorithms avoid some of the problems that may occur with MCMC.

3 Sequential Monte Carlo Samplers

In this section, I review SMC samplers as developed by DDJ (2006). Del Moral *et al.* (2006b) and Jasra *et al.* (2005b) are slightly easier introductions to SMC samplers than the original article. Theoretical convergence results based on the theory of Feynman-Kac flows can be found in Del

Moral (2004).

3.1 Importance Sampling

The posterior density of the DSGE model $p(\Theta|y_{1:T})$ can be considered more generically as a target density $\pi(x)$ from which random samples cannot be directly taken. It is assumed that $\pi(x)$ can be evaluated pointwise up to a normalizing constant. Denote the target density by

$$\pi(x) = \frac{\gamma(x)}{Z}$$

where $\gamma(x)$ can be calculated for any realization x of X and Z is a normalizing constant that typically includes integrals that cannot be solved analytically.

IS works by drawing N samples $\left\{X^{(i)}\right\}_{i=1}^{N}$ from a different density $\eta(x)$ called the importance density and then reweighting each draw. The reweighted draws $\left\{W^{(i)}, X^{(i)}\right\}_{i=1}^{N}$ known as *particles* are i.i.d. samples from $\pi(x)$. In the context of a DSGE model, each particle will contain values for the parameters $X^{(i)} = \left\{\Theta^{(i)}\right\}$.[3] The unnormalized importance weights $\left\{w^{(i)}\right\}_{i=1}^{N}$ are calculated through

$$w^{(i)}(x) = \frac{\gamma^{(i)}(x)}{\eta^{(i)}(x)} \tag{13}$$

The definition of the importance weights follows from the classic identities

$$\begin{aligned} \mathbb{E}_{\pi}(\varphi) &= Z^{-1}\int \varphi(x)\, w(x)\, \eta(x)\, \mathrm{d}x \\ Z &= \int w(x)\, \eta(x)\, \mathrm{d}x \end{aligned} \tag{14}$$

It is known from Geweke (1989) that the empirical distribution of the weighted particles converges asymptotically to $\pi(x)$ as $N \to \infty$. Meaning that for any $\pi(x)$ -integrable function $\varphi(x)$, one can sample from $\eta(x)$ and calculate any function of interest with respect to $\pi(x)$ using the N particles and their weights

[3] When a DSGE model includes latent variables such as regime switching variables, these can be included in X.

$$\sum_{i=1}^{N} W^{(i)} \varphi\left(X^{(i)}\right) \longrightarrow^{a.s.} \mathbb{E}_{\pi}(\varphi) = \int \varphi(x)\, \pi(x)\, \mathrm{d}x \tag{15}$$

Historically, IS has not been successful at estimating integrals with respect to highly complex densities or densities with high-dimensional supports. IS is only valid if the importance density assigns positive probability everywhere the target density does. This is typically impossible to guarantee in high dimensions. To guard against this problem, researchers often choose an importance density with extremely wide tails. This may cause the importance density to fit poorly elsewhere.

The fit of the importance density to the target determines the statistical efficiency of the method. In higher dimensional spaces, it is difficult to find an importance density that can adequately approximate the target. This is particularly true in the tails of the distribution, although it can also be true when the target is multi-modal. When draws are taken from an importance density that is not a good approximation of the target, IS fails because only a few draws will contribute to the estimator. The approximation of the target density is effectively determined by a few particles with large weights while other particles' weights are insignificant. Consequently, the variance of the importance weights grows and the approximation collapses. In extreme cases, the variance of the importance weights may be infinite and the central limit theorem describing how the IS estimator converges may no longer apply. Recently, Koopman and Shephard (2004) have proposed a frequentist procedure for testing this situation, which I employ later in the paper.

3.2 Sequential Importance Sampling

Instead of trying to approximate $\pi(x)$ immediately, the strategy behind sequential importance sampling (SIS) is to slowly build towards the target density by first sampling from a simple density and then gradually getting more complex. The algorithm starts by approximating a simple density $\pi_1(x)$ and then moves through a sequence of densities $\pi_1(x) < \pi_2(x) .... < \pi_n(x) .... < \pi_p(x)$, where n is an index that denotes the iteration number in the sequence. The goal of the overall algorithm is to have at least one of the densities in the sequence (possibly more) equal to a density in which the researcher is interested. For example, one possible algorithm might have the last density equal to the target of interest $\pi_p(x) = \pi(x)$, i.e. the posterior density of the DSGE model. In this case, the sequence of early densities exist to help build a good importance density for the final iteration.

Given a set of particles $\left\{x_{n-1}^{(i)}\right\}_{i=1}^{N}$ distributed as $\eta_{n-1}(x)$ at iteration $n-1$, each particle is moved according to a forward Markov kernel $X_n^{(i)}|x_{n-1}^{(i)} \sim K_n\left(x_{n-1}^{(i)}, \cdot\right)$. Before being reweighted, the new set of particles $\left\{x_n^{(i)}\right\}_{i=1}^{N}$ are approximately distributed as

$$\eta_n(x_n) = \int \eta_{n-1}(x_{n-1}) K_n(x_{n-1}, x_n) \, \mathrm{d}x_{n-1} \tag{16}$$

Even though the particles were not drawn directly from $\eta_n(x_n)$, drawing them from $\eta_{n-1}(x_{n-1})$ and moving them with the forward Markov kernel means that they have distribution (16). New importance weights can theoretically be computed as the ratio of an unnormalized target density over the new importance density given in (16).

In most settings where $K_n(x_{n-1}, x_n)$ is chosen wisely for the problem at hand, importance weights after the first iteration will be impossible to calculate because $\eta_n(x_n)$ will contain an insoluble integral precluding analytical calculation. Unlike regular importance sampling where a user chooses the importance density explicity, in this framework the user typically does not know what the resulting importance density is. Rather, the user chooses the Markov kernels and the importance density is determined endogenously. One of the major contributions of DDJ (2006) was to describe how algorithms could be built to surmount the problem of calculating the importance weights when they are unknown. I review this in section 3.3.

A weakness of the SIS strategy outlined above concerns the impact on the importance weights over successive iterations. The variance of the importance weights will tend to grow as some particles are not representative of the next density in the sequence and receive little probability mass. This phenomenon is known as degeneracy. A breakthrough came in SIS when Gordon *et al.* (1993) introduced a resampling stage within their original particle filtering algorithm designed to eliminate particles with small importance weights and replicate particles with larger weights. At the end of each iteration, particles are resampled with probability according to their importance weights and the weights are then set equal, $W_n^{(i)} = \frac{1}{N}$. Resampling shuffles particles along the support of the density to areas of higher probability allowing better particles to be simulated at the next iteration. Adding a resampling step to the SIS algorithm results in the SISR algorithm, which does not eliminate degeneracy but does severely decrease its effects.

It is important to recognize, however, that it is actually not optimal to resample at each iteration even though it is beneficial overall. Resampling introduces additional Monte Carlo variation within

the algorithm. Liu and Chen (1998) introduced a statistic called the effective sample size (ESS) to stochastically determine when to resample.[4] The effective sample size

$$ESS = \frac{1}{\sum_{i=1}^{N}\left(W_n^{(i)}\right)^2} \tag{17}$$

is a number between one and N that measures how many particles are contributing to the estimator. When this statistic drops below a threshold, say 0.5-0.75 percent of the particles, then resampling should be run.

3.3 Sequential Monte Carlo samplers

Particle filters are a special case of an SMC sampler and are designed for state estimation or *optimal filtering* in nonlinear, non-Gaussian state space models where the goal is to estimate the conditional density of a state variable $p\left(x_t|y_{1:t}\right)$ sequentially. In this setting, the iteration number n of the sequence $\{\pi_j\left(x\right)\}_{j=1}^{n}$ represents calendar time and each density within the sequence is inherently a target of interest. SMC samplers are then performing sequential Bayesian estimation as an additional observation gets added at each iteration. This is a feature of SMC that is not shared by MCMC as a separate run of an MCMC algorithm would be required with each new observation.

In other cases, the initial densities $\pi_1\left(x\right) < \pi_2\left(x\right) .... < \pi_{p-1}\left(x\right)$ within an SMC sampler exist purely to help build a good importance density for the final target $\pi_p\left(x\right) = \pi\left(x\right)$. SMC samplers are then an alternative to MCMC for estimating static parameters in one batch. Del Moral *et al.* (2006b) estimate two important economic models as applications; a binary probit regression model and the stochastic volatility model of Barndorff-Nielsen and Shephard (2001).

An SMC sampler begins by drawing N particles $\left\{x_1^{(i)}\right\}_{i=1}^{N}$ from an initial importance density $\eta_1\left(x\right)$ and reweighting the particles using standard importance weights

$$\omega_1\left(x_1\right) = \frac{\gamma_1\left(x_1\right)}{\eta_1\left(x_1\right)} \tag{18}$$

which can be computed explicitly for this iteration. Particles for the next iteration are sampled as in SIS from a forward Markov transition kernel $X_2^{(i)}|x_1^{(i)} \sim K_2\left(x_1^{(i)}, \cdot\right)$. DDJ (2006) solve the problem of having to evaluate the unknown importance density $\eta_n\left(x_n\right)$ to compute importance weights

[4] There exist other criterion to determine when to resample; see, e.g. Cappé *et al.* (2005) for details.

beyond the first iteration by introducing new artificial target densities, $\tilde{\pi}_n(x_{1:n})$. The artificial targets are not of interest in themselves but their introduction allows the importance weights to be computed. An artificial target must be defined up to a normalizing constant

$$\tilde{\pi}_n(x_{1:n}) = \frac{\tilde{\gamma}_n(x_{1:n})}{Z_n}$$

where the new target is intentionally designed to admit $\pi_n(x_n)$ as a marginal density. By sampling in a larger space, estimates of the marginal using the particles' locations and importance weights can be computed as a by-product.

DDJ (2006) provide a framework for choosing both the artificial target densities $\tilde{\pi}_n(x_{1:n})$ as well as the forward Markov kernels. They suggest defining the artificial targets as a sequence of backward Markov kernels $L_n(x_{n+1}, x_n)$ which can be written as

$$\tilde{\gamma}_n(x_{1:n}) = \gamma_n(x_n) \prod_{k=1}^{n-1} L_k(x_{k+1}, x_k)$$

Given a set of weighted particles $\left\{W_{n-1}^{(i)}, X_{1:n-1}^{(i)}\right\}_{i=1}^{N}$ that approximate the artificial target, the next artificial target can be approximated by sampling from the forward Markov kernel. Immediately after sampling, the particles' distribution is $\eta_n(x_{1:n})$. Reweighting changes the distribution of the particles from $\eta_n(x_{1:n})$ to approximately i.i.d. draws from $\tilde{\pi_n}(x_{1:n})$.

The unnormalized importance weights $\omega_n(\cdot)$ can be written recursively such that at each iteration one only calculates the incremental importance weights $\tilde{\omega}_n(\cdot,\cdot)$ given by

$$\begin{aligned} \tilde{\omega}_n(x_{n-1}, x_n) &= \frac{\gamma_n(x_n) L_{n-1}(x_n, x_{n-1})}{\gamma_{n-1}(x_{n-1}) K_n(x_{n-1}, x_n)} \\ \omega_n(x_{1:n}) &= \omega_{n-1}(x_{1:n-1}) \tilde{\omega}_n(x_{n-1}, x_n) \end{aligned} \tag{19}$$

How easy it is to calculate the incremental weights depends on how one designs the SMC sampler. These unnormalized weights then lead to normalized importance weights

$$W_n^{(i)} = \frac{\omega_n\left(X_{1:n}^{(i)}\right)}{\sum_{j=1}^{N} \omega_n\left(X_{1:n}^{(j)}\right)} \tag{20}$$

Once the normalized importance weights are calculated, estimates of a marginal target distribution can be calculated as

$$\pi_n^N \approx \sum_{i=1}^{N} W_n^{(i)} \delta_{X_n^{(i)}} \left(x_n^{(i)} - x_n \right)$$

where $\delta(\cdot)$ denotes the Dirac measure. The general SMC sampler of DDJ (2006) is provided as algorithm 1.

3.4 Choosing the forward and backward kernels

Choices for the forward and backward kernels are theoretically wide open. In reality though, they must be chosen carefully as they determine the success of the algorithm. Poorly chosen kernels will result in badly behaved algorithms. It is optimal to choose forward and backward Markov kernels $K_n(x_{n-1}, x_n)$ and $L_{n-1}(x_n, x_{n-1})$ such that the importance density $\eta_n(x_{1:n})$ is as close to that iteration's target $\tilde{\pi}_n(x_{1:n})$ as possible. This will keep the variance of the importance weights stable. It is also possible to use the past simulated particles to build new importance distributions that are adapted through time.

Using this variance as a criterion, DDJ (2006) formally establish what the optimal choice for both kernels would be. Not surprisingly, the optimal proposals require calculating $\eta_n(x_n)$ and other densities that we are using SMC to avoid. Instead, the authors present sub-optimal choices that approximate the optimal choice based on experience from the literature with MCMC and particle filtering.

Most applications of SMC will have each particle as a vector containing multiple elements; i.e., latent state variables and/or multiple parameters. Suppose the particle at iteration n is made of 2 components or blocks, $x_n = (v_n, x_n)$, where each block contains a subset of the parameters. At each iteration, the components of each particle can be moved with different forward kernels. Moving components of particles around the state space through Gibbs and M-H moves are attractive. That is, one can use the full conditional distributions, M-H steps, or a combination of the two as the forward Markov kernels for different components. Intuitively, the sweep of components in a particle is similar to a sweep of draws within one iteration of an MCMC algorithm. Although DDJ (2006) discuss many possibilities, I focus only on the moves that are relevant for DSGE models.

As discussed in section 2.2, the only practical choice for the forward Markov kernel in a DSGE

model is a M-H move. This can be an independent M-H move or a random-walk Metropolis move. Approximating the optimal kernel can then be achieved by choosing the backwards kernel as

$$L_{n-1}\left(x_n, \mathrm{d}x_{n-1}\right) = \frac{\pi_n\left(x_{n-1}\right) K_n\left(x_{n-1}, x_n\right)}{\pi_n\left(x_n\right)} \tag{21}$$

which means the incremental weights for this move are

$$\tilde{\omega}_n\left(x_{n-1}, x_n\right) = \frac{\gamma_n\left(x_{n-1}\right)}{\gamma_{n-1}\left(x_{n-1}\right)} \tag{22}$$

As mentioned in DDJ (2006), this choice can be viewed as an "all-purpose" move and will be appropriate as long as the forward Markov kernel is an MCMC move that leaves the particles π_n-invariant. The incremental weights in (22) are independent of the particles simulated at iteration n and one should calculate them beforehand. Particles could also be moved using random walk proposals that are always accepted but this is less likely to be successful. The incremental weights (22) would no longer apply either; see DDJ (2006) for a discussion of this case.

Another alternative SMC method one might consider are the Population Monte Carlo (PMC) schemes of Cappé *et al.* (2004), and Douc *et al.* (2007). These authors use a normal mixture proposal distribution with the components of the mixture adapting through iterations. PMC can be shown to be a special case of an SMC sampler for specific choices of the forward and backward kernels. Using a Markov kernel that is a mixture has strong potential for DSGE models.

4 Algorithm settings

4.1 SMC samplers based on simulated tempering

The first SMC sampler I design uses all of the data at each iteration and the densities in the sequence $\pi_1\left(x\right) < \pi_2\left(x\right) .... < \pi_p\left(x\right)$ differ only based on a simulated tempering sequence $0 \leq \zeta_1 < ... < \zeta_p = 1$ known as a cooling schedule.[5] Simulated tempering raises a function, here the posterior density, to a power less than one.

$$\pi_n\left(x\right) = \left[\pi\left(x\right)\right]^{\zeta_n} = \left[p\left(\Theta|y_{1:T}\right)\right]^{\zeta_n}$$

[5]See Gelman *et al.* (2004) for an introduction to simulated tempering.

At smaller values of ζ_n, the density $\pi_n(x)$ is flatter and particles can move around the state space more freely. In the simulated tempering literature, the particles are "hotter" because of this free movement. As ζ_n gradually gets larger, the densities within the sequence get closer to the posterior density and will be equal to the posterior density when $\zeta_p = 1$. Particles are no longer able to move around the state space as easily and hence they are "cooled."

Choosing the first importance density $\eta_1(x)$ to initialize the particles may differ from one application to the next. Del Moral *et al.* (2006b) and Jasra *et al.* (2005) use the prior distribution for the parameters of their model as the initial importance density. An alternative initialization strategy might be to compute the mode(s) and the curvature at the mode(s) of the posterior density and then approximate this with a heavy tailed density such as a Student's-t. This strategy is often used for the proposal density in independence M-H and IS algorithms; see, e.g. An and Schorfheide (2007). Both strategies require checking that all initial particles are drawn from the determinacy region of the DSGE model. I tried both strategies and found that each of them will result in an immediate reduction in the number of effective particles. Ultimately, the important point is to try to initialize particles from an initial distribution wider than the posterior distribution to help ensure the validity of IS. In the remainder of this paper, I initialize all the simulated tempering algorithms with the latter method. Note that the importance weights can be explicitly computed for the first iteration.

After the initial iteration, choices for the forward and backward Markov kernels are roughly limited to M-H moves for DSGE models. After implementing many alternative algorithms, I concluded that simple random-walk Metropolis moves performed well. Two issues to consider are whether all the components of Θ should be moved at once or in smaller sub-blocks and what are the best covariance matrices for the random walk steps. Moving all the components of Θ individually or in small blocks would be ideal. This will generally increase the diversity of particles. However, this substantially increases the number of times the likelihood must be calculated and the number of tuning parameters. In my experience, moving all the particles at once or in a small number of blocks (2 or 3) provided an efficient use of computing time for the simulated tempering approach. In section 5, my SMC sampling algorithm based on simulated tempering moved all components of Θ in one block.

For the covariance matrices on the normally distributed random walk proposals, I use the particle system to compute the empirical estimates

$$E_n\left[\Theta\right] = \frac{\sum_{i=1}^{N} W_n^{(i)} \Theta^{(i)}}{\sum_{i=1}^{N} W_n^{(i)}} \quad V_n\left[\Theta\right] = \frac{\sum_{i=1}^{N} W_n^{(i)} \left(\Theta^{(i)} - E_n\left[\Theta\right]\right)\left(\Theta^{(i)} - E_n\left[\Theta\right]\right)'}{\sum_{i=1}^{N} W_n^{(i)}} \tag{23}$$

of the covariance matrix $V_n\left[\Theta\right]$ at each iteration. The idea for using this as the covariance matrix comes from Chopin (2002). The acceptance rates for this type of random walk proposal were typically in the 30-50% range. To ensure they remain there, I have the algorithm appropriately adjust a scale parameter on the covariance matrix if successive iterations' acceptance rates are too low or high. Consequently, the SMC sampler I have built is able to tune itself. The only tuning parameters that need to be chosen by the user are the simulated tempering parameters, which I discuss below.

Using the above random-walk Metropolis move means that the incremental weights (22) can be applied, which in this particular case are

$$\tilde{\omega}_n\left(x_{n-1}, x_n\right) = \frac{\pi_n\left(x_{n-1}\right)^{\zeta_n}}{\pi_{n-1}\left(x_{n-1}\right)^{\zeta_{n-1}}} = \frac{p\left(\Theta_{n-1}|y_{1:T}\right)^{\zeta_n}}{p\left(\Theta_{n-1}|y_{1:T}\right)^{\zeta_{n-1}}} \tag{24}$$

All calculations should be done in logarithms for computational stability making (24) reduce to

$$log\left[\tilde{\omega}_n\left(x_{n-1}, x_n\right)\right] = \left\{\zeta_n - \zeta_{n-1}\right\} log\left[\pi\left(x_{n-1}\right)\right] = \left\{\zeta_n - \zeta_{n-1}\right\} log\left[p\left(\Theta_{n-1}|y_{1:T}\right)\right]$$

Implementing this SMC sampler requires only slightly more coding than the original MCMC sampler as one only needs to implement a resampling algorithm and compute the covariance matrices in (23). This sampler is provided as algorithm 2.

4.2 SMC samplers for sequential Bayesian estimation

Algorithms for sequential Bayesian estimation are considerably more difficult to design than the simulated tempering approach. There are several reasons why. Parameters in DSGE models are likely to be highly unstable. The values of some of the parameters may change substantially in a small number of observations. Regardless of whether the data is simulated or real, adding an observation at each iteration is unlikely to change the value of all the components of a particle. However, it will often impact sub-blocks of it.

These facts have several consequences. Using the particle approximation of the distribution at iteration $n-1$ to create an importance density for the next iteration (as in simulated tempering

above) may not work. Two neighboring posterior distributions within the sequence may be quite different. The algorithm needs to create an importance density to approximate the next density rather than the posterior at the last iteration. Moving all the parameters in one block will likely be ineffective as well. Sequential estimators will likely need to be tailored for each model rather than the generic approach with simulated tempering above.

For example, I implemented Chopin's (2002) sequential estimator and unfortunately it did not perform well for the DSGE model considered here. His algorithm moves all the components of a particle in one block by a forward Markov kernel that is an independent M-H proposal. The particles are moved only when the ESS falls below a given threshold. And, the mean and covariance matrix of the M-H move are equal to the mean and the covariance matrix at the previous iteration; i.e. given by (23). I found that the parameters within the DSGE model are too unstable for this forward kernel. Chopin's algorithm did not propose particles far enough into the state space at each iteration. Empirical estimates of the covariance matrix gradually got smaller at each iteration, while the acceptance rates gradually increased. Eventually, the tails of each marginal distribution began to be severely underestimated. I implemented other algorithms that all the components of a particle in one block and each worked poorly.

Consequently, I built an algorithm whose forward Markov kernel moves all the components individually at each iteration. I found that individual moves led to better estimates as it increases the accuracy in the tails of the distribution. Each sub-component of the forward kernel is a random walk Metropolis step with a normal distribution for the proposal. The scales on the proposal of each component are allowed to adapt over time in order for the acceptance rates to remain in the 30-50% range. This can be implemented with a simple conditional statement in the code.

While this improved the estimates substantially, the posterior means for some of the parameters at the final iteration were not in agreement with those from the batch algorithm. The parameters in the model still change too significantly at occasional points in time. The forward kernel currently proposes moves only locally and does not account for large changes in parameters. Consequently, I implement a forward Markov kernel using a mixture of normal distributions. The idea is simply to have one component of the mixture explore locally while another component proposes large moves. This idea originates in work from Cappé *et al.* (2004) who use a mixture kernel in a batch estimation setting.

The mixture I use has two components each of which is determined by an indicator function

that gets drawn first for each particle at each iteration. With probability α, I move all components of a particle in individual blocks using a random-walk Metropolis step as above. In the second component of the new forward kernel, all components of a particle are moved jointly using an independent M-H step. This occurs with probability $1-\alpha$. The proposal distribution for this move is a normal distribution whose mean and covariance matrix are computed at the previous iteration; i.e. given by (23). I set $\alpha = 0.95$ for all iterations, although it is possible to have time-varying random/deterministic probabilities on the components of the mixture. The purpose of this step is to propose large moves to accomodate large changes in parameters. The acceptance rates for this component are expected to be quite low.

The target density at iteration n is now defined as $\pi_n(x) = p(\Theta|y_{1:n})$. As the forward Markov kernel is an MCMC move that leaves the system π_n-invariant, the incremental weights (19) can be applied. They are

$$\tilde{\omega}_n(x_{n-1}, x_n) = \frac{\pi_n(x_{n-1})}{\pi_{n-1}(x_{n-1})} = \frac{p(\Theta_{n-1}|y_{1:n})}{p(\Theta_{n-1}|y_{1:n-1})} \tag{25}$$

In section 5.3, I initialize the sequential algorithm starting with 35 observations. For an initial importance density, I run the simulated tempering algorithm from section 4.1 using the first 35 observations for a small number of iterations p. This leads to a set of particles that accurately represent the target at observation 35. The incremental weights at the first iteration are equal to 1 as the particles are draws from the initial target.

5 Results

5.1 Simulated tempering SMC sampler on simulated data

I simulated 190 observations out of the linearized EHL model with parameters set equal to the posterior estimates from the actual dataset, which are in Table 3 and will be discussed below. I ran the SMC sampler with simulated tempering (labeled SMC-st) using $p = 500$ iterations and $N = 2000$, 4000, and 6000 particles. The second number of particles results in an algorithm with slightly more computational time than the MCMC algorithm in RR (2005) who used 2 million draws. The systematic resampling algorithm of Carpenter *et al.* (1999) was used to perform resampling when the ESS fell below 50% of the particle size.

For the sequence of cooling parameters, I chose a linear schedule starting at $\zeta_1 = 0$ with differentials $\zeta_n - \zeta_{n-1} = 1/p$ that end with $\zeta_p = 1$. I found that this combination of algorithm parameters resulted in a minor number of resampling steps during a run, typically 6 or less. In general, a user may optimize the performance of the algorithm by altering the schedule. It may be preferrable to have tempering parameters changing slower at the beginning and then gradually increasing. The easiest way to implement this is with a piecewise linear schedule. However, it is interesting to view the performance of the algorithm for the linear choice.

I ran an MCMC algorithm for 5 million draws with an additional burn-in of 200,000 iterations. The MCMC algorithm uses a random-walk Metropolis proposal at each iteration (the choice of the covariance matrix for the proposals is discussed below). I then used every 2,000th draw to compute the estimates shown in Table 1. I also report estimates from a regular IS algorithm with 50,000 draws. The IS algorithm followed An and Schorfheide (2006) who suggested using a Student's-t distribution as an importance density with the mean set at the posterior mode and covariance matrix equal to a scaled version of the asymptotic covariance matrix. The degrees of freedom were set to three while I tried many different scale parameters on the covariance matrix. Setting it equal to 1.75 performed the best.[6]

Table 1 provides estimates from the SMC-st, MCMC, and regular IS algorithms. It is easy to see that the IS algorithm estimates the posterior density poorly for several of the parameters (σ, θ_p, γ, σ_a, σ_λ, σ_g). In particular, the estimates of the standard deviations are poor, which is typical when applying IS on a higher dimensional problem. Looking at the parameter estimates alone does not provide a complete picture of the performance of the algorithm. Table 2 reports the ESS which indicates that only 8 draws out of 50,000 are contributing to the IS estimator. Meanwhile, the ESS values are significantly higher for SMC-st. Paths of the ESS over iterations of the SMC-st algorithm are pictured on the left-hand side of Figure 1. On this simulated dataset, the SMC-st algorithm resampled six times regardless of the particle size but the behavior is quite regular. Interestingly, the mean of the importance weights is almost equal to the theoretical equivalent of $W = \frac{1}{N}$. Convergence of the mean toward this value is plotted recursively in Figure 3. This indicates that the importance density at the final iteration is almost equal to the posterior.[7]

[6]The posterior mode and asymptotic covariance matrix were found using the BFGS quasi-Newton algorithm written by Chris Sims for Leeper and Sims (1994). I also ran Matlab's standard optimizer which resulted in the same estimates.

[7]This can still be misleading. Although the SMC importance density at the final iteration and the target density are close, there is no guarantee that particles exist in all areas of the support as they are not simulated from this

In addition to the ESS, I computed the Wald and score statistics from Koopman and Shephard (2004) (see their paper for the construction of the tests).[8] The tests are designed to detect if the variance of the importance sampling weights is finite. As noted by Koopman and Shephard (2004), the assumption of a finite variance is almost never checked in either frequentist or Bayesian applications of importance sampling in economics. These values are given in the bottom rows of Table 2, where the 1, 5, and 10% levels indicate the percentage of weights used to calculate the tests. The tests reject the null hypothesis of a finite variance for large positive values of the statistic relative to a standard normal random variable. The existence of a finite variance for the IS algorithm is easily rejected by both statistics. As explained in Robert and Casella (2004), the law of large numbers still holds for the IS estimator but the CLT does not. The convergence of the estimates will be highly unstable and painfully slow. In repeated runs of the algorithm on the same dataset, the estimates vary wildly from one run to another. The importance density for the IS estimator is clearly a poor choice for this model. The mean and mode of the posterior are significantly different for several parameters, indicating that the posterior is highly skewed in some dimensions. It should not be surprising that a symmetric importance density does not perform well.

Koopman and Shephard (2004) also suggested three diagnostic graphs for checking the behavior of the IS weights. These graphs are given in Figure 2 for both the regular IS and SMC-st algorithms. The left-hand graphs plot the largest 100 importance weights to see if there are any outliers. The middle graphs are histograms of the remaining weights while the right-hand graphs picture recursive estimates of the standard deviation of the importance weights. It is clear from the top left graph, that there are 8-9 significant outliers in the IS weights. The outliers have an enormous impact on the recursive estimator of the sample standard deviation. Meanwhile, the weights for the SMC-st algorithm are extremely well behaved with the histograms indicating that they are balanced. The recursive estimator of the standard deviation converges smoothly and rapidly for each particle size.

While in general agreement with the estimates from SMC-st, the MCMC estimates differ for several parameters (σ, σ_g). Interestingly, these are the parameters in the MCMC algorithm whose draws have the highest posterior autocorrelations or cross-autocorrelations. One of the critical

distribution.

[8]The test statistics of Koopman and Shephard (2004) use the fact that importance sampling weights are *i.i.d.*. This does not hold for SMC algorithms. Resampling of the particles causes the importance weights to be correlated. The SMC-st algorithms resample rarely enough that *i.i.d.* importance weights may not be a bad assumption while for a particle filter the assumption is unlikely to hold.

components of my MCMC algorithm concerns how I chose the covariance matrix on the random walk proposals. I chose the covariance matrix (scaled) from the final iteration of the SMC-st sampler. Adopting this covariance matrix caused the autocorrelations of all the parameters in the MCMC sampler to drop by at least a factor of 10. Autocorrelations that were once positive after 5000 iterations are now an order of magnitude smaller. This shows how an SMC sampler can help improve the performance of MCMC.

Correlation does impact the performance of any SMC sampler. Like MCMC, correlation impacts how quickly the algorithm can explore the support of the posterior, which is equivalent to poorer mixing capabilities of the forward Markov kernels. When correlation is higher, the SMC sampler will need to be run for a longer number of iterations p allowing particles time to move around the support. The better the forward Markov kernels mix the smaller the number of iterations that are needed. Jasra *et al.* (2005b) discuss the trade-off between particle and iteration size in exactly the same way. When the iteration size is chosen to be too small, it will cause the tails of the distribution not to be estimated well. This is because as $p \to 1$ an SMC-st sampler gets closer to becoming a regular IS algorithm.

5.2 Simulated tempering SMC sampler on actual data

Table 3 reports the means and standard deviations of the posterior distribution on actual data for MCMC, IS, and SMC-st. The simulated tempering parameters were the same as above. The MCMC and IS algorithms were implemented as above.

The posterior means for each of the parameters are relatively close. Posterior estimates of the standard deviation are considerably different for IS, where several parameters (σ, θ_p, σ_λ, σ_g) are severely missestimated. The ESS provided in Table 4 indicates that only an equivalent of 9 draws out of 50,000 are contributing to the statistical relevance of the estimator. The test statistics and graphical diagnostics from Koopman and Shephard (2004) in Table 4 and Figure 4 indicate that the variance of the weights for this IS algorithm does not exist. Alternatively, the number of effective draws has increased substantially using SMC-st. The test statistics and graphical diagnostics from Koopman and Shephard (2004) in Table 4 and Figure 4 support this conclusion. The histograms of the weights indicate that the algorithm behaves reliably.

Comparing the MCMC and SMC-st algorithms, the estimates are close with the exception of a few parameters, (σ, σ_g). As in section 5.2, these are the same parameters which have higher

posterior autocorrelations and cross-autocorrelations in the MCMC output. Repeated runs of the MCMC algorithm vary more for these parameters. Having another method such as SMC-st for comparison will allow researchers to check their MCMC output. Meanwhile, estimates from the SMC-st sampler are stable across the different particle sizes. Although computational speed is not the main point, it is interesting to note that the SMC-st sampler based on $N = 2000$ and 4000 particles results in quality estimates in reasonable computational times.

5.3 Sequential Bayesian estimation on actual data

I ran the SMC-seq sampler described in section 4.2 with $N = 10000$ particles, which is roughly 4-5 times as computationally intensive as the SMC-st algorithm with $N = 4000$ and $p = 500$. The algorithm resampled when the ESS reached half the particle size. Figures 5-7 provide pictures of the time varying posterior means of the model along with dashed lines that depict the final estimates from the SMC-st algorithm with $N = 4000$. In addition to comparing the final estimates, I also compare output from SMC-st algorithms run on smaller batches of data at specified times: $t = 50$, 75, 100, 125, 150. The mean estimates for each of these time points are depicted by small boxes in each figure.

It is apparent that almost all the posterior means from the sequential algorithm end at the same estimates as the batch algorithms. The point estimates reported in Table 3 are all quite close. The time-varying posterior means also agree with most of the batch estimates at the pre-selected time points. Variability of the final estimates from this algorithm relative to the SMC-st and MCMC estimates remains reasonable. Figure 7 depicts the evolution of the ESS over time. It indicates that the posterior distribution of this DSGE model is highly unstable during the periods 1970-1975 and 1980-1984. Degeneracy of the particle system during these periods will cause the parameter estimates to be less precise. The SMC-seq algorithm can be redesigned to correct for this degeneracy but leaving it unchanged (at present) provides additional information on the model.

The posterior paths show that there is significant movement in the posterior distribution earlier in the algorithm. Researchers should hesitate to interpret all of this movement as structural. First, it is possible that the earlier movement may be due to smaller sample sizes as the effect of the prior slowly wears off and contributions to the likelihood are more informative. Despite this possibility, most of the parameters do not begin near their respective prior means at the first iteration (with 35 observations). Secondly, experiments on simulated data indicate that several parameters are not

well-identified (e.g. σ, σ_g). On simulated data, these parameters will vary considerably even when they are known to be fixed.

Over half of the parameters' paths change significantly during the early 1980's. The economics literature has documented that there is a break in macroeconomic volatility commonly called the *great moderation* in roughly 1984; see, e.g. Kim and Nelson (1999) and McConnell and Pérez-Quirós (2000). The monetary and technology shocks break upward significantly during the early 1980's and then beginning in 1984 gradually decline until today. This supports the conclusion of Fernández-Villaverde and Rubio-Ramírez (2007), who argue that the decline in macroeconomic volatility was gradual rather than an abrupt change in 1984. Shocks to preferences and the price level appear to increase gradually. Meanwhile, the persistence of shocks appears to have increased since 1980. This is consistent with the observation made by Kim (2006).

The monetary authority's behavior also changed through time. Their tendency for interest rate smoothing fell abruptly in the early 1980's and gradually increased over Greenspan's tenure. The late 1970's and early 1980's indicate that the Fed switched abruptly to punishing deviations of inflation from its target while becoming less concerned with the output gap. The Taylor rule appears to be relatively stable in the last half of the sample while the persistance of preference shocks appears to have increased since 1980. All of these implications are conditional on this set of priors. I have not experimented with the priors to determine how significant their influence is and how much they impact the results. I leave this to future work as my focus here is on building appropriate SMC algorithms.

6 Conclusion

Researchers interested in applying Bayesian methods to difficult computational problems were often restricted to building MCMC algorithms. In this paper, I developed several SMC samplers and compared their performance with methods in the existing literature. I demonstrated how SMC can be used as an alternative to MCMC or as an exploratory devise to help build better MCMC algorithms. The sequential Bayesian algorithm is entirely new. It provides additional information on the time variation of parameters and stability of the model.

This paper also underscores that implicit assumptions behind the use of standard IS algorithms need checked. Using the ESS as well as the test statistics and graphical diagnostics from Koopman

and Shephard (2004), the variance of the weights in the IS algorithm almost certainly did not exist. Meanwhile, the weights from the SMC-st algorithm are almost equal, indicating that the importance density at the final iteration is close to the posterior.

There are a number of extensions to this paper that can be explored in future work. Demonstrating the effectiveness of this strategy on larger scale models is practically important. An and Schorfheide (2006) note that many central banks are currently building DSGE models containing more observables and parameters than the model considered here. The sequential algorithm in this paper is also a step toward developing Bayesian learning algorithms in the spirit of Evans and Honkapohja (2001). It is possible to give the algorithm a learning interpretation. Conditional on time t information, an agent who is uncertain about the parameters of the model solves the model with N different parameter values (one for each particle). Each model then has a probability placed on it (the importance weight) and parameters are estimated by the agent as averages across all the models. Finally, estimation of regime switching models and other models with endogenous structural breaks is critical. It is quite clear that parameters in DSGE models are not constant after WWII and even perhaps since the 1980s. Econometricians often estimate regime switching models assuming the number of regimes is known. However, SMC samplers can be used to determine the number of regimes endogenously. Work in this direction has begun with Jasra *et al.* (2005a).

References

[1] An, S., and Schorfheide, F. (2006). Bayesian analysis of DSGE models. *Econometric Reviews*, forthcoming.

[2] Barndorff-Nielsen, O. E. and Shephard, N. (2001). Non-Gaussian Ornstein-Uhlenbeck-based models and some of their uses in financial economics (with discussion). *Journal of the Royal Statistical Society*, Series B, Vol. 63, No. 2, pp. 167-241.

[3] Calvo, G. (1983). Staggered prices in a utility maximizing framework. *Journal of Monetary Economics*, Vol. 12, pp. 383-398.

[4] Cappé, O., Guillin, A., Marin, J. M., and Robert, C. P. (2004). Population Monte Carlo. *Journal of Computational and Graphical Statistics*, Vol. 13, pp. 907-930.

[5] Cappé, O., Moulines, E., and Rydén, T. (2005). *Inference in Hidden Markov Models*. New York, NY: Springer-Verlag.

[6] Carpenter, J., Clifford, P. and Fearnhead, P. (1999). An improved particle filter for nonlinear problems. *IEE Proceedings. Part F: Radar and Sonar Navigation*, Vol. 146, No. 1, pp. 2-7.

[7] Chopin, N. (2002). A sequential particle filter for static models. *Biometrika*, Vol. 89, No. 3, pp. 539-551.

[8] Christiano, L., Eichenbaum, M., and Evans, C. (2005). Nominal rigidities and the dynamic effects of a shock to monetary policy. *Journal of Political Economy*. Vol. 113, pp. 1-45.

[9] Del Moral, P. (2004). *Feynman-Kac Formulae: Genealogical and Interacting Particle Systems with Applications*. New York, NY: Springer-Verlag.

[10] Del Moral, P., Doucet, A., and Jasra, A. (2006). Sequential Monte Carlo samplers. *Journal of the Royal Statistical Society*, Series B, Vol. 68, no. 3, pp. 1-26.

[11] Del Moral, P., Doucet, A., and Jasra, A. (2006b). Sequential Monte Carlo for Bayesian computation. *Bayesian Statistics 8*, Oxford: Oxford University Press.

[12] Del Negro, M., Schorfheide, F., Smets, F., and Wouters, R. (2007). On the fit and forecasting performance of New Keynesian models. *Journal of Business and Economic Statistics*, forthcoming.

[13] Douc, R., Guillin, A., Marin, J. M., and Robert, C. P. (2007). Convergence of adaptive sampling schemes. *Annals of Statistics*, forthcoming.

[14] Doucet, A., de Freitas, J. F. G., and Gordon, N. J. (Eds.) (2001). *Sequential Monte Carlo Methods in Practice*. New York, NY: Springer-Verlag.

[15] Durbin, J., and Koopman, S. J. (2001). *Time Series Analysis by State Space Methods*. Oxford, UK: Oxford University Press.

[16] Erceg, C., Henderson, D., and Levin, A. (2000). Optimal monetary policy with staggered wage and price contracts. *Journal of Monetary Economics*, Vol. 46, pp. 281-313.

[17] Evans, G., and Honkapohja, S. (2001). *Learning and Expectations in Macroeconomics*. New Jersey: Princeton University Press.

[18] Fernández-Villaverde, J., and Rubio-Ramírez, J. (2005). Estimating dynamic equilibrium economies: linear versus nonlinear likelihood. *Journal of Applied Econometrics*, Vol. 20, pp. 891-910.

[19] Fernández-Villaverde, J., and Rubio-Ramírez, J. (2007). Estimating macroeconomic models: a likelihood approach. *Review of Economic Studies*, forthcoming.

[20] Gelman, A., Carlin, J., Stern, H., and Rubin, D. (2004). *Bayesian Data Analysis*. Boca Raton, Florida: Chapman & Hall.

[21] Geweke, J. (1989). Bayesian inference in econometric models using Monte Carlo integration. *Econometrica*, Vol. 57, pp. 1317-39.

[22] Gilks, W. R., and Berzuini, C. (2001). Following a moving target - Monte Carlo inference for dynamic Bayesian models. *Journal of the Royal Statistical Society*, Series B, Vol. 63, No. 1, pp. 127-146.

[23] Gordon, N. J., Salmond, D. J., and Smith, A. F. M. (1993). A novel approach to nonlinear/non-Gaussian state estimation. *IEE Proceedings - F*, Vol 140, No. 2, pp. 107-113.

[24] Harvey, A., and Koopman, S. J. (1997). Multivariate structural time series models. In: Heij, C., et. al. (Eds.), System Dynamics in Economic and Financial Models. Chichester: Wiley Press.

[25] Harvey, A., Trimbur, T., and van Dijk, H. (2006). Trends and cycles in economic time series: a Bayesian approach. *Journal of Econometrics*, forthcoming.

[26] Jasra, A., Doucet, A., Stephens, D., and Holmes, C. (2005a). Interacting sequential Monte Carlo samplers for trans-dimensional simulation. *Technical report*, Department of Mathematics, Imperial College London.

[27] Jasra, A., Stephens, D., and Holmes, C. (2005b). On population-based simulation for static inference. *Technical report*, Department of Mathematics, Imperial College London.

[28] Kim, C.-J. (2006). ????? Reference to come shortly.

[29] Kim, C.-J., and Nelson, C. R. (1999). Has the U.S. economy become more stable? A Bayesian approach based on a Markov-Switching model of the business cycle. *Review of Economics and Statistics*, Vol. 81, pp. 608-616.

[30] Kim, S., Shephard, N. and Chib, S. (1998). Stochastic volatility: likelihood inference and comparison with ARCH models. *Review of Economic Studies*, Vol. 65, No. 3, pp. 361-393.

[31] Koopman, S. J., and Shephard, N. (2004). Estimating the likelihood of the stochastic volatility model: testing the assumptions behind importance sampling. *Working paper*, Department of Econometrics and Operations Research, Vrije Universiteit.

[32] McConnell, M. M., and Pérez-Quirós, G. (2000). Output fluctuations in the United States: What has changed since the early 1980's? *American Economic Review*, Vol. 90, pp. 1464-1476.

[33] Leeper, E., and Sims, C. (1994). Toward a modern macroeconomic model usable for policy analysis. *NBER macroeconomics Annual 1994*, ed. by S. Fischer and J. Rotemberg. Cambridge, MA: MIT Press.

[34] Liang, F. (2002). Dynamically weighted importance sampling in Monte Carlo computation. *Journal of the American Statistical Association*, Vol. 97, pp. 807-821.

[35] Liu, J., and Chen, R. (1998). Sequential Monte Carlo methods for dynamic systems. *Journal of the American Statistical Association*, Vol. 93, pp. 1032-1044.

[36] Rabanal, P., and Rubio-Ramírez, J. (2005). Comparing New Keynesian models of the business cycle: a Bayesian approach. *Journal of Monetary Economics.* Vol. 52, pp. 1151-1166.

[37] Ristic, B., Arulampalam, S., and Gordon, N. J. (2004). *Beyond the Kalman filter: particle filters for tracking applications.* Boston: Artech House Press.

[38] Robert, C. P., and Casella, G. (2004). *Monte Carlo Statistical Methods.* Second Edition. New York, NY: Springer-Verlag.

[39] Smets, F., and Wouters, R. (2005). Comparing shocks and frictions in U.S. and Euro area business cycles: a Bayesian DSGE approach. *Journal of Applied Econometrics*, Vol. 20, pp. 161-183.

A Sequential Monte Carlo algorithms

Algorithm 1 Sequential Monte Carlo sampler

1. Initialization
- Set $n = 1$
- For $i = 1, ..., N$ draw $X_1^{(i)} \sim \eta_1$.
- Evaluate $\left\{\omega_1\left(X_1^{(i)}\right)\right\}$ using (18) and normalize these weights to obtain $\left\{W_1^{(i)}\right\}$.

Iterate steps 2 and 3.

2. Resampling
- If ESS$<N_{eff}$ resample the particles and set $W_n^{(i)} = \frac{1}{N}$.

3. Sampling
- Set $n = n + 1$, if $n = p + 1$ stop.
- For $i = 1, ..., N$ draw $X_n^{(i)} \sim K_n\left(X_{n-1}^{(i)}, \cdot\right)$.
- Evaluate $\{\tilde{\omega}_n(x_{n-1}, x_n)\}$ using (19) and normalize the weights using (20).

Algorithm 2 Sequential Monte Carlo sampler with simulated tempering

1. Initialization
- Set $n = 1$
- For $i = 1, ..., N$ draw $\Theta_1^{(i)}$ from either the prior or an approximation of the posterior.
- Compute the initial set of importance weights as in standard IS.

Iterate steps 2 and 3.

2. Resampling
- If ESS$<N_{eff}$ resample the particles and set $W_n^{(i)} = \frac{1}{N}$.

3. Sampling
- Set $n = n + 1$, if $n = p + 1$ stop.
- Compute estimates of the covariance matrix for the random walk move via (23).
- Evaluate the incremental weights using (24) and normalize the weights using (20).
- For $i = 1, ..., N$ draw $\Theta_n^{(i)}$using a random-walk Metropolis move.

B Detrending method

I detrend the real wage and output using a bivariate structural time series model; see, e.g. Durbin and Koopman (2001). The model consists of trend, cycle, and irregular components

$$y_t = \mu_t + \psi_t + \varepsilon_t \;\; \varepsilon_t \sim N\left(\left[\begin{array}{c} 0 \\ 0 \end{array}\right], \left[\begin{array}{cc} \sigma^2_{\varepsilon-y} & 0 \\ 0 & \sigma^2_{\varepsilon-rw} \end{array}\right]\right)$$

The trend μ_t follows a random walk with time-varying slope

$$\begin{aligned} \mu_t &= \mu_{t-1} + \beta_{t-1} \\ \beta_t &= \beta_{t-1} + \zeta_t \qquad \zeta_t \sim N\left(\left[\begin{array}{c} 0 \\ 0 \end{array}\right], \left[\begin{array}{cc} \sigma^2_{\zeta-y} & 0 \\ 0 & \sigma^2_{\zeta-rw} \end{array}\right]\right) \end{aligned}$$

And, I specify a first-order similar stochastic cycle for each series

$$\begin{aligned} \left[\begin{array}{c} \psi_t \\ \psi^*_t \end{array}\right] &= \rho \left[\begin{array}{cc} \cos\lambda & \sin\lambda \\ -\sin\lambda & \cos\lambda \end{array}\right] \left[\begin{array}{c} \psi_{t-1} \\ \psi^*_{t-1} \end{array}\right] + \left[\begin{array}{c} \kappa_t \\ \kappa^*_t \end{array}\right] \\ \left[\begin{array}{c} \kappa_t \\ \kappa^*_t \end{array}\right] &\sim N\left(\left[\begin{array}{c} 0 \\ 0 \end{array}\right], \left[\begin{array}{cc} \sigma^2_{\kappa-y} & 0 \\ 0 & \sigma^2_{\kappa-y} \end{array}\right]\right) \end{aligned}$$

The *similar* stochastic cycle taken from Harvey and Koopman (1997) means that ρ and λ are shared across the output and real wage series. The model was estimated using the MCMC algorithm described in Harvey *et al.* (2006). The MCMC algorithm for this model is well-behaved and a SMC-st sampler using the same forward Markov kernels gave similar results. Priors on the variance parameters are all diffuse while the prior on ρ, the rate of decay of the cycle, was taken as uniform on $[0, 1]$. For λ, I used the "intermediate prior" of Harvey *et al.* (2006), which corresponds to a beta distribution with mode equal to $\frac{2\pi}{20}$. This implies a business cycle with period equal to five years.

C Tables

	Prior distr.	Prior	MCMC	IS	SMC-st	SMC-st	SMC-st
	-	-	-	$N = 50000$	$N = 2000$	$N = 4000$	$N = 6000$
σ^{-1}	$\Gamma(2,1)$+1	2.5 (1.76)	5.68 (1.84)	5.82 (3.30)	5.86 (1.83)	5.73 (1.88)	5.77 (1.91)
$\frac{1}{1-\theta_p}$	$\Gamma(3,1)$+1	3.00 (1.42)	4.65 (0.29)	4.93 (0.17)	4.65 (0.29)	4.65 (0.29)	4.65 (0.29)
$\frac{1}{1-\theta_w}$	U[0, 1)	4.00 (1.71)	2.78 (0.23)	2.93 (0.10)	2.78 (0.23)	2.78 (0.23)	2.78 (0.23)
γ	N(1, 0.5)	1.0 (0.5)	1.60 (0.28)	1.52 (0.15)	1.62 (0.28)	1.62 (0.28)	1.62 (0.28)
ρ_r	U[0, 1)	0.5 (0.28)	0.76 (0.03)	0.75 (0.04)	0.76 (0.02)	0.76 (0.02)	0.76 (0.03)
γ_y	N(0.125, 0.125)	0.125 (0.125)	0.30 (0.05)	0.32 (0.03)	0.30 (0.05)	0.30 (0.05)	0.30 (0.05)
γ_π	N(1.5, 0.25)	1.5 (0.25)	1.16 (0.12)	1.19 (0.11)	1.18 (0.12)	1.18 (0.12)	1.18 (0.12)
ρ_a	U[0, 1)	0.5 (0.28)	0.76 (0.04)	0.80 (0.04)	0.76 (0.05)	0.76 (0.04)	0.76 (0.04)
ρ_g	U[0, 1)	0.5 (0.28)	0.83 (0.03)	0.83 (0.02)	0.83 (0.03)	0.83 (0.03)	0.83 (0.03)
$\sigma_a(\%)$	U[0, 1)	50.0 (28.0)	3.78 (1.00)	4.02 (0.72)	4.25 (1.09)	4.24 (1.08)	4.23 (1.07)
$\sigma_z(\%)$	U[0, 1)	50.0 (28.0)	0.35 (0.02)	0.35 (0.02)	0.34 (0.02)	0.34 (0.02)	0.34 (0.02)
$\sigma_\lambda(\%)$	U[0, 1)	50.0 (28.0)	33.77 (4.22)	34.76 (2.34)	33.69 (4.21)	33.70 (4.16)	33.69 (4.16)
$\sigma_g(\%)$	U[0, 1)	50.0 (28.0)	8.87 (2.25)	12.88 (3.49)	9.11 (2.28)	8.94 (2.30)	8.98 (2.35)

Table 1: Mean and standard deviation of the prior and posterior from the EHL model on simulated data.

	IS	SMC-st	SMC-st	SMC-st
	$N = 50000$	$N = 2000$	$N = 4000$	$N = 6000$
ESS	7.21	1938	3775	5766
Wald (10%)	9.20	-14.47	-20.28	-26.18
score (10%)	10.31	-3.76	-5.32	-6.51
Wald (5%)	9.29	-10.64	-16.16	-22.54
score (5%)	11.26	-2.67	-3.77	-4.62
Wald (1%)	9.56	-7.02	-9.63	-13.16
score (1%)	12.58	-1.19	-1.68	-2.07

Table 2: Effective sample size (ESS) and the Wald and score tests from Koopman and Shephard (2004) on simulated data. Percentages are the percentage of weights used to construct the test.

	Prior	MCMC	IS	SMC-st	SMC-st	SMC-st	SMC-seq
	-	-	$N = 50000$	$N = 2000$	$N = 4000$	$N = 6000$	$N = 10000$
σ^{-1}	2.5	9.12	10.81	9.08	9.06	9.06	9.12
	(1.76)	(2.53)	(5.47)	(2.44)	(2.42)	(2.38)	(2.53)
$\frac{1}{1-\theta_p}$	3.00	4.20	4.17	4.19	4.19	4.18	4.16
	(1.42)	(0.36)	(0.41)	(0.36)	(0.36)	(0.36)	(0.33)
$\frac{1}{1-\theta_w}$	4.00	2.26	2.23	2.25	2.25	2.24	2.24
	(1.71)	(0.22)	(0.25)	(0.22)	(0.22)	(0.22)	(0.21)
γ	1.0	1.91	1.83	1.90	1.91	1.91	1.92
	(0.5)	(0.33)	(0.30)	(0.33)	(0.32)	(0.33)	(0.33)
ρ_r	0.5	0.81	0.80	0.81	0.81	0.81	0.81
	(0.28)	(0.02)	(0.01)	(0.02)	(0.02)	(0.02)	(0.02)
γ_y	0.125	0.13	0.12	0.13	0.13	0.13	0.14
	(0.125)	(0.04)	(0.03)	(0.04)	(0.04)	(0.04)	(0.04)
γ_π	1.5	1.19	1.20	1.19	1.20	1.19	1.20
	(0.25)	(0.13)	(0.08)	(0.12)	(0.12)	(0.12)	(0.13)
ρ_a	0.5	0.80	0.81	0.81	0.81	0.81	0.80
	(0.28)	(0.04)	(0.04)	(0.04)	(0.04)	(0.04)	(0.05)
ρ_g	0.5	0.88	0.89	0.88	0.88	0.88	0.88
	(0.28)	(0.02)	(0.01)	(0.02)	(0.02)	(0.02)	(0.02)
$\sigma_a(\%)$	50.0	2.56	2.50	2.54	2.54	2.53	2.50
	(28.0)	(0.79)	(0.86)	(0.76)	(0.75)	(0.75)	(0.72)
$\sigma_z(\%)$	50.0	0.30	0.30	0.30	0.30	0.30	0.30
	(28.0)	(0.02)	(0.02)	(0.02)	(0.02)	(0.02)	(0.02)
$\sigma_\lambda(\%)$	50.0	22.09	21.81	21.98	21.93	21.81	21.59
	(28.0)	(4.29)	(4.51)	(4.21)	(4.24)	(4.16)	(3.89)
$\sigma_g(\%)$	50.0	10.98	12.48	10.90	10.88	10.81	10.97
	(28.0)	(2.67)	(5.36)	(2.57)	(2.54)	(2.54)	(2.67)

Table 3: Mean and standard deviation of the prior and posterior from the EHL model on the actual data series.

	IS	SMC-st	SMC-st	SMC-st
	$N = 50000$	$N = 2000$	$N = 4000$	$N = 6000$
ESS	8.42	1192	2220	3279
Wald (10%)	34.46	-8.44	-15.23	-13.18
score (10%)	24.94	-3.67	-5.20	-6.28
Wald (5%)	9.54	-6.56	-12.70	-10.50
score (5%)	9.50	-2.62	-3.73	-4.49
Wald (1%)	5.24	-3.99	-8.65	-3.86
score (1%)	5.99	-1.17	-1.68	-1.17

Table 4: Effective sample size (ESS) and the Wald and score tests from Koopman and Shephard (2004) on actual data. Percentages are the percentage of weights used to construct the test.

D Figures

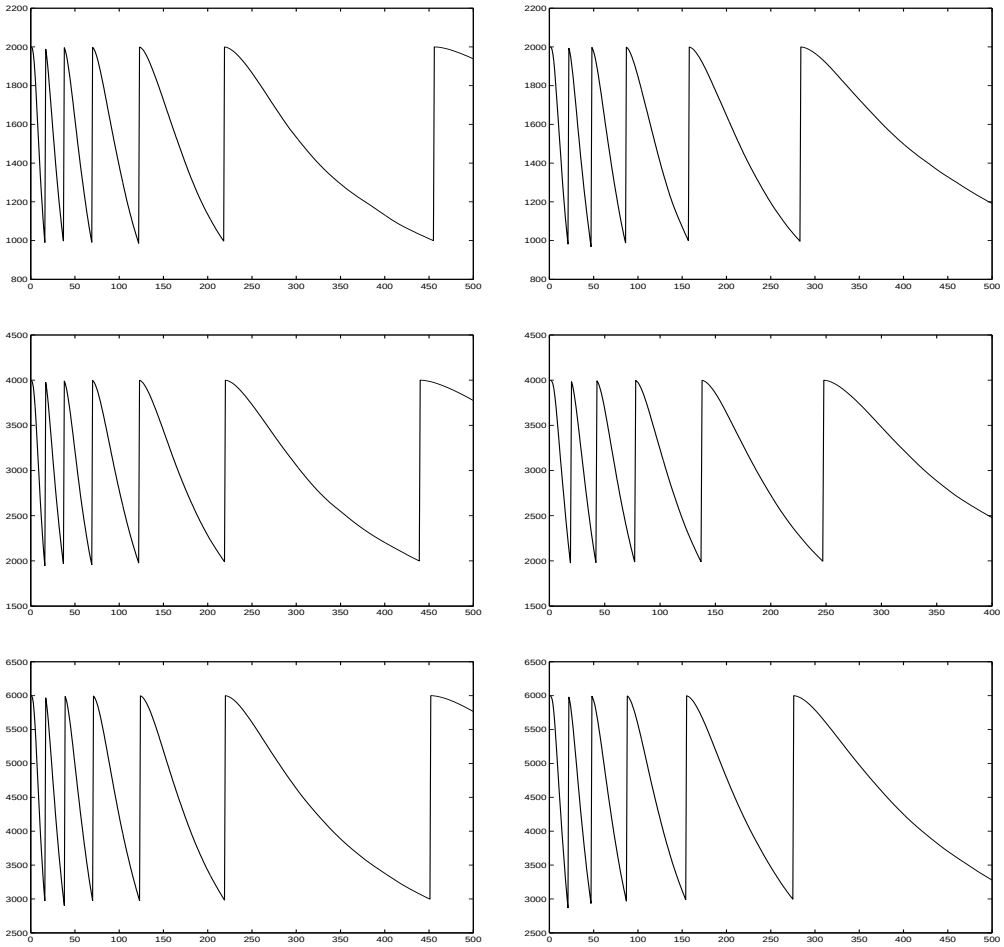

Figure 1: Path of the effective sample size (ESS) for the SMC sampler with simulated tempering on simulated data (left) and actual data (right). Top to bottom: $N = 2000$, 4000, 6000.

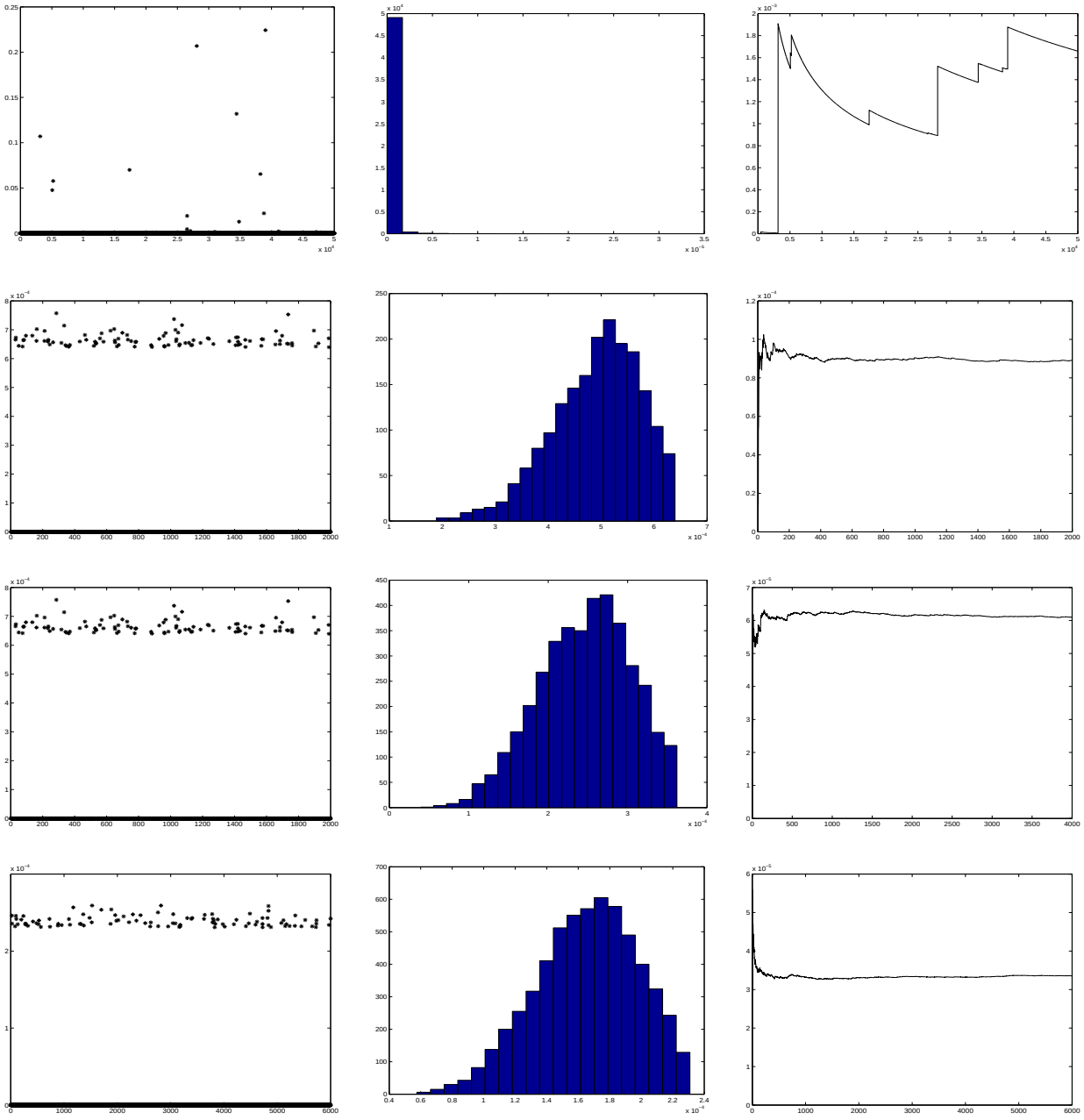

Figure 2: Koopman and Shephard (2004) diagnostics for simulated data. Left: Plots of the largest 100 importance weights. Middle: Histogram of the remaining weights. Right: Recursive estimator of the standard deviation of the weights. Results for regular IS are on top and SMC-st with $N = 2000$, 4000, and 6000 are the second through final rows.

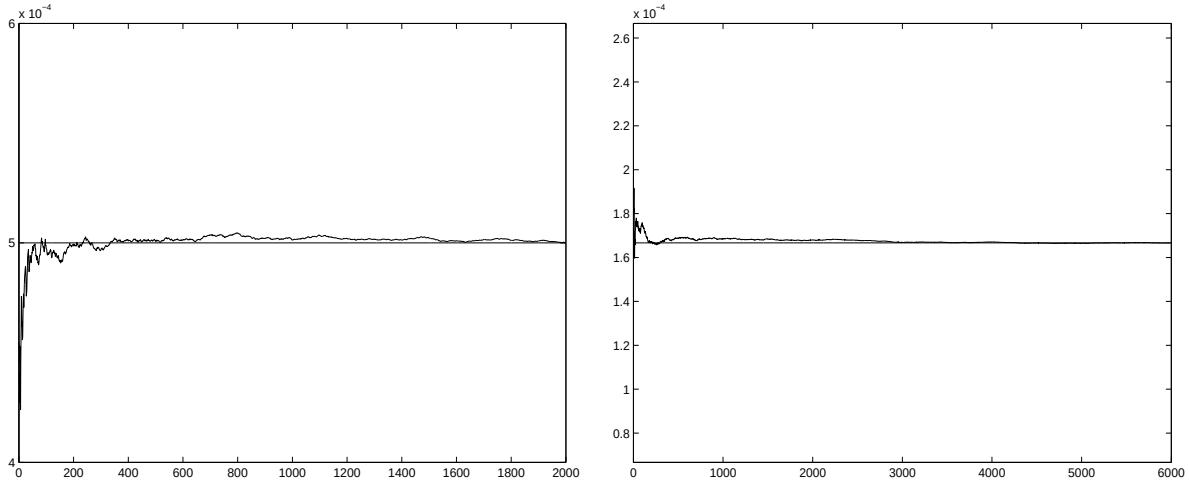

Figure 3: Resursive estimator of the mean of the importance weights for $N = 2000$ and 6000 particles on simulated data. The line indicates weights of $W = \frac{1}{N}$.

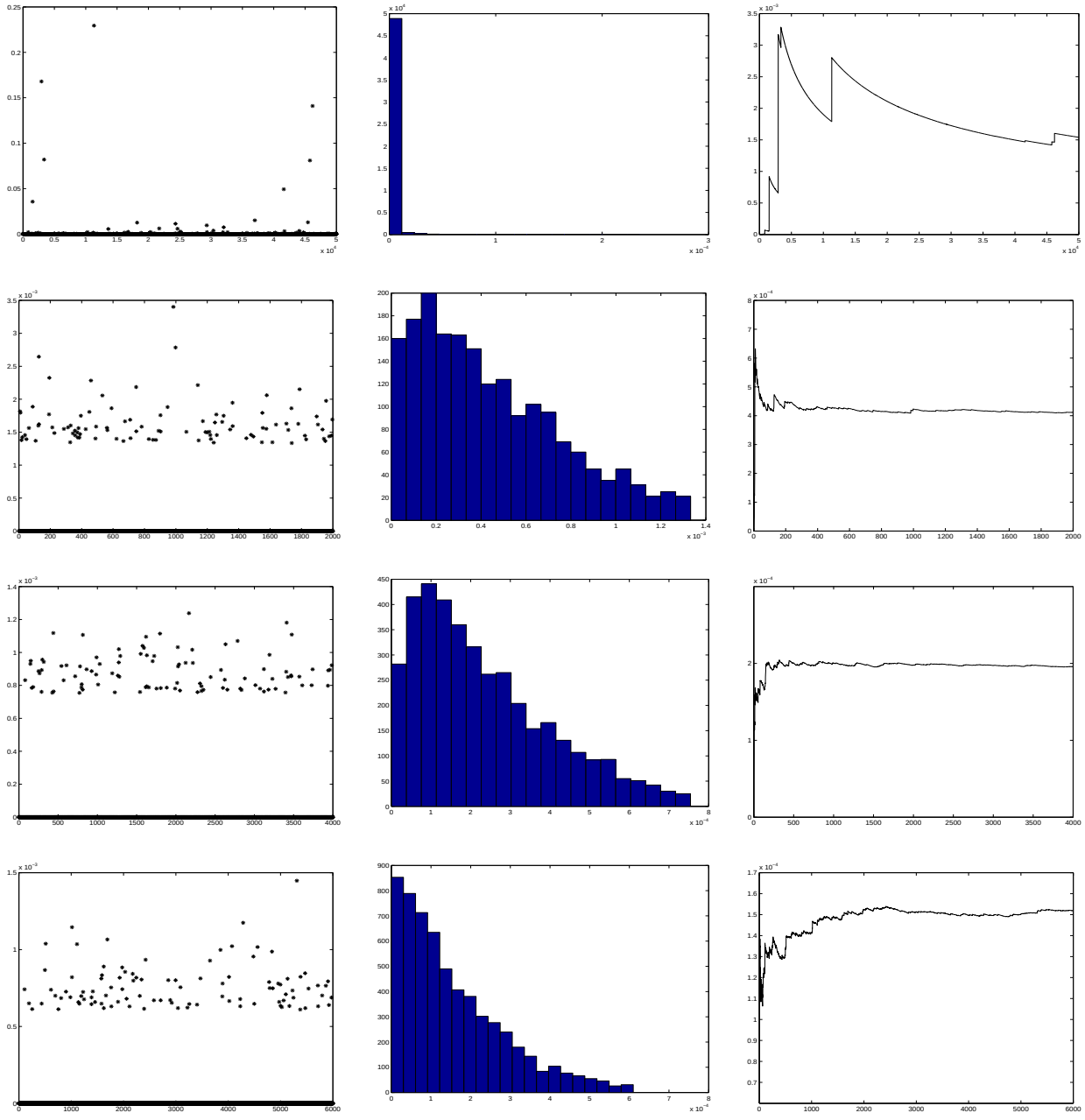

Figure 4: Koopman and Shephard (2004) diagnostics for actual data. Left: Plots of the largest 100 importance weights. Middle: Histogram of the remaining weights. Right: Recursive estimator of the standard deviation of the weights. Results for regular IS are on top and SMC-st with $N = 2000$, 4000, and 6000 are the second through final rows.

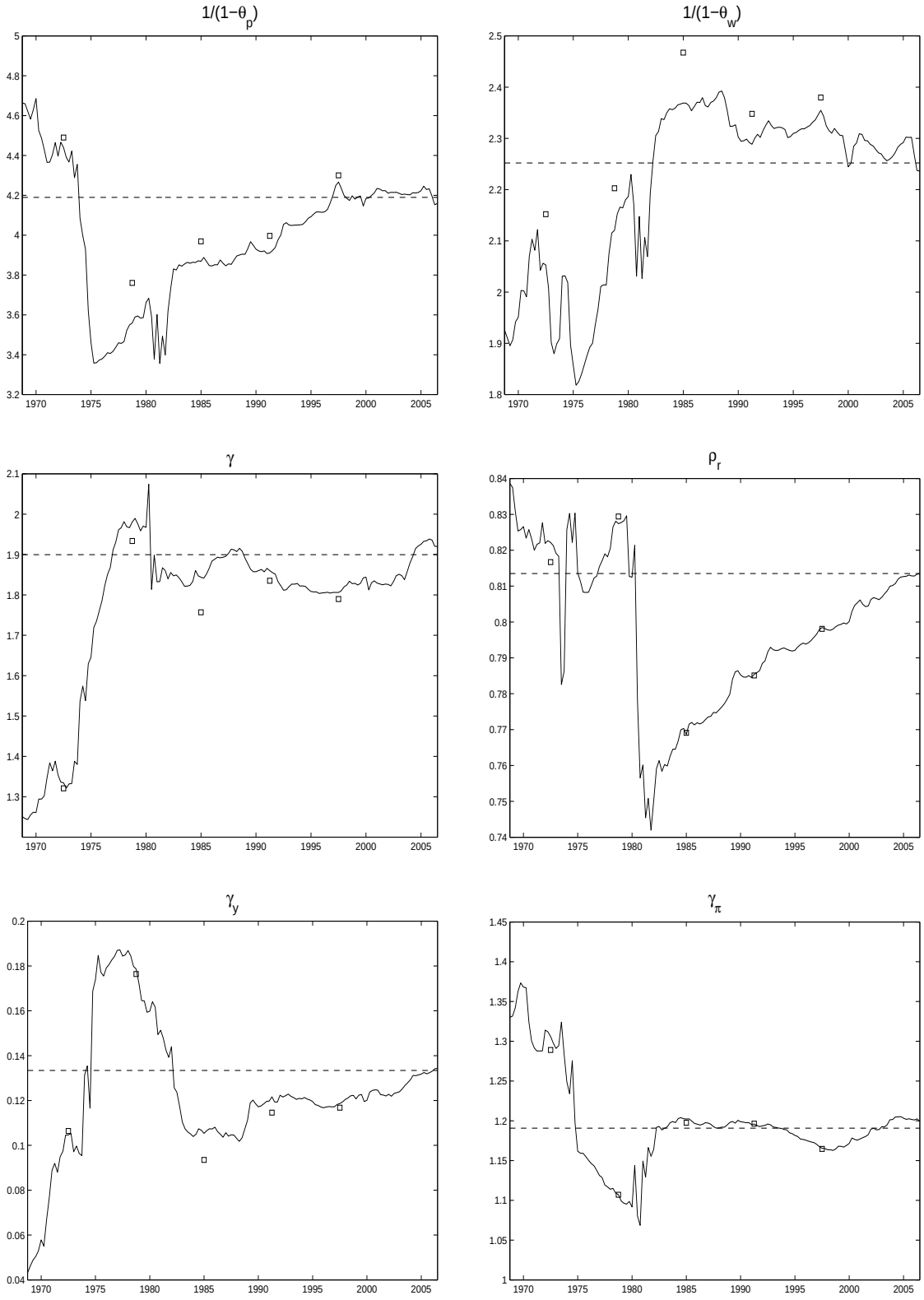

Figure 5: Paths of posterior means from the SMC-seq algorithm on actual data. The dashed line is the final posterior estimate from SMC-st. The algorithm was initialized with 35 observations.

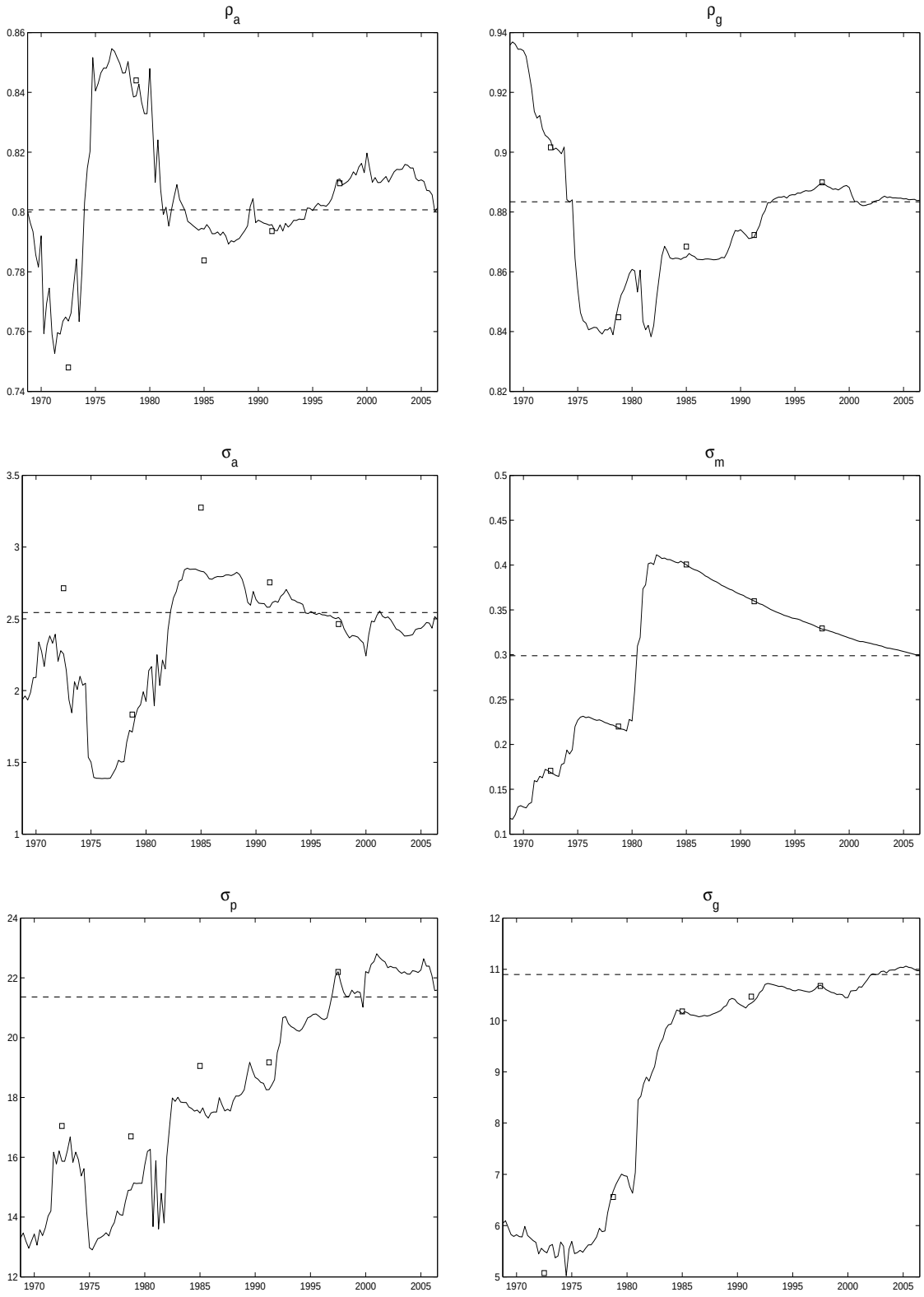

Figure 6: Paths of posterior means from the SMC-seq algorithm on actual data. The dashed line is the posterior estimate from SMC-st. The algorithm was initialized with 35 observations.

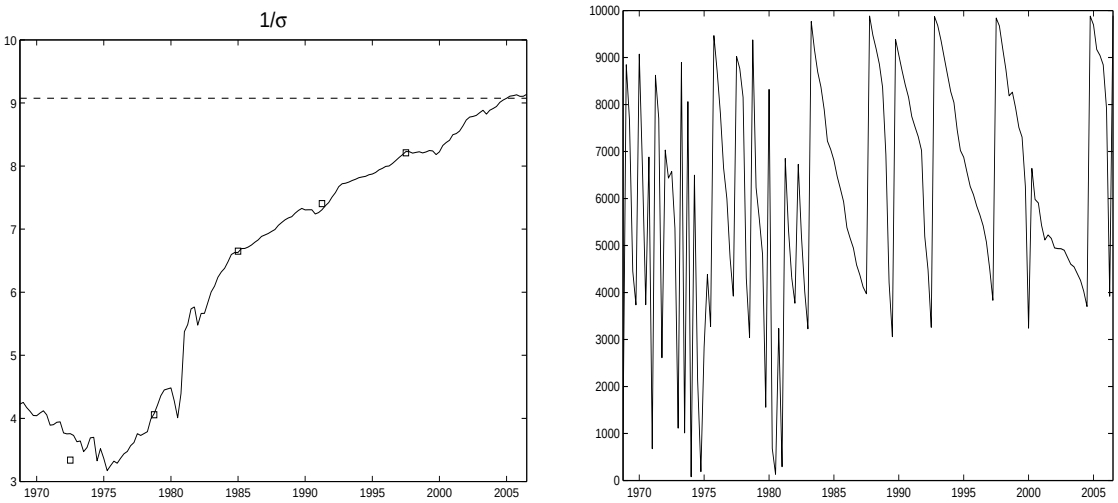

Figure 7: Path of the posterior mean from the SMC-seq algorithm on actual data (left). Path of the ESS over time (right).